Datenschutz und Datensicherheit

Prof. Dr. Klaus Pommerening
Johannes-Gutenberg-Universität Mainz

1991

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Einleitung

Daten werden in zunehmendem Maße elektronisch gespeichert und übermittelt, auf Zentralrechnern und Arbeitsplatzrechnern, in öffentlichen und lokalen Netzen. Dabei sind sie vielen Gefahren ausgesetzt: Ausspähung, Verfälschung, Zerstörung. Die Täter hinterlassen kaum Spuren.

Die Informationstechnik entwickelt sich stürmisch. Die Ziele waren zunächst Leistungssteigerung und weltweiter Zugang zu Daten und Informationen. Dabei blieb die Sicherheit der Systeme oft unbeachtet, der Datenschutz geriet unter die Räder der Computer-Euphorie; insbesondere verstößt die Datenspeicherung auf PC-Systemen nach Ansicht der Datenschutz-Beauftragten in aller Regel gegen Datenschutzbestimmungen. Hier gibt es viel Nachholbedarf; das Vertrauen der Anwender muß wiederhergestellt werden.

Vergleicht man den Datenschutz mit dem Problem, sicher zu wohnen, so stehen wir am Beginn des Mittelalters. Wegen umherziehender Banden ist das Wohnen in freier Landschaft zu gefährlich geworden. Die Menschen verbarrikadieren sich in Burgen und befestigten Städten, deren Sicherheitsmaßnahmen immer perfekter werden und dennoch niemals wirklichen Schutz bieten. Der freie Umgang der Menschen miteinander, die Kommunikation und die freie Entfaltung bleiben dabei auf der Strecke, das allgemeine Mißtrauen regiert. Vielleicht gelangen wir auch beim Datenschutz einmal in ein höheres Stadium der Zivilisation, wo nicht mehr solch martialische Schutzmaßnahmen nötig sind.

Das Thema „Datenschutz und Datensicherheit“ hat gesellschaftspolitische, volkswirtschaftliche, rechtliche, organisatorische und technische Aspekte. Es betrifft eines der Hauptprobleme der modernen Industriegesellschaft, die man ja oft schon als „Informationsgesellschaft“ bezeichnet. Daten und Informationen sind der Lebensnerv von Unternehmen und Behörden, von Forschung und Militär. Datenschutzgesetze gibt es in verschiedenen Staaten seit Ende der 70er-Jahre. Gesetze und Vorschriften helfen aber ohne wirksame technische Abwehr- und Kontrollmaßnahmen wenig. Jeder Verkehrsteilnehmer weiß, daß Regeln übertreten werden, wenn keine Überwachung erkennbar ist.

Dieser Text ist das ausgearbeitete Manuskript einer Vorlesung an der Johannes-Gutenberg-Universität in Mainz im Wintersemester 1989/90. In der Vorlesung ging es in erster Linie um die allgemeingültigen Probleme und Lösungsversuche, unabhängig von einem bestimmten Rechnertyp,

Betriebssystem oder Netz. Allerdings erfordern Beispiele oft, auf bestimmte Konstellationen mit spezifischen Problemen hinzuweisen. Viele der Maßnahmen beruhen auf kryptographischen Protokollen und damit auf mathematischen Methoden. Hier lag ein gewisser Schwerpunkt der Vorlesung, aber auch die anderen Aspekte des Datenschutzes werden behandelt. Das erste Kapitel ist etwa zum großen Teil den rechtlichen und gesellschaftspolitischen Aspekten gewidmet, das zweite der Organisation eines sicheren Rechnerbetriebs. Je ein Kapitel handelt von den Problemen des Datenzugriffs und den speziellen Datensicherheitsproblemen in Netzen. Das „klassische“ Rechenzentrum ließ sich mit einem gewissen Aufwand an Baumaßnahmen zu einem „geschlossenen System“ umwandeln, das hinlängliche Sicherheit bot. Dagegen sind moderne Datenverarbeitungssysteme offen, vernetzt und verteilt; ihre Umgebung ist prinzipiell unsicher. Mit herkömmlichen Methoden sind sie nicht zu sichern. Zu ihrem Schutz braucht man unbedingt kryptographische Maßnahmen. Auf diesem Gebiet verläuft die technische Entwicklung zur Zeit rasch voran, aber auch die wissenschaftlichen Grundlagen sind an manchen Stellen noch ausbaubedürftig.

Viele Sicherheitslücken werden ziemlich konkret beschrieben. Selbstverständlich werden die Verfahren zum Brechen von Sicherheitsvorkehrungen, wenn sie nicht sowieso auf der Hand liegen, nicht so detailliert ausgeführt, daß sie direkt nachgemacht werden können. Es geht ja nicht um Anleitungen zum Einbruch, sondern um eine Schärfung des Sicherheitsbewußtseins der Verantwortlichen für die Datenverarbeitung. *Mit Ignoranz ist kein wirksamer Datenschutz zu betreiben.*

Dieses Buch soll allen helfen, die für das ordnungsgemäße Funktionieren einer Datenverarbeitungsanlage zuständig sind. Für sie sind einerseits Zusammenstellungen der wichtigsten Maßnahmen wichtig, die sie verwirklichen müssen oder können; darüber hinaus brauchen sie auch umfassende Kenntnis von Sicherheitsproblemen, deren Behebung nicht in ihrem Bereich liegt, etwa bei öffentlichen Netzen, bei der Hardware, bei Standard-Software, um die Sicherheit ihres Betriebs beurteilen und Erweiterungen sachgemäß planen zu können. An die Hersteller sind einige Forderungen zu richten. Die Situation wird in [51] mit der Einführung von Sicherheitsgurten in den Autos verglichen: Nur ein geschärftes Sicherheitsbewußtsein bei Kunden und Herstellern zusammen mit gesetzlichen Regelungen kann die Einführung von Sicherheitsmaßnahmen auf dem Markt durchsetzen.

Zur Erleichterung bei der Anwendung enthält der Anhang „Checklisten“ zum Datenschutz. Diese bestehen weitgehend aus Fragen, die bei der Aufstellung eines betrieblichen Sicherheitskonzepts zu beantworten sind. Sie sollen helfen, Sicherheitsprobleme zu erkennen und gegebenenfalls die geeigneten Maßnahmen zu veranlassen. Auch eine kleine Liste von Sicherheitsprodukten im PC-Bereich ist dort aufgeführt, ebenso die Beschreibung von zwei kleinen PC-Programmen, die zwei besonders ärgerliche Sicherheitslücken stopfen sollen.

Ich habe versucht, das Buch nicht theoretisch zu überfrachten. Abstraktion wird nur dort eingeführt, wo sie für das Verständnis nötig oder förderlich ist.

Das ist vor allem im Kapitel über Verschlüsselung der Fall. Aber auch dort ist der Formalismus so gering wie möglich gehalten; für die Frage, was möglich ist, reicht an solchen Stellen ein diagonales Lesen. Wer allerdings zu einer fundierten Bewertung der Verfahren kommen will, muß nach dem Warum fragen, und dazu ist die theoretische Vertiefung nötig, vielleicht sogar anhand der weiterführenden Literatur.

Ich danke allen, die durch Diskussion, Literaturhinweise und Verbesserungsvorschläge mitgeholfen haben, vor allem H. Hill, P. Kaatsch, J. Michaelis, K.-H. Schicketanz und G. Wetter. Mein Dank gilt auch A. Beutelspacher, von dem ich eine Menge über kryptographische Protokolle gelernt habe, und Frau G. Klein fürs Korrekturlesen.

Kapitel I

Grundprobleme der Datensicherheit

Vor der Planung oder Bewertung konkreter Maßnahmen zum Datenschutz steht das Erkennen der möglichen Probleme. Welche Gefahren drohen den Daten? Welche Gefährdung geht von ihnen aus? Welche Vorfälle gab es in der Vergangenheit? Wie reagiert der Gesetzgeber? Welche Daten müssen überhaupt geschützt werden? Aus welchem Grund und vor wem?

1 Bekanntgewordene Vorfälle

Immer wieder liest man in der Presse von Fällen, in denen in der einen oder anderen Weise gegen den Datenschutz verstoßen wurde oder Mängel bei der Datensicherheit offenbart wurden. Hier folgt eine ziemlich unsystematische Aufzählung, die mehr oder weniger zufällig zustandegekommen und weder vollständig noch auch nur repräsentativ ist. Die meisten Fälle stammen aus der jüngsten Zeit und sind der Tages- oder Wochenpresse entnommen, wobei Der Spiegel und Die Zeit besonders ergiebig waren. Eine weitere wichtige Quelle ist eine elektronische Zeitschrift, die man als EARN- oder Internet-Teilnehmer beziehen kann: Schreibt man sich für 'RISKS' beim 'LISTSERV at FINHUTC' (EARN) oder bei 'RISKS-Request@CSL.SRI.COM' (Internet) ein, so erhält man regelmäßig, etwa ein- bis zweimal pro Woche, den 'RISKS-FORUM Digest', im folgenden kurz „Risks-Digest" genannt. Dies ist ein öffentliches Diskussionsforum, zu dem jeder Beiträge liefern kann. Die Veröffentlichung ist allerdings „moderiert", das heißt ein verantwortlicher Redakteur unterdrückt unwichtige, unseriöse oder wiederholte Beiträge.

Nun zum ersten Fall: Anfang der sechziger Jahre bearbeitete auf dem CTSS-System am MIT ein Systemverwalter die Paßwortdatei, ein anderer gleichzeitig eine Meldung an die Benutzer. Durch einen Fehler im Betriebssystem wurden die

temporären Editor-Files beim Abspeichern vertauscht, so daß eine Zeitlang auf jedem Terminal nach der Anmeldung die Paßwortliste erschien [86]. Problem: Die Benutzerbereiche werden vom Betriebssystem nicht sauber getrennt.

Für die nächsten beiden Fälle von Computerkriminalität aus den 70er-Jahren kenne ich die genauen Daten nicht. In einem solchen Fall legte der Täter in Banken Einzahlungsformulare aus, auf denen im Feld für den Klarschriftleser schon eine Kontonummer eingedruckt war – seine natürlich. Kunden, die irgendwelche Einzahlungen mit diesen Formularen machten, konnten ein beliebiges Konto angeben; es wurde nicht berücksichtigt, da ja im Klarschriftleserfeld schon eines stand. Hier wurde also der Computer auf Grund eines fehlerhaften Verfahrensablaufs überlistet.

In einem anderen solchen Fall automatisierte ein Manager einer amerikanischen Versicherungsgesellschaft die Produktion von Scheinpolicen und verkaufte fiktive Versicherungsverträge an einen Rückversicherer. Er wurde erwischt und zu acht Jahren Freiheitsstrafe verurteilt. Das Prinzip dieses Verbrechens war, daß man mit einem Computer ganz leicht und schnell große Mengen authentisch aussehenden Papiers erzeugen kann, das dazu noch ein in sich konsistentes Bild bietet.

Im November 1983 setzte Fred Cohen an der University of Southern California auf einem VAX-Rechner das erste Computervirus in die Welt. Schon eine Woche später mußte die Universitätsleitung einer ganzen Reihe von Studenten die Rechenerlaubnis entziehen. Hier wird das Problem der moralischen Einstellung der Benutzer deutlich.

Im November 1984 wies der „Chaos Computer Club“ mit seinem „Btx-Hack“ auf Sicherheitslücken im deutschen Btx-System hin, nachdem die Bundespost seine vorherigen Eingaben als Unsinn abqualifiziert hatte. Die Hacker entschleierten angeblich Paßwörter durch gezieltes Überlaufen von Datenpuffern und riefen dann Btx-Seiten zu Lasten einer Hamburger Sparkasse ab. Innerhalb von dreizehn Stunden erzeugten sie damit ein Schuldkonto von 134000 DM zu ihren Gunsten. In einem neuen Fall benutzten Btx-Hacker die „Paßwortfalle“, um Paßwörter auszuspähen – sie gaben als Btx-Anbieter eine nachgemachte Zugangsmaske aus, leiteten das eingegebene Paßwort an sich selbst weiter und kassierten sogar noch Gebühren für die ausgegebene Seite [122]. Die Hacker deckten damit auf, wie technischer Fortschritt blind für Gefahren voran getrieben wird. Gründliche Überlegungen zur Sicherheit dauern zu lang und kosten zuviel Geld.

Zum Jahreswechsel 1984/85 machte ein eingebautes Verfallsdatum das Programm „Gurugs“ an der Bundeswehrhochschule München unbrauchbar. Sein Schöpfer hatte es wegen einer Streitigkeit mit der Hochschulleitung eingebaut. Die Abhängigkeit einer Institution von ihren Fachleuten kann eine gefährliche Verwundbarkeit bedeuten.

Im September 1987 wurde der „NASA-Hack“ bekannt; deutsche Hacker waren über das Datex-P-Netz ins Datennetz SPAN (‘Space Physics Analysis Network’) der NASA eingedrungen und hatten sich lange Zeit dort herumgetrieben.

An dieses Netz sind etwa 1200 Rechner angeschlossen. Auf 135 Rechnern erlangten die Hacker Zugang zu zwar nicht sensitiven, aber auch nicht für die Öffentlichkeit bestimmte Daten. Dabei nützten sie einen Fehler im VAX-Betriebssystem VMS aus. Dieses ist ein Großrechner-Betriebssystem. Mit seiner Wartung und Weiterentwicklung sind ständig etwa 300 Leute beschäftigt; die Dokumentation umfaßt etwa 500000 Seiten. Bei Software-Projekten dieser Größenordnung schleichen sich notwendigerweise viele Fehler ein.

Das Weihnachtsvirus 1987 wurde von Clausthal-Zellerfeld aus gestartet. Es ließ Weihnachtsbäume auf den Bildschirmen infizierter Systeme erscheinen, richtete aber keinen direkten Schaden an. Ein indirekter Schaden allerdings war eine zeitweise Verstopfung im Netz.

Am 2. November 1988 setzte der Student Robert Morris an der Cornell University in Ithaka den Internet-Wurm aus, der in kurzer Zeit etwa 6000 amerikanische Computer infizierte, und zwar Sun- und VAX-Rechner mit dem Betriebssystem 4 BSD UNIX. Er nützte dabei zwei verschiedene Systemlücken aus und verwendete ein Paßwort-Entschlüsselungsprogramm. (Einzelheiten werden später behandelt.) Der „Wurm“ war „gutartig“ in dem Sinne, daß er, obwohl das leicht möglich gewesen wäre, Daten weder zerstörte noch beschädigte, Privatpost unberührt ließ, Paßwörter nicht weitergab und keine „Trojanischen Pferde“ hinterließ. Allerdings richtete er doch einigen Schaden an, indem er Ressourcen blockierte (‘denial of access’), den Netz-Verkehr behinderte, Benutzer-Zulassungen mißbrauchte und Systempersonal unter Streß setzte.

Im März 1989 wurde der „KGB-Hack“ bekannt: Deutsche Hacker waren über Datex-P, Tymnet X.25 und Internet in amerikanische Rechner der US-Army und von Rüstungsfirmen eingedrungen und hatten ihre Erkenntnisse an den sowjetischen Geheimdienst KGB weitergeleitet. Es gibt aber keine Hinweise, daß dabei wirklich geheime Daten ausgespäht wurden. Die Hacker nützten bekannte Lücken in UNIX, VMS, MVS/TSO und anderen Betriebssystemen aus, ferner die Nachlässigkeit einiger Systemverwalter und einen Fehler in der Mail-Option des Editors Gnu-Emacs, der teilweise ermöglichte, sich Systemverwalter-Privilegien zu verschaffen. Interessant an diesem Fall ist, daß aufmerksames Personal am Lawrence Berkeley Laboratory diese Hacker seit 1986 immer wieder beobachtete, ihre Sitzungen an diesem Rechner, der oft als Durchgangsstation benutzt wurde, zum großen Teil mitprotokollierte und sie schließlich in eine Falle lockte, die ihre Überführung ermöglichte.

Im Mai 1989 gab die Bundesregierung als Antwort auf eine kleine Anfrage der Grünen zu, daß das Bundesamt für Verfassungsschutz Daten von Besuchern terroristischer Strafprozesse sammelt und speichert [67]. Es gibt offenbar keine gesetzliche Grundlage, dieses zu unterbinden. Im Oktober 1989 gab die Bundesregierung auch zu, daß dieselbe Stelle die Daten von Aus- und Übersiedlern aus Osteuropa und der DDR speichert, einschließlich früherer Adressen und Arbeitsplätzen. Dabei wurden auch illegale Tricks verwendet, um an diese Daten zu kommen [123].

Im Juni 1989, auf der Messe CeBIT in Hannover, half ein Computer-

Sicherheitsberater der Kriminalpolizei, einen mutmaßlichen Wirtschaftsbetrüger zu überführen. Dessen Computer war zwar beschlagnahmt worden, die Daten aber durch das Schutzprogramm „PC-Lock“ geschützt. Das Knacken dauerte 30 Minuten. Um die gleiche Zeit knackte der Experte Georg Krause, Geschäftsführer einer Elektronik-Firma, das Konkurrenzprodukt „Oculis“ in 60 Minuten. In letzter Zeit ist ein gewaltiger Markt für Sicherheitsprodukte gerade im PC-Bereich entstanden, auf dem manche Firmen mit billig gemachten Produkten das schnelle Geld verdienen wollen.

Am 3. Oktober 1989 kam über den Risks-Digest die Meldung, daß ein unbekannter Hacker in DEC-Rechner mit dem Betriebssystem ULTRIX 3.0 eingebrochen ist. Aufgrund einer Lücke im Filetransfer-Programm ‘tftpd’, die in der Version 3.1 behoben ist, konnte er sich Systemverwalterprivilegien verschaffen, sobald er nur irgendeine normale Benutzeridentität erschlichen hatte. Das Problem ist hier, daß bekanntgewordene Sicherheitslücken oft nicht schnell genug geschlossen werden. Auf demselben Wege wurde wenige Tage später die Beobachtung eines Wurms mitgeteilt, der sich auf VAX-Systemen im SPAN über das DECnet-Protokoll ausbreitete und eine Friedensbotschaft als Systemmeldung absetzte (‘WORMS AGAINST NUCLEAR KILLERS – WANK – Your System Has Been Officially WANKed – You talk of times of peace for all, and then prepare for war.’) Das wirft wieder einmal die ethische Frage auf: Darf man aus übergeordneten moralischen Gründen beschränkte Regelverstöße in Kauf nehmen?

Am 5. Oktober 1989 wurde im Regensburger Einkaufszentrum feierlich der Startschuß für „electronic cash“ gegeben, das elektronische Bezahlen an der Ladenkasse mit Eurocheque-Karte und zusätzlicher Geheimnummer. Peinlicherweise hatten wenige Tage zuvor zwei EDV-Spezialisten die Schwäche des Systems vorgeführt, indem sie ein Duplikat einer Eurocheque-Karte anfertigten und damit an einem Bargeldautomaten von einem fremden Konto Geld abhoben. Das Problem hierbei ist, daß Betreiber von technischen Anlagen oft betriebsblind vor Lücken in ihrer Konstruktion stehen, die von sachkundigen Außenseitern aufgespürt und ausgenützt werden können.

Im Oktober 1989 stellte die Verbraucherzentrale Rheinland-Pfalz bei Testeinkäufen fest, daß Scanner-Kassen sehr fehlerhaft arbeiten. Es kamen abweichende und doppelt oder gar nicht erfaßte Preise vor. Auch dies ist wieder ein Beispiel, wo der technische Fortschritt ohne Rücksicht auf entstehende Gefahren vorangetrieben wird.

Am 16. Oktober 1989 wurde ebenfalls im Risks-Digest eine Meldung der New York Times vom Vortag verbreitet, in der es hieß: Die Opposition in Indien protestierte gegen die Verwendung von Computern für die Auswertung von Wahlen, weil dadurch der Regierungspartei die Wahlmanipulation zu sehr erleichtert würde.

Ebenfalls im Oktober 1989 bemängelte der Bundesrechnungshof Sicherheitslücken in Rechenzentren der Bundeswehr:

- fehlende Risikoanalysen für eingesetzte Verfahren,
- fehlende Sicherheitskonzepte,
- unzureichende Regelungen für den Katastrophenfall,
- mangelnde Zugangskontrollen zu Sicherheitsbereichen, etwa Datenträgerarchiven,
- Kabelschächte mit Kabeln unbekannter Herkunft,
- private Antennenanlage auf einem Dach.

Der „Stern“ berichtete am 26. Oktober 1989 von Fällen, wo Datenverarbeitungsaufträge an Strafanstalten vergeben wurden und dabei kartonweise persönliche Daten hinausgeschmuggelt wurden [144]. Hier tritt das Problem der Auftrags-Datenverarbeitung in besonders krasser Form auf.

Eine dpa-Meldung aus der Mainzer Allgemeinen Zeitung vom 19. Oktober 1989: „Ein 14 Jahre alter amerikanischer Hacker hat mit einem nur 200 Dollar teuren Computer den Code einer Bank in New York geknackt und mit Hilfe der dabei erlangten Kreditkarten bei Versandfirmen Waren im Wert von mehr als 11000 Dollar bestellt. Die Polizei im kalifornischen Fresno mußte ihn wieder freilassen, weil sie keine rechtliche Handhabe gegen den Minderjährigen hatte. Allerdings werden seine Eltern vermutlich die Rechnung für die Einkäufe begleichen müssen.“ Diese Meldung ist recht ungenau, was das Vorgehen betrifft; mehr kann man aber von der Tagespresse nicht erwarten. Im Risks-Digest wurde genauer ausgeführt, daß der Junge Paßwörter für Kreditauskünfte bei der TRW-Kredit-Bank in Hackerkreisen aufgeschnappt hatte. Er suchte dann zufällig Personen aus dem Telefonbuch heraus und gab sich gegenüber der Bank als Firma aus, die die Bonität von Kunden prüfen wollte. In den Kreditauskünften fand er die Kreditkarten-Nummern.

Das Wirtschaftsmagazin „DM“ berichtete im März 1990 von Plänen, in Supermärkten den „großen Bruder“ zu installieren in Form einer Überwachung des Käuferverhaltens. Ziel ist, die Waren im Supermarkt zum Vorteil des Handels optimal zu plazieren. Dazu sollen an den Einkaufswagen kleine Sender montiert werden; Sensoren an der Decke registrieren dann jeden Schritt des Kunden.

Allen Beispielen mehr oder weniger gemeinsam ist der leichtsinnige Umgang mit Daten, ungetrübt von Sicherheitsüberlegungen und ohne Sensibilität für drohende Gefahren oder für den Schutz der Privatsphäre von Bürgern.

2 Aspekte des Datenschutzes

Nach diesem Blick in die Praxis der (jüngsten) Vergangenheit sollen nun die rechtlichen und gesellschaftspolitischen Aspekte des Datenschutzes systematisch, wenn auch nur oberflächlich dargestellt werden. Zuvor ist aber noch eine Festlegung der verwendeten Begriffe nötig.

2.1 Grundbegriffe

Zunächst also eine Definition wichtigsten Begriffe:

Datenschutz ist der Schutz von Daten vor Mißbrauch, unberechtigter Einsicht oder Verwendung, Änderung oder Verfälschung, aus welchen Motiven auch immer. Im engeren Sinne, etwa in der Gesetzgebung, handelt es sich dabei nur um personenbezogene Daten; im allgemeinen Sprachgebrauch, und so auch hier, werden aber alle Daten, die irgendwo gespeichert sind, einbezogen. Auch der Schutz der Integrität eines Systems gehört dazu, und der ist in vielen Fällen wichtiger als der Schutz der Vertraulichkeit der gespeicherten Daten. Einzubeziehen ist auch der Schutz vor Fehlern und der Schutz vor Folgefehlern im Falle eines Fehlers.

Katastrophenschutz ist der Schutz von Daten vor Zerstörung durch äußere Gewalten oder Sabotage.

Datensicherung ist die Gesamtheit aller organisatorischen und technischen Vorsorgemaßnahmen gegen Verlust, Fälschung und unberechtigten Zugriff auf Grund von Katastrophen, technischen Ursachen, menschlichem Versagen oder mutwilligen Eingriffen. Der Begriff „Datensicherung“ wird auch im engeren Sinne gebraucht als Anfertigung von Sicherheitskopien, auf die man im Notfall zurückgreifen kann.

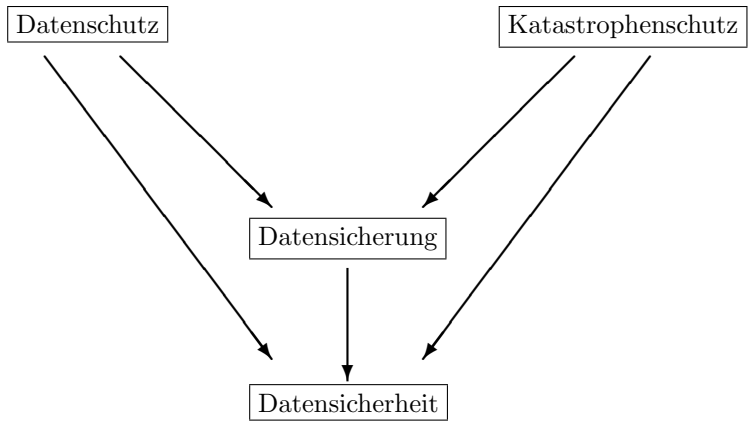
Datensicherheit ist der angestrebte Zustand, der durch alle diese Maßnahmen erreicht werden soll, aber letztlich nicht vollkommen erreicht werden kann.

Im Sinne dieser Definitionen ist der *Datenschutz* der hauptsächliche Gegenstand dieser Abhandlung. Die Beziehungen zwischen den vier Begriffen sind in Abbildung I-1 graphisch dargestellt.

Daten oder im allgemeineren Sinne Informationen haben einige typische Eigenschaften, die sie von materiellen Besitztümern unterscheidet. Die Unterschiede beruhen hauptsächlich auf der leichten Duplizierbarkeit. Daten lassen sich vervielfältigen, ohne geteilt zu werden – zwei verschiedene Besitzer besitzen jeweils das Ganze. Das heißt aber nicht unbedingt, daß sich der Wert verdoppelt hat wie etwa bei einem duplizierten Tonband mit Musikaufnahmen; die Daten können auch für den Eigentümer völlig wertlos werden, wenn ein anderer sie in seinen Besitz bringt – zum Beispiel die Geheimnummer für den Geldautomaten oder Unterlagen für ein noch nicht angemeldetes Patent. Schlimmer noch ist, daß das Duplizieren der Daten unbemerkt vom rechtmässigen Besitzer vor sich gehen kann. Man kann sich das am Beispiel eines Zahlenschlosses verdeutlichen; den Verlust einer Ziffernkombination braucht man nicht zu bemerken, im Gegensatz zum Verlust eines richtigen „harten“ metallenen Schlüssels.

Die elektronische Datenverarbeitung bringt drei wesentliche Neuerungen gegenüber der manuellen mit sich:

Abbildung I-1: Komponenten der Datensicherheit



- Aufhebung der zeitlichen Schranken – viele Vorgänge, die bei manueller Bearbeitung Jahre dauern würden, wie das Entschlüsseln eines Paßworts oder das Durchsuchen riesiger Dateien, gehen elektronisch in Sekundenschnelle.
- Aufhebung räumlicher Entfernungen – mit Hilfe von weltumspannenden Netzen kann man praktisch jederzeit an jede beliebige Information gelangen, egal wo sie gespeichert ist.
- Minderung der menschlichen Unzulänglichkeit – bei einer Datenbankanfrage etwa wird kein Fall übersehen.

Der Datenschutz hat 3 wesentliche Aspekte:

- Rechtlich/politisch, wozu beim Datenschutz in Betrieben auch Betriebsinteresse und -politik gehört.
  - Gesellschaftspolitische Forderungen, etwa das „informationelle Selbstbestimmungsrecht“.
  - Gesetzliche Rahmenbedingungen, Datenschutzgesetze.
  - Technikfolgenabschätzung, etwa bei der Steuerung industrieller Anlagen oder beim Geldtransfer.

- Organisatorisch.
  - Einbindung der Datensicherheit in das allgemeine EDV-Konzept.
  - Benutzergruppen, Definition von Zugriffsrechten, Paßwort-„Politik“.
  - Katastrophenplanung, Checklisten, Sicherheitsnormen.
  - Personalpolitik, Betriebsklima, Überwachungssysteme.
  - Dienstvorschriften, Zuständigkeiten.
  - Dokumentation, Datenschutzbericht. Es gilt das Prinzip der Revisionsfähigkeit.
  - Entscheidung über das grundsätzlich anzustrebende Sicherheitsniveau, über den Arbeitsfaktor für einen Angreifer (wieviel Zeit und Geld wird er voraussichtlich investieren, um Sicherheitsschranken zu überwinden?), über Offenheit oder Geschlossenheit des Systems.
  - Sicherheitsprobleme bei Zentralisierung oder Dezentralisierung.
  - Abwägen von Schutzanforderungen und Leistungsanforderungen, Prinzip der Verhältnismäßigkeit „Absolute Sicherheit ist nur bei Stillstand des Systems zu erreichen.“ [136]
- Technisch – Umsetzung der rechtlichen und politischen Anforderungen und der organisatorischen Definitionen in konkrete Maßnahmen.
  - Physische Schutzmaßnahmen und Baumaßnahmen: Zugang zu Geräten und Übertragungsleitungen, Abhörsicherheit.
  - Schutzmaßnahmen im Betriebssystem: Erlaubnisse zur Benutzung eines Rechners oder zur Kommunikation über Netze, Identifikationskontrolle, Aufzeichnung von Ereignissen zur Beweissicherung, Fehlerüberbrückung.
  - Kryptographische Schutzmaßnahmen: Verschlüsselung von Dateien, Protokolle zur sicheren Datenübertragung, Authentisierung, elektronische Unterschrift, Anonymität.

Diese drei Aspekte entsprechen ungefähr der Einteilung

Anforderungsdefinition – Entwurf – Implementierung,

aus den Ingenieurswissenschaften (etwa Software-Engineering). Die gesellschaftspolitischen und rechtlichen Forderungen nach Datenschutz und Datensicherheit sind durch organisatorische und technische Maßnahmen in der Praxis durchzusetzen. Die gesellschaftlichen, politischen, rechtlichen, betrieblichen Rahmenbedingungen definieren die zu schützenden Daten und die Grundsätze des Umgangs mit ihnen. Durch organisatorische Entscheidungen werden Rechte,

Pflichten und Verantwortlichkeiten festgelegt und Grundsatzfragen über Aufwand, Sicherheitsniveau und Verhältnismäßigkeit geklärt. Die technischen Maßnahmen sorgen dafür, daß Rahmenbedingungen und organisatorische Entscheidungen nicht nur auf dem Papier stehen, und erzwingen, soweit möglich, ihre Einhaltung.

Der Betreiber eines Systems ist oft nicht der Besitzer der Daten. Er hat fremde, dem System anvertraute Daten nach seinen Möglichkeiten zu schützen. Das gilt für den Leiter eines Rechenzentrums, das Daten im Auftrag verarbeitet, genauso wie für einen Netzbetreiber, der sein Netz zum Datentransport zur Verfügung stellt, oder den Arzt, der auf seinem Praxis-PC Patientendaten speichert. Der Betreiber hat seinem „Kunden“ mitzuteilen, welchen Schutz er nicht gewährleisten kann; erforderlich ist eine klare Verteilung der Verantwortung für die Daten zwischen Eigentümer, Bearbeiter und Systembetreiber. Eine typische Verteilung der Verantwortungsbereiche zeigt Abbildung I-2.

Für die Datensicherheit gilt das Prinzip, daß jede Kette nur so stark ist wie ihr schwächstes Glied. Es hat keinen Sinn, ein Rechenzentrum militärisch abzusperren, wenn man leicht über ein öffentliches Netz als Systemverwalter Zugang bekommen kann, ebensowenig, Daten auf einer Festplatte zu verschlüsseln, wenn Algorithmus und Schlüssel in einem Klartextprogramm enthalten sind. Eine gründliche Schwachstellenanalyse empfiehlt sich immer, wenn Datensicherheitsmaßnahmen geplant werden. Besonders nützlich, aber auch kostspielig ist eine Angriffs-Simulation durch ein „Tiger-Team“.

2.2 Gefahren auf gesellschaftspolitischer Ebene

In einer Abhandlung, die sich letzten Endes mit vielen technischen Details beschäftigt, darf zumindest am Anfang nicht ein Überblick über die politischen Aspekte des Themas Datenschutz fehlen. Hier einige Gesichtspunkte:

1. Gläserner Bürger, Persönlichkeitsprofile, Schlüsse auf Lebensumstände, die in den Daten explizit nicht enthalten sind, durch Zusammenführung von Daten und Datenabgleich, erleichtert durch zunehmende Vernetzung.

