1 Einführung

In diesem Buch möchte ich Software von Daniel J. Bernstein (kurz auch einfach DJB genannt) vorstellen und ihren Einsatz diskutieren. Im Zentrum steht hierbei das E-Mail Paket Qmail, aber auch die Daemontools, UCSPI sowie DJBDNS werden in einiger Ausführlichkeit erläutert. DJB-Software läuft in der Regel unter Unix, eine Portierung auf andere Betriebssysteme ist mir nicht bekannt; obwohl prinzipiell möglich.

Im Zusammenspiel zwischen Usabiltity, Simplicity und Safety orientiert sich Daniel J. Bernstein mit seinen Softwareentwicklungen ganz klar in Richtung Sicherheit. D.J. Bernstein verfolgt mit seiner akademisch geprägten Software die Philosophie "klein aber fein". Als Mathematikprofessor analysiert er Anwendungen in einem Ansatz, den man als getragen von "first principles" also erstrangigen Prinzipien verstehen kann – und entwickelt seine Lösungen auch entsprechend. Dies beinhaltet, dass seine Softwareprodukte – heissen sie nun Qmail oder DJBDNS – einerseits genau das tun, was sie müssen und gefordert ist, andererseits dass sie das genau tun, was zu tun ist. Als Mathematiker würde man formulieren, dass DJB-Software für eine gegebene Anforderung eine *hinreichende* und *vollständige* Lösung darstellen.

Sich auf DJB-Software einzulassen, bedeutet somit zunächst, sich mit den Anforderungen und den Randbedingungen auseinanderzusetzen und diese zu verstehen. Erst in diesem Kontext kann die Lösung verstanden und beurteilt werden. Betrachten wir z.B. E-Mail Software. Hier steht eine "Lösungs-mannigfaltig" in Form von Sendmail, Qmail, Exim, Postfix und vielen anderen kommerziellen Produkten für Unix bereit. Analysieren wir die Produkte aber bis auf ihren Kern, d.h. die Implementierung in der Sprache C, muss konstatiert werden, dass Qmail die sicherste Kode-Basis mitbringt. Produkte wie Sendmail können sicherlich mehr, sind also mächtiger; Postfix bietet sicherlich Vorteile in der Usability. Kommt es aber auf die saubere Implementierung an, punktet Qmail. Dies betrifft in der Regel das Softwaredesign und die Kodierung.

Diese Unterschiede schätzen, beurteilen und erfolgreich einsetzen zu können, soll einer der Aufgaben von "Qmail & Co" sein.

1.1 Aufbau und Inhalt des Buchs

Dieses Buch richtet sich an unterschiedliche Qmail- (und Nicht-Qmail-)Benutzer. Es dient sowohl als Referenzwerk über die besprochenen Software-Pakete von Dan Bernstein, als auch als praktischer Leitfaden.

- In diesem Kapitel 1 (O) steht nach einer Orientierung über den Software-Autor Daniel J. Bernstein und sein Tätigkeitsfeld zunächst im Vordergrund,

wie Qmail und die anderen Software-Pakete am besten bezogen werden können. Qmail und alle anderen Entwicklungen von Dan Bernstein sind *Freeware*. Häufig stellt sich aber speziell für den kommerziellen Einsatz die Frage: Darf ich die Software überhaupt einsetzen und ggf. nach meinen Bedürfnissen modifizieren? Dies soll anhand der Diskussion über das Urheberrecht geklärt werden. Dem schliesst sich eine Argumentation über Patente und im besonderen Software-Patente an.

- Das Kapitel 2 (E) soll als Einstieg in das Betriebssystem Unix dienen. Hier werde ich besonders mit den Eigenheiten von Linux und FreeBSD auseinandersetzen. Selbstverständlich gibt es viel bessere und umfangreichere Darstellungen über Unix; mir kommt es aber darauf an, die Diskussion in Bezug auf die *Systemadministration* zu führen. Dies ist notwendig, damit man über die Fallstricke hinsichtlich der Installation und dem Aufsetzen von Qmail & Co. weiss.

- Im Kapitel 3 (E) findet sich ein Abriss über die besprochenen Anwendungen E-Mail und DNS (Domain Name System). Auch dies soll nur als begleitende Information verstanden wissen; wobei ich aus eigener Erfahrung aus weiss, dass gute Bücher über TCP/IP und die darauf basierenden Applikationen selten sind und vom Inhalt her auch schnell veralten.

- Kapitel 4 (R) fungiert als Einstieg in die Welt der DJB-Software. Hier stelle ich das Pakete UCPSI (Unix Client/Server Program Interface) vor. Typischerweise werden Teile hiervon als Ersatz für den sog. INETD unter Unix eingesetzt.

- Aufgabe des Kapitel 5 (R) ist die Vorstellung der Daemontools von Dan Bernstein. Mit den Daemontools wird eine neue Qualität der Systemadministration erzielt. Dies ist die (mir bekannte) erste und umfassende Darstellung der Daemontools.

- Kapitel 6 (R) widmet sich unserem Schwerpunkt: Qmail. Hierin wird das Einrichten und Aufsetzen sowie die Arbeitsweise von Qmail en datail besprochen.

- Das Kapitel 7 (A) ergänzt diese Informationen um konkrete Anwendungen. Hierunter zählen z.B. der Einsatz von Qmail für Internet E-Mail Gateways. Massnahmen zur Spam- und Virenabwehr, Analyse der Logfiles, Aufsetzen von virtuellen Domains und anderes.

- Kapitel 8 (R) beschreibt das Paket DJBDNS, d.h. der Ersatz für den Berkeley Internet Name Daemon BIND. DJBDNS besteht aus einer Sammlung von Programmen die jeweils spezifische Aufgaben von BIND ersetzen. Das Aufsetzen von DJBDNS verlangt die vorige Installation der Daemontools.

- In Kapitel 9 (A) schliesslich soll der Einsatz des "Gesamtpaket" der DJB-

Software unter besonderer Berücksichtigung von Sicherheitsaspekten besprochen werden.

Zur leichteren Orientierung habe ich bei der Aufzählung der Kapitel die Buchstaben, A, E, R und O vergeben. O steht herbei für Orientierung, E für Einführung, R für Referenz und A für Anwendung.

1.1.1 Typographische Konventionen

qmail oder Qmail? In diesem Buch geht es häufig um technische Beschreibungen von Software. Dies korrekt und konsistent in Sprache abzubilden (speziell der Deutschen) ist nicht ohne Risiko. Häufig wird sich bereits darum gestritten, ob qmail, nun Qmail heisst oder qmail; auf keinen Fall jedoch QMAIL oder gar QMail. Eine korrekte Benennung mache ich jedoch nicht zur Philosophiefrage, sondern vom Kontext abhängig:

- Produkte, das heisst eine gebündelte Ansammlung von Software mit einem Ziel (Anwendungssoftware), sollen mit grossen Anfangsbuchstaben anfangen. Also z.B. Qmail, Sendmail, Fetchmail, etc. Ausnahmen werden dort gemacht, wo der Name bereits eine Abkürzung beinhaltet, wie bei DJBDNS oder UCSPI.
- Unix-Dateien werden normalerweise "monospaced" (gesperrt) dargestellt, also z.B. `qmail-send.c`.
- Unix-Verzeichnisse sind natürlich auch Dateien, enden aber immer mit einem abschliessenden Slash "/" beispielsweise `/var/qmail/` . Eine Ausnahme muss ich da machen, wo Verzeichnisse als Argument im Aufruf von Funktionen auftreten; dort werden sie i.d.R. ohne abschliessendes "/" referiert.
- Ausführbare Dateien werden im Gegensatz zu normalen Unix-Dateien fett dargestellt. Dies unterscheidet z.B. `qmail-send.c` von **qmail-send**. Der Unterschied zwischen beiden Dateien besteht normalerweise im Bit "Ausführbarkeit" der Datei ("x"). Für Skripte ist es manchmal sinnvoll, die Datei "als solche" zu benennen, anstatt ihre Ausführbarkeit in den Vordergrund zu stellen.
- Benutzer-Accounts werden hingegen kursiv geführt, so z.B. bei den User *root* oder *qmaill*.
- Protokollkommandos, z.B. beim SMTP werden in Grossbuchstaben und monocspaced dargestellt, z.B. `MAIL FROM:`.
- Die E-Mail Header-Felder werden hingegen wie normaler Text behandelt und durch Anführungszeichen gekennzeichnet, beispielsweise "From:", "Delivered-To:".

- Häufig werden auch URLs, also Internet-Links angeführt. Diese werden wie in folgendem Beispiel dargestellt: `http://www.fehcom.de/qmail.html`

1.1.2 Beispiele und ihre Verwendung

In diesem Buch wird ausführlich von Beispielen Gebrauch gemacht. Diese treten uns in verschiedenen Formen entgegen:

- **Skripte:** Die Skripte entstammen meistens meinem eigenen Fundus und wurden für dieses Buch neu überarbeitet und getestet (Linux/System V und BSD). Die Skripte sind entweder Plattform-unabhängig oder als Plattform-spezifisch gekennzeichnet.
- **Beispiele:** Ich habe mir erlaubt — um dem Leser den Vergleich zu vereinfachen — viele Beispiele von Dan Bernstein, soweit sie in seinen Dokumenten zu finden sind, zu übernehmen und zu kommentieren. D. J. Bernstein verwendet in seinen Beispielen konsequenterweise fiktive Adressen und Namen "foo-bar", "example.com", etc..
- **Listings:** Hierunter verstehe ich Bildschirmlistings, die von meiner Test-Umgebung einfach per "paste&copy" in das Buch übertragen wurden. Dem Leser wird immer wieder der Rechnername "qmailer" begegnen, der sich in meinem privaten Netzwerk befindet.
- **Shell**: Viele der Beispiele erfordern den Einsatz der Kommandos durch den Benutzer *root*. Die Root-Shell soll durch "#" kenntlich gemacht werden, während eine normale Benutzershell durch ein "%" eingeleitet wird.

1.2 Wer ist Daniel J. Bernstein?

"Bernstein sucks!"[1] Über Daniel J. Bernstein kursieren umfangreich Informationen im Internet, aber keine führt auf einen Hinweise, wie sein zweiter Vorname (engl. *middle initial*) lautet. Doch was ist über Dan Bernstein bekannt?

Dan Bernstein hat an der University of California at Berkeley Mathematik studiert und dort in "Fast Arithmetics" 1995 promoviert ("Detecting perfect powers in essentially linear time, and other studies in computational number theory"). Dan Bernsteins Arbeiten (und Veröffentlichungen) im Bereich der Arithmetik haben mit der (schnellen) Fakturierung grosser Zahlen durch Primzahlen zu tun; einem Bereich, der von existentieller Wichtigkeit für alle (heutigen) Verschlüsselungsalgorithmen ist. Seine Ausführungen begleitet er durch ihre parallele Realisierung in der Programmiersprache C, wobei er Teile

[1] *Bernstein sucks*! — Häufiger Kommentar von Bernstein-Gegnern

der Standard-C-Biblotheken durch seine eigenen Implementierungen ersetzt hat.