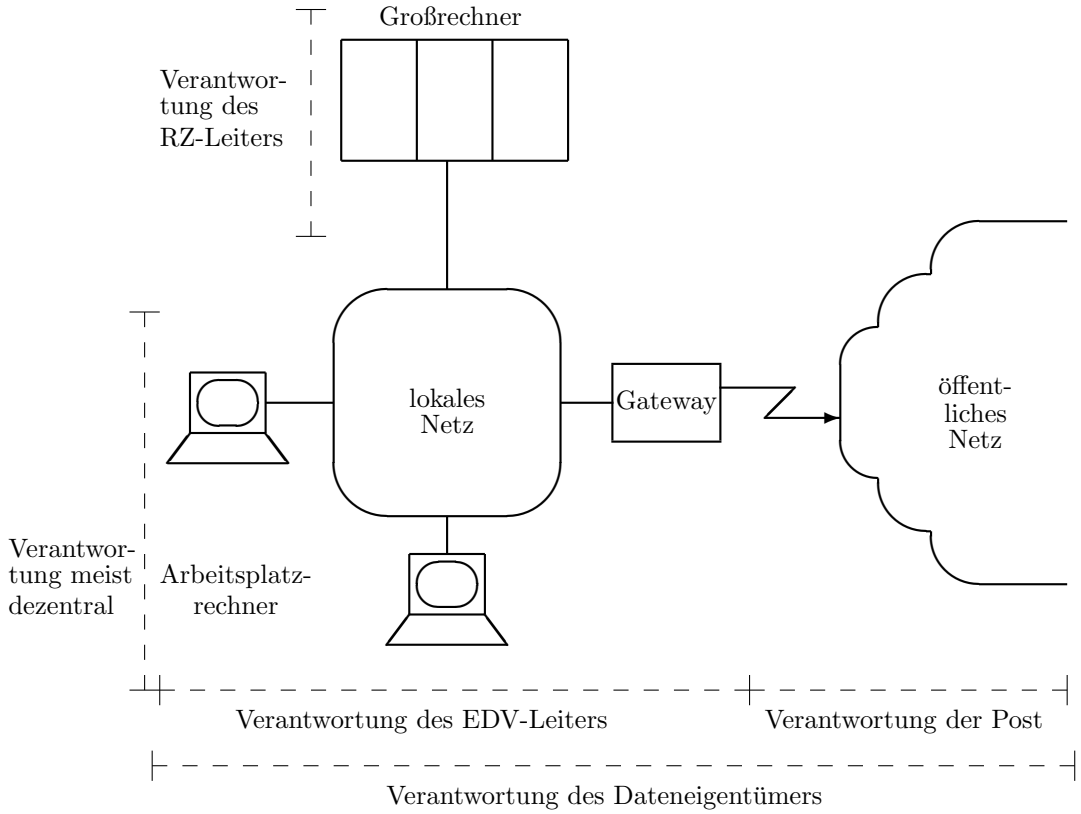
2. Datenschutzprobleme durch Zusammenwachsen der Europäischen Gemeinschaft [121], in den einzelnen Staaten unterschiedliche oder gar keine Datenschutzgesetze, Datenverarbeitung im Ausland.

3. Verstärkte Überwachung, Speicherung von Verbindungsdaten bei Telekommunikation, elektronische Geschäftsabwicklung, Telearbeit, Personal-Informationssysteme.

4. Abhängigkeit von verletzlicher Technik [119].

5. Zugangskontrollen durch Überprüfung persönlicher Merkmale [117], Bewegungsprofile durch Magnetkarten-Überwachung (Risks-Digest 10.54).

Abbildung I-2: Ein typisches EDV-System



6. Unvollkommene Datensicherheit bei der Überwachung gefährlicher industrieller oder militärischer Anlagen.
7. Computerkriminalität, Täter, die keine Spuren hinterlassen, zahlreiche Mißbrauchsmöglichkeiten.
8. Kollisionen von Datenschutz mit anderen Interessen.
9. Allgemeines Mißtrauen, Ende des freien Gedankenaustauschs unter Wissenschaftlern.

Einige dieser Punkte werden noch näher erläutert.

Zu 1: Der gläserne Bürger wird geschaffen durch Datensammlung verschiedener öffentlicher Stellen, Speicherung von Daten elektronischer Vorgänge wie Geschäftsabwicklung und Postverkehr, Speicherung von Kundendaten bei Händlern und Kreditauskunfts-Organisationen und von Personaldaten in Betrieben. Besondere Gefahren lauern in der versehentlichen Speicherung falscher Daten. Durch die Informationstechnik werden ungeheure Machtinstrumente geschaffen, die nur darauf warten, von einem totalitären Regime verwendet zu werden. Bedenklich ist auch, wenn Datensammlungen öffentlicher Stellen im Auftrag von privaten Firmen bearbeitet werden, wie es im Risks-Digest 9.44 aus den USA berichtet wird. Ein großes Problem wirft auch das Krebsregister der DDR auf [32].

Mindestens so umfangreich wie die Datensammlungen bei Behörden sind wohl privatwirtschaftliche Datensammlungen, etwa von Handelsauskunfteien oder Versicherungsunternehmen (geheime „Warndateien“). Wer die Klausel nicht unterschreiben will, daß er seine früheren, gegenwärtigen und zukünftigen Ärzte von ihrer Schweigepflicht entbindet, braucht erst gar keinen Antrag auf Lebensversicherung zu stellen. Das Datenschutzgesetz bietet gegen solche Sammlungen kaum Handhaben, nicht einmal Kontrollmöglichkeiten. Auch Krankenkassen speichern viele sensible Daten ihrer Mitglieder und werden oft von den Datenschützern für ihren Umgang damit gerügt, so etwa von der rheinlandpfälzischen Datenschutzkommission in ihrem Bericht für die Jahre 1988 und 1989. Allgemein unterhalten die Institutionen der „Sozialverwaltung“ umfangreiche Datensammlungen [37, S.195ff.]: „In den Datenbanken der Sozialversicherung sind sowohl personenbezogen als auch lückenlos erfaßt:

- Name, Anschrift, Bankverbindung, Beitragsgruppe, Versicherungszeiten, Beitrags-, Ersatz- oder Ausfallzeiten (u. a. Zeiten des Wehr- oder Zivildienstes), Beitragsgruppe und Staatsangehörigkeit,
- Angaben zu Größe und Gewicht, Art und Schwere von ärztlich behandelten oder einem Gutachter bekannt gewordenen Erkrankungen, die Diagnosen, Befunde (u.a. EKG, Röntgen, Nuklear), psychologische oder psychiatrische Testergebnisse, Datum und Stelle der Untersuchung, Krankheitsverläufe, Abhängigkeiten von Suchtmitteln, Geschlechtskrankheiten, Daten zur Klinikeinweisung und -entlassung, behandelnder Arzt, Therapien,

Arznei- und Heilmittelverbrauch, Art der Behinderung, Wohnverhältnisse bei Kranken und Behinderten, Maßnahmen der Eingliederung, Grad der Erwerbsfähigkeit, Gewährung von Kuren, Massagen u. ä., medizinische Hilfsmittel, vertrauensärztliche Gutachten, Klinikentlassungsberichte,

- Schul- und Berufsausbildung (einschl. Abschlüsse und Praktika), Arbeitsstätten, Betriebsnummer, Krankenversicherung, Verdienst, Arbeitszeiten, Arbeitsunterbrechungen und Fehlzeiten, Art und Ausmaß von Arbeitsunfällen, Stellenwechsel, Teilnahme und Art der Berufsförderungsmaßnahme, Bezug von Vorruhestandsgeld, Rente, Arbeitslosigkeit,
- Empfänger von (welchen?) Sozialleistungen (wie Sozialhilfe, Wohngeld, Arbeitslosengeld oder -hilfe),
- straftatbegründete Unfallzusammenhänge, Haftstrafen,
- Familienstand, Datum der Eheschließung, Name des Ehepartners, Ehescheidung, Versorgungsausgleich, Kinderzahl, Kindschaftsdaten, Höhe der Unterhaltszahlung,
- Schwangerschaftsberatung oder -abbruch, Inanspruchnahme von Drogenberatung,
- Sterbedaten.

Die Aufzählung ist keine erschöpfende Darstellung aller Datensätze in der Sozialversicherung.“ [37, S.202f.] Wie soll sich ein Bürger gegen eine „Datenschutzklausel“ der folgenden Art wehren?

> „Ich willige ein, daß der Versicherer im erforderlichen Umfang Daten, die sich aus den Antragsunterlagen oder der Vertragsdurchführung (Beiträge, Versicherungsfälle, Risiko-/Vertragsänderungen) ergeben, an den . . . -Verband und andere Versicherer zur Beurteilung des Risikos und der Ansprüche übermittelt.
>
> Ich willige ferner ein, daß die Versicherer der . . . -Versicherungsgruppe, soweit das der ordnungsgemäßen Durchführung meiner Versicherungsangelegenheit dient, allgemeine Vertrags-, Abrechnungs- und Leistungsdaten in gemeinsamen Datensammlungen führen und an ihre Vertrauensleute weitergeben.
>
> Auf Wunsch werden mir zusätzliche Informationen zur Datenübermittlung zugesandt.“

Verschärft wird das gesellschaftspolitische Problem der großen Datensammlungen durch die Möglichkeit des Zusammenführens und Abgleichens verschiedener Datenbestände. Auf diese Weise lassen sich die verstreuten Daten zu „Profilen“ mit hohem Informationsgehalt verdichten:

Bewegungsprofile aus Daten der Verkehrsüberwachung, Ausweiskontrolle und sonstigen „elektronischen Spuren“.

Käuferprofile aus Daten von bargeldlosen Zahlungsvorgängen.

Mitarbeiterprofile an Rechenanlagen und elektronischen Arbeitsplätzen durch Zugangsregistrierung und Logdateien.

Benutzerprofile an Informationssystemen.

Kommunikationsprofile aus Verbindungsdaten verschiedener Kommunikationseinrichtungen.

Allein schon das Bewußtsein, daß Daten über mich in hunderten von Datenbanken existieren, muß mich motivieren, intensiv über Datenschutz-Maßnahmen nachzudenken.

Zu 2: Die Gefahr, daß die in der Bundesrepublik Deutschland verbotenen Handlungen vom Ausland aus betrieben werden, wächst mit dem geplanten Zusammenwachsen der EG weiter. Die Datenübertragung ins Ausland ist schon bisher kaum zu verhindern oder auch nur zu kontrollieren, die Datenverarbeitung im Ausland dann völlig legal und unbeschränkt. Neben den Steueroasen entstehen auch Datenoasen. Der Zwang, international wettbewerbsfähig zu bleiben, treibt die Wirtschaft in diese Datenoasen.

Zu 3: Die Erfassung von Verbindungsdaten bei Telekommunikation wird begründet durch die Notwendigkeit zur Steuerung und zur Abrechnung des Kommunikationsprozesses. Schon allein die Information, wer mit wem zu welcher Zeit kommuniziert hat, kann Rückschlüsse zulassen und unterliegt grundsätzlich dem Datenschutz. Die Speicherung dieser Daten liegt nahe und bedeutet so gut wie keinen zusätzlichen Aufwand. Hier liegt einer der wesentlichen Kritikpunkte an der Einführung des ISDN-Netzes in der Bundesrepublik Deutschland [52][107][120][78][80][129]. Der Anrufer kann einen Einzelgebühren-Nachweis verlangen, was die Speicherung der Verbindungsdaten zur Folge hat; der Angerufene wird davon nicht einmal unterrichtet. „Bei Ärzten, Anwälten und Beratungsstellen könnte die neue Technik das Vertrauensverhältnis zu Patienten und Kunden jedenfalls empfindlich stören.“[80]

Zu 4: Die Verletzlichkeit der Technik äußert sich vergleichsweise harmlos bei Stromausfall an der Kaufhauskasse. Kritischer ist möglicherweise Sabotage von Leitungen, das ziemlich einfache Abhören öffentlicher Netze oder ein Fehlalarm in einem militärischen System. Viele Unternehmen könnten den Ausfall ihres Rechenzentrums um höchstens 2 bis 5 Tage überleben. Das Lahmlegen eines Computers kann verursachen, daß wichtige Termine versäumt werden oder kritische Steuerungsfunktionen im entscheidenden Augenblick versagen, etwa an einem Krankenbett in der Intensivstation. Im Risks-Digest 9.45 wurde von einem Software-Fehler in einem Bestrahlungsgerät berichtet, der mindestens vier Menschenleben kostete.

Zu 5: Zugangskontrollen über persönliche Merkmale wie Fingerabdrücke oder Netzhautbilder („Eye-Dentify“) erfordern die Speicherung dieser Merkmale, die wiederum medizinische Diagnosen gestattet oder nur unbefriedigend von polizeilichen Ermittlungsmethoden abgegrenzt ist. Außerdem verleitet diese Art von Kontrolle vielleicht zu Gewaltmaßnahmen – der Fingerabdruck ist auch auf einem abgehackten Finger noch perfekt zu erkennen. Das informationelle Selbstbestimmungsrecht ist mit solchen Sicherheitsmaßnahmen nicht in Einklang zu bringen.

Anlagen zur elektronischen Datenverarbeitung werden immer sensibler und erfordern daher immer strengere Sicherheitsmaßnahmen, da sie lebensnotwendige soziale Funktionen ausüben. Dadurch entsteht eine Eigendynamik, die weiter in die Richtung des totalen Überwachungsstaats führt. Das Personal solcher Anlagen, und das ist ein immer größerer Anteil der Bevölkerung, muß sorgfältig ausgesiebt werden [119]:

- Prüfung der politischen Einstellung bis hin zum faktischen Berufsverbot. (Gehört er dieser oder jener Partei an? Demonstriert er?)
- Kontrolle des Lebensstils. (Trinkt er? Hat er Schulden?)
- Überprüfung persönlicher Kontakte. (Hat er dubiose Bekannte? Reist er ins Ausland?)
- Ausleuchtung der Vergangenheit. (Ist er erpreßbar? Ist er als ehemaliger Hacker als zuverlässig einzustufen? Hat er sich als Student politisch betätigt?)

Wegen solcher Ausforschungsmaßnahmen ist möglicherweise bei Projekten der Informations- und Kommunikationstechnik in Zukunft mit ähnlichen gesellschaftlichen Widerständen zu rechnen wie bei anderen technischen Großprojekten.

Zu 6: Die große Komplexität der Softwaresysteme, etwa bei militärischen Anwendungen, bringt ebenfalls eine sehr große Gefahr der Sabotage mit sich. Schließlich gibt es ja Viren, die lange Zeit unauffällig in großen Systemen schlummern und nur im Ernstfall ihre Wirkung entfalten. „Was ist mit den Tausenden von Menschen, die am Aufbau und der Wartung solcher Systeme beteiligt sind? Wie stellen wir sicher, daß wir ihnen vertrauen können, daß sich nicht ‘zwanghafte Programmierer’ mit einem Todestrieb in diesen Systemen verewigen? Welche polizeistaatlichen Maßnahmen sind erforderlich, um ein solches komplexes und verteiltes System beispielsweise vor Hackern zu schützen?“ [60]

Zu 7: Auch Betrug und Spionage lassen sich mit Computerhilfe rationalisieren. Die Einführung des elektronischen Zahlungsverkehrs mit Plastikgeld und Kreditkarten eröffnet für kreative Kriminelle vielfältige Möglichkeiten [79]. Fälschungen sind vielleicht nicht ganz leicht, aber lohnend. Ausgespähte Paßwörter und Geheimnummern versprechen reichlichen Gewinn, Operation aus dem Ausland sichert vor Festnahme.

Eine Schätzung von 1980 besagt, daß nur 1% aller Fälle von Computerkriminalität überhaupt entdeckt werden, davon nur 7% angezeigt werden, und davon wieder nur 3% zu einer Verurteilung des Täters führen. Vielleicht liegen diese Zahlen heute höher, nachdem die Verantwortlichen der Datensicherheit mehr Gewicht beimessen und die Gesetzgebung verbessert ist. Beim Bundeskriminalamt wurden von 1980 bis 1984 insgesamt 53 Fälle, für 1988 allein 3355 Fälle registriert [44], darunter 2777 Fälle von Betrug, 169 von Spionage, 72 von Sabotage und 49 Hacker-Fälle [46]. Auffallend ist, daß ausgerechnet aus Banken fast keine Fälle bekannt werden.

Zu 8: Der Datenschutz kollidiert sehr oft mit anderen berechtigten Interessen, wie etwa dem Verbraucherschutz oder der wissenschaftlichen Forschung, die ja zumindest im Hochschulbereich der gesamten Gesellschaft dient. Besonders betroffen ist zum Beispiel die epidemiologische Forschung, die auf der Suche nach Krankheitsursachen möglichst vollständige Daten über das geographische und soziale Umfeld der Patienten braucht. Auch in Betrieben entstehen Interessenkonflikte zwischen verschiedenen Personengruppen wie etwa Management, EDV- bzw. Rechenzentrumsleiter, Sachbearbeitern, Betriebs- oder Personalrat, Kontrollorganen, und nach außen hin zum Gesetzgeber und zu Herstellern.

Auf der anderen Seite schützt der Datenschutz auch kriminelle Aktionen und erschwert deren Aufdeckung. („Der Datenschutz schützt die Ganoven.“) Hier ist das Interesse des Staates und der Gesellschaft an wirksamer Verbrechensbekämpfung gegen die Einschränkung der Rechte unbescholtener Bürger abzuwägen. Die Polizei darf ja auch nicht in die Menge schießen, nur weil sich einige Verbrecher unter sie gemischt haben.

Aber auch verschiedene Datenschutzinteressen können miteinander kollidieren, etwa bei Staat und Bürgern, bei Industriebetrieben und Umweltschutz, bei Prominenten und Presse. Es gibt schließlich in einem demokratischen Staat auch ein Recht auf Zugang zu Information, etwa zu Umweltdaten [140]. Es ist zu befürchten, daß im Kollisionsfall kein Konsens herbeigeführt wird, sondern sich der Mächtigere einfach durchsetzt.

Zu 9: Gerade Wissenschaftsnetze sind meist sehr offen konzipiert, um einen schnellen Gedankenaustausch rund um die Welt zu ermöglichen. In ein solches Netz unbefugt einzudringen, ist etwa so originell, wie nachts in den Zeitschriftenlesesaal einer Bibliothek einzubrechen, um dort die neuesten Fachartikel zu lesen. Andererseits sollten Texte und Dokumente, die nicht für fremde Augen bestimmt sind, tabu sein. Wenn mein Nachbar sein Haus nicht abschließt, habe ich noch lange nicht das Recht, es zu durchsuchen. Ein anderes analoges Beispiel ist das Briefgeheimnis.

Ein besonders kritischer Punkt ist der Schutz der Privatsphäre. Dazu hat das amerikanische Ministerium für Gesundheit, Erziehung und Wohlfahrt schon 1973 Richtlinien vorgeschlagen (hier zitiert nach [136]):

- Es darf keine Informationssysteme zur Bearbeitung personenbezogener Daten geben, deren Existenz geheim ist.
- Jede Person muß die Möglichkeit haben herauszufinden, was über sie gespeichert ist und wie diese Daten verwendet werden.
- Jede Person muß die Möglichkeit haben zu verhindern, daß Daten über sie selbst, die sie zu einem bestimmten Zweck gegeben hat, für andere Zwecke gebraucht werden.
- Jede Person muß die Möglichkeit haben, falsche Information, die so abgespeichert ist, daß sie als Person anhand dieser Information identifiziert werden kann, korrigieren oder verbessern zu lassen.
- Jede Organisation, die personenbezogene Informationen, deren Bezugspersonen anhand dieser Informationen identifizierbar sind, erzeugt, bearbeitet, benutzt oder weiterverbreitet, muß die Zuverlässigkeit dieser Daten für den beabsichtigten Zweck sicherstellen, und sie muß geeignete Vorsichtsmaßnahmen ergreifen, um Mißbrauch dieser Daten zu verhindern.

In diesen Zusammenhang gehört auch die Diskussion in [25], [110] und [70].

2.3 Gesetze

In den letzten Jahren wurde eine fast unübersichtliche Fülle von Gesetzen geschaffen, die den Datenschutz dennoch nur unvollkommen regeln und kaum noch für Experten durchschaubar sind. Hier ist eine Auswahl von Gesetzen, die den Datenschutz betreffen:

- Bundesdatenschutzgesetz (BDSG).
- Landesdatenschutzgesetz.
- Bundesstatistikgesetz.
- Landesstatistikgesetz.
- Hochschulstatistikgesetz.
- Meldegesetz.
- Bundesverpflichtungsgesetz.
- Fernmeldeanlagengesetz.
- Urheberrechtsgesetz.
- Strafgesetzbuch.

Für die medizinische Forschung zum Beispiel kommen noch dazu:

- Landeskrankenhausgesetz.
- Heilberufegesetz.
- Gesundheitsreformgesetz

Im Bereich der Wirtschaft sind unter anderem zu beachten

- die Grundsätze ordnungsgemäßer Speicherbuchführung, die das Bundesfinanzministerium 1978 herausgegeben hat,
- das Warenzeichengesetz,
- das Gesetz über den unlauteren Wettbewerb.

Ferner gibt es bereits Dutzende von einschlägigen OLG-Entscheidungen. Diese Gesetze werden in diesem Buch nur sehr oberflächlich oder (meistens) gar nicht behandelt. Als Kuriosität sei noch vermerkt, daß selbst noch die Bestattungsgesetze Datenschutzregelungen enthalten, indem sie den Zugriff auf den personenbezogenen Teil der Todesbescheinigungen verhindern.

Das Bundesdatenschutzgesetz erfaßt grundsätzlich (nur) personenbezogene Daten. Es wird zur Zeit neu gefaßt. Dabei soll auch das Urteil des Bundesverfassungsgerichts zur Volkszählung berücksichtigt werden. Dieses stellte fest, daß das allgemeine Persönlichkeitsrecht auch den Schutz des Einzelnen gegen unbegrenzte Erhebung, Speicherung, Verwendung und Weitergabe seiner persönlichen Daten umfaßt. Paragraph 1, Absatz 1 des Bundesdatenschutzgesetzes (Entwurf von 1988) heißt:

> „Zweck dieses Gesetzes ist es, den einzelnen davor zu schützen, daß er durch die Verarbeitung oder Nutzung seiner personenbezogenen Daten in oder unmittelbar aus Dateien in seinem Persönlichkeitsrecht beeinträchtigt wird."

Die Verarbeitung ist grundsätzlich unzulässig, außer

- sie wird durch eine Rechtsvorschrift geregelt *oder*
- es liegt eine Einwilligung des Betroffenen vor (in der Regel schriftlich).

Weiter heißt es in §5 (Datengeheimnis):

> „Den bei der Datenverarbeitung beschäftigten Personen ist untersagt, personenbezogene Daten unbefugt zu verarbeiten oder zu nutzen (Datengeheimnis). Diese Personen sind bei der Aufnahme ihrer Tätigkeit hierauf zu verpflichten. Das Datengeheimnis besteht auch nach Beendigung ihrer Tätigkeit fort."

Unterschieden wird zwischen der Datenverarbeitung durch

- Behörden und sonstigen öffentlichen Stellen,
- Gesellschaften und anderen Personenvereinigungen des privaten Rechts.

Nach §29, Absatz 1, sind Dateien mit personenbezogenen Daten beim Datenschutzbeauftragten anzumelden. §36 regelt die Verarbeitung und Nutzung personenbezogener Daten für die wissenschaftliche Forschung, §37 die Verarbeitung und Nutzung personenbezogener Daten durch die Medien. In der Anlage zu §8, Satz 1, sind die „zehn Gebote des Datenschutzes“ formuliert:

> „Werden personenbezogene Daten automatisiert verarbeitet, sind Maßnahmen zu treffen, die je nach der Art der zu schützenden personenbezogenen Daten geeignet sind,
>
> 1. Unbefugten den Zugang zu Datenverarbeitungsanlagen, mit denen personenbezogene Daten verarbeitet werden, zu verwehren (**Zugangskontrolle**),
> 2. zu verhindern, daß Datenträger unbefugt gelesen, kopiert, verändert oder entfernt werden können (**Datenträgerkontrolle**),
> 3. die unbefugte Eingabe in den Speicher sowie die unbefugte Kenntnisnahme, Veränderung oder Löschung gespeicherter personenbezogener Daten zu verhindern (**Speicherkontrolle**),
> 4. zu verhindern, daß Datenverarbeitungssysteme mit Hilfe von Einrichtungen zur Datenübertragung von Unbefugten genutzt werden können (**Benutzerkontrolle**),
> 5. zu gewährleisten, daß die zur Benutzung eines Datenverarbeitungssystems Berechtigten ausschließlich auf die ihrer Zugriffsberechtigung unterliegenden Daten zugreifen können (**Zugriffskontrolle**),
> 6. zu gewährleisten, daß überprüft und festgestellt werden kann, an welche Stellen personenbezogene Daten durch Einrichtungen zur Datenübertragung übermittelt werden können (**Übermittlungskontrolle**),
> 7. zu gewährleisten, daß nachträglich überprüft und festgestellt werden kann, welche personenbezogenen Daten zu welcher Zeit von wem in Datenverarbeitungssysteme eingegeben worden sind (**Eingabekontrolle**),
> 8. zu gewährleisten, daß personenbezogene Daten die im Auftrag verarbeitet werden, nur entsprechend den Weisungen des Auftraggebers verarbeitet werden können (**Auftragskontrolle**),

9. zu verhindern, daß bei der Übertragung personenbezogener Daten sowie beim Transport von Datenträgern die Daten unbefugt gelesen, kopiert, verändert oder gelöscht werden können (**Transportkontrolle**),
10. die innerbehördliche oder innerbetriebliche Organisation so zu gestalten, daß sie den besonderen Anforderungen des Datenschutzes gerecht wird (**Organisationskontrolle**).“

Obwohl das Datenschutzgesetz nur für personenbezogene Daten direkt zutrifft, hat es Auswirkungen auf das *gesamte* System, in dem *auch* personenbezogene Daten verarbeitet werden, denn auch Anwendungsprogramme und Systemsicherheit müssen berücksichtigt werden. Die zehn Gebote sind auch als grobe Checkliste für die Sicherheit anderer Daten geeignet, die aus welchen Gründen auch immer von ihrem Besitzer als schutzwürdig betrachtet werden. Eine Checkliste für die Zulässigkeit der Datenübermittlung ist in [33, S. 24] abgedruckt.

Mit dem „Zweiten Gesetz zur Bekämpfung der Wirtschaftskriminalität“ wurden einige neue Paragraphen ins Strafgesetzbuch aufgenommen, so zum Beispiel der Paragraph 202a:

> „Wer unbefugt Daten, die nicht für ihn bestimmt und gegen unberechtigten Zugang besonders gesichert sind, sich oder einem anderen verschafft, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.“

Unter Strafe gestellt sind seither in der Bundesrepublik Deutschland:

Ausspähen von Daten (§202a StGB): Softwarediebstahl, Ausspähen von Daten, Wirtschaftsverrat, Verschaffen von Unternehmensgeheimnissen.

Computerbetrug (§263a StGB): Jeder Eingriff in einen Datenverarbeitungsvorgang, der Vermögensschäden verursacht. Darunter fallen etwa Kontenmanipulationen in Bankcomputern oder das Erschwindeln von Sozialleistungen.

Fälschung beweiserheblicher Daten (§269 StGB): Veränderung von Urkunden, die in Rechenanlagen gespeichert sind („elektronische Urkundenfälschung“), zum Beispiel bei elektronischer Buchhaltung.

Datenveränderung (§303a StGB): Veränderung oder Vernichtung von Daten, auch durch Viren.

Computersabotage (§303b StGB): Anschläge auf die Datenverarbeitung durch Veränderung oder Vernichtung von Computerdaten, Datenträgern oder Anlagen.

Die Weitergabe von statistischen Daten wird vom Bundesstatistikgesetz (BStatG) geregelt. In §11 werden die Vorschriften nach dem Grad der Anonymisierung unterschieden. So lautet etwa §11, Absatz 5:

> „Einzelangaben, die so anonymisiert werden, daß sie Auskunftspflichtigen oder Betroffenen nicht mehr zuzuordnen sind, dürfen vom Statistischen Bundesamt oder von den Statistischen Landesämtern übermittelt werden."

Wie wirksam diese Vorschrift ist, wird in Kapitel III. 2 behandelt.

3 Subjekte und Objekte des Datenschutzes

Als erster Schritt bei der Planung von Datenschutzmaßnahmen, aber auch ständig im bereits laufenden Betrieb, ist festzustellen, welche Objekte, Daten und Informationen Datenschutzgesetzen oder -vorschriften direkt oder indirekt unterliegen. Worauf muß sich der Schutz erstrecken? Welche Angreifer sind denkbar? Beim Datenschutz wie auch anderswo gilt, daß man sich nur gegen bekannte oder erkannte Gefahren wappnen kann.

Weck [136] gliedert diese Überlegungen in die folgenden abstrakten Grundbegriffe:

Verwundbarkeit – Eigentum, etwa an Datenverarbeitungsanlagen, aber auch an Daten und Informationen, macht verwundbar. Daher sind die Schutzobjekte zu spezifizieren.

Bedrohung – Wer verwundbar ist, ist in seiner Sicherheit bedroht; er ist sogar gefährdet, wenn es konkrete Gefahren gibt. Zu spezifizieren sind Angreifer und Motive.

Schutz – Die Verwundbarkeit ist gegen Bedrohung und Gefährdung zu schützen. Die Spezifikation der Schutzmaßnahmen ist unser Thema. Welche verwundbare Stelle wird vor welcher Bedrohung durch welche Maßnahme geschützt?

Ein **Angriff** ist ein nicht erlaubter Zugriff auf das System. Die Schutzmaßnahmen erstrecken sich auf das Erkennen von Angriffen, das Verhindern von Angriffen oder die Begrenzung des Schadens, den Angriffe bewirken können. **Vertrauliche Daten** sind alle Daten, die aus irgendeinem Grunde nicht für jedermann zugänglich sein sollen, sei es auf Grund von Rechtsvorschriften oder einfach, weil ihr Besitzer es so will. Im Zweifelsfall, oder wenn nicht ausdrücklich etwas anderes festgelegt ist, sollten Daten stets als vertraulich behandelt werden — das entspricht dem Prinzip vom geschlossenen System.

3.1 Angreifer und Motive

Ein unerlaubtes Interesse an vertraulichen Daten können haben:

- Zugelassene Benutzer („Insider“).
- Hacker.
- Spione.

Die Insider können reine Anwender ohne besondere Systemkenntnisse, Programmierer mit guten Systemkenntnissen und Systemprogrammierer mit sehr genauen Systemkenntnissen und besonderen Privilegien sein (**Ausnützen von Insider-Wissen**). Neben versehentlichen Einblicken in vertrauliche Daten durch Bedienungs- oder Systemfehler kommen als Motive Neugier, Schabernack, Rache und Habgier oder finanzielle Probleme in Betracht. Weitaus die meisten Verstöße gegen die Datensicherheit werden von Insidern begangen; Schätzungen schwanken zwischen 80% und 98%. Sie sind ungenau, weil hier vermutlich die Dunkelziffer sehr hoch ist. Viele Unternehmen sind völlig abhängig von ihren Systemspezialisten und dadurch von diesen erpreßbar. Allerdings darf man nicht übersehen, daß in den meisten Fällen (nach Schätzungen mindestens zwei Drittel) Daten nicht mit krimineller Absicht vernichtet, unbefugt eingesehen oder weitergeleitet werden, sondern durch Unachtsamkeit, Fahrlässigkeit oder schiere Unwissenheit der berechtigten Anwender.

Hacker sind Betriebsfremde, die sich ins System **einschleichen**, um ihre Fähigkeiten zu beweisen, eventuell um Rechenzeit zu stehlen. Sie haben unterschiedlich gute Systemkenntnisse, meistens aber Kenntnisse von Sicherheitslücken. Sicherheitslücken sind eine Herausforderung für Tüftler. Häufig ist die **Maskerade** (oder Identitätstäuschung) – der Hacker nützt ungenügende Identifikationsprozeduren aus, um sich als berechtigter Benutzer auszuweisen. Der Übergang vom Hacker zum Spion ist fließend; dieser legt es darauf an, persönliche Daten auszuspähen oder Projekte auszuspionieren oder zu sabotieren, in der Regel für fremde Auftraggeber. Man kann in der Regel sehr gute Systemkenntnisse unterstellen. Die Auftraggeber können politische, wirtschaftliche oder private Motive haben. Gewaltanwendung ist nicht ausgeschlossen. Hacker werden in der Regel als Amateure agieren; man kann davon ausgehen, daß sie keine Ausrüstung zur Verfügung haben, die 10000 DM oder mehr kostet, und daß ihre intellektuelle Leistungsfähigkeit nicht über dem Niveau eines Universitätsdiploms liegt. Im Falle der Spionage muß man mit professionellem Vorgehen rechnen, also eventuell mit besserer Ausstattung, aber auch mit ausgeprägtem Kosten-Nutzen-Denken.

Ein besonders tückischer Angriff ist der **Fischzug**. Hier hat der Angreifer es nicht auf ganz bestimmte Daten abgesehen, sondern er sucht irgendwelche Lücken im Datenschutz und was immer er an Daten bekommen kann. Hacker gehen oft nach diesem Muster vor. Denkbar ist aber auch ein findiger Journalist, der die Datenschutzmaßnahmen eines Betriebs oder einer Behörde diskreditieren

will. Auch die **Müllverwertung**, eine Variante des Fischzugs, ist als Angriffsmethode zu beachten (Papiermüll oder elektronischer Müll im Speicher oder auf Datenträgern). Über dieses Thema berichtete das ZDF am 10.7.1990: „Fundort Mülltonne".

Am gefährlichsten ist der **technologische Angriff**, bei dem Kenntnisse von Schwächen oder Fehlern eines Systems ausgenutzt werden. Er kann von Insidern oder von Außenstehenden kommen, wobei Insider natürlich viel bessere Gelegenheit und in der Regel auch tiefere Kenntnisse besitzen.

Generell müssen drei Voraussetzungen für einen Angriff auf die Datensicherheit, wie bei anderen Vergehen auch, zusammenkommen:

Bedarf – Der Täter versucht, sich einen Vorteil zu verschaffen.

Gelegenheit – Der Täter schätzt die Tat als ausführbar ein.

Einstellung – Der Täter hat keine moralischen Hemmungen, die Tat zu begehen.

Die Zusammenstellung der Anteile verschiedener Vergehen in Tabelle I-1 stammt ebenfalls von 1980 [136]; es ist klar, daß diese Zahlen wegen der hohen Dunkelziffer nicht exakt sein können.

Wer Daten zu schützen hat, muß sich über mögliche Angreifer und deren Motive Gedanken machen. Systeme sind so zu konstruieren, daß sie einen Mißbrauchsversuch möglichst abweisen – Vorschriften nützen nichts ohne Maßnahmen zu ihrer Durchsetzung. Verhindern ist sicherer als Verbieten.

3.2 Schutzobjekte

Zu schützen sind:

- Personenbezogene Daten vor versehentlichen Einblicken, Ausspähung und Manipulation (Datenschutz im Sinne der Datenschutzgesetzgebung).
- Private Daten (Schutz der Privatsphäre),
  - elektronische Korrespondenz vor Neugier,
  - Dokumente, Entwürfe, Gutachten, kurz alles, was der Besitzer der Daten gern für sich behalten möchte, vor unkontrolliertem Einblick,
- Betriebs- oder Geschäftsgeheimnisse vor der Konkurrenz:
  - Pläne,
  - Kosten und Ertragslage,
  - Probleme,

Tabelle I-1: Anteile an der Computerkriminalität

Manipulation von Eingabedaten:		
1.	Hinzufügen betrügerischer Daten	21%
2.	Verändern gültiger Daten	5%
3.	Entfernen gültiger Daten	3%
Manipulation von Programmen:		
4.	Falsche Duchführung einer Transaktion	4%
5.	Erzeugung gefälschter Daten	3%
6.	Stehlen vieler kleiner Beträge	3%
7.	Unterdrückung oder Änderung von Ausgaben	3%
8.	Hinzufügen oder Ändern von Datensätzen	1%
9.	Umgehen interner Prüfungen	2%
10.	Hinzufügen gewollter Fehler	2%
11.	Teilweise Durchführung einer Transaktion	$< 0.5\%$
Manipulation von Ausgaben:		
12.	Zerstörung gedruckter Ausgaben	3%
13.	Diebstahl gedruckter Ausgaben	2%
14.	Unterschieben gefälschter Ausgaben	2%
Manipulation von Stammdaten:		
15.	Diebstahl von Stammdaten	16%
16.	Zerstörung der Stammdatei	4%
17.	Temporäre Manipulation von Stammdaten	$< 0.5\%$
18.	„Kidnapping“ der Stammdatei	2%
Sonstiges:		
19.	Ausnützen von Systemfehlern	1%
20.	Physische Sabotage der Geräte	15%
21.	Diebstahl von Rechenleistung	10%

- Programme (Urheberschutz),
  - Schutz kommerzieller Software vor Raubkopie, wozu der Systembetreiber durch gesetzliche und vertragliche Regelungen verpflichtet ist,
  - Schutz eigener Software-Entwicklungen vor nichtautorisierter Weitergabe.
- Betriebssystem (Systemintegrität),
  - Schutz vor Manipulationen am System,
  - Schutz vor unbefugten Autorisierungsänderungen,
  - Schutz vor Unfug, vor Viren,
  - Schutz vor Sabotage von Projekten.
- DV-Leistung (Systemverfügbarkeit),
  - Schutz des Prozessors vor Blockade durch unnütze Arbeit,
  - Schutz der Massenspeicher vor Blockade durch unnütze Daten,
  - Schutz der Datennetze vor Blockade durch unnütze Datenübertragung,
  - Schutz des Bedienungspersonals vor unnützer Arbeit und Streß.

Die zu schützenden Daten können sich auf

- Bildschirmen,
- Arbeitsplatzrechnern,
- Kommunikationsleitungen,
- Disketten,
- Platten,
- Bändern und Kassetten, auch Sicherungs- und Archivbändern,
- Druckerlisten,
- und mitsamt den Datenträgern auf Müllkippen

befinden. Unter Datenträgern im eigentlichen Sinne versteht man Festplatten, Disketten, Bänder, Kassetten und Druckerlisten (Lochkarten und -streifen haben ausgedient). Die notwendigen oder möglichen Schutzmaßnahmen sind Gegenstand des nächsten Kapitels.

3.3 Angriffe und Schäden

Drei Typen des Angriffs lassen sich grob unterscheiden:

- abhören (passiv),
- fälschen (aktiv),
- zerstören.

Die Angriffstaktiken

- Ausnützen von Insiderwissen,
- Einschleichen,
- Maskerade,
- Fischzug,
- Müllverwertung,
- technologischer Angriff

wurden schon erwähnt. Wirtschaftliche Schäden können entstehen durch

- Schäden an Geräten,
- Unterbrechung des EDV-Betriebs,
- Auftreten von Fehlern im System mit zeitraubender Fehlersuche und Reparatur,
- Weitergabe sensitiver Daten,
- Diebstahl von Geld oder Wirtschaftsgütern durch Manipulation,
- Diebstahl von EDV-Ressourcen, zum Beispiel Zugang zu weiteren Netzen auf Kosten des Gastgebers,
- öffentliche Diskreditierung des Betriebs durch findige Journalisten.

Alle diese Schadensmöglichkeiten und ihre Kosten sind gegen die Kosten von Schutzmaßnahmen abzuwägen nach dem Prinzip der Verhältnismäßigkeit. Oft ist es einfacher und letzlich auch billiger, alle Daten zu schützen, als eine komplizierte Auswahl. Das Verhindern mancher Angriffe ist zu teuer oder vielleicht sogar unmöglich. In solchen Fällen kann aber oft eine geeignete billigere Abschreckungsmaßnahme die Hemmschwelle erhöhen; insbesondere für die Abwehr von Gelegenheitstätern kann das ausreichend sein. Solche Abwägungen gehören in das Datensicherheitskonzept des Betriebs, das Teil des allgemeinen EDV-Konzepts ist. Weitere Gesichtspunkte zu Kosten-Nutzen-Abwägungen finden sich in Kapitel II. 1.6.

Kapitel II

Sicherheit beim Rechnerbetrieb

Im ersten Kapitel wurden das gesellschaftliche und politische Umfeld des Datenschutzes und die Probleme der Datensicherheit aus allgemeiner Sicht behandelt. Jetzt geht es um die Sicht dessen, der für den Betrieb eines Rechnersystems verantwortlich ist. Es werden konkrete Überlegungen angeregt, Maßnahmen aufgezählt und Check-Listen aufgestellt, wobei einige Punkte in späteren Kapiteln noch im Detail behandelt werden. Also: Welche Überlegungen muß der Betreiber anstellen, um seiner Verantwortung gerecht zu werden? Was ist der Stand der Technik? Für die Datensicherheit gibt es drei Stufen, die man alle kennen sollte:

- die durchschnittliche Realität,
- den Stand der Technik,
- den Stand der Wissenschaft (das theoretisch erreichbare Wunschziel).

Das Problem der Datensicherheit ist so komplex, daß man mit improvisierten, unsystematischen Maßnahmen nicht zu einer ausreichenden Lösung kommt. Konzeptionslosigkeit führt zu Lücken und unbemerkten Gefährdungen. Wirksame Datensicherheit kann nur auf der Grundlage eines umfassenden Konzepts erreicht werden, das systematisch alle möglichen Schwachstellen erfaßt. Wie ein solches Konzept zu gestalten ist, wird in diesem grundlegenden, aber zugegebenermaßen sehr trockenen Kapitel ausgeführt.

1 Umgebungsbedingungen

Jedes Datenverarbeitungssystem ist in eine Umgebung eingebettet, die geprägt wird durch die Art des Betriebs, etwa Firma oder Behörde, und der

Aufgaben der Datenverarbeitung, durch die baulichen Gegebenheiten und Möglichkeiten, auch im geographischen Umfeld, und durch die Art der Benutzer und des Personals. Das muß berücksichtigt werden, wenn man konkrete Maßnahmen plant. Die Sicherheitsmaßnahmen für ein Institutsnetz in einer Universität werden sich wesentlich von denen in einem Bundeswehr-Rechenzentrum unterscheiden. Auch die Größe des Betriebs ist mit ausschlaggebend für den Formalisierungsgrad der Regelungen.

Die Umgebung des Datenverarbeitungssystems muß so gestaltet werden, daß die physische und logische Unversehrtheit des Systems wirksam geschützt werden kann und die Spezifikation der Sicherheitsmaßnahmen nicht durch unkontrollierte Eingriffe ungültig gemacht werden kann. Jedes System ist höchstens so sicher wie sein Umfeld.

Das Sicherheitskonzept ist Teil des gesamten EDV-Konzepts des Betriebs, und dieses besteht in der ersten Stufe aus

- Datenmodell,
- Benutzerspezifikation,
- Systemspezifikation,
- Entwicklungsperspektiven.

Ohne ein brauchbares Gesamt-EDV-Konzept sind die Sicherheitsprobleme nicht in den Griff zu bekommen; umgekehrt erfordern die bestehenden Gefahren und Probleme, daß das EDV-Konzept ein ausführliches Sicherheitskonzept enthält. Zwei Praxisbeispiele für ein solches Sicherheitskonzept sind in [33, Kapitel 6] zu finden.

1.1 Organisation des Rechnerbetriebs

Hier ist vor allem das Gebot „Organisationskontrolle“ des Bundesdatenschutzgesetzes zu erfüllen; andere Gebote werden natürlich auch tangiert. Die drei wesentlichen Punkte sind *Gerätebeschaffung*, *räumliche Organisation* und *personelle Organisation*. („Was kaufen wir, wo stellen wir es hin, und wer bedient es?“) Bei der Gerätebeschaffung ist über die Art der Rechner und Netze auch unter Sicherheitsaspekten zu entscheiden. Zur räumlichen Organisation gehören die Überlegungen:

- Baupläne und Pläne der Systemkonfiguration,
- Sicherheitsbereiche und Zugangswege,
- Aufstellung von Geräten, insbesondere Rechnern, Netzservern und Peripheriegeräten,
- Kabelwege, Verteilerschränke, Anschlußpunkte,

- Doppelböden, abgehängte Decken,
- Lagerung von Datenträgern,
- Ausgabe von Datenträgern, etwa Druckerlisten,
- Erfordernisse zum Transport von Datenträgern,
- Organisation des Transports von Datenträgern.

Bei der personellen Organisation ist zu überlegen:

- Aufgabenverteilung und Funktionstrennung:
  - EDV-Leitung, Rechenzentrumsleitung,
  - Systemverwaltung (Benutzerverwaltung, Ressourcenverwaltung, Betriebsleitung),
  - Systemprogrammierer,
  - Operatoren (für Rechner, Netz, Bänder, Drucker),
  - Techniker,
  - Reinigungspersonal,
  - Anwendungsprogrammierer,
  - Endbenutzer(-gruppen),
  - Revisoren, Kontrolleure.

  Es müssen alle Verantwortlichkeiten und Berechtigungen klar definiert und nachprüfbar sein.
- Arbeitsplatzbeschreibungen,
- Richtlinien und Arbeitsanweisungen, Standards und Normen,
- Organisation von Benutzergruppen, Projektleiter,
- Ordnung im Geräteraum, im Bandarchiv, ...,
- ordentliche Dokumentation,
- Schlüsselregelung,
- Richtlinien und Zuständigkeiten für Notfälle, Krisenstab.

Erhöhte Aufmerksamkeit ist besonderen Betriebszuständen zu widmen, in denen notwendigerweise Sicherheitssperren abgeschaltet und fremde Personen anwesend sind – das kann bei Wartungsarbeiten oder in Notfällen zutreffen.

Auch sollte in Notfällen nicht durch unbedachtes Handeln inkompetenter Personen der Schaden vergrößert werden (indem etwa der Sicherheitskopie einer zerstörten Datei auch noch der Garaus gemacht wird).

Das Vieraugenprinzip läßt sich durch eine sinnvolle Aufteilung der Verantwortung approximieren, indem etwa Systemanalyse von Systemprogrammierung getrennt wird, und ebenso Entwicklung von Wartung und von Betrieb, insbesondere von der Bedienung der Anlage durch einen Operator bei jeweils gegenseitiger Kontrolle durch die jeweiligen Funktionsträger. Selbstverständlich ist das nur bei einer genügenden Personalausstattung zu verwirklichen. Bei kleinen Betrieben ist es oft besser, auf Vertrauen zu setzen, oder man beschäftigt mehrere Bedienstete „nebenamtlich“ mit Systemaufgaben.

Die bedeutendsten sicherheitsrelevanten Aufgabenbereiche in der Betriebsabteilung sind:

Systemverwalter – Er (oder sie oder die Gesamtheit der Personen, die diese Funktion ausüben) ist quasi allmächtig. Ihm gehören alle System-Definitionstabellen, alle Sicherungsbänder usw. Er hat auch im Betriebssystem besondere Privilegien (‘Super User’). Er definiert alle Sicherheitsmaßnahmen und kann sie nach Bedarf außer Kraft setzen – und muß das auch können; anders ist kein Rechenbetrieb möglich. Er ist verantwortlich für den ordnungsgemäßen Betriebsablauf und muß im Störungsfalle geeignete Maßnahmen zur Fehlerbehebung treffen können. Als Verwalter der Sicherungsbänder hat er auch ohne sonstige Privilegien Zugriff auf das gesamte System: Er definiert und startet Sicherungsläufe und darf alle Daten für alle Benutzer restaurieren (sonst ist die Datensicherung kaum sinnvoll zu betreiben). Damit kann er auch beliebige Dateien in die Sicherung einbringen und dann an beliebige Stellen zurückspielen und hat so beliebige Manipulationsmöglichkeiten.

Operator – Er startet und stoppt das System und dessen Subsysteme, manipuliert Warteschlangen, ordnet Peripheriegeräte, etwa Bänder und Laufwerke, den Benutzern zu, verteilt Druckerlisten .... Ein aus dem Leben gegriffenes Beispiel, welche Probleme bei der Zuordnung von Bandlaufwerken entstehen können: Der Operator erhält von XYZ einen Anruf mit der Bitte, das Band ABC aufzulegen. Er ordnet XYZ das Bandlaufwerk A zu. Gleichzeitig ruft der Systemverwalter an, und will eine Restaurierung aus der letzten Sicherung vornehmen. Der Operator ist unter Streß und legt die Bänder jeweils auf das falsche Laufwerk. XYZ hat unvermutet Zugriff auf Dateien, die er schon immer mal sehen wollte.

1.2 Benutzerberechtigungen

Einige hierher passende Gebote des Datenschutzgesetzes sind:

- Zugangskontrolle:

  - Festlegung befugter Personen,
  - Berechtigungsausweise,
  - Vieraugenprinzip,
  - Regelung für Fremde, Besucherbuch,
- Anwendungskontrolle:
  - Zuordnung zwischen Benutzergruppen und Anwendungen,
  - Verantwortung von Projektleitern,
  - Verfahrensdokumentation bei kritischen Anwendungen,
  - Programmier-Regeln für kritische Anwendungen,
  - Prüfregeln für kritische Anwendungen.
- Auftragskontrolle.

Frei nach Weck [136] muß man dazu folgende Fragen beantworten:

- Wer darf mit dem System arbeiten?
- Was darf mit den Informationen und Daten gemacht werden oder nicht?
- Wer darf bestimmte Informationen lesen oder verändern?
- Warum muß eine bestimmte Operation ausgeführt werden?
- Wann darf eine bestimmte Operation ausgeführt werden?
- Wo darf eine bestimmte Operation ausgeführt werden?
- Wer darf einen Auftrag zu einer bestimmten Operation geben?

Organisatorisch festzulegen ist auch, wer sich hinter einer formalen Benutzerberechtigung verbirgt und wer gegebenenfalls für ihre Verwendung verantwortlich ist. Benutzer kann sein:

- eine eindeutige Person („Klaus Pommerening“),
- ein Stellvertreter („der Sekretär im Auftrag der Chefin“),
- ein Funktionsträger („der diensthabende Operator“),
- eine Rolle („Materialausgabe“).

Das Problem der Mehrfachbenutzung einer Identität kann zwar umgangen werden, indem jede natürliche Person nur unter einer eindeutigen Identifikation Zugang zum Datenverarbeitungssystem bekommt. In der Praxis ist das aber oft zu kompliziert, etwa wenn die gleiche Funktion notwendig auf verschiedene Personen verteilt ist (sonst bräuchte man ständige An- und Abmeldevorgänge

oder viele zusätzliche Terminals); problematisch ist auch stets die Regelung des Mehrfachzugriffs auf einen Datensatz. In solchen Fällen erhalten die Fragen des Wann und Wo besonderes Gewicht.

Weitere Gesichtspunkte zu den Benutzerberechtigungen werden in den Abschnitten 2.2, 4.3, 4.6 und in Kapitel III behandelt.

1.3 Datensicherung

Die Datensicherung dient der Wiederherstellung von Dateien, die durch Katastrophen, Geräteschäden, Sabotage oder (weitaus am häufigsten) durch Bedienungsfehler verloren gegangen sind. Das Standardverfahren in Rechenzentren besteht aus vier Sicherungstypen:

Katastrophensicherung – physikalische Sicherung der Teile des Betriebssystems, die man zur Wiederherstellung der Lauffähigkeit nach einem Totalausfall. Alle übrigen, normal gesicherten Daten können dann anschließend mit Hilfe von Betriebssystemfunktionen zurückkopiert werden.

Langfristiger Zyklus – Grundsicherung aller Dateien etwa monatlich, Aufbewahrungsfrist mindestens ein Jahr.

Kurzfristiger Zyklus – Grundsicherung aller Dateien etwa zum Wochenende, Aufbewahrung mindestens bis zur nächsten Langfristsicherung,

Inkrementelle Sicherung – Sicherung aller Veränderungen seit der letzten Grundsicherung etwa täglich abends, Aufbewahrung mindestens bis zur nächsten Grundsicherung.

Die inkrementelle Sicherung hat den Vorteil guter Effizienz, da gerade bei großen Datenbeständen immer nur vergleichsweise geringe Änderungen auftreten. Ein Nachteil ist, daß man bei der Rekonstruktion verlorener Daten meist mehrere Datenträger verwenden muß, und zwar in der richtigen Reihenfolge. Sinnvoll ist bei der Datensicherung auch eine Unterscheidung zwischen statischen, dynamischen und transienten (temporären) Datenbeständen, die man unterschiedlich behandeln kann. Das Aufheben mehrerer Generationen von Sicherungsdaten empfiehlt sich auch für den Fall einer Virusinfektion.

Nach Möglichkeit sollten bei der Datensicherung stets Zwillingskopien angefertigt werden, von denen ein Exemplar an einem sicheren Ort *außer Haus* (außerhalb des „Schuttkegels“) aufbewahrt wird. Sie müssen dort und auch auf dem Transportweg genau so gut geschützt sein wie die Originaldaten.

Es ist völlig klar, daß ein solches Sicherungskonzept einen großen Aufwand an Arbeitszeit und Datenträgermaterial kostet. Im Rechenzentrum mit Schichtdienst können die Operatoren die Sicherung nebenbei laufen lassen; als Datenträger verwendet man in der Regel Bänder. Kassettensysteme mit Zuführungsschacht oder Bandarchive mit Roboterarm bieten sogar die Möglichkeit, Sicherungsläufe über Nacht ohne menschlichen Eingriff ablaufen zu lassen. Im PC-

oder PC-Netzbereich ist die Datensicherung oft noch ein ungelöstes Problem. Die Sicherung auf Disketten ist zwar billig, aber zeitraubend, und wird daher wegen Zeitdruck oft unterlassen. Bequemer geht es mit Kassettenlaufwerken ('Streamer') oder optischen Platten; bei einer Vernetzung sollte man stets die Möglichkeit einer zentralen Sicherung ins Auge fassen.

1.4 Einstellung der Benutzer

Alle Benutzer des Datenverarbeitungssystems sollten ein ihrer Aufgabe angemessenes Datensicherheitsbewußtsein haben. Die Verantwortung des Managements liegt darin, die Motivation hierzu zu vermitteln. Das beginnt damit, daß das Management selbst die Datensicherheit ernst nimmt und nicht durchgehen läßt, wenn sich Nachlässigkeiten einschleifen, oder daß Mitarbeiter explizit auf den Datenschutz verpflichtet werden. Akzeptanzprobleme bei Mitarbeitern können aus der Unbequemlichkeit von Datensicherheitsmaßnahmen entstehen und führen zum Unterlaufen der Maßnahmen. Hier sind Entscheidungen über die Verhältnismäßigkeit zu treffen. Weck [136] macht das an drei Beispielen deutlich:

- „Wenn der Zugang zum Rechnerraum durch eine Tür erfolgt, die nur mit mehreren Schließvorgängen zu öffnen ist, und wenn häufig Zugang benötigt wird, besteht die Gefahr, daß sehr bald ein Karton in der Türöffnung steht, um die Tür am Zuschnappen zu hindern und damit den unbequemen Öffnungsvorgang zu vermeiden."

- „Wenn (...) eine verdeckte Eingabe von (...) Paßwörtern nur dadurch geschehen kann, daß man von Hand die Helligkeit des Bildschirms herunterdrehen und den Bildschirm nach der Eingabe, ebenfalls von Hand löschen muß, ehe man die Helligkeit wieder aufdrehen darf, so wird es sehr schwer, dem Benutzer eine verdeckte Eingabe nahezubringen."

- „Wenn die Vergabe und die Veränderung von Paßwörtern umständlich ist und eventuell noch die Mitarbeit eines Systemverwalters erfordert, ist zu erwarten, daß einmal vergebene Paßwörter höchst selten verändert werden, was ihre Schutzwirkung praktisch aufhebt."

Die Benutzer dürfen nicht daß Gefühl bekommen, daß die Sicherheitsmaßnahmen von „denen da oben" nur zur Schikane ausgedacht worden sind. Auch sind ihre Persönlichkeitsrechte zu respektieren. Nach Möglichkeit sollen sie unterstützt werden, wenn sie Datensicherheitsmaßnahmen ausführen, etwa durch benutzerfreundliche Identifikationsprozeduren oder durch Automatisierung fehleranfälliger Routine-Tätigkeiten. Die Arbeit sollte so weit wie möglich erleichtert werden, um streßbedingte Flüchtigkeitsfehler vermeiden zu helfen. Die Arbeitsbedingungen sind so zu gestalten, daß die Gefahr des menschlichen Versagens möglichst gering gehalten wird. Wichtig ist natürlich auch die Schulung

des Personals. Und in vielen Fällen ist Vertrauen besser als rigide formale Vorschriften.

Noch vor wenigen Jahren gab es so gut wie kein Sicherheitsdenken in der Datenverarbeitung, und zwar auf allen Ebenen – vom DV-Manager bis hin zum Homecomputerbesitzer; eine Ausnahme bildeten oft die Rechenzentrumsleiter, deren Mahnungen aber überhört wurden. Auch heute noch wird das Datensicherheitsbewußtsein in der Fachpresse oft als unterentwickelt bezeichnet. Dennoch kann man zur Zeit ein starkes Ansteigen des Interesses feststellen.

1.5 Personalprobleme

Die größte Gefahr für die Datensicherheit geht vom eigenen Personal, den „Insidern“ aus. Folglich muß diesem Punkt auch die größte Aufmerksamkeit gewidmet werden. Zu beachten sind:

- Einstellungspolitik (Überprüfung, Probezeit),
- Entlassungspolitik (Gefahr von Racheakten),
- Zuverlässigkeit des Personals (persönliche Probleme, Alkoholismus),
- Kompetenzniveau des Personals (Wahrscheinlichkeit von Bedienungsfehlern),
- Betriebsklima, Interessenkonflikte zwischen Management und Mitarbeitern,
- Kommunikationsverhalten des Managements,
- Abwerbung durch die Konkurrenz, die gängigste Methode der Wirtschaftsspionage,
- Beschäftigung von Fremdfirmen (Unternehmensberatung, Wartung, Reinigung), Gefahr des Einschleichens nicht zugehöriger Personen.

Wie kritisch in anderer Hinsicht die Personalüberprüfung ist, wurde im Abschnitt über die gesellschaftspolitischen Gefahren erläutert.

1.6 Kosten-Nutzen-Abwägungen

Datensicherheit verursacht Kosten. Das können direkte Kosten für bauliche Maßnahmen oder zusätzliche Software sein oder Kosten für zusätzliches Personal mit Sicherheitsfunktionen; manche Risiken können auch durch Spezialversicherungen abgedeckt werden — ein weiterer Kostenfaktor. Kosten entstehen aber auch indirekt in Form von Zeit und Mühe. Zeit braucht man für die Planung, aber auch im täglichen Umgang mit den Sicherheitsmaßnahmen. Mühe verursachen die ständig geforderte Aufmerksamkeit oder lästige Identitätskontrollen.

Nicht zu vergessen ist auch, daß einige Schutzmaßnahmen wie die Verschlüsselung oder die Überprüfung von Zugriffsberechtigungen auf Datenfeld-Ebene Datenverarbeitungsleistung kosten und die Antwortzeiten am Terminal erhöhen.

Demgegenüber steht der Wert der zu schützenden Daten und Informationen, die Wahrscheinlichkeit eines Schadensfalls und der Verlust, der im Schadensfall entsteht. Hier muß man sich auch klar machen, daß etwa ein sehr schneller Datentypist, der mit 250 Anschlägen pro Minute schreibt, 4000 Minuten, also 70 Stunden, für 1 Megabyte braucht (das ja etwa 500 DIN-A4-Seiten Text entspricht). Der Wert der Daten ist nicht notwendig materiell bestimmbar, wie das Beispiel von personenbezogenen Daten, etwa Patientendaten im Krankenhaus oder Personaldaten im Betrieb zeigt. Hier ist unabhängig von Kostengesichtspunkten der nach dem Stand der Technik mögliche und nach dem Ausmaß der Bedrohung nötige Schutz zu leisten.

Kosten-Nutzen-Abwägungen sind auch aus dem Blickwinkel des potentiellen Angreifers durchzuführen. Lohnt es sich für ihn, den Aufwand zur Überwindung der Schutzbarrieren auf sich zu nehmen? Ein typisches Beispiel für eine solche Überlegung ist die Bestimmung des Aufwands zum Entschlüsseln verschlüsselter Daten. Die Kosten für den Angreifer sind auch wieder nicht nur materiell zu sehen; die Gefahr, entdeckt zu werden, gehört ebenfalls dazu. Erschwert wird die Rechnung dadurch, daß sein Nutzen die Befriedigung nichtmaterieller Bedürfnisse sein kann, für die ihm kaum ein Aufwand zu hoch ist.

Bei der Abwägung von organisatorischen Maßnahmen statt eingebauter Schutzmechanismen ist zu bedenken, daß organisatorische Maßnahmen (z. B. Verbote) zwar oft wenig kosten – wenn man davon ausgeht, daß der „Organisator“ sonst nichts zu tun hat –, aber mit der wachsenden Komplexität der Systeme auch immer unübersichtlicher und schwerer zu überwachen werden und von der Zuverlässigkeit der Mitarbeiter abhängen. Die Kosten für systeminterne Schutzmechanismen sinken durch den Preisverfall der Hardware, die immer weitere Verfügbarkeit von Standard-Software-Lösungen und die immer größere Leistungsfähigkeit der Systeme, die einen Leistungsverlust durch Schutzmaßnahmen verschmerzbar macht.

Zu den Kosten-Nutzen-Abwägungen gehört auch die Einschätzung der Qualität der Maßnahmen:

einfache Maßnahmen – sie schützen vor versehentlichen Einblicken und verdeutlichen dem gutwilligen Benutzer des Systems, wo seine Befugnisse enden.

mittelstarke Maßnahmen – sie sind nur mit Spezialausrüstung oder Spezialkenntnissen zu durchbrechen.

sichere Maßnahmen – sie sind nach dem Stand der Technik und der Wissenschaft mit den existierenden Ressourcen nicht zu durchbrechen.

Eine einfache Maßnahme ist zum Beispiel ein Paßwortsystem für eine PC-Festplatte; sie kann durch „booten“ von einer Diskette umgangen werden. Eine

mittelstarke Maßnahme ist der Paßwortschutz auf einem geschlossenen System, sofern er mit einer geeigneten Überprüfung verbunden ist, die etwa Trivialpaßwörter verhindert. Als sichere Maßnahme darf man die Verschlüsselung mit DES oder RSA betrachten.

2 Physischer Schutz

Alle technischen Schutzmaßnahmen beginnen beim physischen Schutz. Ein noch so sicheres Betriebssystem eines Großrechners nützt nichts, wenn die Anlage für jedermann frei zugänglich ist. Ein geschütztes Zugriffssystem für eine PC-Festplatte ist in der Regel leicht zu umgehen, wenn ein Angreifer sich unbemerkt einige Zeit am Gerät zu schaffen machen kann. In einem leicht zugänglichen Regal, in dem Druckerlisten ausgelegt werden, findet der interessierte Spion vielleicht schon alles, was er wissen will, ohne erst in ein Datenarchiv einbrechen zu müssen.

2.1 Schutz vor Katastrophen

Der Begriff „Katastrophe“ bedeutet für eine Datenverarbeitungsanlage einen Ausfall ihrer Funktion. Zunächst einmal ist nötig, den Anspruch an die Ausfallsicherheit zu klären. Welche Komponente darf wie lange ausfallen? Welche Kosten entstehen durch ihren Ausfall? An diesen Größen müssen sich die Schutzmaßnahmen orientieren.

Hier geht es zunächst um den Schutz vor Katastrophen durch „höhere Gewalt“ und physische Zerstörung. Solche Katastrophen sind kaum vorhersehbar oder verhinderbar, wohl aber sind die Auswirkungen steuerbar und begrenzbar. Zu erwägen sind:

- Risikofaktoren:
  - Feuer, Sturm, Erdbeben,
  - Wasser (Brauchwasser, Regenwasser, Hochwasser, Löschwasser),
  - Schmutz,
  - Störungen der Infrastruktur (Stromausfall, Klimaanlage),
  - elektronische Störungen, Überspannungen [27],
  - Bedienungsfehler, menschliches Versagen,
  - Hardware- und Softwarefehler,
  - Sabotage, Zerstörung, Vandalismus,
  - Kriminalität, Mißbrauch,
  - Einbruch, Diebstahl,

  wobei Bedienungsfehler die häufigste Schadensursache sind.

- Gefahren in der Umgebung der Gebäude.
- Ausstattung und Umfeld der Räume:
  - feuersichere Baumaterialien,
  - Brandschutztüren,
  - feuerhemmende Datentresore,
  - Schutz vor Wasserschäden, etwa Rohrbrüchen in höheren Stockwerken,
  - Sicherheit vor Hochwasser und anderen Naturkatastrophen,
  - Meldesysteme für Rauch, Feuer, Wasser,
  - Notausschalter.
- Brandschutzmaßnahmen:
  - Sprinkler und andere Feuerlöscheinrichtungen,
  - sichere Lagerung brennbarer Stoffe (auch Druckpapier und Datenträger),
  - Rauchverbote,
  - Schutz vor Kabelbränden,
- Ausfallsicherheit der Geräte, Notstromversorgung.

Alle eingeführten und geplanten Maßnahmen sowie die Regelungen für das Vorgehen im Notfall sollten in einem **Katastrophenplan** zusammengefaßt sein.

2.2 Zugangssperren

Je nach Art des Betriebs lassen sich unerwünschte Eindringlinge schon im Vorfeld abfangen. Bei Behörden mit Publikumsverkehr, Kliniken und Universitäten ist das allerdings kaum möglich. Dennoch lassen sich auch hier zugangsbeschränkte Sicherheitsbereiche schaffen; und gerade wegen des mehr oder weniger freien Zugangs zu den Gebäuden sind abgetrennte Sicherheitszonen besonders wichtig. Eine Aufzählung möglicher Maßnahmen:

- Schutz des Geländes und der Gebäude,
- Festlegung von Sicherheitsbereichen:
  - Maschinenraum,
  - Stromversorgungs-, Hausanschlußraum,
  - Klima-Anlagen-Raum,
  - Datenarchiv,

  - Operatorräume,
  - Räume der Systemabteilungen,
  - Räume für Benutzer und Benutzergruppen,

- Schließanlagen und Schleusen für Sicherheitsbereiche:
  - Türsicherung mit Schlüsselregelung oder Zugangskontrollsystem,
  - Personalschleusen mit Ausweis- oder Gesichtskontrolle,
  - Schalter mit Sicherheitsglas, Durchreiche und Gegensprechanlage zur Datenträgerausgabe,

- Maßnahmen zur Objektsicherung:
  - Videoüberwachung,
  - Sicherung durch Alarmanlage, besonders außerhalb der Dienstzeit,
  - einbruchsicheres Glas in den Fenstern der Sicherheitsbereiche,
  - Stahltüren zu den Maschinenräumen,
  - Sicherung von Zugangsmöglichkeiten zu Kellerräumen und benachbarten Geschossen,
  - Zugangssicherung zu Mitarbeiterräumen.

Anzustreben ist, je nach Einschätzung der Verhältnismäßigkeit, ein „Closed-Shop-Betrieb“: Zugang zu einem bestimmten Ort hat nur, wer ihn auf Grund seiner Aufgabe braucht. Solch eine Regelung ist in der Industrie weitgehend üblich, in Universitäten dagegen nur vereinzelt anzutreffen.

Zugangssperren müssen in Notfällen auch von autorisierten Personen abschaltbar sein, damit Rettungsmaßnahmen nicht verzögert werden.