Zuvor ist er bekannt geworden als Schöpfer des Online-Verschlüsselungsprogramms "Snuffle", für das er 1990 (als "graduate Student") versuchte, eine Exportlizenz zu bekommen. Diese wurde ihm (damals) von der US-Behörde ITAR[2] verweigert. Diesen Maulkorb liess er sich selbstverständlich nicht gefallen. Seit dieser Zeit führt Dan Bernstein einen juristischen Kampf gegen die ihm auferlegten Exportbeschränkungen, wie ich weiter unten dokumentieren möchte. Den Namen bzw. die URL seiner eigenen Webseite — cr.yp.to —kann daher durchaus als ironische Anspielung auf die Exportbeschränkungen der USA für Verschlüsselungsalgorithmen angesehen werden.

Andererseits unternimmt er juristischen Anstrengungen, gegen — seiner Meinung nach — ungerechtfertigte Patente im Bereich der Verschlüsselungstechnologien. Wir müssen berücksichtigen, dass zur Zeit seiner Ausbildung an der University of Berkely dies ein "heisses Eisen" war. So wesentliche Durchbrüche wie die Diffie-Hellmann Verschlüsselung und auch das asymmetrische Verschlüsselungsverfahren nach RSA (benannt nach Ronald Rivest, Adi Schamir und Leonard Adleman) wurden erst kurz zuvor entwickelt.

Seine Frustration im Bereich der E-Mail Kommunikation mündeten 1995 im Design der Software *Quick Mail* (Qmail), die er bis zum Jahr 1998 zur "Serienreife" und zu der Alternative zu Sendmail werden liess.

Zu diesem Zeitpunkt hatte er bereits einen Ruf als "Associate Professor" des Fachbereichs für Mathematik, Statistics und Computer Science an der University of Illinos in Chicago erhalten, wo er heute noch im Stand eines "Assistant Professor" lehrt.

Anfang des Jahres 2002 machte er auf sich Aufmerksam, indem er behauptete, dass in Bälde auch der 1024 Bit starke Schlüssel gemäss Diffie-Hellman (DH) und RSA mit spezieller Hardware relativ leicht zu "knacken" sei (`http://cr.yp.to/papers/nfscircuit.ps`).

In einer Kurzbiographie über ihn ist zu lesen: "He is the recipient of three National Science Foundation grants, including a CAREER award on Computational Number Theory, Cryptography, and Computer Security. He currently has seven computers on this desk and doesn't think that's enough." (`http://www.butler.edu/cs/seminars/oldstuff/Bernstein.shtml`)

Dan Bernstein's mitunter harsche Kritik an den bestehenden Implementierungen im Unix-System haben ihm nicht nur Freunde geschaffen. Bekannt sind z.B. seine Dialoge mit dem Softwareautor (u.a. von Postfix und SATAN) Wietse

[2] ITAR — International Traffic in Arms Regulations

Venema, dem er eine eigene Webseite gewidmet hat (`http://cr.yp.to/qmail/venema.html`).

1.2.1 Software

Dan Bernstein versorgt uns mit einem umfangreichen Schatz von Unix-Software für unterschiedlichste Anwendungen. Diese möchte ich nun zunächst in einem schnellen Überblick vorstellen. Kennzeichnend für DJB-Software ist, dass er in der Regel die Standard-Unix Bibliotheken (typischerweise unter `/usr/include/`) nicht verwendet, sondern dass er diese durch seine eigenen substituiert, die einerseits sicherer und andererseits effizienter sind. Zudem macht dies die Portierung seiner Software für unterschiedliche Unix-Derivate relativ unproblematisch.

Dan Bernstein stellt eine umfangreiche Liste von Software auf seiner Home-Page zur Verfügung. Neben Mathematik-Programmen (z.B. *fast arithmetics*), unterscheidet er die Kategorien, Datenbanken, Zeit, Unix und natürlich DNS und E-Mail. Aus dem grossen Fundes der DJB-Programme, wollen wir uns in diesem Buch mit folgenden Softwarepaketen beschäftigen:

- *UCSPI* — das Unix Client/Server Program-Interface; häufig als Ersatz für den INETD genutzt, soll hier als Einstieg in die Welt der DJB-Software genutzt werden.
- *Daemontools* — Programmpaket zur Kontrolle und Management sog. Daemon-Prozesse im Unix-System. Unter Einsatz der Daemontools werden Programme wie Qmail, Squid, Apache oder auch Samba hochverfügbar.
- *Qmail* — Die Alternative zum Standard-E-Mail-Programm Sendmail. Qmail 1.03 ist seit 1998 verfügbar und hat sich in dieser Zeit als robuste, schnelle und sichere E-Mail Plattform bewährt. Zudem existieren für Qmail eine Reihe von Erweiterungen, von denen hier zunächst nur die von Dan Bernstein genannt sein sollen:
  - *Checkpassword* — Die Schnittstelle zur Übertragung des Benutzer-Passworts an den Qmail POP3-Daemon.
  - *Fastforward* — Ein Programmpaket zum Erstellen und Verwalten sog. Alias-Listen für E-Mail-Adressen.
  - *Qmailanalog* — Das Analyse-Programm für Qmail.
  - *EZMLM* — Der Easy Mailing Listen Manager.
  - *Serialmail* — Effiziente Nutzung von Qmail über Dial-Up-Verbindungen.
- *DJBDNS* — Der sichere und leistungsfähige Ersatz für den Berkeley Internet

Name Daemon (BIND).

Dan Bernstein beschränkt sich auf die saubere Implementierung der Core-Funktionen seiner Programme. Im Gegensatz zu anderer *Public Domain Software* vollziehen sich die Update-Zyklen sehr gemächlich. Insbesondere gibt es aber in der Regel keine notwendigen Security-Updates, die als CERT[3]-Advisories kursieren. Manche Anwender sehen darin eine mangelnde Pflege seiner Produkte. Für den Anwender bringt dies aber erhebliche Vorteile: Einerseits kann er beim Einsatz von DJB-Software sicher sein; andererseits kann er langfristig eigene Erfahrungen aufbauen sowie auf das (archivierte) Wissen und die Erfahrung der vielen DJB-Anwender setzen.

Gerade im Falle von Qmail gibt es eine überwältigende Anzahl von Ergänzungen, Patches und Implementierungsvorschlägen, so dass gerade neue Anwender den "Wald für lauter Bäumen" nicht sehen. Dave Sill hat in seinem Buch "Life with Qmail" und seinen öffentlichen Dokumenten versucht, eine Art "Rezeptesammlung" zu erstellen, die häufig als Konsens für die Qmail-Installation betrachtet wird.

Ziel des Buchs

In diesem Buch möchte ich die entgegengesetzte Methode anwenden: Statt zu sagen: "Lies die Strassenkarte und Du wirst ankommen!", versuche ich folgendes: "Überlege wohin Du willst, lerne die Strassenschilder zu lesen und Dein Auto zu beherrschen; dann erreichst Du Dein Ziel!". Die Strassenschilder sind in diesem Zusammenhang die Internet-Standards bzw. RFCs[4]. Daher wird sich Kapitel 3 mit diesen befassen. Das Auto steht in dieser Analogie für die (Software-)Werkzeuge. Diese lauten natürlich Qmail, DJBDNS, Daemontools und andere, denen der Schwerpunkt dieses Buchs zufällt. Auch ist es sinnvoll, ein Wort über die Strassen zu verlieren, die vom Auto zu befahren sind. Daher habe ich im Kapitel 2 einen kurzen Überblick über das Betriebssystem Unix vorgesehen. Eins kann ich Ihnen — dem Leser — natürlich nicht abnehmen: Was ist Ihr Ziel? Software von Dan Bernstein kann sehr nutzbringend auf einer einzelnen Unix-Workstation eingesetzt werden, die beispielsweise per DSL-Dial-Up ins Internet geschaltet wird, sie eignet sich ebenso für Firmen, wo es darum geht, auf der Grundlage unterschiedlicher Unix-Betriebssysteme eine gewisse Einheitlichkeit (der Administration) zu erzielen, sie ist aber gerade auch für Internet Service Provider (ISP) interessant, die hohe Anforderungen and Sicherheit, Ausfallsicherheit und Durchsatz stellen.

Mein Ansatz ist der, dass (theoretisches) Wissen und Erfahrung häufig zwei paar Schuhe sind. Wissen ist notwendig, um eine Aufgabe *effektiv* lösen zu können, Erfahrung zeigt uns, wie die Lösung auch *effizient* zu realisieren ist. Erfahrung geht somit mit einer technischen Umsetzung konform. Im Englischen würde ich

---

[3] CERT — Computer Emergency Response Team

[4] RFC — Request for Comment

statt Wissen *knowledge* schreiben und statt Erfahrung *know-how* oder *skill* (im Deutschen häufig mit Fähigkeit gleichgesetzt). Wir sehen, dass Begriffe trotz scheinbarer sprachlicher Kongruenz oft (kulturelle) unterschiedlich besetzt sind. Diese Problematik wird uns noch öfters im Buch begegnen, wo es doch einerseits um technische Produkte geht, die andererseits noch zusätzlich in englischer (Programmier-) und Fachsprache definiert sind.

Ergänzende Software

Bei der Recherche über die hier vorgestellten Software-Produkte von Dan Bernstein (z.B. über google) findet der zukünftige Anwender häufig eine Vielzahl von Ergänzungen, von denen auch ich einige beigesteuert habe. Es ist mir naturgemäss nicht möglich, alle vorzustellen. Häufig bieten bereits Dan Bernstein's Basisprodukte selbst soviel Flexibilität, geeignete Erweiterungen auch mit relativ geringem Aufwand selbst zu entwickeln bzw. beizusteuern. Auf einige dieser zusätzlichen Tools werde ich jedoch speziell im Umfeld von Qmail eingehen.

Graphische Aufsätze

Für die meisten der genannten DJB-Programme gibt es graphische Front-Ends zur Administration und auch zur Auswertung der Loginformation. Die meisten Aufsätze stützen sich auf CGI- oder Perl-Skripte die über ein HTTP-Server angesprochen werden. Meiner Erfahrung nach stellen die meisten Programme nur Basis-Funktionen bereit, sodass sicherlich nicht der Komfort einer "Windows-"Administration erreicht werden kann.

Für viele Unix-Produkte trifft ebenfalls zu, dass über das graphische Werkzeug sich bestenfalls 90% der Aufgaben realisieren lassen; geht es ums "Eingemachte", muss die Kommandozeile herhalten. Daher ist es m.E. unabdingbar, dass ein Systemadministrator seine Werkzeuge gut kennt. Hierzu zählen natürlich die allgemeinen Unix-Kommandos, aber auch die Shell, mindestens ein "höhere" Programmiersprache (z.B. Perl, Python oder PHP) und ein profundes Wissen über das zu administrierende Programm.

1.2.2 Kryptographie

Bei der Verschlüsselung von Daten sind allgemein zwei Verfahren bekannt und genutzt:

- Symmetrisch (auch *private key* genannt) Verfahren mit einem Schlüssel zum Ver-schlüsseln und Ent-Schlüsseln der Daten. Diese Verfahren sind seit den 60er Jahren gebräuchlich und als DES (Data Encrpytion Standard) geführt. Das Problem hierbei ist die gesicherte Übermittlung des Schlüssels zwischen der Partei die verschlüsselt und derjenigen, die entschlüsselt.
- Asymmetrische oder *public key* Verfahren, wo es einen (privaten) Schlüssel zum Ver-schlüsseln und einen öffentlichen (public) Schlüsseln zum Ent-schlüsseln gibt. Dieses Verfahren bringt zudem mit, dass eine Autentisierung des Verschlüsslers bzw. Private-Key Inhabers gewährleistet ist. Das RSA-

Verfahren wurde Anfang der 80er Jahre am MIT entwickelt und anschliessend in den USA (und in Kanada) patentiert. Eine Patentierung ausserhalb der USA wurde versagt, weil der Algorithmus bereits einschlägig veröffentlicht wurde.