2.3 Sicherung von Datenträgern

Zur Sicherung der Datenträger gehört die sichere Aufbewahrung, aber auch die Abgangs- und Transportkontrolle:

- Aufbewahrung:
  - Lagerung der Datenträger in Sicherheitsbereichen, Wegschließen von Disketten,
  - Sicherheitsschränke, Tresore,
  - klare Definition der Befugnis zur Datenträgerverwaltung,
  - Bestandskontrolle der Datenträger,

- Abgangskontrolle:
  - Ausgabe von Datenträgern nur an befugte Personen,
  - Kontrollierte Löschung oder Vernichtung von Datenträgern,
  - Abgabemöglichkeit für zu vernichtende Druckerlisten, Reißwolf,
  - Arbeitsplatzrechner ohne Diskettenlaufwerke,
- Transportkontrolle:
  - Verpackungs- und Versandvorschriften, zum Beispiel Verwendung verschlossener Transportkoffer,
  - Transport nur durch ausdrücklich befugte Personen,
  - Nutzung eines gesicherten Eingangs und von Schaltern und Schleusen für An- und Ablieferung.

2.4 Schutz der Datenleitungen

Die verschiedenen Typen von Datenleitungen bieten unterschiedliche Sicherheit:

Verdrillte Kupferkabel (Telefondraht) sind elektromagnetisch abhörbar und störbar; es gibt aber auch abgeschirmte Kabel dieses Typs.

Koaxialkabel sind elektromagnetisch sicher, lassen sich sich aber leicht anzapfen.

Lichtwellenleiter (Glasfaserkabel) sind abhörsicher und etwas schwerer anzuzapfen; das Anzapfen erfordert ein Unterbrechen oder starkes Biegen der Leitung und wird daher durch geeignete Überwachung leicht entdeckt.

Als Maßnahmen zum physischen Schutz sind geeignet:

- Elektromagnetische Abschirmung von Kabeln und Bildschirmen,
- Zugangssperre zu Kabelschächten und Kabelwegen,
- Aufstellung von Verteilereinrichtungen nur in Sicherheitsbereichen,
- Verlegung von Leitungen in Gasrohren mit Überdruck.

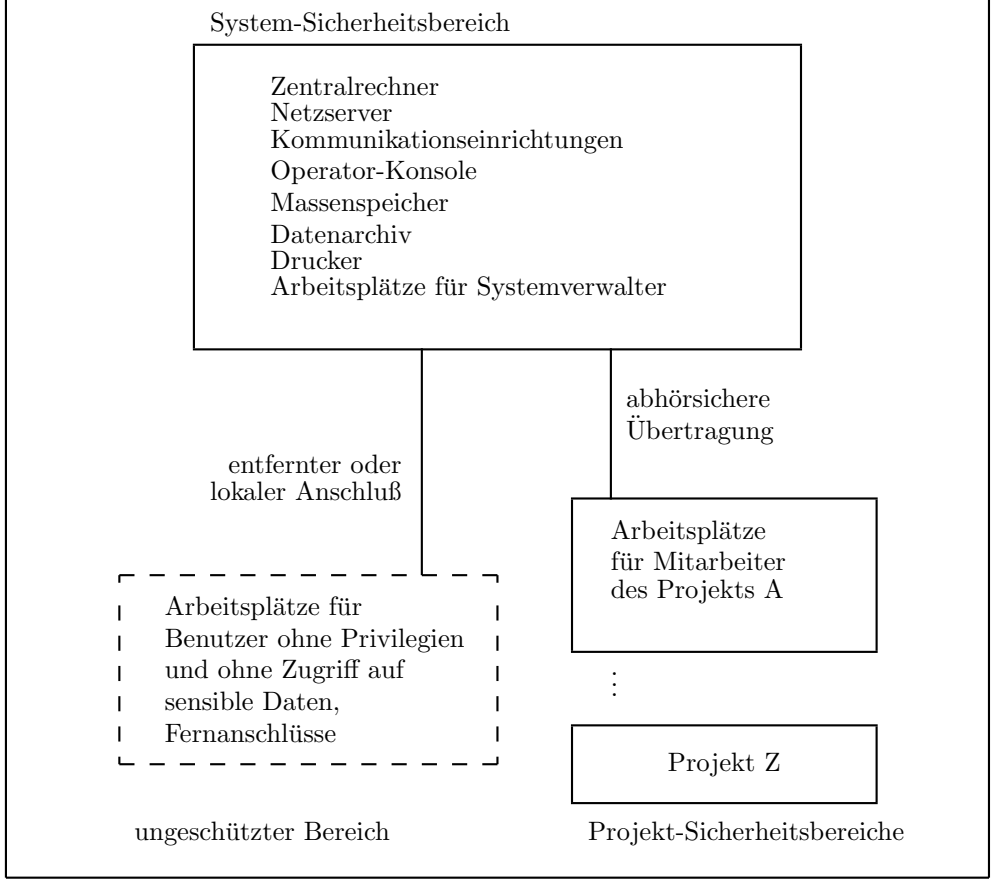
Das Problem der Leitungssicherheit wird im Kapitel IV über Netze noch ausführlicher behandelt.

Allgemein gilt, daß Zentraleinheiten, Plattenspeicher und parallele Schnittstellen schwer abzuhören sind wegen des dort auftretenden Signalgemischs. Leicht abzuhören sind die Videosignale von Terminals, serielle Schnittstellen, Überlandleitungen, Richtfunkstrecken und Satellitensignale, wobei in den letzten vier Fällen auch Verfälschungen, also aktive Angriffe, möglich sind.

2.5 Ein Modell für die physische Sicherheit

Ein Modell für eine gute physische Absicherung eines Rechnersystems wird in Abbildung II-1 vorgestellt.

Abbildung II-1: Ein Modell für physische Sicherheit

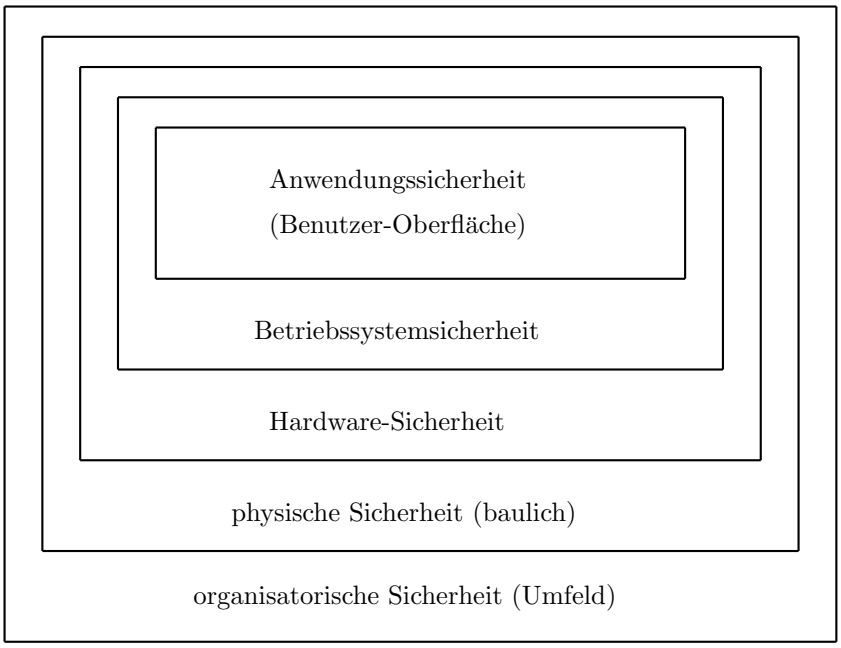


Es enthält einen System-Sicherheitsbereich, in dem alle Ressourcen für den Systembetrieb untergebracht sind und in dem auch die Systemverwaltung und -bedienung untergebracht sind. Getrennt davon existieren Projekt-Sicherheitsbereiche für die einzelnen Projektgruppen, die sensible Daten bearbeiten. Außerhalb dieser Bereiche dürfen nur Benutzer ohne irgendwelche Systemprivilegien und ohne Zugriff auf sensible Daten arbeiten; im begründeten Fall sind zeitlich und örtlich beschränkte Ausnahmegenehmigungen möglich.

Die Verbindungen zwischen den verschiedenen Bereichen laufen über abhörsichere Leitungen oder verschlüsselt. Als minimaler Schutz für frei zugängliche Arbeitsplätze, etwa im PC-Saal eines Universitätsrechenzentrums, ist eine übersichtliche Anordnung und die Anwesenheit einer Aufsichtsperson zu empfehlen.

Was ein geschlossenes System ist und wie in ihm die Daten geschützt werden, läßt sich am einfachsten durch ein Schalenmodell verdeutlichen, siehe Abbildung II-2.

Abbildung II-2: Die Sicherheitsschalen eines geschlossenen Systems



Ist eine der äußeren Schalen durchlässig, so ist der Schutz der inneren Schalen zu umgehen, das System nicht mehr geschlossen. Das verdeutlichen die Beispiele:

1. Die physische Sicherheit, die ein abschließbarer Raum bietet, nützt nichts ohne organisatorische Sicherheit – wenn sich niemand für das Abschließen zuständig fühlt.
2. Die Hardware-Sicherheit durch ein Schloß am PC nützt nichts ohne physische Sicherheit: Man kann den PC einfach wegtragen und zu Hause in

aller Ruhe aufbrechen.

3. Die Betriebssystem-Sicherheit eines Paßwortschutzes für die Festplatte nützt nichts ohne eine Hardwaresicherung, die das „Booten“ von einer Diskette verhindert.

4. Die Anwendungs-Sicherheit durch Vergabe von Zugriffsrechten in einem Datenbanksystem nützt nichts, wenn das Betriebssystem gestattet, die Festplatte sektorenweise zu analysieren.

In einem geschlossenen System reicht es tatsächlich, Zugriffsrechte auf bestimmte Daten durch Einträge in System-Tabellen zu definieren – das ist eine sichere Maßnahme. Die wichtigen Systemkomponenten sind physisch geschützt; auf dieser Grundlage lassen sich logischer Zugang zum System und Zugriff auf Daten wirksam durch das Betriebssystem überwachen. Dieser Schutz ist heute typisch für eine Großrechner-Umgebung.

3 Hardware

Zum eigentlichen Datenverarbeitungssystem gehören Hardware und Software. Die Trennung dieser Komponenten ist oft nicht so klar, wie man auf den ersten Blick meinen könnte, da die Hardware meist von „Microcode“ (auch „Firmware“ genannt) gesteuert wird. Das ist nicht ganz unproblematisch, da der Microcode auf unterster Ebene maschinennah programmiert und für den Systembetreiber nicht dokumentiert ist. Daraus resultiert in der Praxis eine nicht zu vernachlässigende Fehleranfälligkeit.

Allgemein gehört zur Hardware:

- Prozessoren,
- Hauptspeicher,
- Peripherie- oder Massenspeicher,
- Ein- und Ausgabegeräte.

Schutzmechanismen der Hardware dienen meist gleichzeitig dem Schutz vor Fehlfunktion und der Datensicherheit, sind also Nebenprodukte der Bemühungen, die Datenverarbeitungsanlage überhaupt in einen einigermaßen stabilen Betriebszustand zu bringen. Weitergehende Schutzfunktionen der Hardware sind hauptsächlich auf Großrechnern anzutreffen.

Sicherheitsmaßnahmen auf der Hardware-Ebene sind für den Systembetreiber über die reinen Beschaffungsentscheidungen hinaus kaum möglich; der Handlungsbedarf liegt beim Hersteller. Dennoch ist auch hier ein Kriterienkatalog notwendig, einmal für die Beschaffung, aber auch, um Stärken und Schwächen des eigenen Systems zu kennen.

Manche Schutzmaßnahmen lassen sich sowohl durch Hardware-Einrichtungen als auch mit Software verwirklichen. Ein Vorteil von Schutzmaßnahmen im Hardwarebereich ist, daß sie in der Regel die Rechenleistung des Systems nicht vermindern. Software-Schutzmaßnahmen sind meistens billiger und leichter zu konfigurieren, verbrauchen aber oft beträchtliche Rechenleistung und sind für Systemspezialisten auch leichter und unauffälliger zu umgehen. Hardwareschutz ist dagegen durch Programme nicht auszuhebeln.

Wie unerwartet und schwer durchschaubar sich Hardware-Fehler auswirken können, zeigt ein Bericht im Risks-Digest 9.37, wo Fälle beschrieben wurden, in denen Anwendungsprogramme plötzlich temperaturabhängig arbeiteten. Der Grund wurde nach langem Rätselraten gefunden: Ein Coprozessor fiel unbemerkt immer bei erhöhten Temperaturen aus. Man stelle sich ein Sicherheitssystem (etwa in einem Chip) vor, das abhängig von den Umgebungsbedingungen arbeitet!

3.1 Schutz des Hauptspeichers

Folgende Einrichtungen sind üblich:

- Verhinderung oder Korrektur von Übertragungsfehlern und Speicherfehlern durch Paritätsprüfung und fehlerkorrigierende Codes. Mathematische Grundlage dieser Verfahren ist die Codierungstheorie.
- Grenzregister, die beim Mehrprozeßbetrieb sicherstellen, daß ein Prozeß nicht auf Daten außerhalb des ihm zugewiesenen Speicherbereichs zugreift.
- Speicherschutzschlüssel aus mehreren Schutzbits, die differenzierte Zugriffsrechte (lesen/schreiben) auf Speicherbereiche für Prozesse absichern (realisiert in der IBM-/370-Architektur).
- Virtuelle Speicherverwaltung und Adressierung. Hierbei können Programme gar nicht auf reale Adressen zugreifen, sondern haben ihr eigenes Adressierungssystem, das vom Prozessor mit Hilfe spezieller Register in reale Adressen umgesetzt wird. Auf diese Weise ist sehr leicht ein sicherer Schutz der verschiedenen Prozesse gegeneinander zu verwirklichen. Deshalb und aus anderen Gründen hat sich die virtuelle Speicherverwaltung heute weitgehend durchgesetzt.
- Typgebundene (TR440) oder objektorientierte (IBM-/38 und AS/400) Speicherverwaltung, bei der Speicherwörter oder gespeicherte „Objekte“ einen bestimmten Typ haben, der die zulässigen Funktionen festlegt.

Eine detaillierte Beschreibung dieser Maßnahmen würde hier viel zu weit führen; sie gehört in ein Buch über Betriebssysteme wie etwa [139]. Auch Probleme und Schutzmethoden neuerer Speicherkonzepte wie schneller Zwischenspeicher (‘Cache’) oder Adreßumwandlungsspeicher (‘translation lookahead buffer’) sollen hier nicht behandelt werden.

3.2 Prozessor-Operation

Ein Benutzer, der die Kontrolle über die CPU hat, kann zunächst einmal jede beliebige Operation ausführen. Er kann den ganzen Hauptspeicher durchsuchen und alle Ein-Ausgabe-Ports ansprechen. Es ist Aufgabe der Hardware, dies zu verhindern. Das wird erreicht, indem der Prozessor verschiedene Betriebszustände ('Modi') kennt und dadurch an seinem Zustand ablesen kann, in wessen Auftrag er gerade arbeitet. Daraus kann er ermitteln, welche Speicheradressen und Portadressen er ansprechen darf und in welcher Weise. Das Betriebssystem selbst muß natürlich alles dürfen. Minimal werden ein privilegierter Zustand („System-Modus", „Supervisorstatus") und ein unprivilegierter („User-Modus", „Problemstatus") unterschieden.

Wichtig ist, daß es keine unkontrollierten Zustandsänderungen geben darf. Andererseits müssen Benutzerprogramme ja Leistungen des Betriebssystems in Anspruch nehmen können. Zu diesem Zweck gibt es einen 'Supervisor Call', der den Wunsch nach einer Operation an das Betriebssystem übergibt, wo er vor der Ausführung auf Berechtigung überprüft wird.

Werden nur zwei Zustände unterschieden, besteht die Gefahr, daß privilegierte Programme keinen Schutzmaßnahmen mehr unterliegen. Ein Fehler oder ein Trojanisches Pferd in einem Prozeß, der im privilegierten Zustand abläuft, kann großen Schaden anrichten. Bei solchen Systemen (etwa IBM /370) muß dann ein ziemlich hoher Aufwand in der Software betrieben werden, um mögliche Fehler zu verhindern. Aber auch eine Unterscheidung von mehr Zuständen löst das Problem nicht grundsätzlich; auch hier arbeiten Systemprozesse immer noch meistens mit mehr Privilegien, als sie im Moment brauchen, also nicht nach dem Prinzip der minimalen Rechte. Manche Betriebssysteme lassen sich durch provozierte Fehler gezielt dazu bringen, zu einem Benutzerprozeß zu springen, ohne ein Privileg-Bit zurückzusetzen; das wird aus den 70er-Jahren vom PDP-10 TENEX berichtet, soll aber in manchen UNIX-Systemen auch heute noch möglich sein. Viele moderne Systemarchitekturen arbeiten mit besser differenzierten Zuständen, etwa statischen, hierarchisch angeordneten oder gar dynamisch erzeugbaren. Die feinere Regelung ist dann aber Sache der Software.

3.3 Schutz der Ein- und Ausgabemedien

Bei der Ein- und Ausgabe verlassen die Daten den schützenden Bereich des Rechners und bekommen Kontakt mit der Umwelt. Dieser Kontakt ist der Hauptansatzpunkt des Datenschutzes. Die wichtigsten Medien sind:

- Terminal (Bildschirm und Tastatur),
- Drucker,
- Datenübertragungsleitungen,
- Platten und Bandspeicher.

Die Hardware sollte das Betriebssystem möglichst weitgehend bei der Aufgabe unterstützen, die nötigen Datenübertragungsvorgänge korrekt abzuwickeln. Gegen Übertragungsfehler werden auch hier Prüfbits und fehlererkennende Codes eingesetzt. In der Regel haben Ein- und Ausgabegeräte heute eigene Prozessoren ('Controller') und Zwischenspeicher (Puffer oder 'Buffer'), die von Microcode gesteuert werden. Auf den Microcode kann vom System aus nur in einem besonderen Wartungszustand zugegriffen werden. Die Auslagerung komplizierter Gerätesteuerungen auf spezielle Prozessoren führt zu einer Modularität im Hardwarebereich und damit zu einer Verringerung der Komplexität und zu erhöhter Sicherheit – vorausgesetzt die einzelnen Teile arbeiten so, wie sie sollen.

Auch die Datenträger selbst sind durch Hardware-Vorkehrungen schützbar. Das einfachste Beispiel ist der Schreibschutz auf Disketten (durch Schließen einer Schreibschutzkerbe) oder Bändern (durch Entfernen des Schreibrings). Immer wenn möglich sollte man mit diesem Schreibschutz arbeiten. Sind mehrere Drucker vorhanden, so sollte die Zuordnung von Sicherheitsklassen zu Druckern durch Druckersicherheitsmasken unterstützt werden.

Arbeitsplatzrechner ohne Diskettenlaufwerk in einem Netz haben den Vorteil, daß Daten nicht so leicht unbefugt wegtransportiert werden können. Außerdem lassen sich nicht so leicht Viren ins System kopieren – man muß sie von Hand eingeben.

Wechselbare Festplatten (Data-Pac-System von Tandon) erlauben, auch größere Datenbestände sicher aufzubewahren, und bieten, sofern das System zwei solche Platten aufnehmen kann, zusätzlich eine unkomplizierte Möglichkeit zur Anfertigung von Sicherungskopien.

3.4 Sicherheitshardware

Die einfachste Sorte von Sicherheitshardware sind abschließbare Terminals; als Steigerung lassen sich manche Terminals auch mit Lesern für maschinenlesbare Ausweiskarten versehen. In die Kategorie Sicherheitshardware gehören auch abgeschirmte Terminals und Kommunikationsleitungen sowie einmal beschreibbare optische Platten ('WORM') zur manipulationsgeschützten Protokollierung von Vorgängen; diese Technik wird hierfür in der Praxis bisher allerdings noch kaum genutzt.

Einfache Magnetkarten-Systeme bieten in den meisten Anwendungsfällen keine ausreichende Sicherheit. Wer einen PC hat und zusätzlich etwa 5000 DM für ein Kartenlese- und -schreibsystem mit passender Software anlegt, kann Magnetkarten beliebig kopieren oder ändern. Für eine ausführliche Beschreibung sei auf [42] verwiesen.

Deutlich erhöhte Sicherheit bietet dagegen ein neuer Typ von maschinenlesbaren Ausweiskarten, die Chipkarten. Hier werden Speicher- oder Prozessorchips in Karten verpackt, die der DIN-Norm für Kredit- und Scheckkarten entsprechen, insbesondere 0.76 mm dick sind; um den Bruchtest zu bestehen, dürfen

eingebaute Chips nicht größer als $20mm^2$ sein. Der Einbau in Karten mit den Ausmaßen einer Eurocheque-Karte ist noch nicht gelöst. Auch die Verschlüsselung bereitet Probleme, da DES-Chips noch zu sperrig sind. Eine solche Karte braucht

- freien (ungeschützten) Speicherplatz,
- geschützten Speicherplatz, der nur nach Eingabe einer Geheimnummer (PIN) lesbar ist,
- geheimen Speicherplatz, der überhaupt nicht auslesbar ist,

außerdem Kontakte zur Datenübertragung an Kartenleser; es wird auch mit „drahtloser“ Übertragung experimentiert (Mikrowellen, Induktion), diese Systeme sind aber noch nicht praktikabel. Nötige Software wird in ein ROM eingebaut. In Frankreich werden Chipkarten schon in größerem Umfang eingesetzt. Anwendungen sind:

- Bargeldloser Zahlungsverkehr (‘Point Of Sale’, POS),
- Geldausgabeautomaten,
- Btx-Anwendungen, zum Beispiel ‘teleshopping’,
- öffentliche Telefone,
- Zugangskontrollsysteme,
- Berechtigungskarten, auch als Schutz vor Raubkopien.

Es lohnt sich also, hier einige Gedanken an Datensicherheit zu verschwenden. Die Chipkarte als Zugangskontrollsystem eröffnet neue Möglichkeiten, den Datenschutz zu verbessern. Auf der anderen Seite ist sie aber als Angriffsobjekt gefährdet. Der Versuch, die in ihr gespeicherten Daten zu lesen, führt zur Zerstörung (solange niemand eine anderes Verfahren erfunden hat). Trotzdem ist ein zusätzlicher Paßwortschutz als Diebstahlsicherung unbedingt nötig. Dieses Paßwort (‘PIN’) wird aber immer nur lokal benützt und geht niemals über ein Netz. Eine große Gefahr ist die automatische Erfassung von (etwa) Zahlungsvorgängen oder Bewegungsprofilen. Eine genauere Beschreibung der Chipkarten findet man in [138]. Die in der Chipkarte ablaufenden Prozesse sind kryptologischer Natur und werden im Kapitel V besprochen, ebenso Ansätze, die bestehenden Gefahren zu vermeiden.

Als Zugangskontrollsysteme werden biometrische Meßgeräte angeboten, die etwa folgendes erkennen können:

- Fingerabdrücke,

- Sprachcharakteristika,
- Netzhautbilder,
- Unterschriftsdynamik (also charakteristische Druckpunkte und ähnliches).

Für Sicherheitszwecke, etwa zur Überwachung von Netzen, zur Zugangskontrolle oder zum Zugriff auf Datenbestände oder Kommunikationsleitungen, können separate Rechner oder Prozessoren eingesetzt werden.

Eine wichtige Klasse von Sicherheitshardware bilden Verschlüsselungs-Chips: Kryptographische Verschlüsselung von Daten kostet Rechenzeit. Wenn möglich sollte man dafür gesonderte Einsteckkarten verwenden, die einen eigenen Prozessor mit RAM-Speicher und den Algorithmus in einem ROM haben. DES-Chips verschlüsseln bis zu einigen 100 Kbit pro Sekunde; das reicht aus, um bei den meisten Anwendungen keine Verzögerung spüren zu lassen – zum Vergleich: bei ISDN werden 64 Kbit/sec übertragen. Allerdings wird das DES-Verfahren in letzter Zeit von der Regierung der Vereinigten Staaten mit Exportbeschränkungen belegt. Für das RSA-Verfahren gibt es inzwischen Chips, die über 10 Kbit/sec schaffen.

Eine Verschlüsselungskarte läßt sich zu einem brauchbaren Sicherheitssystem für den PC ausbauen. Sie kann zum Beispiel einen nicht zu umgehenden Paßwortschutz gewährleisten, eventuell gekoppelt mit einem mechanischen Schlüssel. Die Daten auf der Festplatte müssen verschlüsselt sein. Damit man die Karte nicht einfach ausbauen kann, sollte sie mit einem Selbstzerstörungsmechanismus versehen sein, etwa einem Lichtfenster im ROM, das für dessen Löschung sorgt, indem beim Abhebeln eines Verschlusses eine Fotozelle ausgelöst wird. Ferner sollten Lötstellen nicht zugänglich sein, damit nicht an ihnen Informationen abgezapft werden können. Auch ein System zum Schutz vor Raubkopien ließe sich nach diesem Prinzip verwirklichen, würde aber zu einer deutlichen Verteuerung der Software führen und wird sich wohl deshalb auf dem Markt nicht durchsetzen. Sicherheitssysteme für PCs werden in Abschnitt 5.2 noch einmal behandelt.

Auch die großen Hersteller bieten inzwischen einiges an Sicherheitshardware an, etwa das IBM-Transaktionssicherheitssystem, das einen Netzwerk-Sicherheitsprozessor, einen kryptographischen Adapter und eine Sicherheits-Zugangseinheit enthält.

4 Software

Mit Softwaremethoden lassen sich sehr differenzierte und zuverlässige Schutzmaßnahmen erreichen. Sie können allerdings nur im Rahmen eines ausreichenden physischen Schutzes und auf hinreichend sicherer Hardware wirksam werden. Sie schützen den logischen Zugang zum System und seinen Daten, wenn

der physische Zugang so gesichert ist, daß niemand die Maßnahmen durch physische Gewalt oder Basteleien aushebeln kann. Logischer Zugang bedeutet, daß das System den Benutzer als berechtigt akzeptiert.

Drei wichtige Prinzipien, die beim Softwareschutz zu verwirklichen sind, sind

das Prinzip des geschlossenen Systems oder der minimalen Rechte: Jeder Benutzer darf nur die Funktionen verwenden, die er ausdrücklich für seine Tätigkeit braucht. Hiermit kann man dem Ideal des „verifizierten Schutzes gegen alle nichtspezifizierten Zugriffe“ ziemlich nahe kommen.

das Prinzip der minimalen Schnittstellen oder das Modularitätsprinzip: System- und Anwendungssoftware muß modular aufgebaut sein; jeder Modul hat eine möglichst schmale Schnittstelle, die seinen gesamten Kontakt zur Außenwelt regelt. Auf diese Weise kann die Gefahr von Fehlern minimiert und die Geschlossenheit des Systems optimiert werden.

das Server-Prinzip: Dienstprozesse laufen unabhängig von Benutzerprozessen und überprüfen bei ihrem Aufruf die Berechtigung des Aufrufenden.

Das Server-Prinzip kann man als Ausprägung des Prinzips der minimalen Schnittstellen ansehen: Der Kern eines Prozesses wird von der Außenwelt abgekapselt; als Verbindung dient eine möglichst enge, gut kontrollierbare Schnittstelle. (Als Vergleich kann man sich den Zugang zu einer mittelalterlichen Burg vorstellen.) Als übergeordnetes Prinzip erkennt man das Prinzip des geschlossenen Systems: Machbar ist nur, was ausdrücklich erlaubt ist, und auch das ist nur unter Kontrolle machbar. Natürlich ist das Server-Prinzip auch im Hardware-Bereich von Bedeutung, wenn in einem Netz logische Funktionen auch physisch verteilt werden.

Die Zuverlässigkeit der Software könnte das Hauptproblem der Informationsgesellschaft sein. Anwender und Betreiber müssen sich darauf verlassen können, daß die Software korrekt abläuft und sicher vor Manipulationen ist. Auch im Falle von Mängeln müssen die Änderungen korrekt eingebaut werden. Das Thema der Korrektheit von Software wird in dem Buch [43] eingehend behandelt.

4.1 Betriebssystem

Das Betriebssystem ist eine Sammlung von Prozessen und Prozeduren, die den Umgang mit der Hardware für Anwendungsprogramme erleichtern. Zunächst handelt es sich dabei um das Ansprechen von Peripheriegeräten, vor allem von Platten- oder Diskettenlaufwerken und Terminals, und um den Aufbau und die Verwaltung des Dateisystems. Bei Großsystemen kommt als wichtige Aufgabe noch die Verwaltung des Mehrbenutzer- und Mehrprozeßbetriebs hinzu. Den Aufbau eines Betriebssystems detailliert zu beschreiben, würde hier viel zu weit führen; eine geeignete Darstellung ist [139]. Auch in [136] wird das Thema ziemlich ausführlich behandelt. Die wesentlichen Aufgaben des Betriebssystems sind:

- Verwaltung von Objekten (Benutzern, Daten, Peripheriegeräten),
- Zugriffsmechanismen für Objekte (Synchronisation, Zugriffsschutz),
- Kommunikation zwischen Objekten,
- Erzeugung, Überwachung und Beendung von Prozessen,
- Fehlerkontrolle und -behandlung, Wiederaufsetz-Mechanismen.

Unter dem Gesichtspunkt der Datensicherheit sind vor allem Betriebssysteme für den Mehrprozeßbetrieb zu behandeln; das wird im Abschnitt 4.2 gemacht. Im Moment geht es um einige allgemeine Probleme.

Da es beim Betriebssystem besonders auf die Zuverlässigkeit ankommt, muß es sehr sorgfältig unter Beachtung aller Techniken zur Software-Erstellung konstruiert werden. Dennoch werden die meisten Betriebssysteme zur Erzielung einer möglichst hohen Ablaufgeschwindigkeit in Assembler programmiert. Selbstverständlich enthalten sie daher massenhaft Fehler; bei großen Betriebssystemen werden jeden Monat Hunderte von Fehlern neu bekannt – die meisten haben zum Glück keine Auswirkungen auf die Datensicherheit. Die Komplexität von Betriebssystemen macht es praktisch unmöglich, auf ein unsicheres Betriebssystem nachträglich Sicherheitssoftware aufzupfropfen. Es ist Aufgabe des Herstellers, das Betriebssystem *von Grund auf* sicher zu entwerfen.

Unter den höheren Programmiersprachen wird vor allem C zur Systemprogrammierung verwendet. Leider ist C vom Standpunkt der Sicherheit eher ein neues Problem als eine Lösung. Das wurde durch den Internet-Wurm deutlich demonstriert: Das *finger*-Programm dient in vielen UNIX-Systemen dazu, Informationen über andere Benutzer auszugeben, etwa ob sie im Moment arbeiten (so daß man ihnen Nachrichten schicken kann). Es enthält einen Hintergrundprozeß („Dämon“) *fingerd*, der für solche Anfragen über Netzverbindungen zuständig ist. Dieser liest eine Eingabezeile mit Hilfe der C-Prozedur *gets*. Diese legt die Eingabezeile in einen Puffer ab, ohne dessen Grenzen zu überwachen – eine Spezialität veralteter Programmiersprachen. (Dieses Problem tritt auch bei anderen Prozeduren der Standard-Bibliothek von C auf und ist sogar durch die ANSI-Norm geadelt.) Der Internet-Wurm übergab nun dieser Prozedur eine speziell konstruierte Kette von 536 Bytes, die über den Eingabepuffer hinausreichte und den anschließenden Programmstapel (‘Stack’) überschrieb, und zwar speziell die Rücksprungadresse. Auf VAX-Anlagen mit 4 BSD-UNIX wurde so ein fatales Programmstück angesprungen, auf anderen Systemen ergab sich einfach eine Fehlersituation mit Abbruch des Prozesses. Eine interessante Diskussion der Sicherheitsproblematik von C und (folglich) UNIX wurde im Risks-Digest 11.15 und 11.16 geführt. Tückische Fehlermöglichkeiten in C werden auch in [142] sehr anschaulich erklärt. Eine alarmierende Untersuchung über die Fehler in UNIX ist [85].

Meiner Meinung nach ist die optimale Korrektheit nur durch MODULA-2 zu erreichen; selbst ADA bietet mehr Möglichkeiten für unbeabsichtigte Nebenwirkungen und Fehler. Wie elegant man ein Betriebssystem in MODULA-2 programmieren kann, wird in [139] vorgeführt.

Um das Ausnützen von Fehlern im Betriebssystem zu erschweren, wird von einigen Herstellen der Quellcode der Systemprogramme nicht mehr ausgeliefert, etwa von IBM ('Object Code Only'; vielleicht ist der eigentliche Grund aber auch der Schutz vor Nachahmern). Das wirkt sicher auf die gewünschte Weise, widerspricht aber dem Prinzip der Revisionsfähigkeit – mit mangelhafter Dokumentation kann die Sicherheit eines Systems weder nachgewiesen noch überprüft werden.

Fehler- und Ausnahmesituationen werden in Abschnitt 4.9 behandelt. Vorweg zwei typische Beispiele: Durch Unterbrechen des Ladevorgangs treten oft Sicherheitslücken auf. Das Betriebssystem sollte dafür sorgen, daß Eingabevorgänge erst registriert werden, wenn es selbst einschließlich aller Sicherheitsvorkehrungen initialisiert ist. Ein spezielles, nicht ganz leicht zu lösendes Problem tritt auf Arbeitsplatzrechnern auf, wenn ein Benutzer sein eigenes Betriebssystem von einer mitgebrachten Diskette lädt und damit das offizielle Betriebssystem umgeht.