Hinsichtlich der Schlüssel können ebenfalls zwei zentrale Merkmale festgemacht werden:

1. Wie lange ist der Schlüssel?
   Dies korreliert streng mit dem Aufwand den Schlüssel zu erzeugen; aber vor allem ihn zu "knacken". Im Vergleich zu einem "gemeinen" Schloss ist die Länge des elektronischen Schlüssels mit der Anzahl der "Einkerbungen" des Schlüsselbarts zu vergleichen. Typischerweise beträgt die Anzahl der Bits zur Darstellung des elektronischen Schlüssels 128, 256, 512, 1024 oder 2048 Bits. Die Schlüssellänge sollte in der Regel mit der "Lebensdauer" der zu schützenden Daten zusammenhängen. Kurzfristig relevante Daten können durch schnelle, kurze Schlüssel ausreichend geschützt werden. Langfristig zu schützende "Geheimnisse" verlangen einen hoch-bittigen Schlüssel.

2. Wie ist der Schlüssel generiert worden?
   Dies ist eine Frage der mathematischen Methode. Typische Verfahren basieren auf der Faktorisierung "grosser Zahlen" durch Primzahlen oder diskreten Logarithmen. Ist das Verfahren zur Schlüsselerzeugung bekannt, kann natürlich sehr viel effizienter beispielsweise über eine sog. "brute force" Attacke ein Schlüssel ermittelt, d.h. "geknackt" werden.

1.2.3 Bernstein vs. US

Anfang der 90er Jahre erfand Dan Bernstein als Doktorand an der University of Berkeley "Snuffel": Eine Einweg-Hash-Funktion mit symmetrischer Verschlüsselung und "Zero-Delay" ("The Snuffel Encryption System"). Snuffel sollte dazu dienen dazu, in Online-Dialogen den Inhalt der Übertragung zu verschlüsseln, sofern beide Kommunikationspartner über die geeigneten Schlüssel verfügen. Sein Vorhaben war, diesen Algorithmus im Rahmen einer "elektronischen Konferenz" in "sci.crypt" international bekannt zu machen.

Nach einigen Konsultationen reichte er jedoch 1992 dieses Verfahren dem State Department zur Begutachtung ein. Die USA hatten zu dieser Zeit "ihren Daumen" auf jeder Art von Verschlüsselungsverfahren, wie dies im "International Traffic in Arms Regulations" (ITAR) formuliert ist. Der Export von Verschlüsselungsmethoden fällt daher unter die gleichen Regularien wie die Ausfuhr von Waffen selbst. Verstösse hiergegen werden mit $ 1 Million, 10 Jahren Haft und zivilrechtlicher Wiedergutmachung bestraft.

Als Resultat hiervon konnte Dan Bernstein seine "Erfindung" weder Publizieren, Vermarkten noch lehren. Das Ende der "freien Wissenschaft".

Am 30. Juni 1992 versucht Dan Bernstein eine Freigabe von Snuffel 5.0 (einschliesslich Dokumentation) zu erwirken: "In effect what I want to export is a description of a way to use existing technology in a more effective manner. I do not forsee military of commericial use of Snuffle by anyone who does not already have access to the cryptographic technology contained in ... I do not forsee practical use of Snuffel by those who do have such access in particular for the purpose of interactively exchanging encrypted text."

Der Direktor des Office of Defence Trade Controls, William B. Robinson antwortete ihm am 20. August 1992: "This ... stand-alone cryptograpic algorithm ... is designated as a defence article under U.S. Munition List Category XIII (b) (1). Licenses issued by this office are required prior to export."

In der berechtigten Annahme, dass mit dieser Ablehnung seine Arbeiten an den ITAR-Beschränkungen scheitern würde, unternahm er im September 1993 einen erneuten Versuch, indem er seine Ergebnisse in fünf Teilbereiche zu unterteilte (Code, Beschreibung, Verschlüsselungs- und Entschlüsselungsbeschreibung sowie das ursprüngliche Papier), um hierfür separate Erlaubnisse zu erreichen. Hierzu bemerkte er: "For the crime of setting Snuffle down on paper, I am an arms manufacturer, and I must register with the State Department — or so I am told by DTC, the Office of Defense Trade Controls. DTC also insists that it would be a felony for me to publish Snuffle without their blessing and approval" (Bernstein 1993-07-30).

Obwohl die ITAR-Regularieren vorschreiben, eine (wie auch immer geartete) Antwort nach 30 Tagen dem Antragsteller zuzustellen, erfolgte auch nach 15 Monaten keine Rückmeldung. Anschliessend nahm Dan Bernstein Kontakt zur Electronic Frontier Foundation[5] (EFF - `http://www.eff.org/`) auf, die im zusagte, im Rahmen eines Musterprozesses juristischen Beistand (speziell durch die Rechtsanwältin Cindy Cohn) für sein Anliegen, d.h. einer Klage gegen die Beschränkung von Veröffentlichungen über Verschlüsselungssoftware- und Dokumentation, bereitzustellen (`http://www.eff.org/bernstein/`).

Somit wurde am 21. Februar 1995 eine (nach deutschem Recht würde man sagen Normen-) Klage (#C95-0582-MHP) am Northern District of California gegen das Department of States, Defence and Commerce, die National Security Agency und die Arms Control und Disarmement Agency eingereicht, was als *Bernstein vs. Dept. State* bekannt geworden ist. Das Verfahren zog sich mehrere Jahre hin und wurde schliesslich am 9th Circuit Court of Appeals am 9. Dezember 1997 verhandelt. Zuvor wurde schon in einigen Vorverhandlungen wesentliche Meilensteine erreicht (in englisch)[6]:

---

[5] Laut einer Selbstdarstellung: "The Electronic Frontier Foundation is the leading civil liberties organization working to protect rights in the digital world."

[6] Zitiert nach John C. Mitchell: "Cryptography and Public Policy"

- Bernstein I, 15. April 1996: "Source code is speech protected by First Amend"
- Bernstein II, 6. December 1996: "Export control laws on encryption are unconstitutional prior restraint on speech"
- Bernstein III, 25. August 1997: "Restrictions on publication are unconstitutional prior restraint on speech even as written under the new Commerce Department regulations"

Schliesslich entschied der 9th District Court of Appeals in San Francisco am 6. Mai 1999: "Export restrictions against encryption are an unconstitutional prior restraint of free expression, impermissible under the First Amendment."

Laut einer CNN-Meldung vom 11. Mai 1999 zufolge hat Dan Bernstein diese Entscheidung mit folgenden Worten kommentiert: " I feel great. I'm bouncing off the ceiling right now. The court is telling me that I'm free to publish my work."

Mit dem Wechsel der US-Regierung vom Demokratischen Präsidenten Bill Clinton zum Republikaner George Bush, wurden die zunächst etwas liberaleren Exportvorschriften für Verschlüsselungstechnologien und know-how wieder angezogen und damit effektiv alle von Dan Bernstein erreichten Verbesserungen zurückgenommen. Hiergegen reichte er am 7. Januar 2002 zusammen mit dem EFF beim Federal District Court of the Northern District of California erneut Klage ein (`http://www.eff.org/bernstein/20020107_amended_complaint.html`): "I'm trying to help protect computer systems against terrorists and other criminals. It's inexcusable that the government is continuing to interfere with my research in cryptography and computer security."

1.2.4 Fighting Patents

Neben der oben dargestellten und andauernden gerichtlichen Auseinandersetzung von Dan Bernstein mit dem US State Department, führt er auch noch an einer anderen Stelle eine mustergültigen Kampf: Gegen die Vergabe (seiner Meinung nach ungerechtfertigter) Patente von Verschlüsselungsverfahren und Algorithmen (`http://cr.yp.to/patents.html`):

- US patent 4200770, Hellman Diffie Merkle, public-key cryptography
- US patent 5159632, Crandall, elliptic-curve cryptography
- US patent 5222140, Beller Chang Yacobi, session keys
- US patent 5271061, Crandall, elliptic-curve cryptography

- US patent 5299262, Brickell Gordon McCurley, exponentiation
- US patent 5463690, Crandall, elliptic-curve cryptography
- US patent 5673318, Bellare Guerin Rogaway, message authentication
- US patent 5848159, Collins Hopkins Langford Sabin, RSA with several primes
- US patent 5999627, Lee Lim, exponentiation
- US patent 6141420, Vanstone Mullin Agnew, elliptic-curve cryptography
- US patent 6185681, Zizzi, file encryption

Die Problematik hierbei ist, das relativ allgemeine mathematische Ideen nun in den Kontext eines Verschlüsselungsverfahren gerückt werden und damit patentierbar sind. Hierdurch werden diese Verfahren der Forschung entzogen, da nun lizenzpflichtig. Basieren die patentierten Algorithmen allerdings auf bereits veröffentlichten Verfahren, ist das Patent ohnehin hinfällig, wie dies Dan Bernstein an einigen der oben genannten Fällen demonstriert. Die Patentvergabe ist dann somit obsolet.

Ein schönes Bespiele nicht auf dem Gebiet der Verschlüsselung sondern dem der Optimierung, stellt folgende Meldung der Zeitschrift c't vom 13.10.2002 (`http://www.heise.de/newticker/data/vza-12-10.02-002/`) dar: IBM verzichtet auf ein Patent zur computergesteuerten Reservierung für Toiletten- und Waschräume in Flugzeugen (USPTO 6,329,919). Hierzu meint ein Sprecher von IBM: "Wir haben das Patent der Öffentlichkeit übergeben, damit wir uns auf unser hochwertigeres Patent-Portfolio konzentrieren können". Ohne Kommentar.

1.3 Quellen

Sämtliche Software von Dan Bernstein und auch alle die in diesem Buch genannten Ergänzungen sind im World Wide Web verfügbar. Es lohnt sich in jedem Fall, die Sourcen aktuell und online herunter zu laden. Einige Software von Dan Bernstein sind auch als Pakete unter den verschiedenen freien Unix-Derivaten verfügbar. Dies bedeutet aber dann, dass sich die Distributoren den bestehenden Installationsanforderungen unterworfen haben. Prinzipiell würde ich mich nicht darauf verlassen. Die Installation via Paket bringt bei Software von DJB sowieso kaum (Zeit-)Vorteile. Es ist in der Regel immer besser, von der Quelle zu installieren und hierdurch ein erstes Verständnis für das Produkt zu erzielen.

Zur Orientierung möchte ich aber noch die folgenden Hauptquellen benennen:

- cr.yp.to (`http://cr.yp.to/`) — Dan Bernsteins Webseite mit allen

Sourcen und einer vorzüglichen Dokumentation gerade im Bereich von E-Mail. Interessant sind auch seine Aussagen über Computerhardware.

- qmail.org (`http://www.qmail.org/`) — Gepflegt von Russell Nelson und mittlerweile umfangreich gespiegelt, ist die Referenzseite, wenn es um Qmail geht. Über dies Seite ist auch das Qmail Mailinglisten-Archiv verfügbar. Die hier verfügbaren Informationen sind aber für den Anfänger ziemlich unübersichtlich gestaltet.
- ezmlm.org (`http:/www.ezmlm.org/`) — Die leider kaum mehr gepflegte Referenzseite zum *Easy Mailing Listen Manager* und der Erweiterung ezmlm-idx von Fred Lindberg.
- djbdns.org (`http://www.djbdns.org/`) — Russell Nelsons Seite zu DJBDNS; erst im Aufbau begriffen.

1.3.1 Qmail Ergänzungen

Ein Patch für alle Fälle, könnte man meinen, wenn man die nachfolgende Liste von Qmail-Ergänzugen liest. Aber keine Angst, nur die wenigsten Programme sind für "Normal-Anwender" gebräuchlich. Dies ist ein kleiner Auszug aus der Patchsammlung von qmail.org.

- Michele Beltrame (`http://sourceforge.net/projects/qmhandle`) — stellt uns das unverzichtbare Hilfsprogramm **qmHandle** zur Anzeige und Löschen von E-Mails in der Queue zur Verfügung.
- Charles Cazabon (`http://www.qcc.ca/~charlesc/software/getmail-3.0/`) — Autor the *Fetchmail*-Ersatzes *Getmail* und die "Antwort-Maschine" der Qmail Mailing-Liste.
- Bruce Guenther (`http://www.vmailmgr.org/`) — hat den Qmail basierenden Virtual Domain Manager *VMailMgr* realisiert. Ferner stammt von ihm ein Qmail-Autoresponder.
- André Opperman (NRG4U, `http://www.nrg4u.com/`) — macht uns "The big qmail picture" verfügbar. André Opperman ist auch Schöpfer der *Qmail/LDAP-Kopplung* und einiger weiterer Qmail-Patche für grosse E-Mail-Volumina.
- Will Herris (`http://will.harris.ch/`) — stellt uns eine Sammlung von Patches zur Verfügung.
- Inter7 (`http://www.interseven.com/`) — hosten den Virtual Domain Manager *Vpopmail* für Qmail.

- Sam Varshavchik (`http://www.flounder.net/~mrsam/maildrop/`) — liefert uns eine umfangreiche Programmsammlung zur Bearbeitung und Filtern von E-Mails: *Maildrop*. Maildrop ist Teil des Mails Delivery Agents *Courier* Mail-Server, aber auch separat nutzbar.
- Krzysztof Dabrowski (`http://members.elysium.pl/brush/qmail-smtpd-auth/index.html`) — hat das *Qmail SMTP Authentication Patch* von "Mrs. Brisby" umfangreich erweitert.
- Frederik Vermeulen (`http://inoa.net/qmail/qmail-1.03-tls.patch`) — liefert uns eins der mittlerweile verfügbaren *Qmail Start-TTLS* Ergänzungen, die auf der OpenSSL-Bibliothek aufbauen.
- Ismail Yenigul (`http://www.enderunix.org/isoqlog/`) — hat *ISOQLog*, eine Sammlung von Analyseprogrammen für E-Mail Logs und Ausgabe im HTML-Format, geschaffen.
- William E. Baxter (`http://www.superscript.com/qtools/intro.html`) - ist Autor der *qtools*, eine Sammlung von Programmen zum Bearbeiten von E-Mails; vergleichbar Dan Bernstein's *mess822* Paket.

1.3.2 Dokumentation über Qmail

Links zu einer weiterführenden Dokumenten (in unterschiedlichen Sprachen) über Qmail (und seinen manchmal sehr spezifischen Einsatz) kann über die qmail.org Webseite erreicht werden. Hier eine Auswahl:

- Dave Sill Life With Qmail (`http://www.lifewithqmail.org/`)
- Adam McKenna "The qmail HOWTO" (`http://www.flounder.net/qmail/qmail-howto-v2.html`)
- Magnus Bodin "Qmail Cookbook" (`http://x42.com/qmail/`)
- Hennig Bauer "Life With qmail-ldap" (`http://www.lifewithqmail.org/ldap/`)
- FAQTS: qmail.faqts (`http://www.faqts.com/knowledge_base/index.phtml/fid/139/lang/en`)
- Allan Fair: "Qmail Application Developer's Guide" (`http://www.cyberdesk.com/qmail/`)

1.3.3 Weitere Informationsquellen

- Gerrit Pape (`http://smarden.org/pape/djb/`) — seine man-pages zu UCSPI, DJBDNS und Daemontools sind m.E. ein Muss
- Henning Bauer's "Life With DJBDNS" (`http://www.lifewithdjbdns.com/`)
- Felix von Leitner's Ergänzungen, speziell für IPv6 (`http://www.fefe.de/`)
- FAQTS domains.faqts (`http://www.faqts.com/knowledge_base/index.phtml/fid/699`) —interessante Wissensammlung zu DJBDNS

1.3.4 Mailing Listen

Wer sich langfristig ernsthaft mit Software von Dan Bernstein beschäftigen will, sollte von der Möglichkeit Gebrauch machen, sich in die bestehenden E-Mail-Listen einzuschreiben, und somit quasi online an der aktuellen Diskussion um Qmail&Co. teilzunehmen. Zur Zeit sind folgende E-Mail-Listen aktiv:

- Qmail (qmail@list.cr.yp.to) — mit täglich über 50 Postings die aktivste Mailingliste über Qmail.
- Multilog (log@list.cr.yp.to) — hier wird über **multilog**, dem Syslog-Ersatz aus dem Fundus der Daemontools diskutiert.
- DJBDNS (dns@list.cr.yp.to) — die Mailingliste um und über DJBDNS.

Es ist klar, dass für diese unmoderierten Mailinglisten natürlich EZMLM herangezogen wird. Für Qmail besteht auch ein Archiv älterer Postings, das aber leider nicht sehr gut zugänglich ist. Beim Einstellen einer Anfrage wird man mit dem Tool *qsecretary* von Dan Bernstein konfrontiert. Nachdem in der Vergangenheit auch diese E-Mail-Listen zum Ziel für Spam-E-Mails genutzt wurden, hat sich dies nach Einführung von *qsecratary* auf Null reduziert.

Die umfangreichen Qmail-Ergänzungen wie Vpopmail und Qmail-LDAP besitzen jeweils eigenständige Mailing-Listen, die zum zielgerichteten Nachfragen für Probleme in diesem Zusammenhang herangezogen werden können.

Gerade die Qmail-Mailingliste ziert sich nicht unbedingt vornehmer Zurückhaltung. Gelegentlich werden richtige Flame-Threads ausgetragen. Zudem ist oft das Niveau der Fragen, aber leider auch der Anworten nicht unbedingt erspriesslich. Kennzeichen hiervon ist ein Auszug von Postings, die als "Worst of Qmail" bekannt geworden sind.

Generell sollte man — speziell als Anfänger oder *Newbi* — vor einem Posting,

d.h. einer Frage in die entsprechende Mailing-Liste zunächst die vorhandene Dokumentation, das Archiv aber die Suchmaschine *Google* bemühen. Häufig findet man dann bereits eine vergleichbare Situation mit qualifizierter Antwort. Newbis fehlt aber häufig dieses systematische Herangehen. Auftretende Fehler werden nicht zunächst im eigenen Verständnis oder Herangehen, sondern in der eingesetzten Software vermutet; um diese dann sofort in die E-Mail-Verteilerliste zu posten. Dies (dis-)qualifiziert in erster Linie den Fragesteller.

Als Haupt-Beitragender kann Charles Cazabon hiervon umfangreich berichten. Häufig lauten seine Antworten daher: "Gehört nicht hierher", "wurde vor einigen Tagen bereits besprochen" etc. Dies hat er in einem Kodex zur Nutzung der (speziell Qmail-) Liste zusammengefasst `http://www.qcc.ca/~charlesc/writings/12-steps-to-qmail-list-bliss.html`.

1.4 Urheberrechte, Lizenzen und Patente

Eigentlich ist es ein Treppenwitz, dass Software von Dan Berstein kaum in den aktuellen Distributionen der freien Unix-Derivaten zu finden ist. Dies trifft im besonderen auf Qmail zu. Häufig wird von den Distributoren die "restriktive" Lizenzpolitik von Dan Bernstein genannt. Die Vorstellungen und Einschränkungen hinsichtlich der Nutzung und der Verbreitung seiner Software hat Dan Bernstein auf den URL `http://cr.yp.to/softwarelaw.html` und `http://cr.yp.to/distributors.html` hinterlegt.

Freeware

Es ist interessant zu bemerken, dass bereits der Begriff *Freeware* — mit dem ich die Software von Dan Bernstein bezeichne — politisch besetzt ist. Als Schildträger sind hier die Free Software Foundation zu nennen (`http://www.fsf.org/`), die ihren Begriff von *Freeware* auf einer eigenen Web-Seite hinterlegt haben (`http://www.fsfeurope.org/documents/freesoftware.en.html`) und die darüber hinaus auf Richard M. Stallman's GNU-Projekt (`http://www.gnu.org/`) verweisen, das seinerseits ein noch *restriktiveres* Verständnis von *Freeware* dokumentiert (`http://www.gnu.org/philosophy/free-sw.html`). *Freeware* versteht sich hierbei nicht als kostenlose sondern als beliebig modifizierbare und publizierbare Software. Wir konstatieren hier: Dan Bernstein's Software ist *kostenlos*, sie ist *frei veränderbar*, aber als Software-Paket *nicht in geänderter Form distributierbar* (dies gilt aber nicht für Patche).

Dan Bernstein versteht sich auch als Kämpfer gegen ungerechtfertigt erteilte Software-Patente (`http://cr.yp.to/patents.html`) sowie gegen Exportbeschränkungen der US-Bundesregierung (Bernstein v. United States). Letztlich war es seine eigenen Erfahrungen, die in die Web-Seite "Crypto" (cr.yp.to) eingeflossen sind.

Um dies zu beleuchten, ist ein Blick in die aktuelle Diskussion um das Nutzungsrecht von Software, die Frage von Software-Lizenzen und -Patente notwendig. Zuvor möchte ich jedoch einige Randbedingungen feststellen:

- Meine Ausführungen sind nicht die eines Juristen, der mit ausreichendem Kenntnisstand die Inhalte der Gesetzgebung rezitieren oder gar interpretieren kann. Letztlich bin ich lediglich ein "interessierter Laie".
- Die Rechtsprechung und Gesetzgebung ist zur Zeit im Umbruch begriffen. Was noch vor einigen Jahren als verbindlich betrachtet wurde, wird im Hinblick auf die Europäische Union (EU) und die damit einhergehende Angleichung der Gesetzgebung modifiziert.
- Einen starken Impuls auf die internationale Gesetzgebung geht derzeit vom TRIPS[7]-Abkommen aus, wichtige Änderungen des Urheberrechts ergeben sich ab April 2003.
- Es gibt beachtliche Unterschiede zwischen dem germanisch/französischen Recht einerseits und dem anglo-amerikanischen Recht andererseits (Rechtstraditionen).
- Die Tendenzen der EU-Gesetzgebung gehen eher in Richtung des anglo-amerikanischen Rechts, da aus dieser Richtung der grösste wirtschaftliche wie politische Druck für einen faktischen Klärungsbedarf zu verzeichnen ist.
- Abweichend von Legislative, beeinflusst die Jurisprudenz das "gelebte" Recht massgeblich. Dies trifft im besonderen z.B. hinsichtlich der Patentierbarkeit auf die Interpretation des "technischen Beitrags" zu.
- Insbesondere ist nicht ausreichend definiert, was unter "Software" zu verstehen ist. Ist dies Quell-Code oder Maschinen-übersetzter Code? Wie verhält es sich mit abstrakten Progamm-Definitionen, z.B. in UML (Unified Modelling Language)? Hier gibt es in der Rechtsprechung sehr stark abweichende Interpretationen.