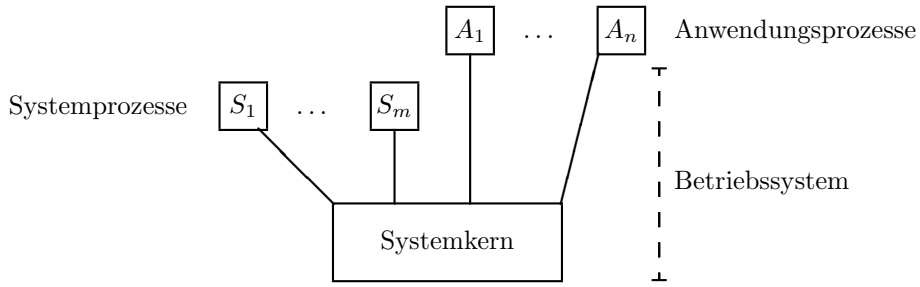
4.2 Mehrbenutzerbetrieb

Auf Großrechnersystemen arbeiten meistens viele Benutzer gleichzeitig; viele Aspekte dieses Sachverhalts treffen auch auf Netze zu. Das Hauptproblem ist die Trennung der verschiedenen Benutzer (oder allgemeiner der verschiedenen Prozesse) voneinander, einschließlich der Trennung der Adreßräume und der Zugriffe auf Peripheriespeicher. Ein wichtiger Aspekt dieser Trennung ist der Schutz von Prozessen vor Fehlern in anderen Prozessen.

Das Betriebssystem besteht selbst auch aus einer Anzahl von separaten Prozessen. Unter diesen muß einer, der Systemkern, besonders ausgezeichnet sein. Seine Aufgabe ist die Verwaltung der übrigen Prozesse und ihrer Privilegien, deren Synchronisation und Kommunikation und die Synchronisation von Ein- und Ausgabe-Operationen. Das ist schematisch in Abbildung II-3 dargestellt.

Der Systemkern muß besonders zuverlässig arbeiten; Fehler in ihm führen oft zum Absturz des gesamten Systems oder zu undefinierten Zuständen, in denen die Sicherheit nicht mehr gewährleistet ist und möglicherweise auch Dateien in einem inkonsistenten Zustand hinterlassen werden. Fehler im Systemkern sind auch Ansatzpunkte für einen technologischen Angriff [136]: „Wenn einem Angreifer bekannt ist, daß er durch bestimmte Befehlsfolgen, etwa durch Übergabe einer geeigneten falschen Adresse beim Aufruf eines Systemdienstes, eine Datenstruktur des Betriebssystems mit eigenen Daten überschreiben kann, so hat er hier ein Mittel in der Hand, um schwerste Schäden zu verursachen, bis hin zum Unterlaufen jeglichen Datenschutzes oder zur Zerstörung des Systems." Daß diese Gefahr nicht nur in der Theorie besteht, hat der Internet-Wurm gezeigt.

Abbildung II-3: Der Systemkern



Der Systemkern ist auch dafür verantwortlich, daß bei einem Prozeßwechsel die Registerinhalte des alten Prozesses gelöscht werden, ebenso Haupt- oder Cachespeicherbereiche, die einem anderen Prozeß zur Verfügung gestellt werden. Der Systemkern sollte auf der Hardwareebene durch eine Privilegienklasse oder einen Prozessorzustand abgesichert sein, die oder der ihm alleine zusteht. Alle Software-Sicherheitsmaßnahmen sollten im Zweifelsfall im Systemkern verankert sein, da sie dort besonders zuverlässig gegen Manipulation geschützt sind.

Einige weitere kleinere Aufgaben des Betriebssystems, die relevant für die Sicherheit im Mehrbenutzerbetrieb sind, sollen hier aufgezählt werden:

- Überwachung des Ressourcenverbrauchs.
- Erzeugung von deutlich sichtbaren Drucker-Trennseiten, um Fehlleitung von Ausdrucken zu verhindern.
- Überwachung der Stapelverarbeitung ('Batch jobs', 'Remote Job Entry').
- Automatische Löschung von Plattenbereichen und Bändern vor einem Besitzwechsel.
- Schutz von Datenbändern vor falscher Zuordnung durch interne Markierungen ('Labels') und deren Überprüfung.
- Ein System zur Datensicherung ('Backup'), das Zuordnungsfehler beim Restaurieren verhindert.
- Automatische Löschung von temporären Daten, die bei verschiedenen Arbeitsvorgängen erstellt werden, zum Beispiel in „Spool-Bereichen“ (Warteschlangen vor Ausgabegeräten), von Benutzern angelegten temporären

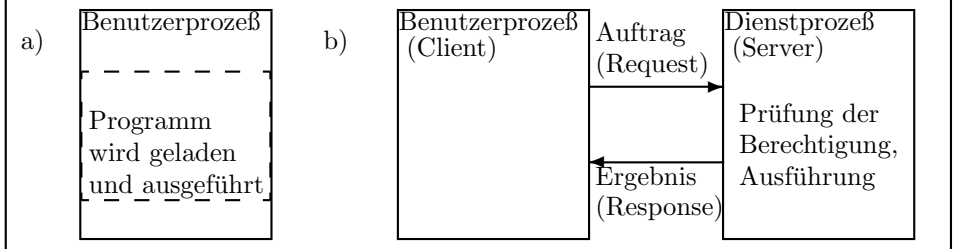
Plattendateien, Hauptspeicherbereichen, auch beim 'Paging' auf Platte ausgelagerten Hauptspeicherbereichen.

- Angriffsversuche auf Daten sind sowohl dem Systemverwalter als auch dem Besitzer zu melden.

Ein sehr eleganter Ansatz zur Verwirklichung eines Mehrbenutzerbetriebs ist das Konzept der „virtuellen Maschine", etwa im IBM-Betriebssystem VM, das auf /370-Maschinen läuft. Hier simuliert das Betriebssystem für jeden Benutzer (und für jeden „höheren" Systemprozeß) einen eigenen Rechner mit Hauptspeicher und Peripherie. Dadurch ist eine sichere vollständige Trennung der verschiedenen Prozesse von vornherein gegeben; zum Beispiel sind alle Adressen virtuell und können sich real nur auf die der virtuellen Maschine gehörigen Speicherbereiche beziehen. Wegen der Simulation der Benutzer-Prozesse durch ein Kontroll-Programm können diese auch niemals die volle CPU-Kontrolle erreichen. Ein Nachteil dieser Methode ist der Verwaltungsaufwand, der etwas höher als bei anderen Systemen liegt. Komplikationen entstehen, weil die verschiedenen Prozesse miteinander kommunizieren müssen.

Für die Ausführung von Betriebssystemprozeduren, etwa Zugriff auf eine Datei, gibt es zwei Modelle, die in Abbildung II-4 dargestellt sind. Im Mehrbenutzerbetrieb müssen insbesondere Systemprozesse von mehreren Benutzern oder Prozessen *gleichzeitig* nutzbar sein. Solche Prozesse heißen **'reentrant'** (ablaufinvariant, simultan nutzbar). Eine besondere Gefahr entsteht, wenn sie gleichzeitig von Benutzern mit verschiedenen Privilegien aufgerufen werden.

Abbildung II-4: Aufruf von Systemprozessen



Der Regelfall ist, daß die Prozedur in den Speicherbereich des Benutzers geladen wird und das Anwendungsprogramm an den Start der Prozedur springt. (Das Anwendungsprogramm kann auch die nackte Betriebssystem-Oberfläche sein.) Dies ist für unkritische Prozeduren eine schnelle und bequeme

Ausführungsart; das Problem der simultanen Nutzung entsteht nicht, da für jeden Anwender eine eigene Kopie des Prozesses erzeugt wird. Zusätzliche spezielle Sicherheitsvorkehrungen lassen sich aber nicht einbauen, da der Benutzer das in seinem Adressbereich stehende Programm beliebig manipulieren kann (es sei denn, die Hardware verhindert dies). Ein gewisser Schutz besteht allenfalls darin, daß längere Maschinencode-Passagen nicht leicht zu entziffern sind. Im zweiten Modell läuft ein Systemprozess ('Server') in einem eigenen Adressbereich permanent (sinnvollerweise mit Mehrprozeß- = 'mulititasking'-Fähigkeit) und wartet auf einen Auftrag ('Request') eines Benutzerprozesses ('Client'). Erhält er einen solchen, prüft er dessen Berechtigung, durchläuft, was auch immer an Sicherheitsvorkehrungen in ihn eingebaut ist, und führt dann die verlangte Prozedur aus, deren Ergebnis er an den Auftraggeber zurückmeldet ('Response'). Für kritische Systemprozeduren ist dies die einzig sichere Art der Ausführung; jede gewünschte Sicherheit läßt sich so verwirklichen. Nach diesem Prinzip werden in der Regel auch Datenbanken verwaltet (**„Server-Requester-Prinzip“**). Diese Art, Systemprozeduren auszuführen, läßt sich besonders leicht mit dem Konzept der virtuellen Maschinen verwirklichen. Eine Variante, die sehr verbreitet ist, ist, Serviceprozesse nicht permanent eigenständig laufen zu lassen, sondern auf Antrag des Benutzers erst speziell für ihn zu starten. Das ist zwar theoretisch genauso sicher, der Überprüfungsprozeß muß jetzt aber im Systemkern ablaufen. In der Praxis schleichen sich hier durch unsaubere Programmierung oft Fehler ein, da der Benutzer doch etwas mehr Einfluß auf den Ablauf hat.

Besondere Probleme bringt der Mehrbenutzerbetrieb auf einem PC mit sich; hier ist natürlich nicht die gleichzeitige mehrfache Benutzung gemeint, die auf Standard-PCs noch nicht möglich ist, sondern einfach die Verwendung des Geräts durch mehrere Personen zu verschiedenen Zeiten. Falls diese Personen unterschiedliche Rechte haben sollen, sind spezielle Sicherheitsmaßnahmen nötig, die im Abschnitt 5 behandelt werden.

4.3 Identifikation und Authentisierung

Mit der **Identifikation** beim Anmeldevorgang sagt der Benutzer, wer er zu sein vorgibt, mit der **Authentisierung** *beweist* er, daß er das auch wirklich ist, in der Regel durch ein Paßwort. Dafür geeignet ist auch ein unveränderliches, eindeutiges persönliches Merkmal wie der Fingerabdruck oder ein Geheimnis, das nur der Benutzer und der Computer kennen. Die drei Prinzipien

wissen (zum Beispiel ein Paßwort),

haben (einen Ausweis oder einen Schlüssel),

sein (nachgewiesen durch biometrische Merkmale)

können einzeln oder kombiniert angewendet werden. Der Aufwand für einen Angriff ist entsprechend zu bewerten; er besteht aus:

- Erlangen des Wissens,
- Entwenden oder Fälschen des Objekts,
- Fälschen der Merkmale,

je nachdem.

Strenggenommen ist die Identifikation in der Authentisierung enthalten; trotzdem ist eine Trennung der beiden Vorgänge von Bedeutung:

- Ein zufällig erratenes Paßwort nützt nur, wenn es auch der richtigen Identität zugeordnet werden kann.
- Benutzer sollten ihr Paßwort selbst wählen können. Wäre die Identifizierung kein getrennter Vorgang, so müßte verhindert werden, daß zwei Benutzer zufällig das gleiche Paßwort wählen. Aus der Ablehnung eines Paßworts könnte ein Benutzer schließen, daß dieses schon existiert.

Der Versuch, sich mit einer falschen Identifikation ins System einzuschleichen, wird als **Maskerade** bezeichnet. In vielen Fällen reicht dazu ein enttarntes Paßwort. Die Authentisierung kann zusätzlich durch eine Kennkarte oder eine Chipkarte abgesichert werden; dieses wurde schon im Abschnitt 3.4 beschrieben. Man darf aber nicht vergessen, daß einfache Magnetkarten leicht zu kopieren sind; Lesegeräte sind billig und leicht erhältlich. Eine interessante Variante, die im militärischen Bereich verwendet wird, ist das Paßwort (besser nur einen Teil davon) auf der Karte abzuspeichern. Bei der Abmeldung wird ein vom Rechner zufällig erzeugtes neues Paßwort auf die Karte geschrieben. Wird die duplizierte Karte vor der echten benützt, trägt die echte dann noch das alte Paßwort. Daher ist das unbemerkte Duplizieren einer Karte zumindest nachträglich feststellbar; wird die duplizierte Karte erst nach der echten verwendet, nützt sie nichts mehr, wird aber trotzdem entdeckt.

Wichtig sind Maßnahmen bei Fehlversuchen zur Authentisierung:

- Alarm,
- Meldung ungültiger Versuche beim nächsten korrekten Logon unter der betroffenen Benutzer-Identität,
- Stillegen des Anschlusses.

Etwas ausführlicher werden diese Maßnahmen im nächsten Abschnitt behandelt.

Eine Alternative zum Paßwortschutz ist ein Erkennungsdialog, in dem der Computer den Benutzer zufällig einige aus einer Liste gewählte Fragen stellt, etwa persönliche Fragen, besser aber einige Paßwörter. Das Verfahren hat sich in der Praxis nicht durchgesetzt, obwohl es bei sorgfältiger Implementierung durchaus Vorteile hat.

Auf jeden Fall kann man vom Betriebssystem verlangen, daß es die Identifikationsprozeduren und die Paßwortpolitik unterstützt, ohne daß man zusätzliche Software einsetzen oder umfangreiche Modifikationen vornehmen muß.

4.4 Paßwörter

Paßwörter sind der klassische Schutzmechanismus und das klassische Sicherheitsrisiko. Es ist sehr sorgfältig zu überlegen, wie die Paßwortpolitik des Systems aussehen soll. Die Handhabung von Paßwörtern ist per Vorschrift zu regeln; wichtigster Punkt:

Niemals ein Paßwort aufschreiben.

Drei Stufen des systematischen Angriffs auf Paßwörter lassen sich unterscheiden:

1. Ziel: *Irgendein Zugang* zum System. Hier verspricht ein **Fischzug** Erfolg. Häufige Methode: Der Angreifer kennt (oder errät) eine Benutzer-Identifikation und probiert (per PC-Programm) Tausende von Paßwörtern. Oft lassen sich Paßwörter aber auch mit wenigen Versuchen erraten, wenn man die Vornamen der nächsten Angehörigen eines Benutzers kennt oder wenn der Systemverwalter zu faul war, die von der Herstellerfirma voreingestellten Paßwörter für Systemprozesse zu ändern. Häufig verwendet der Benutzer XYZ auch die Paßwörter XYZ, ZYX oder XYZXYZ. Technisch aufwendiger ist das Abfangen von Paßwörtern durch Abhören von Kommunikationsleitungen. Allerdings lassen sie sich dann aus den empfangenen Datenmassen ziemlich leicht herausfiltern. Schließlich lassen sich Paßwörter auch durch Erpressung oder Gewalt beschaffen.

2. Ziel: *Privilegierter Zugang* zum System. Einfachste Methode, falls man Zugang zu einem allgemein benutzten Terminal hat oder gar weitere Terminals für einen Prozeß requirieren darf: Ein zugelassener oder unbefugt ins System eingedrungener Benutzer stellt eine **Paßwortfalle** ('spoofing program') auf: Er programmiert einen Prozeß, der genau wie der offizielle Anmeldungsdialog wirkt, und läßt diesen für andere Benutzer stehen. Das dann eingegebe Paßwort wird in eine Datei umgeleitet und die Sitzung mit einer glaubhaften Fehlermeldung abgebrochen oder, falls möglich, der richtige Anmeldevorgang mit den gespeicherten Eingaben nachgeholt. Eine andere Methode bietet sich an, wenn die Paßwortdatei zwar verschlüsselt, aber allgemein zugänglich ist. Hier ist ein „Fischzug“ mit einer „Klartextstück-Attacke“ oft erfolgreich. Mehr dazu später.

3. Ziel: *Völlige Kontrolle* über das System. Dazu braucht der Angreifer eine genügend privilegierte Benutzer-Identifikation samt Paßwort. Die meisten Betriebssysteme bieten ohne zusätzliche Maßnahmen dann kaum noch Schutz.

Mit der Paßwortfalle arbeiteten auch, zumindest in dem zuletzt bekannt gewordenen Fall, die Btx-Hacker und die POS-Hacker, die den Startschuß zum elektronischen Bezahlen in Regensburg verdarben. Der Geschäftsmann, bei dem per

Euroscheck-Karte bezahlt wird, kann die Information auf der Karte und die eingegebene Geheimnummer („Persönliche Identifikationsnummer“, PIN) auf einen PC umleiten und später für unlautere Zwecke mißbrauchen.

Ein Problem besteht darin, daß man sich immer mehr Paßwörter merken muß, einschließlich PINs für Scheckkarten. Die Chancen, in einem Notizbuch einige Paßwörter zu finden, werden für Kriminelle also immer größer, zumal viele davon „von oben“ zugeteilt werden und nicht leicht merkbar sind. Wer kann sich schon ein Zufallspaßwort wie „X19AQ34B“ merken! Im Risks-Digest 9.38 berichtete ein Teilnehmer am 31.10.1989 von einem Werbeschreiben von American Airlines, das für Buchungen über die Kreditkarte warb und für den Fall des Vergessens die zugehörige Geheimnummer gleich mit angab. Sie war also nicht nur an einer unerwarteten Stelle gespeichert, bei einer Firma, die für die Kreditkarte nicht zuständig ist, sondern auch noch, zumindest für die Werbeabteilung, frei zugänglich.

Noch einmal zusammengefaßt: Paßwörter sind gefährdet durch

- Nachlässigkeit des Besitzers,
- ungenügende Schutzvorkehrungen des Systems,
- Abhören von Leitungen, Bildschirmen oder PCs,
- Paßwortfallen,
- systematisches Probieren und Fischzüge.

Hier einige Planungshilfen:

- Paßwörter dürfen bei der Eingabe nicht auf dem Bildschirm erscheinen; dazu setzt etwa das System im Eingabefeld die Schreibfarbe gleich der Hintergrundfarbe.
- Die Anzahl der zulässigen Falscheingaben muß beschränkt werden. Ein bewährtes Verfahren: Ein Fehlversuch ist frei, nach dem zweiten Versuch gibt es einen Alarm, hörbar oder als Meldung an eine Aufsichtsperson, nach dem dritten Versuch werden das Terminal und die Benutzer-Identifikation gesperrt, und zwar so lange, bis der Systemverwalter die Sperre ausdrücklich aufhebt. Bei Fern- und Wählanschlüssen ist diese rigorose Sperre des Anschlusses vielleicht nicht praktikabel, weil es keine festen Terminaladressen gibt. In diesem Fall sollte man wenigstens Zeitsperren einbauen, deren Länge mit jedem Fehlversuch stark zunimmt. Auch die Rückrufmethode kann zusätzlichen Schutz bieten.
- Paßwörter müssen leicht änderbar sein, und zwar vom Benutzer selbst, in Notfällen aber auch vom Systemverwalter. (Im Normalfall geht den Systemverwalter ein fremdes Paßwort nichts an.)

- Der Systemverwalter darf das Paßwortverzeichnis nicht lesen können; er muß jedes beliebige Paßwort ändern können, aber das darf nicht unbemerkt geschehen. Die gängige Methode, dieses zu erreichen, ist die Einweg-Verschlüsselung.

- Der Systemverwalter kann sich auch aus einem Einweg-verschlüsselten Verzeichnis noch zufällige Paßwörter nach der Fischzugmethode verschaffen. Um dies zu erschweren, muß ein nichtmanipulierbares Protokoll über Lesezugriffe mitgeführt werden, das nach dem Vieraugenprinzip ausgewertet wird. Allerdings dürften sich da in jedem Betriebssystem noch Lücken finden.

- Das Paßwortverzeichnis darf auch in verschlüsselter Form für unprivilegierte Benutzer nicht lesbar sein. Gerade hier boten bisherige UNIX-Systeme ein Einfallstor für Hacker; auch der Internet-Wurm nutzte es für einen Fischzug aus, wobei er den offiziellen, absichtlich langsamen Verschlüsselungs-Algorithmus durch eine eigene, schnelle Version ersetzte.

- Ein Paßwort darf niemals der einzige Schutz für kritische Daten sein; ein enttarntes Paßwort darf noch nicht der „Schlüssel zum Königreich“ sein. Zum Beispiel konnte der Internet-Wurm Programme auf weiteren Rechnern starten, da manche Benutzer das dürfen; der fremde Rechner geht einfach davon aus, daß die Authentisierung vom sendenden Rechner gesichert worden ist. Ein Paßwort reichte also, um viele neue Möglichkeiten zu erschließen.

- Paßwortverletzungen müssen protokolliert werden.

- Paßwortänderungen müssen leicht zu erzwingen sein, sowohl gezielt als auch automatisch per Verfallsdatum.

- Zu einfache Paßwörter müssen verhindert werden können, Standards müssen automatisch überwacht werden. Sinnvolle Kriterien:

  - Mindestens fünf Zeichen.
  - Nicht gleich Benutzer-Identität.
  - Nicht gleich Benutzer-Identität rückwärts gelesen.
  - Nicht von der Form „xyzxyz“.
  - Nicht von der Form „xyzzyx“.
  - Nicht in der expliziten Negativliste enthalten.
  - Nicht gleich einem vom selben Benutzer bereits verwendeten Paßwort.

- Paßwörter sollten auf einzelne Personen beschränkt sein; das spricht dagegen, einzelne Dateien durch Paßwörter zu schützen (Sicherungsmaßnahmen für Datenzugriffe sind der Gegenstand des Kapitels III). Ein geteiltes Paßwort ist schon halb verdorben und doppelt so schwer änderbar. Auch soll sich jeder Benutzer möglichst nur ein Paßwort merken müssen.

- Nach Programmvorführungen sind Paßwörter sofort zu ändern.

Das Problem der besonders tückischen Paßwortfalle ist, daß keine sichere Identifizierung des Systems gegenüber dem Benutzer vorgesehen ist. Ein gewisser Schutz wird gewährleistet, wenn es eine feste Zuordnung zwischen Terminals und Benutzergruppen gibt. Insbesondere für privilegierte Funktionen ist dieser Schutz zusätzlich zum Paßwort unbedingt einzuführen. Zukünftige Systeme mit dem Anspruch der Sicherheit werden aber die gegenseitige Identifizierung von Benutzer und Zielsystem verwirklichen müssen.

Damit ein Paßwort nicht der einzige Schutz für ein sensitives System ist, kommen als Zusatzmaßnahmen in Frage:

- physische Zugangssperren,

- logische Terminalsperren,

- Zeitsperren,

- Hardwareschlüssel,

- Ausweiskarten mit einer Information, aus der zusammen mit Benutzernamen und Paßwort der eigentliche Zugangsschlüssel errechnet wird.

Mit diesen Zusatzmaßnahmen werden auch die Gefahren der Paßwortfalle, des systematischen Probierens und des Abhörens entschärft. Man darf aber nicht vergessen, daß Zusatzmaßnahmen eventuell ebenfalls durch Abhören unwirksam gemacht werden können — so kann ein Angreifer etwa die Information, die nach der Überprüfung von biometrischen Merkmalen wie Fingerabdrücken von Terminal ans Zentralsystem gesendet wird, abfangen, speichern und später wiederverwenden.

4.5 Anwendungssoftware

Eigentlicher Sinn des gesamten Datenverarbeitungssystems sind die Anwendungen, die auf ihm durchgeführt werden. An erster Stelle bei den Sicherheitsüberlegungen steht daher die genaue Spezifikation der verwendeten Daten und die Dokumentation der Verfahrensabläufe. An zweiter Stelle steht die Frage, welche Software eingesetzt werden soll. Der Trend der Zeit läuft weg von der selbsterstellten Software und auch von der individuell angefertigten Fremdsoftware hin zur von der Stange gekauften Standard-Software, die allerdings oft noch

einigen Anpassungsaufwand erfordert. Typisch hierfür sind Datenbanksysteme, Compiler und Textverarbeitung.

Auf PC- oder UNIX-Systemen ist oft eine verwirrende Vielfalt von Software aus verschiedenen Quellen implementiert, auch aus dem Public-Domain-Bereich. Für die Sicherheit des Systems ist dies natürlich sehr ungünstig. Erstens sind die einzelnen Komponenten oft nicht gut dokumentiert, zweitens ist ihr Zusammenwirken kaum noch kontrollierbar. Eine typische, wenn auch noch leicht durchschaubare Konstellation ist ein Datenbanksystem und ein Disketten-Inspektionsprogramm, das die Sicherheitsmechanismen des Datenbanksystems locker unterläuft. Verantwortung für die Integrität des Gesamtsystems kann in einer solchen Umgebung niemand ernsthaft übernehmen.

Bei selbsterstellter Software sind Kriterien zur Qualitätssicherung zu beachten, die möglichst unabhängig zu überprüfen sind, also nicht von den Software-Entwicklern selbst:

- Programmierregeln für kritische Anwendungen,
- Programmiersprache und ihre Sicherheitslücken,
- Tests:
  - Formale Verfahrensprüfung,
  - Sachlogische Programmprüfung,
  - Testdaten,
  - Schnittstellenprüfung zwischen Programmteilen,
  - Spezielle Prüfprogramme,
- Freigabe zur Anwendung.

Die Qualitätssicherung muß auch den Einbau Trojanischer Pferde verhindern, die beim konventionellen Testen nicht entdeckt werden. Der Abschnitt 2.4 von [43] behandelt das Thema der Manipulationsmöglichkeiten bei der Software-Erstellung und deren Verhinderung sehr ausführlich. Die EG fördert das Projekt PCTE ('Portable Common Tool Environment') zur Erstellung einer Software-Entwicklungsumgebung, in die auch Sicherheitsanforderungen eingearbeitet werden; es gibt bisher eine Implementierung auf UNIX-Basis.

Bei der individuell erstellten Fremdsoftware ist vor allem die Zuverlässigkeit des Erstellers und die Verteilung der Verantwortung zwischen Ersteller und Anwender zu regeln. Die geringsten Probleme gibt es (hoffentlich!) bei Standard-Software. Zu beachten sind:

- Art und Herkunft der eingesetzten Standard-Software, Vertrauenswürdigkeit des Vertriebs, Fehler- und Manipulationsmöglichkeiten auf dem Vertriebsweg,
- Entscheidung über Anschaffung und Einsatz,

- Fehlermöglichkeiten beim Einspielen der Software und bei der Systemgenerierung und ihre Auswirkungen auf die Sicherheitsfunktionen, Selbsttestmechanismen,
- Anpassung ('Customizing'), Gewährleistung der Sicherheitsfunktionen unabhängig von der Konfiguration,
- Möglichkeiten zur Meldung von Fehlern und Problemen an den Hersteller oder Vertreiber ('Hot Line'?),
- Wartung der Software, Fehlerbehebung, Nebenwirkungen auf die Integrität des Systems, sicherer Wiederanlauf,
- Dokumentation der sicherheitsrelevanten Funktionen,
- Verbreitete Kenntnisse von Sicherheitslücken bei potentiellen Angreifern.

4.6 Benutzerprofil und Benutzer-Oberfläche

Die Frage der Offenheit oder Abgeschlossenheit des Systems stellt sich beim logischen Zugang zum System und seinen Ressourcen genauso wie beim physischen Schutz. In einem geschlossenen System ist nur erlaubt und möglich, was bei der Erfüllung einer bestimmten Aufgabe im Moment benötigt wird.

Die Autorisierung des Benutzers legt fest, welche Rechte er nach einem erfolgreichen Systemzugang hat (Benutzerprofil). Diese sind über die Benutzer-Oberfläche zugänglich. Dazu gehören:

- spezielle Betriebssystem-Versionen,
- eine Startprozedur,
- die verfügbaren Betriebssystem-Funktionen, auch Tastenfunktionen (etwa 'break', 'escape', 'system request', 'print screen'),
- Privilegien,
- die verfügbaren Anwendungsprogramme,
- Voreinstellungen für Parameter und Datei-Zugriffspfade,
- Berechtigungen zum Dateizugriff (Thema von Kapitel III),
- Beziehungen zu anderen Benutzern (Kommunikation, Gruppenzugehörigkeit),
- Ein- und Ausgabemöglichkeiten,
- Verbrauchsrechte.

Anzustreben ist eine sichere Benutzer-Oberfläche mit jederzeit genau spezifiziertem Funktionsumfang. Das ist technisch möglich, entspricht aber nicht der gängigen Praxis. Eine solche Benutzer-Oberfläche ist die beste Absicherung gegen Bedienungsfehler, die zu unvorhergesehenen Zuständen führen, und stellt sicher, daß der Benutzer jederzeit nur im Rahmen seiner Rechte agiert. Da dieser Schutz auch in Fehlersituationen oder im Falle eines absichtlichen Abbruchs ('escape', 'break', „Dreifingergriff") erhalten bleiben muß, muß er ins Betriebssystem integriert sein und kann nicht durch aufgepfropfte Maßnahmen, etwa in Anwendungsprogrammen, erreicht werden. (Allerdings kann ein Anwendungsprogramm in sich eine geschlossene Benutzer-Oberfläche bieten, die so lange gilt, wie der Benutzer nicht ins Betriebssystem aussteigt.) Diese Maßnahme hat sogar den Vorteil, daß sie ausgesprochen benutzerfreundlich ist, solange der Benutzer im Rahmen seiner Aufgabe bleibt. Sie würde allerdings ein gründliches Umdenken bei den Betriebssystem-Herstellern erfordern.

Hergestellt wird die Arbeitsumgebung für den einzelnen Benutzer durch Parameter im Benutzerverzeichnis, vor allem aber durch die Startprozedur ('autoexec', 'profile'). Diese ist zu Sicherheitszwecken nur geeignet, wenn sie vom Benutzer nicht selbst geändert, abgebrochen oder umgangen werden kann.

Wichtig ist auch die Gestaltung der Benutzer-Oberfläche nach ergonomischen Gesichtspunkten. Ein unübersichtlicher Bildschirm provoziert Fehler, ein langweiliger Programmablauf führt zu Schlamperei, etwa wenn wichtige Fehlermeldungen in einem Wust von unwichtiger Information untergehen.

Bedenklich ist, daß Betriebssysteme oft großzügig Auskunft geben. So kann man Konfigurationsdetails abfragen und schnell einen Überblick über verwendete Software bekommen, deren Sicherheitslücken sich dann gegebenenfalls ausnützen lassen. Besonders bedenklich erscheinen Abfragen der angeschlossenen Benutzer ('who', 'query names'); mit solchen Auskünften kann ein Hacker oft etwas anfangen. Er erfährt Benutzeridentitäten, die er mißbrauchen kann – je mehr er davon kennt, desto leichter erwischt er eine mit einem unzulänglichen Paßwort. Und er erfährt auch privilegierte Identitäten und Identitäten von Serviceprozessen. Auch der Internet-Wurm begann auf jedem neuen Rechner damit, Informationen über Rechner, Benutzer und Netzanschlüsse zu sammeln.

Ein Sicherheitsproblem kann auch entstehen, wenn vergangene Eingaben zur bequemen Wiederholung in einem Puffer gespeichert werden, von wo man sie mit einer Funktionstaste abrufen kann ('Retrieve'-Funktion). Solche Puffer sollten zumindest bei Benutzerwechsel geleert werden.

4.7 Überwachung und Beweissicherung

Die beiden wesentlichen Teilaufgaben sind Überwachung gerade ablaufender Vorgänge und Aufzeichnung zur späteren Analyse. Alle sicherheitsrelevanten Vorgänge sind zu überwachen und zu protokollieren (manipulationsgeschützt! – hier eröffnet sich ein geeignetes Anwendungsfeld für einmal beschreibbare optische Platten); die Beweissicherung muß untäuschbar und vollständig sein.

Übeltäter werden dadurch gezwungen, Spuren zu hinterlassen. Aufzeichnungen verhindern zwar Übergriffe nicht, lassen aber erkennen, wo die Sicherheit verletzt ist und in Zukunft bessere Maßnahmen zu treffen sind, und sie gestatten je nach den Umständen des Falls den Schaden wieder rückgängig zu machen. Darüberhinaus erhöhen sie das Vertrauen in das System, indem sie in Zweifelsfällen dokumentieren, daß im Moment die Daten sicher sind.

Protokolliert werden sollten alle An- und Abmeldevorgänge, vor allem falsche Paßworteingaben, Zugriffe auf Dateien, Benutzung von Programmen, Durchführung von Transaktionen, Zugriffen auf Systemtabellen, Änderungen von Systemparametern, Ressourcenverbrauch. Die entsprechenden Schlagwörter sind:

Logging – Aufzeichnung aller Aktionen und Meldungen der Systemkonsole oder eines bestimmten Benutzers, insbesondere von Start und Stop von Untersystemen und Prozessen, und alle Fehlermeldungen.

Auditing – Aufzeichnung von An- und Abmeldevorgängen und Datenzugriffen, natürlich mit Zeitangaben; Aufzeichnung von Transaktionen und Änderungen von Systemparametern und Sicherheitsdefinitionen; Kontrolle, ob festgelegte Regeln eingehalten werden.

Accounting – Aufzeichnung des Ressourcenverbrauchs zum Zwecke der Abrechnung; natürlich lassen sich mit einem solchen System auch mißbräuchliche Zugriffe auf Ressourcen aufdecken.

Monitoring – laufende Überwachung des Ressourcenverbrauchs, um Engpässe zu erkennen und unbefugte Systemaktionen aufzudecken; schließlich läßt sich durch Blockade wichtiger Betriebsmittel (etwa CPU oder Ein- und Ausgabekanäle) das System ganz oder weitgehend lahmlegen. Ein Monitor-System sollte sowohl gezielte Beobachtung einzelner Benutzer und Betriebsmittel erlauben als auch automatisch Meldungen an die Systemkonsole oder an Verantwortliche geben, also ein Alarmsystem enthalten.

Es versteht sich von selbst, daß diese Prozesse und die von ihnen angelegten Daten auf der höchsten Sicherheitsstufe angesiedelt sein müssen. Sie müssen natürlich für Sicherheitsbeauftragte einsehbar sein. Da sich hier leicht riesige Datenberge ansammeln, sind Auswertungsprogramme nützlich, die ihrerseits auch wieder vor Manipulation geschützt werden müssen. Um aus der Datenflut die ungewöhnlichen Vorgänge herauszufiltern, wird in neuester Zeit auch die Expertensystem-Technik eingesetzt [124].

Alle diese Arten von Systemüberwachung konfrontieren uns mit dem gesellschaftspolitischen Problem der Sammlung von personenbezogenen Daten und der Überwachung von Arbeitsabläufen. Aus Gründen des Datenschutzes sollten diese Daten nicht langfristig gesammelt, sondern baldmöglichst gelöscht werden.

Hilfreich sind Prozeduren zur automatischen Auswertung solcher Daten, die ungewöhnliche Zustände und Ereignisse an die Verantwortlichen melden und ihnen (und den Überwachten) das lückenlose Durchlesen ersparen.

Zu den Überwachungsmaßnahmen gehören auch das Timeout für inaktive Terminals oder Kommunikationsverbindungen, ferner Tastatursperren, die nur mit einem Paßwort zu lösen sind, verbunden mit einer Abdunklung des Bildschirms. Es ist unter Umständen praktisch, wenn sich diese Abschaltung vom Benutzer absichtlich aktivieren läßt. Diese Maßnahme ist auch zum Schutz von PC-Sitzungen sinnvoll.

Der Schwachpunkt bei jeder Überwachungsmaßnahme ist der Systemverwalter. Wichtige Daten können vor ihm nur durch Verschlüsselung verborgen werden; ansonsten sind organisatorische Maßnahmen wie das Vieraugenprinzip zur Überwachung unumgänglich. Vor allem muß er an der Manipulation von Überwachungsdateien gehindert werden.

4.8 Viren und andere Schadprogramme

Der Begriff „Virus“ wird oft als Oberbegriff für eine ganze Klasse von Sabotage-Programmen gebraucht, die unbemerkt in ein System kommen und unbeabsichtigt ausgeführt werden. Zu diesem Zweck gibt es eine Reihe von ganz primitiven, aber auch von ziemlich raffinierten Methoden; auch „KI-Viren“, die die Regelbasis eines Expertensystems modifizieren können, wurden schon in die Welt gesetzt. All dies gehört in die Kategorie „manipulierte Software“; die eigentliche ernsthafte Gefahr sind die „Trojanischen Pferde“; alle anderen Formen sind mehr oder weniger intellektuelle Spielerei.

Ein **Virus** (im engeren Sinne) ist ein unselbständiges Programmstück, das sich in andere Programme einschleust und, wenn es irgendwann einmal ausgeführt wird, Kopien von sich selbst erzeugt und diese in noch nicht infizierte Programme injiziert. Soweit ist das nur ein harmloser Spaß. Gefährlich wird es, wenn das Virus neben diesem Reproduktionsteil auch noch einen Wirkteil hat. Besonders geeignet für die Verbreitung sind Netze, aber auch die Verteilung von Software (meistens ‘Public Domain’) über klassische Datenträger wie Disketten. Der Vermehrungsmechanismus kann, wie im Beispiel des Internet-Wurms, nach den Gesetzen des exponentiellen Wachstums sehr schnell zu einer Verstopfung eines Netzes führen, auch wenn das Virus ansonsten harmlos ist. Ist das Virus entdeckt, ist es oft nur schwer unschädlich zu machen, weil man kaum erkennt, wie weit es sich schon verbreitet hat. Viren nützen in der Regel nicht irgendwelche Systemfehler aus. Das ist aber kein Grund zur Beruhigung — ganz im Gegenteil: Viren können ihre schädliche Wirkung auch in fehlerfreien Betriebssystemen entfalten. Zu ihrer Erkennung und Bekämpfung sind zusätzliche Maßnahmen nötig; die Integrität des Systems genügt nicht.

Ein **Trojanisches Pferd** ist ein Programm, das im Innern versteckt Operationen ausführt, also unbemerkte Nebenwirkungen hat, die nicht dokumentiert sind, zum Beispiel Dateien des augenblicklichen Anwenders an den Programmer-

steller schickt. Man sagt auch, daß in dem Programm eine **Falltür** versteckt ist. Besonders gefährlich ist die Kombination mit einem Virus. Ein typisches Trojanisches Pferd ist die Paßwortfalle. Trojanische Pferde nützen oft Systemfehler aus, ja sie bestehen selbst aus einem absichtlich eingebauten Fehler.

Eine **logische Zeitbombe** ist ein Programm (vielleicht in einem Virus oder einem Trojanischen Pferd versteckt), das zu einem bestimmten programmierten Zeitpunkt unerlaubte Aktivitäten entfaltet, etwa Dateien oder Programme löscht. Kombiniert mit einem Virus hat es sich vielleicht schon sehr weit verbreitet, bevor es seine Anwesenheit durch eine zerstörerische Aktion verrät.

Ein **Wurm** schließlich ist ein Typ von Programm, der als Methode der verteilten Datenverarbeitung erfunden wurde [56] und auch für Multiprozessorsysteme geeignet ist. Es handelt sich um ein Programm, das in mehrere „Segmente“ unterteilt ist, über mehrere Systeme verteilt läuft und sich auf einem System, das im Moment nicht ausgelastet ist, auch vervielfältigt und freie Systeme im Netz sucht, die es ebenfalls für die Arbeit einspannen kann. Solch ein Programm muß natürlich sorgfältig kontrolliert werden, damit es nicht ein Netz verstopft. Da der Internet-Wurm nach diesem Schema arbeitete, bekam er seinen Namen.

Alle diese Arten von Programmen sind heimtückisch, weil sie sich unbemerkt einschleichen und vermehren können; die Entdeckung ist ohne besondere Vorkehrungen kaum möglich. Die Anwesenheit eines Trojanischen Pferdes bedeutet, daß das System nicht vollständig unter der Kontrolle seines Besitzers läuft, sondern daß ein früherer Besitzer des Systems oder eines Teils der Software einen Teil der Kontrolle ausübt.

Viren finden ihren Nährboden in offenen Systemen. Das Eindringen ist nicht vollständig zu verhindern. Ein insgesamt sicheres, möglichst geschlossenes System ist aber für Virus-Attacken weniger anfällig. Besondere Schutzmaßnahmen sind nach [57]:

- Zugangsbeschränkungen,
- Schreibschutz wo immer möglich, Verwendung von RO-Medien (PROMs, WORMs),
- Aufzeichnung von Ereignissen, sofortiges Verfolgen ungewöhnlicher Aktivitäten,
- sofortige Quarantäne für infizierte Systeme,
- sofortige Entfernung infizierter Programme,
- strenge Kontrolle der eingeführten Software,
- Originaldatenträger vor Installation mit Schreibschutz versehen, danach sicher verwahren,
- Quarantäne für Software unsicheren Ursprungs, Test auf einem völlig isolierten System,

- Datensicherung mit Aufbewahrung mehrerer Generationen von gesicherten Daten, dabei auch Boot-Sektoren und System-Tabellen nicht vergessen,
- Installation geeigneter Überwachungsprogramme, Anwendung jeweils vor einer Datensicherung.

Seit einiger Zeit gibt es an der Universität Hamburg, Fachbereich Informatik, ein Virus-Test-Center. Dort waren im Dezember 1989 für MS-DOS 15 Viren registriert und 13 weitere in Untersuchung; registriert sind auch Viren für Amiga und Atari. In letzter Zeit häufen sich Berichte, daß sich auf Messen und Ausstellungen Viren ausbreiten. Ein Beitrag von K. Brunnstein im Risks-Digest 10.13 ist diesem Thema gewidmet.

Einige drastische Beispiele dafür, was Insider durch das Einpflanzen von Trojanischen Pferden in Programme anrichten können, stammen aus [43]:

„Eine Funktion, die digitale Unterschriften verifiziert, könnte eine Hintertür enthalten, durch welche die angebliche Unterschrift, die zu einem bestimmten Namen gehört, immer als ‚korrekt' gemeldet wird, auch wenn sie gefälscht ist. Das Verfahren zum Beweis der Integrität von Partnern und Daten mag vollständig und korrekt durchgeführt werden, es übersieht dennoch Integritätsverletzungen bei Sendungen von diesem Namen, solange eine derart manipulierte Funktion verwendet wird. Und daher wäre das Verfahren insgesamt wertlos."

„Eine Funktion zur Verschlüsselung vertraulicher Texte könnte als Nebeneffekt, bevor sie einen Text verschlüsselt, zusätzlich und heimlich den Klartext einem unautorisierten Leser zustellen. Dann wäre die Vertraulichkeit der Nachricht verloren, obgleich das Verfahren zum Schutz der Vertraulichkeit vollständig durchgeführt werden würde."

„Ein Message Transfer Agent ist eine Betriebskomponente in einem Nachrichtenvermittlungssystem, welche für die Weitervermittlung von Nachrichten im Netz verantwortlich ist. Wenn ein Message Transfer Agent derart manipuliert ist, daß er zwar seine Vermittlungsfunktionen korrekt ausführt, aber zusätzlich alle Absender- und Empfangsadressen einem interessierten Spion meldet, dann würden alle Verfahren zum Schutz vor unerlaubter Verkehrskontrolle unterlaufen werden."

Schutz vor Trojanischen Pferden kann nur eine optimal kontrollierte Software-Entwicklung bieten.

4.9 Fehlersituationen

Besonders kritisch für die Sicherheit eines Betriebssystems sind Fehlersituationen, erstens weil sie oft zu unvorhergesehenen Zuständen führen, zweitens weil während ihrer Behebung Sicherheitsmaßnahmen außer Kraft gesetzt sein können. Wichtig sind die Sperrung von Zugängen und Zugriffen bis zum Wiederanlauf, also die ununterbrochene Integrität der Zugangskontrolle, und geeignete Selbstprüfungsmechanismen beim Wiederanlauf. Viele Systeme produzieren

beim Absturz einen Dump, also einen vollständigen Speicher- und Registerauszug auf Papier, Magnetband oder einem temporären Plattenbereich. Dieser kann hochsensitive Daten enthalten, so daß hier ein erhöhtes Maß an Kontrolle nötig ist. Umgekehrt besteht bei einem Systemabsturz die Gefahr, daß gerade die Protokoll-Dateien, als offene, nicht auf Platte zurückgeschriebene Dateien verloren gehen, so daß man Sicherheitsverstöße, die kurz vor dem Ereignis stattgefunden oder es gar ausgelöst haben, nicht rekonstruieren kann.

Der Erfolg des Internet-Wurms beruhte im wesentlichen auf zwei Systemfehlern; der erste, der das Programm *fingerd* betrifft, wurde schon besprochen. Der zweite war eher eine Nachlässigkeit der Systemverwalter: Das Programm *sendmail* hat einen 'Debug Mode', der bei der Installation dazu dient, Tests auszuführen, ohne tatsächlich Post über das Netz zu schicken. Er erlaubt auch einige privilegierte Kommandos. Bei vielen Systemen wird nach Abschluß der Testphase einfach vergessen, zum Normalmodus zurückzukehren.

Fehlersituationen in einem Betriebssystem sollten möglichst vermieden werden; die geeigneten Maßnahmen sind Software-Engineering und Programmverifikation – das Problem liegt natürlich beim Hersteller. Die zur Erstellung sicherer Software wichtigsten Prinzipien sind:

- Komplexitätsreduzierung durch Zerlegung und Schichtung von großen Programmsystemen in überschaubare Einheiten,
- Modularisierung mit sauberen und möglichst schmalen Schnittstellen,
- Verbergen unnützer Information ('information hiding'), also Verhinderung von unbeabsichtigten oder nicht durch eine Schnittstellendefinition ausdrücklich erlaubten Zugriffen,
- abstrakte Datentypen indexabstrakter Datentyp und Datenobjekte mit genau definierten Zugriffsprozeduren.

Ziel ist der Entwurf einer geeigneten Systemarchitektur mit einem relativ kleinen, aber besonders zuverlässigen Sicherheitskern. In einem geschlossenen System ist die Komplexität deutlich reduziert. Viele Fehler können überhaupt nicht auftreten. Die Fälle unvorhergesehener Benutzeraktionen brauchen bei der Validierung des Systems nicht berücksichtigt zu werden; sie sind für formale Verifikationsverfahren sowieso unzugänglich.

Zur Programmverifikation gibt es Ansätze, aber keine allgemeingültige Lösung. Eine solche würde voraussetzen, daß das zu verifizierende Programmsystem formal vollständig spezifiziert ist; bei größeren Systemen ist das (prinzipiell?) nicht machbar. Durch eine Art von mathematischen Beweis soll dann die korrekte Funktion der erstellten Programme nachgewiesen werden. Selbst dann – jeder Mathematiker weiß, daß die Korrektheit eines Beweises, vielmehr aber noch die Adäquatheit eines mathematischen Modells durch nichts garantiert werden kann. (Das mathematische Modell steckt implizit oder explizit in der Spezifikation des Systems.) Verifikation sollte wo immer möglich durchgeführt

werden, ersetzt aber das Testen nicht. (Man stelle sich vor, man sitzt in einem Flugzeug, dessen Konstruktion zwar verifiziert, das aber noch nicht getestet ist.)

Aber auch mit einer völlig korrekten Spezifikation und Implementation in einer geeigneten Programmiersprache ist noch nicht gewährleistet, daß ein Betriebssystem einwandfrei und ohne Sicherheitslücken funktioniert. Korrektheitprüfende Compiler sollen Programmierfehler verhindern; insbesondere die Feldgrenzenüberwachung ist sehr wichtig, wie bereits demonstriert. Durch gezielt nicht-korrekte Programmierung lassen sich Falltüren in Programme einbauen; ein geschickter Systemprogrammierer wird so zu einem besonders heimtückischen Angreifer. Besonders leicht sind Manipulationen in Assembler möglich, aber auch C bietet gute Möglichkeiten. Eine Methode ist etwa, Programmteile als Datensegmente zu behandeln und sie mit direkten Speicherzugriffen abzuändern.

... und dann bleibt immer noch der 'Debug Mode' oder die voreingestellte Abschaltung von Sicherheitsmaßnahmen für die Installationsphase, die irgend jemand vergessen hat, rückgängig zu machen.

Etwas anders liegt der Fall, wo der Compiler selbst fehlerhaft ist. Wie leicht man mit einem unkorrekten Compiler ein ganzes Betriebssystem korrumpieren kann, hat THOMPSON, einer der Väter von UNIX, in [133] sehr drastisch beschrieben; natürlich ist das vor allem wieder in C möglich, aber auch sonst ein Problem. Da ein C-Compiler bis auf einen kleinen Kern selbst in C programmiert ist, ist es leicht, ihn in die gewünschte Richtung zu verbiegen, etwa ein Trojanisches Pferd einzubauen, das dann in jedem mit diesem Compiler erstellten Programm schlummert. Interessanterweise ist die Software für die französischen Kernkraftwerke hauptsächlich in C programmiert, wie im Risks-Digest 9.52 beschrieben.

Für eine abstrakte Theorie sicherer Systeme und die abstrakten Grundlagen der Programmverifikation sei auf [30] verwiesen.

5 Spezielle PC-Probleme

In einer Großrechner-Umgebung sind die wichtigen Systemkomponenten in der Regel physisch geschützt; auf dieser Grundlage lassen sich logischer Zugang zum System und Zugriff auf Daten wirksam durch das Betriebssystem überwachen. Dagegen steht den Vorteilen des individuellen Arbeitsplatzrechners ein erhöhtes Datensicherheitsrisiko gegenüber. PCs lassen sich so herrlich unbürokratisch anschaffen und benutzen. Der Datenschutzanspruch macht dem ein Ende: Zumindest etwas Bürokratie muß sein.

Wenn ein PC alleinstehend im „Privatbesitz“ ist und in einem nicht frei zugänglichen Raum aufgestellt ist, heißt die wichtigste Sicherheitsmaßnahme:

Rechner abschließen, Disketten wegschließen, Raum zuschließen.

Wenn diese Regel beachtet wird, wird an einem isolierten Arbeitsplatz das Problem des Datenschutzes zwar nicht gelöst, aber durch das Aufstellen des PC auch nicht allzusehr verschärft. Viele Probleme bleiben aber auch in diesem Fall, etwa das Einschleppen eines Virus mit einer Diskette obskurer Herkunft oder Mangel an brauchbaren Sicherungskopien im Katastrophenfall. Und der Trend geht dahin, die freien, alleinstehenden PCs in Netze zu verstricken und so ihrer Freiheit zu berauben. Damit bekommt das Sicherheitsproblem plötzlich eine ganz andere Größenordnung.

5.1 Sicherheitsprobleme im PC-Bereich

Zunächst zwei Beispiele, wie „offen“ PCs tatsächlich sind:

Beispiel 1: Löschen von Daten. Gelöschte Dateien sind nur im Verzeichnis als gelöscht markiert. In Wirklichkeit stehen die Daten noch da, bis sie irgendwann von anderen Schreibvorgängen überschrieben werden. Bei Festplatten (im Gegensatz zu Disketten) werden die Daten nicht einmal beim gewöhnlichen Formatieren gelöscht. Das Datenschutzgebot „Aufbereitung von Datenträgern zur Wiederverwendung“ ist verletzt. Mit einem Disketten-Monitor wie etwa den „Norton Utilities“ kann man diese Daten leicht sehen, oft sogar ganze gelöschte Dateien wiederherstellen.

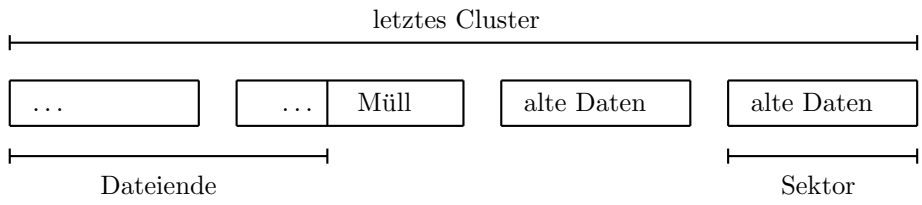
Beispiel 2: Schreiben von Daten. Beim Formatieren wird die Platte oder Diskette in Sektoren eingeteilt, die normalerweise 512 Bytes groß sind; das Betriebssystem MS-DOS bzw. PC-DOS faßt zur Dateiverwaltung je 1 bis 8 Sektoren zu einem ‘Cluster’ zusammen, auf einer Festplatte üblicherweise 4. Jeder Datei wird eine ganze Anzahl von Clustern zugeordnet; falls die Länge der Datei nicht ein ganzzahliges Vielfaches der Clusterlänge ist, bleibt also Platz übrig. Physikalisch werden aber immer ganze Sektoren beschrieben. Der freie Raum hinter dem Ende einer Datei bis zum Ende des Sektors wird also mit Daten vollgeschrieben, die zufällig in einem internen Puffer stehen – das können durchaus Daten sein, die Sie eigentlich geheimhalten wollten, siehe Abbildung II-5. Eventuell übrige Sektoren im Cluster behalten ihren alten Inhalt. Auch diese Daten kann man mit einem Disketten-Monitor leicht sehen.

Gegenüber großen Datenverarbeitungsanlagen in Rechenzentren tritt in einer typischen PC-Umgebung eine Reihe von zusätzlichen organisatorischen und technischen Problemen auf. Es folgt eine Liste solcher Probleme, die einen alleinstehenden PC betreffen, der an einem mehr oder weniger zugänglichen Ort aufgestellt ist; das kann ein PC-Saal in einer Universität, eine Arzt-Praxis, ein kleines Büro oder eine Amtsstube sein. Der zugelassene Benutzerkreis ist in der Regel klein, wenn man vom genannten PC-Saal absieht.

- Die Verantwortung ist kaum geregelt, das Sicherheitsbewußtsein unterentwickelt.

- Es gibt keine geschulte Bedienungsmannschaft.

Abbildung II-5: Datenmüll am Dateiende



- Es gibt keine organisatorische Trennung von Systemverwaltung, Bedienung, Programmierung und Anwendung — der Anwender ist Auftraggeber, Programmierer, Operator, Archivar, ... in einer Person; das Vieraugenprinzip entfällt.
- Systemfunktionen werden von irgend jemand nebenbei ausgeübt.
- Die Benutzer sind unbedarft, in Fehlersituationen hilflos und scheuen Sicherheitsmaßnahmen.
- Arbeitsvorgänge werden nicht überprüft oder überwacht.
- Die Geräte sind oft unbewacht.
- Die Geräte sind wegtragbar. Ein Angreifer, der einen PC samt Festplatte geklaut hat, hat dann sehr viel Zeit, um vorhandene Schutzmechanismen zu studieren und zu knacken.
- Das Innenleben der Geräte ist leicht zugänglich. Zum Beispiel lassen sich leicht Abhöreinrichtungen („Wanzen“) auf Lötkontakte klemmen, mit denen man Paß- und Schlüsselwörter abhören und somit jeden Softwareschutz unterlaufen kann. (Denken Sie auch an den Wartungsdienst – wer hat schon die Zeit, einem Techniker beim Austausch einer Festplatte eine halbe Stunde lang auf die Finger zu schauen.)
- Ein wirksamer physischer Schutz ist im Vergleich zu den geringen Gerätekosten vergleichsweise teuer und daher nicht wirtschaftlich.
- Das Betriebssystem ist offen konzipiert und bietet keine Schutzfunktionen. Es gibt wirksame Programme zur Untersuchung von Hauptspeicher und Massenspeichern, mit denen der Zugriff bis zum letzten Bit möglich ist und Schutzmaßnahmen leicht zu unterlaufen sind.

- Es gibt eine Reihe von undokumentierten Systemfunktionen, die von Schadprogrammen ausgenützt werden können. Die erwähnten Beispiele — Löschen von Daten und Datenmüll am Dateiende — gehören auch in diese Kategorie und können selbst schon als Trojanische Pferde angesehen werden.

- Daten sind leicht auf Disketten zu kopieren und so wegtransportierbar („Datenentnahmestation“). Auf diskettenlosen Arbeitsplätzen bietet sich als Ersatz an, die Daten auf den Druckerausgang umzulenken. Auch die Hardcopy-Taste (‘PrtSc’) kann zu diesem Zweck dienen, und ihre Verwendung ist kaum zu kontrollieren.

- Festplattendaten sind nur softwaremäßig gegen Überschreiben zu schützen.

- Verschlüsselung von Daten auf Softwarebasis ist zu langsam und hindert daher bei der Arbeit.

- Die Anfertigung von Sicherungskopien ist umständlich.

- Gute Systemkenntnisse sind bei möglichen Angreifern weit verbreitet.

- Man kann leicht unsichtbare Dateien erzeugen; hier tritt das Problem der verdeckten Datenkanäle auf, also von Daten und Datenbewegungen, die vom Betriebssystem aus nicht sichtbar sind. Vergleiche dazu auch Risks-Digest 9.55.

- Es lassen sich auf der Festplatte Sektoren als unbrauchbar markieren und dort gefährliche Programme verstecken, die bei Gelegenheit aktiviert werden können.

- Sogar die Tastatur-Codes lassen sich für unseriöse Zwecke umlenken.

Der einzige von den Herstellern vorgesehene Schutz ist das Schloß, das sich an PCs ab der AT-Klasse befindet. Es hält einem Angreifer, der einen Schraubenzieher festhalten kann, allerdings nicht allzu lange stand. Je mehr Benutzer auf dem PC arbeiten, desto unwahrscheinlicher ist es, daß er regelmäßig abgeschlossen wird, falls überhaupt für jeden ein Schlüssel da ist.

Die Hauptangriffe auf einen alleinstehenden PC sind also:

- Festplatte entwenden und lesen (oder ganzen PC entwenden).

- PC mit eigenem Betriebssystem von einer Diskette booten.

- In laufende (verlassene) PC-Sitzung einsteigen (führt zu unbeschränktem Systemzugriff).

Ist der PC in ein Netz eingebunden oder hat er Anschlußmöglickeiten an Großrechner, so ist er nicht nur selbst gefährdet, sondern wird auch umgekehrt zur Gefahrenquelle, wie viele Hacker-Vorfälle gezeigt haben. Eine dieser Gefahren ist die Funktion als „intelligentes Terminal“; der Zielrechner merkt keinen Unterschied zu einem gewöhnlichen „dummen“ Terminal. Kommunikationsprogramme erlauben in der Regel, Tastatureingaben durch Programme zu simulieren. So steht einem blitzschnellen Durchprobieren von Tausenden von Paßwörtern zunächst nichts im Wege. Eine Gefahr durch berechtigte Benutzer ist, daß sie sich lästige Anmeldeprozeduren automatisieren und dabei auch die Paßwörter mit ins Programm schreiben. Der ersten Gefahr läßt sich mit Maßnahmen auf dem Zielrechner begegnen, gegen die zweite Gefahr helfen nur organisatorische Maßnahmen – Steigerung des Sicherheitsbewußtseins, Dienstvorschriften, Kontrolle.

Weitere Gefahren drohen durch die Schwemme der *tragbaren Computer*. Sollen Firmenvertreter abends ihre Daten in die Zentrale übertragen, müssen notwendig Verbindungen über öffentliche Netze mit all ihren Unsicherheiten zugelassen werden. Hacker können von Telefonzellen aus operieren und sind so zusätzlich vor Entdeckung geschützt. Und schließlich sind solche Rechner weitgehend der Kontrolle eines Datenschutzbeauftragten entzogen.

5.2 Sicherheitsprodukte

Es gibt eine Reihe von Produkten auf dem Markt, die je nach organisatorischer Umgebung, physischen Schutzmöglichkeiten und Schutzbedürftigkeit der Daten die Datensicherheit auf einem PC wesentlich verbessern können. Sie bestehen teils aus Hardwarekomponenten, teils aus Software; am sichersten ist die Kombination von beidem. Der Schutz durch Sicherheitshardware wurde schon in Abschnitt 3.4 erwähnt.

Die einfachste Möglichkeit ist ein Paßwortschutz für die Festplatte oder einzelne Teile (Dateien, Programme) davon. Die einfachste Umgehung ist das Laden eines eigenen Betriebssystems von einer Diskette. (PCs ohne Diskettenlaufwerk sind nur in Netzen zu gebrauchen.) In Fällen, wo Daten nicht gegen äußere Angriffe, sondern nur gegen Fahrlässigkeit oder Bedienungsfehler zu schützen sind, kann ein solcher Schutz durchaus ausreichend sein.

Als nächste Idee bietet sich eine Verschlüsselung von Daten auf der Festplatte an. Das ist zeitraubend, wenn es per Software realisiert wird – und gerade die wichtigen Daten sind die, die man ständig braucht. Wird es dagegen von einem Hardwarezusatz erledigt, ist es teuer. Auch sind nicht alle auf dem Markt angebotenen Verschlüsselungsverfahren so einbruchsicher, wie es der Laie glaubt. Dazu kommt, daß man sich ein Schlüsselwort merken muß (sonst taugt das Verschlüsselungsverfahren garantiert nichts), und damit beginnen die gleichen Probleme wie mit den leidigen Paßwörtern. Verschlüsselung schützt nicht vor Zerstörung der Daten. Wie ein brauchbares Verschlüsselungsverfahren aussieht, ist das Thema eines Extrakapitels V. Für den Zeitpunkt der Verschlüsselung

gibt es zwei Konzepte: Das einfachere ist, daß Dateien vor der Bearbeitung entschlüsselt und nach der Bearbeitung wieder verschlüsselt werden. Das behindert bei der eigentlichen Arbeit dann zwar nicht mehr, muß aber in der Regel von Hand ausgeführt werden, so daß sich der Arbeitsbeginn verzögert, und am Ende steht die Gefahr, daß man das Wiederverschlüsseln vergißt. Eine andere Lösung ist, daß die Daten während der Bearbeitung, also auf dem Weg zwischen Platte und Hauptspeicher ent- oder verschlüsselt werden ('online'). Wird per Software verschlüsselt, läßt sich dies mit Hilfe eines Gerätetreibers realisieren.

Trotz aller Vorbehalte können auch einfache Verschlüsselungsprogramme wertvoll sein, wenn man nicht mit professionellen Angreifern rechnen muß. Sie schützen davor, daß geschützte Daten versehentlich offengelegt werden, etwa wenn eine Diskette auf dem Transport verloren geht. Auch die Festplatte eines PCs kann im Falle eines Hardware-Diebstahls so durchaus ausreichend geschützt sein. Verschlüsselte Programme sind übrigens auch gegen die Einnistung von Viren gefeit; ein Virus kann sich zwar ins Programm kopieren, bei der Entschlüsselung wird es aber in „Bytesalat“ verwandelt.

Ohne Verschlüsselung ist echter Datenschutz auf dem PC nicht möglich, da nicht verhindert werden kann, daß ein Anwendungsprogramm volle Kontrolle über die CPU bekommt und somit auf der untersten Maschinenebene, noch unterhalb des Betriebssystems, alle Ein- und Ausgabemedien ansprechen kann. Eine geschlossene Benutzer-Oberfläche, die in jedem Zustand nur genau spezifizierte Operationen zuläßt, ist auf solchen Geräten nicht perfekt zu verwirklichen.

Ein weiterer Typ von Schutzprogrammen soll vor unbefugten Datenveränderungen schützen; diese werden meist in der Kategorie „Anti-Virus-Programme“ verkauft. Das Prinzip ist, Daten und Programme, auch die Systemspuren auf der Festplatte, regelmäßig auf Änderungen zu untersuchen, wobei meistens eine Prüfsumme mit einem Sollwert verglichen wird. Dieses Verfahren kann natürlich nur wirken, wenn der Angreifer es nicht so gut kennt, daß er es austricksen kann. Andere Programme, die zum Virenschutz angeboten werden, suchen nach bekannten Viren oder verhindern deren Verbreitung. Gegen neue Viren helfen sie natürlich nicht.

Schutz vor Einbringen von nicht genehmigter Software und vor allzu leichtem Kopieren und Wegtragen von Daten bieten auch diskettenlose PCs im Netz, die aber wieder als Netzstationen anderen Gefahren ausgesetzt sind und auch nicht das Umlenken von Ausgabedaten auf einen Drucker verhindern. Die umgekehrte Idee liegt den auswechselbaren Festplatten zugrunde; hier sind die Daten besonders leicht zu entfernen, etwa um sie sicher zu verschließen. Welche dieser Einrichtungen sinnvoll sind, muß in der Situation des Einzelfalls entschieden werden.

Zu einem optimalen Schutz durch Kombination dieser Maßnahmen gehört auch eine Absicherung der Kompetenzenaufteilung. Benutzerverwaltung, Kopien auf Diskette, Formatieren der Festplatte, Änderungen an der Systemkonfiguration, etwa die Einstellung der Zeit, und andere sicherheitsrelevante Prozeduren sind zu verhindern, wenn sie nicht von besonders berechtigten Personen

vorgenommen werden. Drucker, Diskettenlaufwerke und andere Peripheriegeräte können gezielt für einzelne Benutzer gesperrt werden. Auch andere auf Software basierenden Schutzmaßnahmen aus dem Großrechnerbereich lassen sich so auf den PC übertragen; sogar das Server-Prinzip läßt sich durch Einsteckkarten mit eigenem Prozessor verwirklichen. Gute Produkte, die dieses leisten, kosten zur Zeit einiges über 1 Kilomark und einige Mühe für die Einarbeitung und laufende Verwaltung. Allerdings wird so ein beträchtliches Sicherheitsniveau erreicht. Eine Produktübersicht ist in Anhang B wiedergegeben.

6 Offizielle Bewertungskriterien

Die Sicherheit von Datenverarbeitungssystemen kann nur zuverlässig und auf vergleichbare Weise geprüft werden, wenn standardisierte Kriterien definiert sind. Solche Kriterienkataloge gibt es seit einiger Zeit in den USA, seit kurzem auch in der Bundesrepublik Deutschland. Ziel ist es, datenverarbeitenden Stellen und Anwendern Maßstäbe zur Verfügung zu stellen, mit denen die „Vertrauenswürdigkeit“ von Rechnersystemen in Hinblick auf Datenschutz und sichere Informationsverarbeitung beurteilt werden kann. Es sollte so auf lange Sicht überflüssig werden, daß jeder Systembetreiber die letzten Winkel seines Systems kennen muß, um wirksame Schutzmaßnahmen durchführen zu können.

6.1 Das amerikanische ‘Orange Book’

Der erste offizielle Kriterienkatalog zur Beurteilung der Sicherheit von Datenverarbeitungssystemen wurde vom ‘Computer Security Center’ des amerikanischen Verteidigungsministeriums entwickelt und 1983 als ‘Department of Defense Trusted Computer System Evaluation Criteria’ veröffentlicht; diese Veröffentlichung wird auch als ‘Orange Book’ bezeichnet. Aufgrund der Kriterien kann eine formale Produktbewertung durchgeführt werden, die zur Aufnahme in die ‘Evaluated Products List’ führt.

Das Ziel ist die Hebung des Sicherheitsstandards von kommerziellen Systemen (in der Regel Hardware und Betriebssystem). Die Betreiber der Anlagen sollen von der Entwicklung eigener Sicherheitssysteme entlastet und bei der Erstellung von Ausschreibungen unterstützt werden.

Das ‘Orange Book’ ist keine leichte Lektüre. Die wichtigsten Aspekte werden hier skizziert; für eine detaillierte Beschreibung siehe [19]. Für die Beurteilung der Sicherheit eines Produkts werden zunächst sechs *Hauptkriterien* formuliert, von denen sich vier direkt auf die Zugriffssicherheit und zwei auf die Vertrauenswürdigkeit der Implementation beziehen:

Sicherheitspolitik ('security policy'): Das Produkt muß eine klar definierte Sicherheitspolitik unterstützen und durchsetzen. Insbesondere sind zwei Arten von Regeln für den Zugriff zu unterstützen:

- *festgelegter Zugriff* ('mandatory access') – also Zugriff auf Grund eines festgelegten „Sensitivitätsgrades",
- *benutzerbestimmbarer Zugriff* ('discretionary access') – also Zugriffserlaubnis durch den Eigentümer der jeweiligen Daten.