Der Grund für die derzeit heftig tobende Debatte um Software ist auch schnell ausgemacht:

- Software ist in immaterielles Gut (Abstraktum). Zu den Abstrakta zählen auch Wissen und manche künstlerische Produkte, z.B. Belletristik und Kompositionen (Musik). Im Gegensatz zu materiellen Produkten (auch die der gestaltenden Kunst), lassen sich Abstrakta im Prinzip unbeschränkt (zumindest solange bis der physikalische Träger erschöpft ist) reproduzieren und multiplizieren.
- Während die Verwertung künstlerischer Abstrakta von der kulturellen

[7] TRIPS — Trade-Related Aspects of Intellectual Property

Rezeption abhängig ist, verlangt (komplexe) Software eine spezifische Computer-Hardware, auf der sie lauffähig ist. Ferner endet der "gewinnbringende Lebenszyklus" künstlerischer Abstrakta mit der Rezeption durch das Individuum. Hingegen weist Software im Produktionsprozess einen potenziell unbeschränkten Lebens- und Verwertungszyklus auf.

- Im Vergleich hierzu unterliegen materielle Ressourcen (Realiter) — wie Grund und Bodenschätzen — einer prinzipiellen Knappheit sowie ggf. einem Verbrauch, d.h. werden stofflich (physikalisch oder chemisch) umgewandelt; Abstrakta werden bestenfalls überflüssig.
- Die Verfügbarkeit von Abstrakta war (und bleibt) immer schon an ein physikalisches Trägermedium (Buch, Schallplatte, Magnetband, optisches Medium) und an dessen Reproduktion gekoppelt. Solange die Reproduktionskosten im Vergleich zum erzielbaren Preis des Abstraktum vergleichbar sind, erfolgt die erfolgt die Marktsteuerung indirekt über den Preis des damit verbundenen Trägermediums. Dieses Paradigma ist aber heute mit dem Personal Computer nicht mehr zutreffend. Software lässt sich im Gegensatz z.B. zu Rolex-Uhren ohne Qualitätsverlust kopieren.

Meiner Meinung nach ist es notwendig, ein vernünftiges Rechts- und Verwertungsmodell (für Software) zu definieren, da beides in der aktuellen Diskussion fehlt. Ich will daher versuchen, dies in den nachfolgenden Abschnitten postum zu liefern und dies auch als Referenz für meine eigenen Überlegungen heranziehen.

1.4.1 Rechtsgrundlagen und Rechtsmodell

Es gibt im deutschen Recht keine spezifischen Gesetze, die den Einsatz oder die Verwertung von Software regeln. Generell kann aber von drei Quellen ausgegangen werden:

- Das Urheberrecht (UrhG). Unter das Urheberrecht fallen in der Regel künstlerische Werke wie Literatur (speziell auch Belletristik), Musik (Kompositionen) und die abbildenden Künste (z.B. Malerei, Fotografie). Erst relativ spät wurde im achten Abschnitt (§§ 69ff) die "besondere(n) Bestimmungen für Computerprogramme" eingefügt und sie an das allgemeine Urheberrecht angeglichen.
- Das Markenrecht (MarkenR) und Geschmacksmusterrecht (GeschmMG). Über das Markenrecht können Begrifflichkeiten und Logos eingetragen und hiermit zu exklusivem Gebrauch geschützt werden (eingetragenes Warenzeichen ® bzw. Trademark ™), z.B. Windows™. Ergänzend zum Markenrecht sind das Geschmacksmusterrecht geltend.
- Das Patentrecht (PatG). Das Patentrecht ermöglicht die Registrierung "technischer" Erfindungen. Einerseits werden diese damit zum öffentlichen

Allgemeingut, andererseits kann der Erfinder die Nutzung seines Patents über die Vergabe von (vergüteten) Lizenzen regeln.

In der internationalen Diskussion über Software kommen häufig die Begriffe "geistiges Eigentum" bzw. "intellectual property (IP)“ zum Einsatz. Obwohl beide Begriffe sprachlich identisch sind, entstammen sie jedoch unterschiedlichen Rechtssystemen und bezeichnen somit unterschiedliche juristisch Sachverhalte:

- Das *geistige Eigentum* ist entsprechend französisch/deutschem Recht ein Naturrecht, dass dem Schöpfer durch den Schöpfungsakt ("Droit d'auteur") Zuteil wird. Es ist wie ein Menschenrecht nicht übertragbar und stellt ein "Band der Vaterschaft" zum Schöpfungsgut dar, das mit spezifischen Rechten versehen ist. Z.B. kann ein Komponist verlangen, dass sein Werk nicht von faschistischen Organisationen (öffentlich) aufgeführt wird — unabhängig davon, welchem Verlag er die Rechte an seinem Werk übertragen hat.
- Nach anglo-amerikanischen Recht obliegt es dem Vertragsabschluss zwischen Schöpfer und Verwerter (z.B. einem Verlag), den Umfang der abgetretenen Rechte zu regeln. Der Verwerter wird hierbei in der Regel zum Inhaber des *intellectual property*. Ähnliche Vorschriften gelten in Deutschland z.B. für Arbeitsverträge bei Software- bzw. IT-Beratungsfirmen: Der Arbeitgeber sichert sich per Arbeitsvertrag das Eigentums- und Verwertungsrecht an der vom Mitarbeiter geschriebenen Software zu.

Im Grunde geht es bei allen gesetzlichen Regelungen bzw. Einschränkungen immer um drei Komplexe:

1. Das Nutzungsrecht. Wie und in welchem Umfang darf ich das Abstraktum nutzen? Bekannt sind z.B. das sog. Copyleft bzw. die GNU Public License (GPL2), oder auch die Schullizenzen für Microsoft Produkte
2. Das Vervielfältigungsrecht. In welchem Umfang darf ich Kopien des Werkstücks (für den privaten Gebrauch) erzeugen?
3. Die Verbreiterungsrecht. In welchem Umfang darf ich Kopien des Werkstücks weiterreichen? Unterschieden werden muss hier selbstverständlich zwischen privaten und gewerblichen Gebrauch.

Eine weitere zentrale Frage stellt sich im Hinblick auf den Vorrang von allgemeinen Vorschriften (Gesetzen) und privaten Abmachungen (Lizenzen).

Dieses Buch als Abstraktum

In der realen Welt sind die Verhältnisse noch komplizierten und können nur durch ein Geflecht bilateraler und multilateraler Beziehungen beschrieben werden. Als Orientierung lässt sich Abbildung 1-1 heranziehen, die z.B. die (impliziten und expliziten, d.h. privatrechtlichen) Rechtsverhältnisse für dieses

Buch beschreibt:

- Zunächst bin ich natürlich Autor und somit Urheber dieses Buchs (kein belletristisches Wunderwerk). Diese Urheberschaft wird vom Staat anerkannt und garantiert.
- Mit meinem Verlag habe ich ein Vertrag, der die konkreten Übergabe- und Verwertungsbedingungen (und natürlich auch mein Honorar) regelt.
- Der Verlag meldet das Buchprojekt bei der Verwertungsgesellschaft VG Wort an. Dort werde ich sodann als Autor für dieses (potenzielle) Werk registriert geführt. Die VG Wort fungiert sozusagen als nebenstaatliche Organisation, an die der Staat Teile seiner hoheitlichen Aufgaben delegiert hat.
- Der Verlag erwirkt für das fertige Manuskript eine ISBN-Nummer, lässt das Buch drucken, vermarktet es und übergibt es dem Fachhandel zum Verkauf.
- Ferner verpflichtet sich der Verlag, einen Teil der Auflage öffentlich in Fachbibliotheken zur Einsicht und zur Ausleihe zugänglich zu machen und dort zu hinterlegen.
- Sie — lieber Leser — erwerben das Buch in der Regel anonym als "Werkstück". Hiermit erhalten Sie das Recht, Teile des Buchs zu rezitieren und sich private Kopien zu erstellen. Was Sie mit dem Buch anstellen, ob Sie die Seiten rosa ausmalen, diese ausschneiden und Schiffe daraus basteln oder Sie sie als Klopapier verwenden, ob sie hilfreiche oder gemeine Ergänzungen ins Buch kritzeln, das ist ganz alleine Ihnen überlassen.
- Unabhängig hiervon, bekomme ich über die VG Wort einen winzigen Bruchteil (proportional der angemeldeten Auflage des Buchs) des Kaufpreises Ihres (und aller anderen in Deutschland) gekauften Hochleistungs-Kopierer, -Drucker und -Scanner. Dies gilt als Ausgleichsgabe für alle (hiermit potenziell möglichen) Kopien dieses Buchs und wird mir in Form von Tantiemen ausgezahlt.

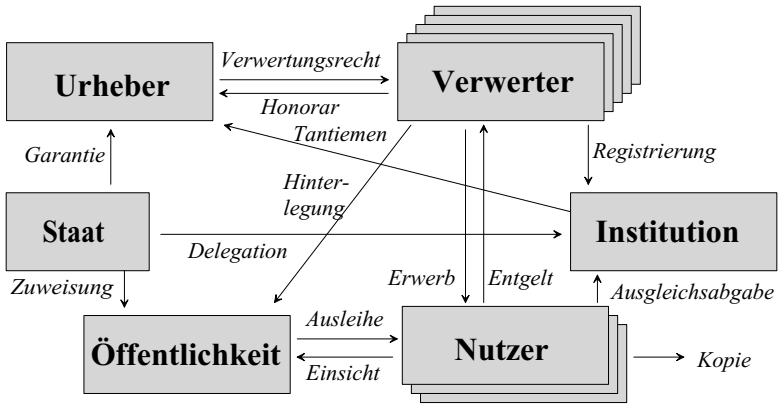

Abbildung 1-1: Rechtsmodell für künstlerische Abstrakta unter Einschluss der bilateralem und multilateralen Rechtsbeziehungen.

1.4.2 Urheberrecht und Copyright

Das komplizierte Rechtsmodell des Urheberrechts mit seinen expliziten und impliziten Rechtsbezügen wurde bislang aus der Sicht des Urhebers beschrieben.

Der Verwerter (also z.B. der Verlag Markt&Technik) kann natürlich auch sein Veröffentlichungsrechts an einen anderen Verlag abtreten. Dies ist z.B. bei Belletristik üblich, wo zu einem späteren Zeitpunkt eine (günstigere) Paperback-Ausgabe bei einem anderen Verlag erscheint. Dies geschieht in Form von Lizenzen an Dritte. Als Urheber habe ich in der Regel hierauf keinen Einfluss.

Der Erwerb dieses Buchs (= *Werkstück*) erfolgt in der Regel *anonym*, hierbei kommt kein explizites Rechtsverhältnis zwischen Erwerber und Verwerter zustande. Mit dem Erwerb des Werkstücks erhalten Sie z.B. keine besonderen Ansprüche hinsichtlich der Richtigkeit des Inhalts oder auch der Fehlerfreiheit. Dies ist häufig auch im Klappentext eines technisch-orientiertem Buches wie diesem explizit ausgeschlossen. Als Erwerber besitzen Sie jedoch das Recht, Kopien des Werkstücks für Ihren eigene Gebrauch zu erstellen; ferner sichert Ihnen das Urheberrecht zu, Teile des Buchs (öffentlich) zu rezitieren.