Kennzeichnung ('marking'): Subjekte und Objekte müssen mit ihrem „Sensitivitätsgrad" gekennzeichnet werden können.

Identifizierung ('identification'): Subjekte müssen zuverlässig identifiziert werden können; die Angaben über Identifikation und Rechte müssen im System sicher gehalten werden können.

Beweissicherung ('accountability'): Protokolle von sicherheitsrelevanten Aktionen müssen manipulationsgeschützt gespeichert werden können.

Funktionsgarantie ('assurance'): Die entsprechenden Teile des Betriebssystems müssen sorgfältig analysiert und getestet sein.

Funktionsschutz ('continuous protection'): Die Sicherheitsmechanismen müssen vor Manipulationen geschützt sein.

Die Gesamtheit der Sicherheitsmechanismen innerhalb eines Produkts wird unter dem Begriff **vertrauenswürdige Rechenbasis** ('Trusted Computing Base' – TCB) zusammengefaßt.

Die Bewertungskriterien werden in sieben Klassen abgestuft zusammengefaßt; diese sind hierarchisch angeordnet, das heißt, die Anforderungen einer Klasse umfassen die der niedrigeren Klassen. Diese Klassen sind:

Klasse A	verifiziertes Design
Klasse B3	Schutz durch Sicherheitsdomänen
Klasse B2	Schutz durch Strukturierung
Klasse B1	Schutz durch Kennzeichen
Klasse C2	Schutz durch kontrollierten Zugriff
Klasse C1	benutzerbestimmbarer Zugriffsschutz
Klasse D	minimaler Schutz

Die Kriterien der Klasse C1 werden von praktisch allen Großrechner- oder Netzbetriebssystemen erfüllt. IBM-/370-Systeme unter MVS oder VM mit dem Sicherheitssystem RACF sind zur Zeit in C2 eingestuft; gleiches gilt auch für VM mit VMSECURE (das zu RACF funktional weitgehend äquivalent ist). Das Bewertungsverfahren für die Stufe B1 läuft, ebenso für die UNIX-Version AIX der IBM. Das UNIX-System V/MLS von AT&T hat im September 1989 den

Rang B1 erhalten. Verschiedene Hersteller arbeiten an UNIX-Systemen, die den Rang B2 erreichen sollen. Von der X/Open-Gruppe wird die Definition eines C2-UNIX-Standards vorbereitet.

Die Kritik am 'Orange Book' setzt an mehreren Stellen an. Zum einen sind die Kriterien sehr von militärischen Gesichtspunkten beeinflußt und in Wirtschaft, öffentlicher Verwaltung und Forschung nicht ohne weiteres anwendbar (sagen die Kritiker). Funktionale und qualitative Kriterien sind gekoppelt. Die Ausrichtung der Kriterien auf Betriebssysteme erschwert die Beurteilung von Gesamtsystemen mit breitem Anwendungsspektrum und von einzelnen Systemkomponenten. Selbstverständlich wird an Verbesserungen und Ergänzungen gearbeitet.

6.2 Das deutsche Grünbuch

Ein ähnlicher Kriterienkatalog, aber weiterentwickelt und verbessert, wurde im Auftrag der deutschen Bundesregierung von der Zentralstelle für Sicherheit in der Informationstechnik (ZSI) (der früheren Zentralstelle für das Chiffrierwesen) erarbeitet; Nachfolgebehörde dieser Institution ist seit Januar 1991 das Bundesamt für Sicherheit in der Informationstechnik. Der Katalog jedenfalls wurde 1989 unter dem Namen „IT-Sicherheitskriterien“ veröffentlicht [145]; „IT“ ist dabei die Abkürzung für „Informationstechnik“. Man bezeichnet das Werk auch als „Grünbuch“. Auch hier ist ein formaler Bewertungsprozeß vorgesehen, der zu einem amtlichen Zertifikat führt. Eine zweite Schrift erschien als „IT-Evaluationshandbuch“ [146]. Beide sind im Bundesanzeiger-Verlag erschienen und kosten je ungefähr 10 DM. Da vor allem das zweite sehr leicht verständliche Beispiele enthält, sind sie als Lektüre für Sicherheitsverantwortliche durchaus zu empfehlen.

Im Gegensatz zum amerikanischen Modell ist das deutsche flexibler, und es hat ein zweidimensionales Bewertungsschema (Funktionalität und Qualität). Die Grundzüge werden hier nach [65] zitiert. Die abzudeckenden *Grundfunktionen der Informationssicherheit* sind:

Identifikation und Authentisierung – etwa durch Besitztum, Wissen, Merkmale.

Rechteverwaltung – Vollständigkeit, Widerspruchsfreiheit, Überschaubarkeit, ...

Rechteprüfung

Beweissicherung – Untäuschbarkeit, Vollständigkeit.

Wiederaufbereitung – insbesondere Löschen von Speichern vor Weitervergabe.

Fehlerüberbrückung

Gewährleistung der Funktionalität

Übertragungssicherung – in Anlehnung an das 'Security Addendum' des OSI-Modells sind das die Funktionen:

- *Authentisierung auf Partnerebene* ('Peer Entity Authentication') – Sicherstellung, daß während einer Datenübertragung auch tatsächlich die gewünschten Partner miteinander kommunizieren.
- *Zugriffskontrolle* ('Access Control')
- *Vertraulichkeit von Daten* ('Data Confidentiality')
- *Integrität von Daten* ('Data Integrity')
- *Authentisierung des Senders* ('Data Origin Authentication')
- *Anerkennung von Daten* ('Non-repudiation') – Möglichkeit für den Empfänger eines Datenstroms zu beweisen, daß dieser ihm von dem angegebenen Sender übersandt worden ist, sowie umgekehrt die Möglichkeit für den Sender zu beweisen, daß der Adressat die Daten auch in Empfang genommen hat.

Einige dieser Funktionen sind nur mit kryptographischen Techniken zu verwirklichen; diese werden im Kapitel V behandelt.

Aus diesen Grundfunktionen werden zehn *Funktionalitätsklassen* abgeleitet. Die ersten fünf davon (F1-F5) entsprechen dem 'Orange Book' und sind hierarchisch geordnet. Die übrigen fünf (F6-F10) sind voneinander unabhängig; F9 und F10 erfordern zwingend kryptographische Protokolle.

F1 – entspricht 'Orange Book' C1: Benutzerbestimmbarer Zugriffsschutz.

F2 – entspricht 'Orange Book' C2: Mechanismen zur Protokollierung.

F3 – entspricht 'Orange Book' B1: Festgelegter Zugriffsschutz.

F4 – entspricht 'Orange Book' B2: Vertrauenswürdiger Zugriffspfad.

F5 – entspricht 'Orange Book' B3/A: Überwachung sicherheitskritischer Ereignisse.

F6 – bezieht sich speziell auf Systeme mit hohen Anforderungen an die Datenintegrität (etwa Datenbanken).

F7 – bezieht sich auf Anforderungen an die Systemverfügbarkeit (etwa bei Prozeßrechnern).

F8 – bezieht sich auf die Integrität von Daten bei der Datenübertragung.

F9 – bezieht sich auf die Geheimhaltung von Daten bei der Datenübertragung.

F10 – bezieht sich auf Vertraulichkeit und Integrität von Daten in vernetzten Systemen.

Die Sicherheitsfunktionen sollen darüber hinaus nach ihrer Qualität beurteilt werden. Zu diesem Zweck gibt es acht formale *Qualitätsstufen*:

Q0 – unzureichende Qualität.

Q1 – getestet.

Q2 – methodisch getestet.

Q3 – methodisch getestet und teilanalysiert.

Q4 – informell analysiert.

Q5 – semiformal analysiert.

Q6 – formal analysiert.

Q7 – formal verifiziert.

Im 'Orange Book' sind Funktionalitätsklassen und Qualitätsstufen vermengt. Die folgende Tabelle dient zur Übersetzung:

Orange Book	C1	C2	B1	B2	B3	A
Grünbuch	F1/Q2	F2/Q2	F3/Q3	F4/Q4	F5/Q5	F5/Q6

Der Sinn eines solchen Kriterienkatalogs ist:

- Vertrauenswürdigkeit von Datenverarbeitungssystemen definieren.
- Standards für Systemhersteller setzen; allgemeine Hebung des Sicherheitsstandards; Qualitätsdruck auf die Hersteller.
- Beurteilungsmaßstäbe für Betreiber und Anwender zur Verfügung stellen; Hilfe bei der Erstellung von Ausschreibungen.
- Prüfrichtlinien für offizielle Bewertungsstellen („objektive Vertrauensbildung“ durch „neutrale und vertrauenswürdige Institution“).

Nach der Etablierung entsprechender Standards sollte folgendes Vorgehen bei der Installation eines vertrauenswürdigen Datenverarbeitungssystems möglich sein:

1. Risikoanalyse (organisatorisches Umfeld, Datenmodell, einzusetzende Verarbeitungsverfahren).
2. Anforderungsdefinition (Funktionalitäts- und Qualitätsanforderungen nach den Maßstäben des Kriterienkatalogs).
3. Auswahl eines entsprechend klassifizierten Systems.

Formale Bewertungen nach dem Grünbuch liegen bisher noch nicht vor. Die Firma Siemens strebt für BS2000 die Bewertung nach F2/Q3 und für SINIX die Bewertung nach F2/Q2 an.

Kapitel III

Zugriff auf Daten

Im vorigen Kapitel II stand die Abwehr unerwünschter Eindringlinge ins Datenverarbeitungssystem im Vordergrund, also der Zugangsschutz. In diesem Kapitel geht es darum, was zugelassene Benutzer, die mit dem System arbeiten, tun dürfen. Nach der äußeren wird also jetzt die innere Sicherheit behandelt. Perfekte äußere Sicherheit reicht nicht aus, um ein sicheres System zu gewährleisten; man denke nur daran, daß die häufigsten Verstöße gegen die Datensicherheit von Insidern kommen. Auch viele Aspekte der inneren Sicherheit wurden im Kapitel II schon abgehandelt. Es bleiben aber noch zwei Themenschwerpunkte, die ausführlicher zu besprechen sind:

- die Spezifizierung der Rechte zum Zugriff auf Daten und ihre Absicherung,
- die speziellen Probleme, die durch den Abgleich von Datenbanken entstehen, die umfangreiche Datensammlungen beherbergen.

Die Rechte der zugelassenen Benutzer kann man in **Funktionsrechte** und **Zugriffsrechte** unterteilen; diese Unterscheidung ist natürlich nicht scharf. Die Funktionsrechte umfassen einerseits das Recht, Funktionen für den Datenzugriff auszuüben (etwa statistische Auswertungen). Ein solches Recht wird als Teil der Spezifikation der Zugriffsrechte angesehen, die im folgenden ausführlich behandelt werden; die Funktion oder Prozedur tritt als verlängerter Arm des Benutzers in Erscheinung und wird bei der Rechteprüfung als „Surrogat“ des Benutzers angesehen. Andererseits umfassen die Funktionsrechte die Rechte zum Aufruf von Systemprozeduren (die etwa durch Privilegien geregelt werden) oder zum Start von Anwendungsprogrammen; dieses Thema wurde schon im Kapitel II ausführlich genug behandelt. An ein wichtiges Prinzip sei noch einmal erinnert: das Prinzip der minimalen Rechte, das selbstverständlich auch auf Funktions- und Zugriffsrechte anzuwenden ist.

Spezielle Probleme ergeben sich beim Zugriff auf Datenbanken. Wie läßt sich verhindern, daß anonymisierte Daten durch Datenabgleich „deanonymisiert“ werden? Wie kann man Informationen, etwa demographische Daten aus

einer Volkszählung, in zusammengefaßter Form für statistische Auswertungen so zur Verfügung stellen, daß Informationen über einzelne Datensätze geschützt bleiben?

1 Zugriffsschutz

Die Maßnahmen zum Zugriffsschutz betreffen die regulär zum Datenverarbeitungssystem zugelassenen Benutzer und zielen darauf, nur berechtigte Zugriffe auf Daten zuzulassen und unberechtigte Zugriffe zu verhindern. Dazu müssen Daten und Personen einander zugeordnet werden, und diese Zuordnung ist fest im System zu verankern.

1.1 Definition der Zugriffsrechte

Zuerst ist die Berechtigung von Personen zum Zugriff auf Daten und Prozeduren auf der organisatorischen Ebene zu definieren. Diese Verfahrensstufe entspricht einer Anforderungsdefinition und ist Teil der Systemanalyse.

Die Besitzverhältnisse der einzelnen Daten müssen eindeutig geklärt sein, ebenso die Berechtigung zur Vergabe von Rechten an ihnen. Besondere Beachtung ist den Systemdateien zu widmen: Benutzerverzeichnis, Log-Dateien und andere Aufzeichnungen, Systemtabellen und Sicherheitsdefinitionen. Besitzer von Betriebsmitteln und Systemdaten ist im allgemeinen der Systemverwalter im Rahmen der organisatorischen Richtlinien, Besitzer der Daten einer Projektgruppe der Projektleiter. Darüber hinaus gibt es Daten im „Privatbesitz“ einzelner Benutzer. Zugriffsrechte werden im allgemeinen vom Eigentümer vergeben (**Prinzip des benutzerbestimmbaren Zugriffs**, ‘discretionary access); in manchen Situationen bietet das **Prinzip des festgelegten Zugriffs** aufgrund von systemweit festgelegten Sicherheitsstufen (‘mandatory access’) geeigneteren Schutz.

Um die Definition der Zugriffsrechte übersichtlich zu gestalten, verwendet man ein rechteckiges Schema, die **Berechtigungsmatrix**. Ihre Zeilen entsprechen den Benutzern, die Spalten den Daten, auf die zugegriffen werden darf – das können Dateien oder auch einzelne Datensätze sein. In jedem Feld der Matrix steht die Art des entsprechenden Zugriffsrechts, siehe Abbildung III-1. Sinnvolle Rechte auf dieser Planungsebene sind

- lesen,
- ändern,
- löschen,
- neu anlegen.

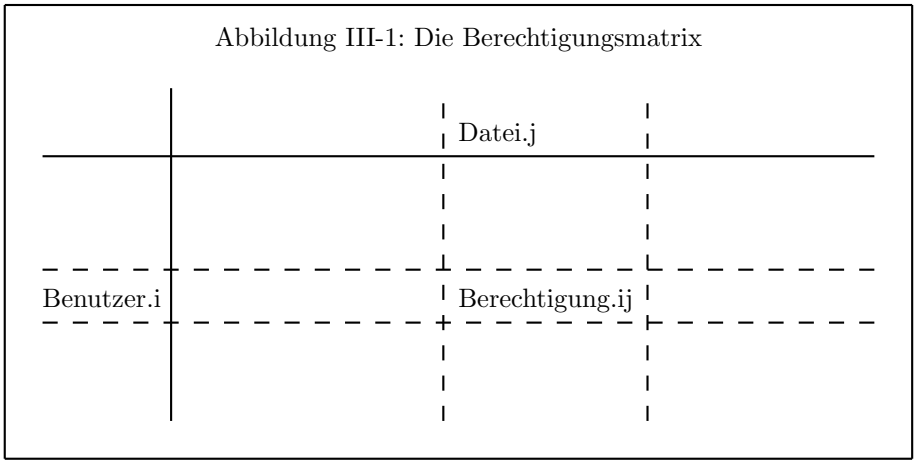


Abbildung III-1: Die Berechtigungsmatrix

Zur praktischen Durchführbarkeit ist es sinnvoll, sowohl Benutzer als auch Daten in Gruppen, eventuell auch in Hierarchien einzuteilen. Zu jeder Gruppe von Daten (etwa einer Datenbank) wird dann einfach eine Liste der Benutzergruppen (etwa Projektgruppen) erstellt, die auf diese Daten zugreifen dürfen, wobei die Art des erlaubten Zugriffs jeweils angegeben ist. In der Praxis wird ein Benutzer gelegentlich mehreren Gruppen angehören. Dann ist es besser, ihm nicht die Vereinigung aller entsprechenden Rechte gleichzeitig zu geben, sondern immer nur das Rechtepaket, das er in seiner momentanen Rolle benötigt.

1.2 Die Zugriffsmatrix

Auf der nächsten Planungsebene, die der Entwurfsphase entspricht, wird die Berechtigungsmatrix zur **Zugriffsmatrix** verfeinert. Ihre Zeilen entsprechen den zugreifenden Subjekten – das können Benutzer oder Prozesse sein. Die Spalten entsprechen den Objekten, auf die zugegriffen werden soll – das können Daten oder Prozesse sein. In jedem Feld der Matrix steht wieder die Art des entsprechenden Zugriffsrechts. Auf dieser Stufe sind die Daten in Form von Datenobjekten mit den jeweils nötigen Zugriffsprozeduren zu organisieren. Ein Zugriffsrecht ist dann die Erlaubnis, eine bestimmte Zugriffsprozedur auszuführen, eventuell mit einer Einschränkung der zulässigen Wertebereiche der Eingabeparameter. Das „Datenobjekt“ ist ein Konzept, das aus dem Software-Engineering stammt und in der Praxis wegen unsauberer Programmierung oder ungeeigneter Programmiersprachen leider selten in methodischer Reinheit vorkommt.

Kompliziert wird die Situation dadurch, daß Prozesse sowohl Subjekte als auch Objekte sein können, was zum Beispiel bei Systemprozessen und Anwendungsprogrammen der Fall ist. Die zu einem Datenobjekt gehörigen Zugriffspro-

zeduren sind in diesem Sinne keine Prozesse, sondern Rechte. Wegen dieser „Zwitter“ aus Objekten und Subjekten muß man Nebenwirkungen sorgfältig untersuchen; mathematisch gesprochen muß man die Relation „A darf auf B zugreifen“ transitiv abschließen. Der typische Gefahrenfall ist der, wo ein Prozeß einen anderen Prozeß mit weitergehenden Rechten aufrufen darf. Eine weitere Gefahrenquelle bilden die Dateien, in denen die Zugriffsmatrix bei der Implementation (in der nächsten Phase) im System abgelegt ist; auch für sie müssen Besitzverhältnisse und Rechte wohldurchdacht sein.

Die Zugriffsmatrix ist ein sehr großes Objekt und läßt sich wohl kaum auf ein Blatt Papier schreiben. Ein großes Rechnersystem hat oft 1000 Benutzer und 100000 Objekte, wobei zu den Benutzern als Subjekte noch die Prozesse kommen. Im allgemeinen ist die Zugriffsmatrix in einem geschlossenen System jedoch sehr dünn besetzt. Zur praktischen Durchführbarkeit ist auch hier wieder eine Modularisierung angebracht, indem sowohl Subjekte als auch Objekte in Gruppen, eventuell in Hierarchien eingeteilt werden. Zu jeder Gruppe von Objekten wird dann wieder eine Liste der Gruppen von Subjekten erstellt, die auf diese Objekte zugreifen dürfen, unter Angabe der Art des erlaubten Zugriffs. Die Gruppen von Objekten können Datenobjekte von einem gemeinsamen Datentyp, also insbesondere mit gemeinsamen Zugriffsprozeduren, sein. In der Praxis handelt es sich in der Regel um eine Zusammenfassung verschiedener Objekte, bei denen nur grob definierte gemeinsame Rechte zu vergeben sind – etwa eine Programmbibliothek, die ein bestimmter Programmierer ändern und mit dem jeweils passenden Compiler übersetzen darf, oder ein Dateiunterverzeichnis, je nach Art der Datei-Organisation im Betriebssystem. Auf der anderen Seite ist möglicherweise ein einzelnes Feld in einem einzigen Datensatz mit einem besonderen Schutz zu versehen („das Gehalt der Chefin“).

Die einfachste Form einer Hierarchie ist eine Einteilung in Sicherheitsstufen: Ein Subjekt darf auf genau die Objekte zugreifen, die höchstens auf der gleichen Stufe stehen. Dieses Konzept ist für sich allein genommen viel zu grob; es ist aber sehr gut geeignet, um systemweite Voreinstellungen zu definieren und wird im militärischen Bereich häufig angewendet, wo man noch gewöhnt ist, hierarchisch zu denken. Außerdem ist es durch die Aufnahme in die Stufe B1 des ‘Orange Book’ beziehungsweise F3 des „Grünbuchs“ amtlich abgesegnet. Allgemeiner spricht man von „globalen Zugriffsmodellen“, wenn sich die Schutzbedürfnisse ganzer Klassen von Objekten einheitlich beschreiben lassen.

Das Dilemma der hierarchischen Sicherheitsstufen wird durch folgende Überlegung deutlich:

- Die Vertraulichkeit verlangt, daß Leserechte nur von höheren auf niedrigere, Schreibrechte nur von niedrigeren auf höhere Stufen gewährt werden. Dabei stellt man sich vor, daß auf den höheren Stufen größere Geheimnisse verwahrt werden, die nicht in die tieferen Stufen gelangen dürfen.

- Die Manipulationssicherheit verlangt genau das umgekehrte: Schadprogramme dürfen nicht über die niedrigen zu den hohen Sicherheitsstufen

gelangen. Daher dürfen Leserechte nur von unten nach oben, Schreibrechte nur von oben nach unten gewährt werden.

Die notwendige Folgerung ist, daß man die Stufen in beiden Richtungen streng separieren muß; man hat dann logisch getrennte Systeme, die man besser gleich auch physisch trennt.

Etwas abgemildert wird das Dilemma durch die Unterscheidung zwischen Daten und Programmen:

- Für Daten hat die Vertraulichkeit Priorität,
- für Programme die Manipulationssicherheit.

Will man diese Idee in die Realität umsetzen, gerät man sofort an die Frage der Abgrenzung zwischen Programmen und Daten. So werden etwa bei der Software-Entwicklung Programme als Daten behandelt, oder aber Programme enthalten Datenblöcke, während umgekehrt Konfigurationsdateien oder Regelbasen von Expertensystemen den Charakter von Programmen haben. Kurz: Das Konzept der Sicherheitsstufen bedarf einer sorgfältigen Differenzierung, wenn es wirkungsvoll eingesetzt werden soll, und genau dazu wurde es nicht erfunden – es sollte vielmehr ein einfaches, idiotensicheres Sicherheitssystem ermöglichen.

1.3 Typen des Zugriffs

Bei der Datenobjekt-gebundenen Zugriffskontrolle sind alle überhaupt möglichen Zugriffe in der Definition des Datenobjekts enthalten. Programmiersprachen, die dieses Konzept unterstützen (ADA und MODULA-2), sind besonders zur Implementation sicherer Systeme geeignet. Auch erhöht eine gute Benutzer-Oberfläche die Sicherheit – Benutzer, die nur die Objekte sehen, mit denen sie arbeiten dürfen, und nur die erlaubten Operationen auch tatsächlich ausführen können, geraten weder selbst in unvorhergesehene Schwierigkeiten, noch können sie im System solche erzeugen.

Allgemein zutreffende Zugriffsrechte sind die folgenden:

- ‘see’ – Es darf festgestellt werden, ob ein Objekt existiert (es wird etwa in einem Inhaltsverzeichnis angezeigt).
- ‘read’ – Das Objekt darf in den Adreßraum des Berechtigten kopiert werden. Damit kann es zum Beispiel auf dem Bildschirm ausgegeben werden.
- ‘change’ – Schreibzugriff; das Objekt darf geändert werden.
- ‘extend’ – Das Objekt darf um weitere Informationen erweitert werden.
- ‘delete’ – Das Objekt darf vernichtet werden.
- ‘control’ – Die Zugriffsrechte auf das Objekt dürfen verändert werden.

- 'execute' – Das ist sinnvoll für Programmdateien, die zwar ausgeführt, aber nicht geändert werden sollen. Ein Schreibschutz alleine nützt nichts, es muß auch ein Leseschutz vorhanden sein, damit sich nicht ein Anwender eine Kopie des Programmtextes anfertigt, diese ändert und dann ausführt. Auf diese Weise könnte er Schutzmechanismen aus dem Programm entfernen.

Die Möglichkeit, für Programmdateien 'execute (only)'-Zugriffsrechte zu vergeben, verursacht den Systemverwaltern auf vielen Systemen erhebliches Kopfzerbrechen, denn Sicherheitsmaßnahmen lassen sich oft sehr einfach in einem Programm unterbringen, wo sie aber selbst nicht ohne weiteres geschützt sind. Dazu sollte die Möglichkeit des 'execute'-Zugriffs im Betriebssystem, am besten sogar in der Hardware verankert sein. Es gibt drei Realisierungen:

- Ein Programm wird nur im Maschinencode zum Lesen zur Verfügung gestellt. Das ist ein gewisser Leseschutz, besonders für längere Programme bei unbedarften Benutzern. Allerdings sind eingebettete Textstellen wie Schlüsselwörter zum Zugriff auf verschlüsselte Dateien oder Sicherheitsabfragen leicht lesbar, und mit Hilfe eines Disassemblers kann ein gewiefter Fachmann zumindest einige besonders wichtige Stellen entschlüsseln. Ohne besondere Vorkehrungen im Betriebssystem ist dies oft schon der einzig mögliche Schutz.

- Besser ist es, wenn das Betriebssystem ein spezielles Dateiattribut zur Verfügung stellt. Hier läßt sich ein Lesezugriff zwar verhindern, aber gegen Speicherdumps ist der Code nicht ohne weiteres geschützt. Zusätzlich sollte also die Hardware auch noch ein entsprechendes Schutzattribut für den Hauptspeicher zur Verfügung stellen.

- Als weitere gute Möglichkeit gibt es in vielen Betriebssystemen die schon in Kapitel II vorgestellte Methode, eigenständige Serviceprozesse aufzusetzen, wo der Programmcode dann völlig außerhalb des Benutzerbereichs liegt und dadurch vor Lesezugriff geschützt ist. Der Nachteil dieser Methode ist der hohe Implementierungsaufwand; oft leidet auch die Abarbeitungsgeschwindigkeit.

Statistische Lesezugriffe auf Gruppen von Datensätzen werden in Abschnitt 2 ausführlich behandelt.

1.4 Zugriffsregeln

Die auf der organisatorischen Ebene spezifizierten Zugriffsrechte sind in konkrete Zugriffsberechtigungen innerhalb des Systems umzusetzen. Das Software-Sicherheitssystem muß die genaue Umsetzung des Konzepts „Zugriffsmatrix“ gestatten. Natürlich muß das Bild nicht formal wie eine Matrix aussehen; denkbar ist auch zu jedem Objekt eine Liste (**Zugriffsliste**) „A darf lesen“, „A darf

nicht schreiben“, „B darf schreiben und lesen“, ... Diese Liste kann auch als „File-Dämon“ (oder ‘User Exit’) implementiert sein, also als Prozedur, die immer dann angestoßen wird, wenn jemand das Objekt anfaßt (also zum Beispiel eine Datei öffnet). Genausogut denkbar, aber letztlich doch weniger geeignet, ist eine Rechteliste für jedes Subjekt. Jedenfalls ist eine möglichst übersichtliche Darstellung anzustreben. Dazu dient auch die Gruppenbildung; im Konzept definierte Gruppen von Subjekten und Objekten sollten auch in der Realisierung verwendet werden können.

Es müssen sowohl Voreinstellungen möglich sein als auch allgemeingültige Regeln, die nicht außer Kraft gesetzt werden können. Ein geschlossenes System entsteht durch die Voreinstellung: „Ist die Erlaubnis für diesen Zugriff nicht ausdrücklich gegeben, so ist er verboten.“ Dagegen erhält man ein offenes System durch die Voreinstellung: „Ist dieser Zugriff nicht ausdrücklich verboten, so wird er zugelassen.“ In beiden Fällen wird jedenfalls eine drastische Reduktion des Aufwands bei der Implementation der Zugriffsmatrix erreicht.

Eine bewährte Hierarchie von Zugriffsregeln ist die folgende:

- System-Regeln – das sind allgemein gültige Zugriffsregeln, die durch andere Regeln nicht außer Kraft gesetzt werden können. Auf diese Weise ist ein festgelegter Zugriffsschutz (‘mandatory access’) auch ohne hierarchische Sicherheitsstufen denkbar, auch wenn das dann nicht mehr dem Konzept des ‘Orange Book’ entspricht.

- Gruppen-Regeln – das sind Zugriffsregeln für Benutzergruppen, die stärkere Gültigkeit haben als Regeln für einzelne Benutzer.

- Einzelfall-Regeln, die auch den benutzerbestimmbaren Zugriffsschutz verwirklichen.

- Gruppen-Voreinstellungen, die zum Tragen kommen, wenn keine explizite Regel formuliert wird.

- System-Voreinstellungen, die alle übrigen Fälle regeln. Hier ist der Platz, um ein System zu schließen, indem die generelle Voreinstellung „nicht erlaubt“ eingesetzt wird. Dadurch wird verhindert, daß aus Vergeßlichkeit Sicherheitslücken offengelassen werden.

Paßwortschutz für Datenzugriffe ist eine veraltete Methode, jedenfalls wenn er die hauptsächliche Maßnahme darstellt. Er kann höchstens als zusätzlicher Schutz eingeführt werden. Immerhin bietet ein Dateipaßwort noch Schutz gegen einen unbefugt unter einer falschen Identität in das System eingedrungenen Angreifer, der sonst sofort alle Rechte des echten Benutzers hätte. Ein weiterer Vorteil ist die Möglichkeit, unerlaubte Zugriffsversuche mit einer Sperre zu beantworten. Im Gegensatz zu einem persönlichen Zugangspaßwort, das auf eine Person beschränkt ist, muß ein Zugriffspaßwort jedoch in der Regel mehreren Berechtigten mitgeteilt werden. Je mehr Personen aber ein Paßwort besitzen,

desto unsicherer ist es und desto schwerer ist es änderbar; durch die Weitergabe verliert der Eigentümer der Daten die Kontrolle über die Zugriffsberechtigungen – ein Geheimnis, das man weitergegeben hat, gehört einem nicht mehr ganz. Ferner führen Dateipaßwörter sehr schnell dazu, daß sich jeder Benutzer eine riesige Menge verschiedener Paßwörter merken muß. Einem Dateipaßwortschutz vergleichbar, aber mit wesentlichen weiteren Vorteilen versehen, ist übrigens auch die Verschlüsselung einer Datei; die Rolle des Paßworts spielt dann der Schlüssel.

Zugriffsberechtigungen müssen automatisch gelöscht werden, wenn das Subjekt (oder das Objekt) aufhört zu existieren, damit sie nicht später auf neugeschaffene Subjekte (oder Objekte) gleichen Namens übergehen.

Werden Datenzugriffe über einen Server verwaltet (auch bei der Datensicherung kann das zutreffen), so ist ebenfalls auf strenge Zuordnung der Daten zu ihren Besitzern und sonstige Zugriffsrechte zu achten. Möglicherweise dient der Server selbst als Kontrollinstanz (etwa bei einer Datenbank), oder er hat die „Surrogat-Fähigkeit“, das heißt, er arbeitet mit genau den Rechten, die sein Auftraggeber hat (etwa beim Restaurieren von Daten aus einer Datensicherung).

2 Sicherheit von Datenbanken

Eine Datenbank ist ein spezielles Datenobjekt mit einem typischen Satz an Zugriffsoperationen. Insofern fällt der Schutz von Datenbanken unter den allgemeinen Zugriffsschutz aus Abschnitt 1. Eine Datenbank hat als Datenobjekt aber auch einige besondere Merkmale:

- Sie enthält besonders viele Informationen, ist also ein lohnendes Angriffsobjekt.
- Die Daten sind oft sehr fein granuliert – Zugriffsrechte beziehen sich manchmal auf einzelne Felder in einzelnen Datensätzen.
- Der Benutzerkreis ist oft sehr groß.
- Durch die Verknüpfung von Informationen aus verschiedenen Datenbanken (**„Abgleich“**) lassen sich oft Schlüsse ziehen, die den Datenschutz verletzen (unzulässige Inferenzen durch zulässige Datenzugriffe).

Die Konstruktion einer Datenbank bringt auch weitere spezielle Probleme mit sich, die hier nicht zur Debatte stehen: die Koordination von Zugriffen, um die Datenintegrität stets zu gewährleisten (eine **Transaktion** ist ein Zugriff unter Bewahrung der Konsistenz), Protokollierung und Datensicherung.

Geklärt werden sollen die Fragen: Wie weit trägt der gewöhnliche Zugriffsschutz im Spezialfall der Datenbanken? Was kann man durch Abgleich an Information gewinnen? Wie weit sind die Daten geschützt, wenn als Zugriffsoperationen nur statistische Prozeduren erlaubt sind? Wie hoch ist das Risiko eines

„anonymisierten“ Datensatzes, identifiziert zu werden? Diese Fragen sind vor allem durch die Diskussion um die letzte Volkszählung in der Bundesrepublik Deutschland aktuell geworden.

2.1 Klassischer Zugriffsschutz

Typisch für Daten, die in einer Datenbank organisiert sind, ist der „inhaltsgesteuerte Zugriff“ – die physische Struktur der Daten muß dem Anwender oder Anwendungsprogramm nicht bekannt sein, sie wird vom Datenverwaltungssystem (‘Data Base Management’) geregelt. Dieses stellt an seiner Schnittstelle die genau spezifizierten Zugriffsoperationen zur Verfügung, wie es dem Konzept eines Datenobjekts entspricht.