Das anglo-amerikanische Copyright hat sich aus anderen Prämissen entwickelt. Hier wurden ursprünglich privatrechtliche Vereinbarungen innerhalb der Buchdruckergilde herangezogen. Um das Erstveröffentlichungsrecht zu gewährleisten, erhält ein geschütztes Werk den Eintrag "Copyright©[Jahr der Erstveröffentlichung][Rechteinhaber]". Das Copyright-Gesetz in den USA wurde in der Vergangenheit um die Rechtsbestandteile "Original Workers", "Fair Use" sowie vor allem die Lebensdauer der Schutzrechte überarbeitet. Hierbei wurden

einerseits die Rechte des Urhebers, andererseits die der Öffentlichkeit ("Fair Use") gestärkt, zum dritten die Lebensdauer des Verwertungsschutzes ("Mickey Mouse Gesetz") umfangreich von 50 auf 70 Jahre (nach Tod des Autors) angehoben.

Das US-amerikanische Copyright steht zwischen dem deutschen Urheber- und dem Markenrecht. Per Copyright geschützte Abstrakta sind z.B. die "Mickey Mouse" und die Gershwin-Komposition "Raphsody in Blue". Abstrakta, die nicht dem Copyright unterworfen sind, gehören zur sog. *Public Domain*. Nach der Ergänzung des US-Copyright-Gesetzes von 1976 fallen unter dieses Gesetz auch automatisch Abstrakta, die nicht explizit registriert, aber auf einem materiellen Ausdrucksmedium fixiert sind. Allerdings sind hiervon nur Werke mit literarischem, dramatischen und musikalischen Inhalt, Pantomimen und choreographische Darstellungen betroffen, sowie bildliche, grafische und skulpturale Werke und ferner Kinofile sowie audiovisuelle Darbietungen[8].

Beim Copyright gibt es keine spezifische Deklaration von Software — sie wird einfach nicht erwähnt. Es bleibt zu bemerken, dass das Copyright-Gesetz besonders für den "Original Worker" kaum einen geeigneten Urheberrechtschutz mit sich bringt. Besonders deutlich wird dies bei der Frage der Vermarktung von kontemporärer Musik. Mittlerweile hat sich eine Organisationen von Musikern etabliert, die gegen diese Disparitäten öffentlich zu Felde zieht (`http://www.recordingartistscoalition.com/`).

Andererseits versucht speziell die US-amerikanische Musikindustrie (unter Federführung der RIAA (`http://www.riaa.org`) die Nutzungsrechte der digitalen Medien für den Nutzer stark einzuschränken und zwar in Form eines "Digital Right Management Systems" DRMS, das die Beschränkung einer privaten digitalen Kopie des Informationsträgers vorsieht. Initiativen dieser Art sind nicht neu. Vergleichbare Aktivitäten wurden bereits Ende der 60er Jahre mit der Verbreiterung der *Compact Cassette* von Philips gestartet, da hiermit zum ersten mal ein Medium geschaffen wurde, preiswert und mit merklicher Marktdurchdringung (analoge) Kopien (allerdings zunächst in schlechter Qualität) der verkaufsträchtigen Musiktitel zu erstellen. Erwähnenswert ist auch das Vorgehen der *Record Industry* gegen sog. *Bootlegs*, also nicht-authorisierter Konzertmitschnitte und deren Veröffentlichung.

Alle diese Angriffe gegen das exklusiver *Vervielfältigungsrecht* von Musik und Musikern hat die *Record Industry* prächtig und mit wachsenden Umsatz- und Gewinnraten überstanden. Insbesondere galt dies nach dem schnellen Ablösen der analogen Schallplatte durch das neue von Sony und Philips Anfang der 80er Jahre ins Leben gerufene optisch-digitale Medium *Compact Disc*[9]. Das Blatt

---

[8] laut Grasmucke

[9] Mit der Sampling-Frequenz von 44kHz und der linearen Abtastrate von 16 Bit,

wendete sich allerdings Ende der 90er Jahre mit dem Aufkommen und Erstellen der preiswerten CD-R, d.h. der (einmalig-)wiederbespielbaren CD per PC. Nachdem sich in diesen Jahren der Verkaufspreis eines Musiktitels auf CD — trotz wesentlich verbilligter Herstellung des Tonträger gegenüber der analogen *Langspielplatte* LP — nahezu verdoppelt hatte, sanken parallel hierzu die Verkaufszahlen der populären Titel auf dem digitalen Medium CD. Angeblich haben die Kids die "Silberscheiben", statt diese teuer im Plattengeschäft zu kaufen, billig "schwarz" kopiert und diese gegen geringes Entgelt den Klassenkameraden auf dem Schulhof überlassen. Mit Indiz hierfür kann gelten, dass in den folgenden Jahren nicht mehr die eigentliche CD im Plattengeschäft "geklaut" wurde, sondern deren Hülle ("Jewel Case"). Mit dem Aufkommen qualitativ hochwertiger und zudem preisgünstiger Scanner und Farb-Tintenstrahldrucker, kann aber auch dieses Risiko vermieden werden.

Eine weitere Bedrohung stellt für die Record Industry die (weitgehend verlustfreie) Komprimierung der Musik im MPEG-3 (MP3) Format dar und die Verbreiterung über das Internet.

Eigentlich hätte dieser Sachverhalt hier — im Zusammenhang mit Software — nicht so umfangreich diskutiert werden müssen, würden sich nicht aus der Sondersituation *eines* Wirtschaftszweiges Einschränkungen für die *allgemeine* Nutzung digital vorliegender Informationen ergeben.

Wie wenig effektiv Beschränkungen gegen die Verbreitung von Informationen und insbesondere Software sind, und wie Paradox und ggf. Konträr sich die anschliessenden Resultate entwickeln, kann aus den US-amerikanischen Exportbeschränkungen für Verschlüsselungssoftware bzw. -algorithmen nachvollzogen werden. Die *Pretty Good Privacy* von Philip Zimmermann durfte bekanntlich aufgrund Exportbeschränkung für die RSA-Verschlüsselung nicht in digitaler Form ins Ausland (und hierunter auch Europa) exportiert werden. Im seinem Buch "PGP source code and internals" (MIT Press, ISBN 0-26224-039-4) beschreibt Phil Zimmerman jedoch das in der Version 2.6.3 vorliegende (lediglich in USA und Kanada patentiere) Verschlüsselungsverfahren. Da ein Buch ja keine Software ist, konnte es exportiert und in der "patentfreien" Rest-Welt eingesetzt werden.

Damit schliesst sich auch der Kreis im Hinblick auf Software von Daniel J. Bernstein:

- Dan Bernstein führt auf seinen Seiten (Softwarelaw) aus, dass es keiner irgendwie gearteten Lizenz bedarf, im Internet frei verfügbare Software für den eigenen Bedarf herunterzuladen, zu nutzen oder zu modifizieren.
- Andererseits gibt er eine Art "Herstellergarantie" auf einige seiner Software

---

stellt die CD für HiFi-Ansprüche die unterste Grenze des Notwendigen dar.

(Qmail, DJBDNS) mit einer Prämie für evtl. gefundene sicherheitskritische Fehler; dies verlangt natürlich vom Vervielfältiger (=Distributor) ein relativ rigoroses und genaues Erstellen der auf *seiner* Software basierenden Installations-Pakete nach *seinen* definierten Bedingungen.

- Es ist zu berücksichtigen, dass Abstrakta, die aus dem öffentlichen Wissenschaftsbetrieb heraus erschienen sind (was auf Software von Dan Bernstein zutrifft), im Hinblick auf ihren Veröffentlichkeitsstatus einen anderen Stellenwert besitzt, als solche, die von kommerziellen Unternehmen oder Privatpersonen stammen. Entsprechend Abbildung 1-1 gilt als Urheber gleichsam der Staat bzw. die Allgemeinheit.
- Letztlich lassen Dan Bernstein's Ausführungen vermuten, dass für ihn das Urheberrecht näher liegt als das Copyright.

1.4.3 Lizenzen

Einmal bin ich per E-Mail angefragt worden, unter welchen Lizenzkriterien ich meine eigene für Qmail geschriebenen Softwareergänzungen anbiete ("Is the license of spamcontrol allow me to distribute the modified patch?"). Allgemein herrscht die Ansicht, dass Software nur unter gewissen Bedingungen eingesetzt und weiter gereicht werden kann. Seit dieser Zeit füge ich meinen eigenen Entwicklungen eine Art "Lizenz" bei; wohl wissend, dass sie rechtlich überhaupt nicht bindend ist.

Das sich diese Meinung so manifestiert hat, ist sicherlich auf zwei Einflussfaktoren zurückzuführen:

- Einerseits der Tatsache, dass praktisch alle kommerziellen Sofware-Produkte mit einer "Lizenz" verbunden sind, z.B. (fast) alle Produkte von Microsoft.
- Andererseits, dass durch die Präsenz der sog. GNU Public License (GPL) versucht wird, die Nutzung auch sog. Public Domain Software zu reglementieren.

Wir betrachten zunächst das von Microsoft angestrebte Lizenzmodell entsprechend Abbildung 1-2. Microsoft kennt (zumindest) zwei Arten von Lizenzen:

- Bei "freiem" Erwerb der Software gilt für den Endbenutzer die sog. EULA (End User License Agreement).
- Hersteller von PCs (sog. *Original Equipment Manufacturer* - OEM) erhalten das Recht, in einem Bündelpaket Hardware und Software zu schnüren. Für den Endbenutzer ist die so erworbene Lizenz nicht nur Personen-, sondern auch Hardwaregebunden ("Das SOFTWAREPRODUKT wird in Verbindung mit der HARDWARE als einheitliches integriertes Produkt

lizensiert. Das SOFTWAREPRODUKT darf in Verbindung mit der HARDWARE nur wie in diesem EULA ausgeführt verwendet werden.") Unklar ist mir hierbei, wie man *gekaufte* Hardware (als Teil des integrierten Produktes) *lizensieren* kann.

Mir geht es hier nicht darum, Microsoft's Lizenzpolitik zu bewerten; ich will aber das Augenmerk auf folgende Punkte richten und offene Fragen in Bezug auf die folgenden vier (möglichen) Säulen diskutieren:

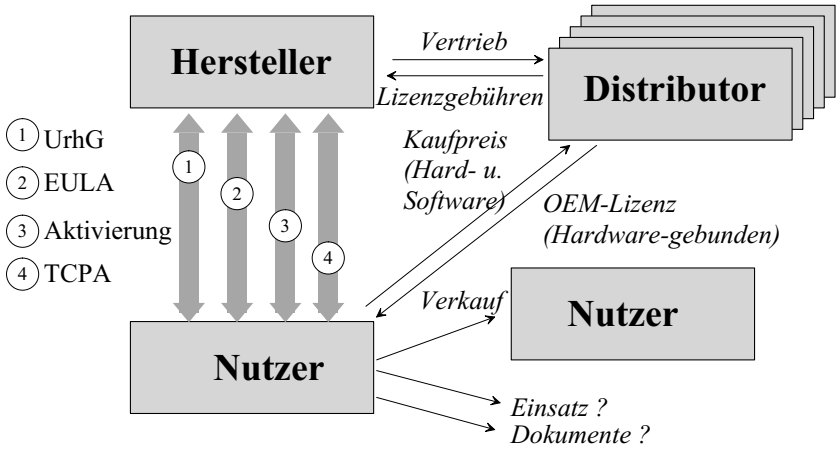

Abbildung 1-2: Die vier Säulen beim Erwerb von Software.

Säule 1 (Urheberrecht):

- Entsprechend dem Urheberrecht erfolgt die Übergabe des Werkstücks anonym, der Nutzer kann eigenverantwortlich über den Einsatz des Werkstücks bestimmen.
- Ist es rechtmässig, die Nutzung eines "Werkstücks" postum, d.h. nach Abschluss des Kaufvertrags zu beschränken? Es würde niemandem einfallen, z.B. einen Kühlschrank zum Verkauf anzubieten, der bei den "Nutzungsbestimmungen" folgendes Statement enthält: "Mit dem Öffnen des Kühlschranks verpflichten Sie sich, hierin keine alkoholischen Getränke zu lagern oder zu kühlen ....".

Säule 2 (Lizensierung/EULA):

- Im UrhG ermöglichen es § 69c (Zustimmungsbedürftige Handlungen) und § 69e (Ausnahmen von den zustimmungsbedürftigen Handlungen), die Nutzung des Softwareproduktes einzuschränken — dies ist Grundlage der Software Lizenzen.
- Die EULA stellt in Abkehr (bzw. in Ergänzung) vom Urheberrecht einen

privatrechtlichen Vertrag zwischen Hersteller und Anwender dar. Häufig wird dargestellt, dass mit dem Öffnen der Umverpackung z.B. der beiliegenden CDs oder aber nach der Installation bzw. vor der Inbetriebnahme die EULA implizit anerkannt wird. Haarsträubend ist teilweise die EULA von Windows 2000: "Sie erklären sich durch die Bestimmungen dieses EULAs gebunden zu sein, indem Sie das Softwareprodukt installieren, kopieren, downloaden, darauf zugreifen oder es anderweitig verwenden. Falls Sie sich damit nicht einverstanden erklären ... sind Sie nicht berechtigt, das Softwareprodukt zu verwenden oder zu kopieren.".

- Die EULA verlangt nach Abschluss des Kaufvertrages (einseitig) die Einhaltung von Nutzungsbestimmungen, ohne dass hierzu i.d.R. seitens des Herstellers ergänzende Rechte eingeräumt werden müssen (z.B. freie Updates).

- Ebenso werden durch die EULA keinerlei (zusätzliche) Ansprüche des Anwenders hinsichtlich der "Fitness" der Software gewährt, die über das entsprechende Produkthaftungsgesetz hinausginge ("Der Hersteller garantiert (a) für einen Zeitraum von neunzig (90) Tagen nach Erhalt der SOFTWARE, dass diese im Wesentlichen gemäß der begleitenden Benutzerdokumentation arbeitet.").

- Hinsichtlich der Nutzungs- und Vervielfältungseinschränkungen lassen sich einige Stilblüten sammeln. So liest sich § 69b (2) folgendermassen: "jede Form der Verbreiterung des Originals eines Computerprogramms oder von Vervielfältigungstücken ....". Bitte schön: Was ist das *Original eines Computerprogramms*? Der Quell-Code? Die erste lauffähige Version?
  Ich zitiere im folgenden aus der Erklärung zur EULA von Windows 2000: "Eine 'Kopie' eines Softwareprogramms erstellen Sie dann, wenn Sie (1) die Software in den temporären Speicher des Computers laden, indem Sie sie von einer Diskette, von einer Festplatte, von einer CD-ROM oder von einem anderen Speichermedium ausführen, (2) die Software auf andere Datenträger kopieren, z. B. auf eine Diskette oder auf die Festplatte des Computers, oder wenn Sie (3) das Programm von einem Netzwerkserver ausführen, auf dem die Software resident ist oder gespeichert wird." Ist das Laden einer (ausführbaren) Computer-Software in den "temporären Speicher" nicht etwa seine offensichtliche Nutzbestimmung?
  Unklar ist auch, was unter "Softwareprogramm" zu verstehen ist: Das Betriebssystem mit seinen Moduln? Das Installationsprogramm (**setup.exe**) oder ggf. ein selbstauspackendes Archiv?

- Die von Microsoft gewünschte Kopplung seiner Software (speziell der Betriebssysteme) an die PC-Hardware kann als gescheitert betrachtet werden (`http://www.heise.de/newsticker/data/psz-`

07.07.00-000/). Somit wurde zugleich der Weg geebnet, wesentlich preisgünstigere Versionen seiner Software in Form sog. OEM[10]-Versionen zu vertreiben und

- Es ist ferner zu berücksichtigen, dass die Wirksamkeit der Lizenzvereinbarungen den AGBG[11] und anderer Verbraucherschutznormen unterworfen sind, die in diesem Zusammenhang ein "höherwertiges" Recht darstellen.

Säule 3 (Registrierung/Aktivierung):

- Unter Registrierung wird üblicherweise die (freiwillige) Übertragung von Informationen an den Software-Hersteller verstanden, in der mitgeteilt wird, welches Produkt (mit der Produkt-Id und ggf. Lizenznummer) von wem (dem Erwerber) auf welcher Plattform installiert wurde. Die Registrierung ist unverbindlich und führt auch dann, falls sie nicht erfolgt, zu keinerlei Einschränkung hinsichtlich der Nutzbarkeit des Werkstücks. Eine zwangsweise Registrierung ist rechtlich unzulässig (OLG München vom 12.12.2002).

- Der offensichtlichen Unwirksamkeit des EULA für Privatanwender versucht Microsoft durch eine eingebaute Zwangsaktivierung entgegenzutreten. Dies liest sich Windows XP Home Edition folgendermassen: "Um die Softwarepiraterie einzudämmen, verwendet Windows XP die Methode der Produktaktivierung. Auf diese Weise kann Windows XP nur auf einem einzigen Computer installiert werden." (Anmerkung: Es gibt sicherlich Leute die wünschten, Microsoft meinte dies im Wortlaut ernst.)

- Der Aktivierungskey, der anonym maximal dreimal generiert werden kann, lässt sich als Software-Dongle bezeichnen. Im Gegensatz zu einer (permanenten) Produkt-Lizenznummer (wie beispielsweise z.Z. bei Adobe-Produkten), ist der Aktivierungskey nicht eindeutig für die Software (das Produkt) sondern generisch für das installierte System und besitzt daher lediglich temporären Charakter.

- Die Hardwarebindung ist hierbei stärker als bei der häufig anzutreffenden Beschränkung der Lauffähigkeit von Software mittels eines Dongels, da ein Umbau/Ergänzung der installierten Hardware-Plattform, die Validität des Aktivierungskeys in Frage stellt. Ferner kann z.B. Windows XP ohne Aktivierungskey nach 30 Tagen lediglich noch im "geschützten Mode" gestartet werden; verliert also wesentliche Teile seiner Funktionalität.

---

[10] OEM — Original Equipment Manufacturer

[11] AGBG — Allgmeine Geschäftsbedingungen Gesetz

- Problematisch ist hierbei, dass nicht der Eigner (d.h. Nutzer) der Software Besitzer des Aktivierungskeys ist, sondern der Hersteller. Dies bedeutet eine lediglich eingeschränkte Nutzbarkeit des Produktes. Eindeutig geniessen hierbei die Interessen des Herstellers Vorrang vor denen des Erwerbers.
- Hardwarekeys werden nach bestimmten Algorithmen auf Grundlage der eingebauten Hardware erzeugt. Zudem findet der Abgleich gegen eine interne Datenbank vor, in der üblicherweise auch die allgemeinen Keys der OEMs hinterlegt sind. Bei allem Bemühen, diese Informationen unter Verschluss zu halten, ist es jedoch nur eine Frage der Zeit, bis auch diese öffentlich werden.

Säule 4 (DRM/TCPA):

- Als neueste Massnahme zur Sicherung der Verwertungsrechte sollen technische Methoden zur Beschränkung der Vervielfältigung (Kopie) von geschützten Werken eingesetzt werden. Die rechtliche Grundlage hierzu ist das in den USA 1998 verabschiedete *Digital Millennium Copyright Act* (DMCA) zu nennen, was in Deutschland in einer Novelle des UrhG (und hier speziell im umfangreichen § 95) Eingang findet.
- Kernstück des DMCA ist Akzeptanz eines *Digital Right Managments* (DRM), dessen Verletzung unter Strafe gestellt wird. Dies ist auch Inhalt einer EU "Richtlinie zur Harmonisierung bestimmter Aspekte des Urheberrechts und der verwandten Rechte in der Informationsgesellschaft", wie er im Entwurf zum UrhG in § 95a zu finden ist.
- Hintergrund für die "Gleichschaltung" ist das sog. TRIPS-Abkommen der WTO[12] zur Angleichung und Pflege der Schutzrechte der Mitgliedsstaaten, der Einhaltung gewisser Mindeststandards und zur Sicherung der Rechte ausländischer Ansprüche auf geistiges Eigentum.
- Alle DRM-Systeme beinhalten einen Hardware/Software mit der einerseits DRM geschützte Werke erkannt, und deren (unberechtigte) Nutzung bzw. Vervielfältigung unterbunden werden sollen.
- Ein weiterentwickeltes DRM-System, das bereits in PCs Einzug (z.B. Notebooks von IBM) gefunden hat, wird von der *Trusted Computing Platform Alliance* (TCPA; `http://www.trustedcomuping.org`) als Zusammenschluss unterschiedlicher Soft- und Hardwarehersteller promotet. TCPA besteht aus zwei Komponenten: Einem speziellen (sog.

[12] WTO — World Trade Organisation (deutsch: WHO - Welt Handels Organisation)

Fritz-)Chip, dem *Trusted Platform Module* (TPM), das TCPA-Funktionen wahrnehmen kann — und einer Software, die das TCPA-API unterstützt. Software Hersteller wie z.B. Microsoft arbeiten bereits an einer TCPA-Unterstützung ihre Systeme (Codename *Palladium*). Kennzeichnend hierbei ist, das sich jede ausführbare Software einer (externen) Zertifizierung unterziehen muss, die ausserhalb der Kontrolle des Eigners abläuft und auf das Goodwill des Herstellers angewiesen ist. Ebenfalls lassen sich unter TCPA zertifizierte Dateien (z.B. Word-Dokumente) erzeugen, die dann gesichert übertragen und nur durch das Zertifikat des Erzeugers gelesen werden kann.

- Als erste Realisierung eines TCPA-System kann Microsoft's Xbox gelten. Die Anstrengungen der "Hackergemeinde" Microsoft's Xbox zu "knacken" und darauf Linux zu installieren, ohne dass Eingriffe in die Hardware des Systems nötig sind, wurden immerhin mit $ 200 000 Prämie bedacht.

- Inwieweit auf zukünftigen TCPA-PCs nicht TCPA-fähige Software lauffähig ist, ist klärungsbedürftig. Wie erläutert, lässt sich mittels TCPA nicht nur die Ausführbarkeit von Software-Programmen steuern, sondern auch auf diesem PC (d.h. mit der geeigneten Software) erstellte Dokumente schützen und verschlüsseln.

- Letztlich führt also DRM/TCPA zu einer informationellen Zweiteilung der Welt: Anwender die Zugang zu den Zertifikaten der Hersteller haben und denen diese gewährt werden, und solche, für die eins von beiden Kriterien nicht zutrifft. Wie sang Wolfgang Niedecken (von der Kölner Gruppe BAP) einst: "Orwell 84 ess naa, ess mittlerweile nur noch vier läppsche jaar."

Wie diese kurze Übersicht zeigen soll, leiten sich die Nutzungs- und Vervielfältigungsbedingungen mittelbar aus dem Urheberrecht ab; oder — um es klarer ausdrücken — die derzeitigen Situation ist eine Folge des Versagens eines adäquaten Urheberrechts für Software. Nicht nur, dass die Legislative die faktische Definition von Software der Jurisprudenz zuweist; die mit der Novelle des UrhG einhergehenden Änderung lesen sich wie ein Flickenteppich. Das Zustandekommen der Inhalte einzelner Paragraphen und Absätze kann nur durch den Druck der (interessierten) Industrie einerseits und durch ein hilfloses (staatliches) Gegensteuern andererseits verstanden werden.

Im Grunde genommen ist ein DRM-System für Software überflüssig. Klarer definierte EULAs und ihre Verankerung im UrhG sind m.E. rechtlich ausreichend. Gingen die Hersteller dazu über, auf jeder Kopie ihrer Software einen Lizenzschlüssel zu hinterlegen (statt eines Generalschlüssels), würde dies auch den "Serialized" Web-Seiten den Boden entziehen und dem Erwerber ein individuelles Stück Software zusprechen.

Digital geschützte Werke der Literatur und Musik sind ebenfalls — meiner Meinung nach — grober Unfug und inhärent unwirksam. Der Wert eines Buchs richtet sich nicht nur nach dem Inhalt, sondern auch nach der Erscheinung (die Musikindustrie hat hierfür einmal die treffende Bezeichnung "message&massage" parat). Wenn die Plattenindustrie aber glaubt, aktuelle Neuerscheinungen auf CDs (für rund 20) in der Ausgabequalität von Taschenbüchern auf dem Markt platzieren zu müssen, hat sie natürlich auch die Folgen dafür zu tragen. Veteranen unter den Musikhöreren denken mit Wehmut an die liebevoll gestalteten LP-Cover von Hipgnosis (u.a. YES), die die Platten — trotz vielfältiger Überspielung auf Compact Cassette — zum Verkaufsschlager und Kultobjekt werden liessen. Die CD[13] hat dem gegenüber die Musik zur "Bitware" degradiert; der Kulturverlust folgt auf dem Fusse. Die kommerziellen Verwertungsbedingungen sollen mittels DRM optimiert werden.

1.4.4 Patente

Während das UrhG einerseits die Rechte des Schöpfers (= Individuums) als geistigem Eigentümer definiert und das Verhältnis zwischen ihm und dem (anonymen) Erwerber, d.h. Nutzer bzw. Konsument regelt, ist es Aufgabe des Patent- und Markenrechts, gewerbliche Rechtsansprüche zwischen dem Schöpfer und möglichen Verwerten festzulegen.

Gewerblicher Markenschutz

Das Marken-, Gebrauchsmuster- und Geschmacksmusterrecht regelt die Eintragung von gewerblichen Schutzrechten und deren Verwertung allgemein. Hingegen beinhaltet das Patentrecht qualitative Überprüfungen hinsichtlich der Realisierung von patentfähigen Lösungen. In allen Fällen werden Werk, Schöpfer und Kategorie öffentlich gemacht, was zugleich mit spezifischen Nutzungs- und Schutzrechten verbunden ist. Diese betreffen i.d.R. nicht den Endnutzer sondern sind spezifisch für den Wirtschaftszweig.

Eine bekannte "Schutzmarke" ist z.B. CocaCola. Name, Namenszug und Zusammensetzung, d.h. letztlich auch Geschmack dieser phosphathaltigen Brause sind international geschützt. Die Herstellung von CocaCola wird in Lizenz vergeben und sichert somit quasi ein Monopol — sieht man von Mitbewerbern wie Pepsi-Cola, oder Afri-Cola ab, die geschmacklich in die gleiche Richtung gehen. Weniger gut bestellt steht es um die Produkte — obwohl ebenfalls geschützt — des Schweizer Uhrenherstellers Rolex. Nur allzu gerne werden Plagiate und billige Nachbauten in Fernost hergestellt und international vertrieben. Im Gegensatz zum Imitieren von CocaCola lassen sich hier wesentlich grössere Gewinnspannen erzielen und zudem dem Käufer bzw. Besitzer eines

---

[13] Die CD ist nur der Träger der Information; damit einher geht aber eine digitale Bearbeitung der Musik, bei der häufig psycho-akustische statt künstlerische Werkbearbeitungen Vorrang geniessen

Plagiats ein wenig vom Renommee einer begehrten (= teuren) Marke abgeben. Erleichtert können wir feststellen, dass uns dies beim Trinken von CocaCola oder Pepsi-Cola erspart bleibt; obwohl die Marketing-Strategen gerne den Genuss dieser Erfrischungsgetränke mit dem Image und Life-Style von Pop-Idolen wie Mariah Carrey oder Michael Jackson verbinden möchten.

Patente dienen — so die häufig anzutreffende Meinung — dem technischen Fortschritt. Innovative Ideen können hierdurch geschützt und ihre Verwertung (über eine Lizenzvergabe) kontrolliert werden. Im amerikanischen Recht lässt sich der Begriff des Patents nicht so eng auslegen wie im (bisherigen) deutschen Recht beispielsweise. So hat die Firma Apple für die Darstellung des (leeren) Papierkorbs bei Max OS X im Februar 2003 in Patent beim USPTO erwirkt.

Bekannt geworden ist auch der Versuch, Musik-Akkorde zu patentieren. Hierzu zählt z.B. der bekannte Nokia-Ton bei Mobiltelefonen. In die gleiche Kategorie fallen auch Abschnitte des genetischen Codes (C-G-A-U). Auf Grundlage der Digitalisierung von Audio- und Videomaterial kann man schnell die Frage stellen, ob z.B. die Stimme eines Bundeskanzlers Gehard Schröders patentierbar ist. Oder, in Bezug auf Hollywood, die digitale 3D-Kopie eines Humphrey Bogart.

Mit dem Verschwimmen der Definitionen von Patent und Gebrauchsmuster verschmelzen zwei bislang getrennte Bereiche:

- Patente können für neue und öffentlich noch nicht bekannte *Herstellungen* erteilt werden.
- Ein Gebrauchsmusterschutz (im weitestgehenden Sinne) kann für eine spezifische Ausprägung von *Darstellungen* gewährt werden.

Diese mangelnde Unterscheidung zwischen *Wie* und *Was* wird im zukünftigen Patentrecht ergänzt um eine fehlende Definition von *Wann*. Bislang konnte kein Patent erwirkt werden für Sachverhalte, die bereits einschlägig veröffentlicht wurden. Dies sieht man als ein unzumutbare Beschränkung der Industrie. In Zukunft sollen auch dann Patente gewährt werden können, wenn die Beschreibung postum nicht mehr als 6 Monate zurück liegt.

Auch in diesem Zusammenhang wird deutlich, die neuen Rechtsvorschriften gemäss amerikanischem Verständnis *utilaristisch* zu deklarieren. Statt normativ Sachverhalte zu definieren, juristisch aufzubereiten und in die Legislative umzusetzen, wird der Verwendungszweck eines Gesetzes in den Vordergrund gerückt. Hiermit geht einher, dass Firmen sich genötigt sehen, ein "Patentpolster" anzulegen, um sich damit gegenüber anderen Firmen in konkurrenzfähige Situation zu versetzen.

Potentielles Know-how wird somit instrumentalisiert und dem allgemeinen Fortschritt entzogen; Wissen also zumindest "teuer" gemacht. Firmen wie Cisco haben bislang ihre Entwicklungen quasi unentgeltlich öffentlich gemacht. Dies

entspricht weitgehend der gesamten Entwicklung des Internet und ist auch massgeblich mitverantwortlich für den Boom in den letzten Jahren. Es ist kaum vorstellbar, dass unter den Bedingungen eines geänderten Paradigmas, der freie und unbeschränkte Austausch von Informationen und Wissen anhält.

Patente und Software-Entwicklung

Problematisch ist die angestrebte Entwicklung hinsichtlich des Patentrechts nicht in erster Linie für den Endanwender, der — bedingt durch den Patentschutz — nicht oder zumindest nicht rechtzeitig in den Genuss einer Entwicklung kommt, sondern für Firmen bzw. Software-Entwickler generell. Häufig "materialisieren" sich Software-Ideen oder Algorithmen nahezu parallel bei verschiedenen Personen/Entwicklern, weil diese entweder auf einen allgemeinen Problemdruck zurückzuführen sind, oder aber bestimmte Entwicklungen (z.B. neue Compiler) eine vergleichbare Lösung praktisch erzwingen. Man könnte sagen, dass hier ein *common understanding* vorliegt. Im Wettlauf um Patente ist derjenige im Vorteil, der als erster eine Patentschrift einreicht; der andere steht bei Veröffentlichung seines vergleichbaren, aber unabhängig gefundenen Algorithmus mit einem Bein beim Kadi. Dies gilt um so mehr, je weniger komplex das zum Patent eingereichte Verfahren ist, bzw. sich die Patentschrift auf einen zentralen Algorithmus stützt.

Zudem mangelt es an einer systematischen Deklaration und Klassifikation von Patentschriften, mit deren Hilfe schnell und effektiv bestehende Patente auf Attribute und *buzzwords* (z.B. in XML) überprüft werden könnten. In der Mathematik ist es überdies so, dass Verfahren aus der einen Disziplin (z.B. Topologie) auch gewinnbringend beispielsweise in der Optimierung eingesetzt werden können. Die Geschichte der Mathematik weist viele solcher Fälle auf, wie z.B. das Drei-Farben-Problem oder Descart's berühmter "letzter Satz". Damit stellt sich zwingend die Frage ob ein Patent z.B. für einen Algorithmus nur für das dargelegt Problem vergeben ist, oder ob es allgemeinen Charakter besitzt. Mit anderen Worten: Ist die *Methode* (= Weg und Ziel) massgeblich oder erzieltes *Ergebnis*?

Steuerung von Informationen

Die Beschränkung der Weitergabe von Wissen und Know-how, die (mitunter) subtile Steuerung von Informationen mögen für einzelne Interessengruppen und Firmen *effektiv* sein, makroökonomisch und gesamtgesellschaftlich ist dies jedoch nicht *effizient*.

Wie Dan Bernstein selbst erfahren hat, hat die NSA[14] den Daumen auf (sicherheits-)relevanten Software-Entwicklungen. Patente dienen zur Monopolisierung von allgemeinem Wissen. Microsoft verfolgt m.E. den Versuch, mit seinen Betriebssystem und Anwendungen den Markt für den Endanwender zu beherrschen. Die Musikindustrie macht z.Z. Vorstösse, den Einsatz und die Kopierfähigkeit von Produktionen einzuschränken. Suchmaschinen werden

[14] NSA — National Security Agency

manipuliert, um die kommerziell wichtigsten Links an die vorderste Stelle zu "bugsieren": Welcome to the real World!