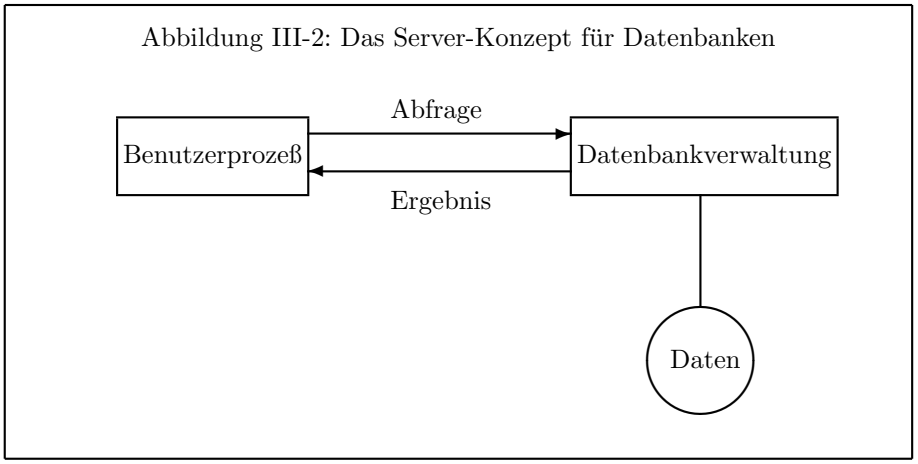
Die Daten müssen nun zunächst vor physischen Zugriffen und vor Zugriffen über das Betriebssystem geschützt sein – das Datenbanksystem kann keinen besseren Schutz gewähren als das zugrundeliegende Betriebssystem; eine gewisse Ausnahme von dieser Regel läßt sich allerdings durch Verschlüsselung erreichen. Jedenfalls soll ein Zugriff auf die Daten nur über das Verwaltungssystem möglich sein. Hier gelten wieder die grundsätzlichen Überlegungen über die Sicherheit von Anwendungsprogrammen in Abschnitt II-4.2 und II-4.6. Bei Datenbanken in Netzen oder auf Großrechnern ist das Verwaltungssystem in der Regel in einem Server untergebracht, in dem dann auch das Zugriffsschutzsystem seinen Platz hat, siehe Abbildung III-2. Natürlich muß der Programm-Code des Servers vor Manipulation geschützt sein. Im Netz ist der Server ein eigener Rechner, der nichts weiteres tut, als die Datenbank zu verwalten; in einem Großrechner ist er dagegen ein eigenständiger Prozeß („virtueller Server“). Als Alternative für kleinere Datenbanken mit schutzwürdigen Daten bietet sich ein „dedizierter“ Arbeitsplatzrechner an, der nur für die Datenbank da ist; er muß mit den in Abschnitt II-5.2 beschriebenen Maßnahmen geschützt sein und darauf eine nicht zu umgehende geschlossene Benutzer-Oberfläche für die Datenbankabfragen zur Verfügung stellen.

In die Kategorie „klassischer Zugriffsschutz“ gehört auch der Vorschlag, die Datensätze oder zumindest ihren Identifikationsteil asymmetrisch zu verschlüsseln [37, S.164]. Das erlaubt die Eingabe für einen größeren Personenkreis mit Hilfe des öffentlichen Teils des Schlüssels; gegen Lesen sind die Daten, auch bei unsicherem Übertragungsweg, geschützt – außer vor den autorisierten Personen, die den geheimen Teil des Schlüssels kennen. Speziell für Krebsregister scheint dies ein guter Vorschlag zu sein.

2.2 Abgleich von Daten

Datenabgleich bedeutet, Informationen aus verschiedenen Quellen, etwa Datenbanken, zusammenzuführen. Am besten stellt man sich das mit personenbezogenen Daten vor, dem Gegenstand des Datenschutzes im engeren Sinne des

Abbildung III-2: Das Server-Konzept für Datenbanken



Datenschutzgesetzes. Als einfaches Modell eines Datensatzes in einer Datenbank sei eine Zerlegung in

frei zugängliche Merkmale	vertrauliche Merkmale

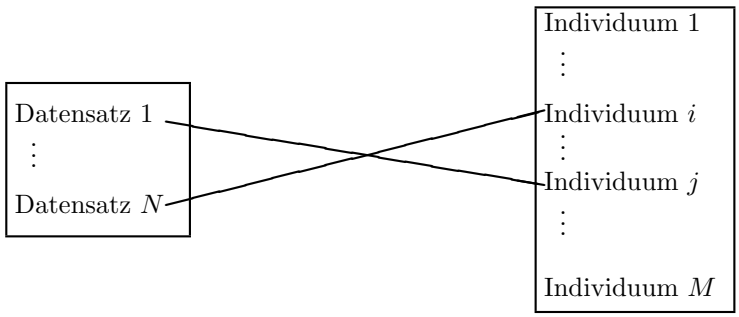
angenommen; die Abgrenzung zwischen beiden Arten von Merkmalen kann natürlich je nach dem zugreifenden Subjekt variieren. Stellt man sich als Beispiel Patientendaten in einem Krankenhaus vor, so sind

- für den behandelnden Arzt alle Daten zugänglich, keine Daten vertraulich,
- für einen Forscher Diagnose- und Therapiedaten zugänglich, Identifikationsmerkmale (Name usw.) vertraulich,
- für die Krankenhausverwaltung (zur Abrechnung) Name und Therapiemaßnahmen zugänglich, viele diagnostische Daten vertraulich.

Die Datenschutzgesetzgebung fordert die Löschung des Personenbezugs von Datensätzen (**Anonymisierung**), wo immer möglich, zum Beispiel wenn Daten für Forschungszwecke benützt werden. Ein typisches Beispiel sind Volkszählungsdaten: Sie werden (formal) anonymisiert durch Löschen der Identifikationsmerkmale; die übrig bleibenden Daten sind (mehr oder weniger) frei zugänglich. Ist die Anonymität der Daten in einer solchen „statistischen Datenbank“ gesichert? Oder kann man vertrauliche Informationen aus zulässigen Zugriffen durch **Inferenz** herleiten und die Datensätze **deanonymisieren**?

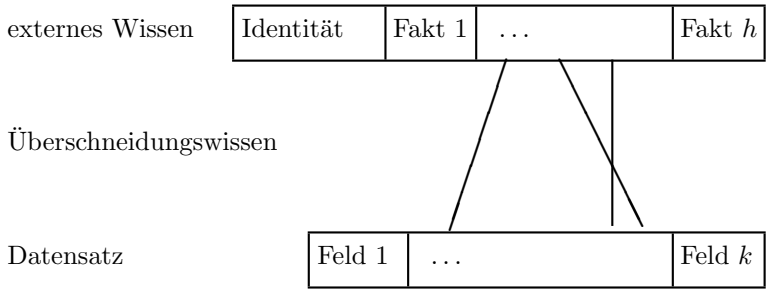
Stellen wir uns modellhaft die Datenbank als eine Sammlung von Datensätzen vor, die einer *Stichprobe* aus einer Grundgesamtheit (etwa aus der Gesamtbevölkerung) entsprechen. Diese Situation ist in Abbildung III-3 dargestellt.

Abbildung III-3: Datenbank als Stichprobe



Ein **Angreifer** hat Teilwissen und sucht weitere Informationen über ein **Zielindividuum**, für welches vielleicht ein Datensatz in der Datenbank enthalten ist. Das Teilwissen kann aus anderen Datenbanken stammen oder aus privaten Quellen; es wird auf jeden Fall mit Bezug auf die Datenbank als **externes Wissen** bezeichnet. Datensatz und externes Wissen enthalten möglicherweise einige gemeinsame Daten (**Überschneidungswissen**) wie in Abbildung III-4 angedeutet. Das externe Wissen besteht dort aus h Fakten plus der Identität des Individuums; der Datensatz enthält k Felder, von denen einige Entsprechungen im externen Wissen haben.

Abbildung III-4: Überschneidungswissen



Ein konkret denkbarer Fall wird in Tabelle III-1 beschrieben; man sieht ganz

deutlich, wie wenige Merkmale im Überschneidungswissen schon zur eindeutigen Identifikation ausreichen können.

Tabelle III-1: Datenabgleich

Merkmale	externes Wissen	Datensatz
Identität:		
Name	Pommerening, Klaus	—
Geburtsdatum	26.8.1946	—
Überschneidungswissen:		
Beruf	Universitätsprofessor	Universitätsprofessor
Familienstand	verheiratet	verheiratet
Zahl der Kinder	2	2
Wohnort	6507 Ingelheim	Ingelheim
Alter	(siehe Geburtsdatum)	44
Zieldaten:		
Schulden	?	241000 DM bei X-Bank
Vorstrafen	?	keine

Ein Datensatz läßt sich umso leichter identifizieren,

- je größer das Überschneidungswissen ist,
- je differenzierter die einzelnen Merkmale oder gewisse Merkmalskombinationen sind.

Im allgemeinen enthält der Datensatz noch genügend Merkmale des Zielindividuums, um dieses eindeutig zu identifizieren, selbst wenn er *formal* anonymisiert ist, also etwa Name, Anschrift und Geburtsdatum gelöscht sind.

2.3 Statistische Abfragen

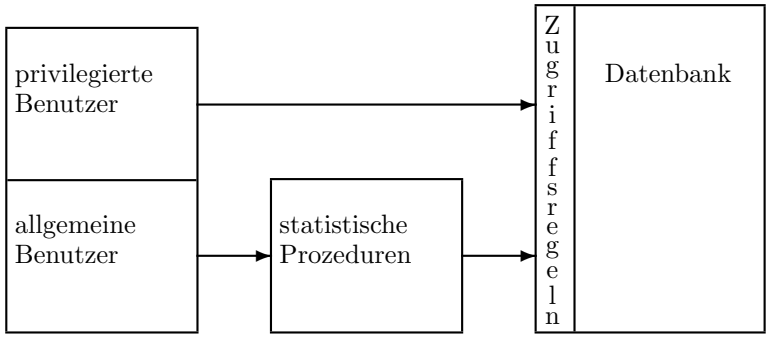
Forscher sind in der Regel nicht an einzelnen Datensätzen interessiert, sondern an statistischen Auswertungen mit Hilfe der gesamten Daten; das gleiche gilt sicher auch für viele andere Nutzer von Datenbanken. Gefragt sind also nicht Lesezugriffe auf einzelne Datenfelder, sondern nur statistische Zugriffsprozeduren. Eine **statistischen Datenbank** ist ein Spezialfall eines Datenobjekts mit spezifizierten Zugriffsprozeduren. Um eine solche zu modellieren, stellen wir uns vor:

- Ein kleiner, privilegierter Nutzerkreis, die Besitzer und Verwalter der Datenbank darf auf die Datensätze direkt zugreifen, differenziert durch übliche Zugriffsregeln wie in Abschnitt 1.4.

- Ein großer, allgemeiner Benutzerkreis darf nur statistische Größen abfragen, wie Häufigkeiten, Mittelwerte usw.

Als Beispiel wäre etwa im Statistik-Paket 'SAS' die Prozedur 'PRINT' nicht allgemein nutzbar, weil sie einzelne Datenfelder ausgibt; allgemein nutzbar wäre dagegen die Prozedur 'MEANS'. Schematisch dargestellt ist das in Abbildung III-5.

Abbildung III-5: Statistische Datenbank



Es ist offensichtlich, daß die Beschränkung auf statistische Prozeduren einzelne Datenfelder nicht wirksam schützt — die folgende Abfrage gibt ziemlich sicher den Wert eines einzigen Datenfeldes aus:

> Mittelwert des Merkmals „Schulden“ über alle Datensätze
> *mit* Beruf = Universitätsprofessor
> *und* Alter = 40 bis 50
> *und* Wohnort = Ingelheim
> *und* Zahl der Kinder = 2.

Wie stellt man sich im allgemeinen eine **statistische Abfrage** vor? Sie besteht aus zwei Teilen:

- Angabe einer auszuwählenden Gruppe von Datensätzen, spezifiziert durch einen Booleschen Ausdruck, zum Beispiel

$$(\text{Geschlecht} = m) \quad \textit{und} \quad (\text{Alter} \geq 50).$$

- Anforderung einer deskriptiven statistischen Größe, zum Beispiel „Häufigkeit“ (bei qualitativen Merkmalen) oder „Summe“ (bei quantitativen Merkmalen).

Es liegt nun nahe, daß man versucht, sensitive statistische Abfragen auszuschließen. Für den Begriff der Sensitivität wird ein Maß definiert: Eine statistische Abfrage heißt (n, t)-**sensitiv**, wenn es n Datensätze gibt, die mehr als $100t\%$ zum Ergebnis (etwa Häufigkeit oder Summe) beitragen. (Es hat nicht viel Sinn, diesen Begriff weiter zu formalisieren.) Beispiele:

- Abfragen über eine Gruppe von n Datensätzen sind stets $(n, 1)$-sensitiv.
- Angenommen in einem Ort gibt es eine sehr große und ein paar sehr kleine Firmen. Dann ist die Summe der Umsätze aller dieser Firmen in einem Jahr (vielleicht) $(1, 0.8)$-sensitiv.

Der kritische, „sensitive“ Fall ist also ein großes t bei kleinem n — dann lassen sich für die entsprechenden n Datensätze Schlüsse auf die eigentlich gesperrten Datenfelder ziehen.

Damit läßt sich folgendermaßen ein Angriff auf gesperrte Datenfelder starten: Man sucht eine Gruppe B zu konstruieren, die klein ist, am besten nur aus dem einen interessierenden Datensatz besteht, und fragt Häufigkeit oder Summe des entsprechenden Merkmals in der Gruppe B ab. Als Schutz vor einem solchen Angriff muß die Ausgabe von Ergebnissen für kleine Abfragegruppen gesperrt werden, vor allem $(1, 1)$-sensitive Abfragen.

Das reicht aber noch nicht. Der Angreifer kann nämlich noch große Abfragegruppen angeben. Daher sucht er sich eine Gruppe B', die alle Datensätze *außer* dem interessierenden enthält, und gibt zwei Abfragen auf: eine über sämtliche Datensätze und eine über die Gruppe B'. Aus der Differenz erhält er sein gesuchtes Datenfeld. Die Datenbankverwaltung muß also auch Abfragen über große Gruppen sperren; die Abfrage über die gesamten Datensätze (also die größtmögliche Gruppe) ist allerdings unkritisch.

Mit dieser **Kontrolle der Abfragegröße** wird ein *versehentlicher* Datenabgleich durch unbeabsichtigtes Wählen von zu kleinen oder zu großen Gruppen recht gut verhindert. Der normale anständige Benutzer weiß ja vor einer Abfrage im allgemeinen nicht, wie groß die ausgewählte Gruppe ist. Besteht sie nur aus einem Datensatz und enthält dieser einige leicht wiederzuerkennende Merkmale, so ist eine versehentliche Identifikation denkbar.

Der Angreifer, der vorsätzlich aus der Datenbank geschützte Daten holen will, wird durch diese Maßnahme aber nur unwesentlich behindert. Die überraschend einfachen Methoden, die ihm zur Verfügung stehen, werden im nächsten Abschnitt 2.4 vorgestellt.

2.4 Tracker-Angriffe

Wir stellen uns jetzt eine statistische Datenbank mit Kontrolle der Abfragegröße vor. Die Anzahl aller Datensätze sei N, und es ist eine Sicherheitsschranke

n festgesetzt: Ein Abfrage-Ergebnis für eine Gruppe B wird nur ausgegeben, wenn für die Gruppengröße, symbolisch als $\#B$ geschrieben, die Ungleichung

$$n \leq \#B \leq N - n$$

erfüllt ist. Damit überhaupt Abfragen möglich sind, muß natürlich $n \leq N/2$ sein. Je näher allerdings n bei $N/2$ liegt, desto weniger brauchbar ist die Datenbank. Im übrigen kann man Abfragen mit $\#B = 0$ oder $\#B = N$ ohne weiteres Sicherheitsrisiko immer zulassen.

Die nun folgende Beschreibung von Angriffsmethoden erfordert elementare mathematische Kenntnisse aus der Mengenlehre. Das Ergebnis einer Abfrage q über eine Gruppe B wird mit $q(B)$ bezeichnet; der Einfachheit halber sei immer angenommen, daß es sich bei einem qualitativen Merkmal um die Häufigkeit und bei einem quantitativen Merkmal um die Summe handelt. Andere statistische Größen sind für Angriffe aber genau so gut geeignet, der Angreifer muß nur eventuell ein bißchen mehr rechnen.

Will der Angreifer Informationen über eine bestimmte, eigentlich gesperrte Gruppe B erhalten, kann er den **individuellen Tracker** verwenden, der 1975 von J. SCHLÖRER vorgestellt wurde. Dazu braucht er nur zwei Gruppen C und D (durch Probieren oder nach einem Algorithmus) zu finden mit

$$B = C \cap D,$$

so daß C und $T = C \cap \neg D$ zulässig sind, siehe Abbildung III-6; $\neg D$ bezeichnet dabei die Komplementärmenge von D. Die gesuchte Größe wird dann einfach aus der Formel

$$q(B) = q(C) - q(T)$$

berechnet.

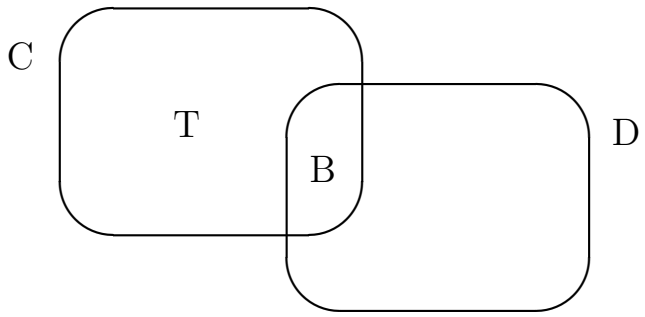
Für den Verteidiger der Datenbank wird es langsam eng. Das „Erfolgsgeheimnis“ des individuellen Trackers liegt in der Abfrage über zwei Mengen von kleiner Differenz. Solche paarweisen Abfragen müssen also gesperrt werden. Was aber, wenn diese Abfragen mit größerem zeitlichen Abstand kommen? Wenn zwei Benutzer zusammenarbeiten und jeder nur eine der Abfragen losschickt? Oder wenn der Angreifer die Menge B auf etwas raffiniertere Weise konstruiert? Selbst umfangreiche Protokollführung über sämtliche Abfragen kann da kaum noch helfen.

Es kommt aber noch schlimmer: M. D. SCHWARTZ stellte in seiner Dissertation an der Purdue University 1977 den **allgemeinen Tracker** vor. Er ist möglich, wenn $n \leq N/4$; aber sonst kann man die Datenbank sowieso vergessen. Er besteht aus einer Abfragemenge T mit

$$2n \leq \#T \leq N - 2n,$$

die ansonsten völlig beliebig ist. *Mehr braucht man nicht, um beliebige gesperrte Abfragen erfolgreich durchzuführen.* Nehmen wir eine unzulässige Abfragemenge

Abbildung III-6: Der individuelle Tracker



B mit $\#B < n$ an. Dann sind die Abfragegruppen $B \cup T$ und $B \cup \neg T$ mit Sicherheit zulässig. Falls die Gesamtmenge A gesperrt ist, erhält man das Ergebnis $q(A)$ aus den zulässigen Mengen T und $\neg T$:

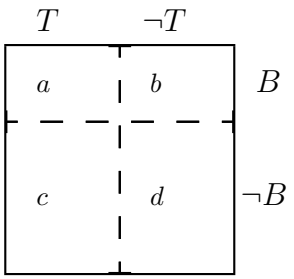
$$q(A) = q(T) + q(\neg T),$$

ansonsten durch direkte Frage. Daraus berechnet der Angreifer

$$q(B) = q(B \cup T) + q(B \cup \neg T) - q(A),$$

vergleiche Abbildung III-7.

Abbildung III-7: Der allgemeine Tracker



$a = q(B \cap T)$ $\quad b = q(B \cap \neg T)$

$c = q(\neg B \cap T)$ $\quad d = q(\neg B \cap \neg T)$

$q(B) = a + b$ $\quad q(A) = a + b + c + d$

$q(B \cup T) = a + b + c$

$q(B \cup \neg T) = a + b + d$

Dieser Angriff kann nur verhindert werden, wenn man die Sicherheitsschranke $n > N/4$ setzt und damit die Datenbank praktisch wertlos macht. Aber das ist auch kein Schutz. Der Angreifer kann mit Hilfe eines m-fachen Trackers mit genügend großem m immer noch alle Informationen aus der Datenbank holen, wenn nicht $n = N/2$ ist — eine absurde Situation, denn dann sind nur noch Abfragegruppen zulässig, die genau die Hälfte aller Datensätze enthalten. Die Details des m-fachen Trackers und andere ähnliche Angriffe werden in [30] beschrieben. Dort wird auch gezeigt, daß das Problem der Erkennung, ob eine Menge von Abfragen sensitiv ist, NP-vollständig, das heißt, nicht praktisch lösbar ist.

Die Tracker-Angriffe zeigen: Wer genügend Merkmale kennt, um ein Zielindividuum zu identifizieren, kann alle Daten über dieses aus der Datenbank holen, selbst wenn nur statistische Zugriffe mit Kontrolle der Abfragegröße erlaubt sind. Denkt man über die Sicherheit einer statistischen Datenbank nach, muß man also dem böswilligen Angreifer den vollen Lesezugriff auf alle Felder aller Datensätze unterstellen. Insbesondere ist die Beschränkung auf statistische Zugriffsoperationen noch keine wirksame Anonymisierungsmaßnahme. „Normale“, gutwillige Benutzer werden ernsthaft behindert, wenn man versucht, Tracker-Angriffe zu unterbinden. Wenn diese Benutzer nicht an der Deanonymisierung von Datensätzen interessiert sind, merken sie in der Regel gar nicht, daß sie in der Lage dazu wären. Hier muß wieder eine Entscheidung nach dem Prinzip der Verhältnismäßigkeit getroffen werden. Es ist sicher sinnvoller, Forscher auf die Einhaltung von Datenschutzregeln zu verpflichten, als Maßnahmen durchzuziehen, die die Arbeit empfindlich behindern, ohne letztlich echte Sicherheit zu gewähren.

Dennoch sollte man die „naiven“ Sicherheitsmaßnahmen für den Schutz von Datenbanken,

- Zugriffskontrolle,
- Kontrolle der Abfragegröße,

in jedem Fall in einem sinnvollen Umfang implementieren. Als zusätzlicher Schutz sind dann, soweit möglich und sinnvoll, noch Anonymisierungsmaßnahmen einzuführen. Die Möglichkeiten hierzu sind Gegenstand der nächsten Abschnitte. Auch hier wird sich zeigen, daß sich kein befriedigender Sicherheitszustand erreichen läßt. Die Sicherheit von Datenbanken bleibt ein schwacher Punkt, der mit viel Fingerspitzengefühl angegangen werden muß.

2.5 Anonymisierung

Für die Forschung in Medizin, Wirtschafts- und Sozialwissenschaften, ebenso wie für die administrative Planung, bieten große Datensammlungen mit einer Vielzahl von Merkmalen zum Zwecke komplexer statistischer Auswertungen erhebliche Möglichkeiten. Einfache Tabellen reichen dazu nicht aus; Zugriff auf

Originaldaten ist nötig, um Hypothesen zu bilden, Modelle zu entwerfen, Hochrechnungen und Simulationen durchzuführen, Prognosen zu geben. Nötig ist der flexible Zugriff auf möglichst repräsentative Stichproben mit Einzelangaben. Natürlich interessiert die Identität der einzelnen Datensätze dabei überhaupt nicht. Der Datenschutz erfordert daher, daß es unmöglich sein soll, Datensätze bestimmten Personen zuzuordnen.

Datensätze sind umso besser geschützt, je geringer der Informationsgehalt der Überschneidungsmerkmale mit dem potentiellen Zusatzwissen ist. Dieser wird auf natürliche Weise gemindert durch **Erhebungsfehler** wie

- irrtümliche Falschangabe,
- absichtliche Falschangabe,
- Übertragungs- und Codierfehler, auch Fehlinterpretationen ungenauer Angaben oder Modifikation von Angaben bei der Aufbereitung oder der Plausibilitätskontrolle,
- ungenaues externes Wissen,

und durch eine eventuelle **Stichprobeneigenschaft** der Datenbank: Wenn der Angreifer nicht sicher ist, ob sein Zielindividuum in der Datenbank enthalten ist, weiß er auch bei einem eindeutigen Ergebnis einer Abfrage nicht unbedingt, ob er vielleicht nur einen Doppelgänger erwischt hat.

Hier setzen die Verfahren zur **Anonymisierung** an – der Informationsgehalt des möglichen Überschneidungswissens wird absichtlich gemindert. Dazu sind die folgenden Maßnahmen geeignet:

- Formale Anonymisierung – offensichtlich mehr oder weniger eindeutige Identifikationsmerkmale wie Name, Adresse, Telefonnummer werden weggelassen.
- Vergröberung der Merkmale etwa durch Rundung oder Klassenbildung; statt „Mainz“ wird „Großstadt“ eingetragen, statt des Geburtsdatums „Alter 40-49“.
- Weglassen von einzelnen Datenfeldern mit extremen Merkmalsausprägungen wie „Größe 2.12 m“ oder „Beruf: Bundeskanzler“.
- Störung der Daten durch absichtliche Fehler, etwa Addition einer zufälligen Größe oder zufällige Rundung.
- Stichprobenziehung – Statistische Prozeduren werden jeweils nur auf eine Stichprobe aus der Abfragemenge angewendet.
- Konstruktion synthetischer Datensätze, so daß die multivariate Verteilung möglichst wenig verändert wird:

– Austausch von Daten zwischen Datensätzen,

– Aggregation – Mittelbildung über jeweils 3 bis 5 Datensätze.

Alle diese Verfahren bringen mehr oder weniger große Nachteile mit sich; die Möglichkeiten zur statistischen Auswertung werden stark eingeschränkt. So erhält der Austausch von Daten zwischen Datensätzen zwar die Mittelwerte der einzelnen Merkmale (wenn sie quantitativ sind), aber er zerstört sämtliche Korrelationen. Als Maßnahme für den Zeitpunkt der Abfrage ist vor allem die Stichprobenziehung geeignet, wobei die Stichproben sogar ziemlich groß sein können (50–90%). Wird für jede Antwort eine andere Stichprobe ausgewählt, ist ein Tracker-Angriff zunächst nicht mehr möglich. Allerdings kann man die wahren Werte durch Mittelbildung über einige identische oder äquivalente Abfragen approximieren und dann doch noch einen Abgleich versuchen.

Die meisten der Maßnahmen sind eher so gedacht, daß aus der „Rohdatenbank“ eine „Arbeitsdatenbank“ erzeugt wird, die dann allgemein zugänglich gemacht wird. Die Originaldaten können im allgemeinen nicht verändert werden, man denke etwa an Patientendaten aus einem Krankenhaus.

Die Anonymisierung beeinträchtigt die Qualität der Daten, sobald sie über die formale Anonymisierung hinausgeht. Das Problem ist nun, einen Maßstab dafür zu gewinnen, wie weit Anonymisierungsmaßnahmen tatsächlich gehen müssen, um wirksam zu sein, und zu einem tragbaren Kompromiß zwischen Datenschutz und Anspruch an die Qualität der Daten zu kommen.

2.6 Das Identifikationsrisiko

Wie läßt sich allgemein das Risiko einschätzen, daß ein Datensatz in einer Datenbank identifiziert, also mit externem Wissen abgeglichen wird? Das Wesentliche dieser Situation wird in Abbildung III-8 dargestellt.

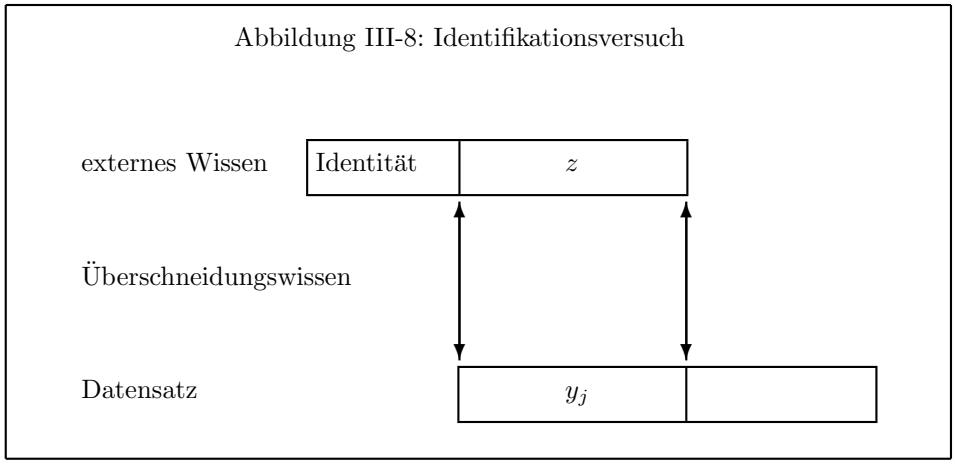
Der Angreifer hat einen externen Datensatz als Kenntnis über ein Individuum und weiß (zunächst sei das so angenommen), daß ein entsprechender Datensatz in der Datenbank existiert. Externes Wissen und Datensatz können beide fehlerbehaftet sein; für die folgenden Überlegungen kann man aber ohne Einschränkung ihrer Gültigkeit annehmen, daß das in z enthaltene externe Wissen exakt ist. Von den Datensätzen der Datenbank interessiert bei dem Identifikationsexperiment nur der Teil (mit y_j für den j-ten Datensatz bezeichnet), der dem Überschneidungswissen entspricht.

Der Angreifer möchte wissen, welcher der Datensätze y_j am wahrscheinlichsten zu seinem externen Datensatz z gehört; er möchte also die Wahrscheinlichkeitsverteilung kennen, deren Werte üblicherweise mit

$$P(y_j|z)$$

bezeichnet werden („die Wahrscheinlichkeit für y_j, wenn z gegeben ist“). Das ist eine Wahrscheinlichkeitsverteilung auf der Datenbank $\Omega = \{y_1, \ldots, y_N\}$ als

Abbildung III-8: Identifikationsversuch



Grundmenge. Der Angreifer kann sich eine Schranke p_0 vorgeben (etwa 90%) und einen Datensatz y_i als z identifiziert ansehen, wenn

$$P(y_i|z) \geq p_0.$$

Das Risiko des durch z beschriebenen Individuums, identifiziert zu werden, wird durch die „Risikofunktion“

$$p_{\max}(z) = \max_{i=1,\ldots,N} P(y_i|z)$$

ausgedrückt.

Die Verteilung $P(\bullet|z)$ kann man mit Hilfe eines BAYESschen Ansatzes durch Größen ausdrücken, die man als bekannt ansieht – obwohl man nur mehr oder weniger plausible Hypothesen über sie aufstellen kann. Diese Größen sind die Wahrscheinlichkeiten dafür, daß ein bestimmter Datensatz y_j zu einer „Beobachtung“ z führt, beschreiben also die Verteilung der Fehler in der Datenbank. Der Grundraum dieser Verteilung ist die Menge aller Beobachtungen, also $\mathbf{R}^k$, wenn der Datensatz aus k stetigen Merkmalen besteht, oder $\{0,1\}^k$, wenn es sich um k binäre Merkmale handelt, allgemein

$$\tilde{\Omega} = \Omega_1 \times \cdots \times \Omega_k,$$

wenn Ω_i der mögliche Wertebereich des i-ten Datenfeldes ist. Diese Verteilung werde mit $P(\bullet|y_j)$ bezeichnet; sie habe eine Dichtefunktion, die mit $f_P(\bullet|y_j)$ bezeichnet wird. Die „a-priori-Wahrscheinlichkeit“, also die Wahrscheinlichkeit, daß eine zufällige Beobachtung z zu y_j gehört, wenn man z noch gar nicht kennt, wird als

$$P(y_j) = \frac{1}{N}$$

angenommen; das heißt, wenn man nichts über z weiß, sind alle Datensätze gleich wahrscheinlich. (Der mathematisch strenge Leser möge die informelle Behandlung, insbesondere die Verwendung des gleichen Buchstabens P für drei verschiedene Verteilungen verzeihen.)

Die vom Angreifer gesuchten Größen lassen sich mit diesen Bezeichnungen durch die Formel von BAYES ausdrücken:

$$P(y_i|z) = \frac{f_P(z|y_i) \cdot P(y_i)}{\sum_{j=1}^{N} f_P(z|y_j) \cdot P(y_j)}.$$

Werden die a-priori-Wahrscheinlichkeiten $P(y_i) = 1/N$ eingesetzt, bleibt übrig:

$$P(y_i|z) = \frac{f_P(z|y_i)}{\sum_{j=1}^{N} f_P(z|y_j)}.$$

Ist die Fehlerverteilung $P(\bullet|y_i)$, oder genauer gesagt, ihre Dichte, für alle Datenfelder der Datenbank bekannt, so kann man daraus die Risikofunktion für jedes der gespeicherten Individuen explizit berechnen und zum Beispiel feststellen, für welchen Anteil das Risiko größer als p_0 ist. Auf diese Weise erhält man die Identifikationsquote als ein Maß für die Identifikationssicherheit der gesamten Datenbank.

In der Wirklichkeit ist die Fehlerverteilung (oder ihre Dichtefunktion) allerdings nicht bekannt, es sei denn, die Fehler sind absichtlich als Anonymisierungsmaßnahme nach einer bestimmten Zufallsverteilung erzeugt. Aber auch sonst kann man plausible Modellannahmen machen, etwa die Unabhängigkeit der Fehler der einzelnen Merkmale und, im Falle stetiger Merkmale, Normalverteilung mit bestimmten Parametern, die auf Erfahrungswerten beruhen. Auch dann ist die explizite Bestimmung des Identifikationsrisikos nur in ganz einfachen Fällen rechnerisch möglich, in komplizierteren Fällen ist sie noch abschätzbar. In allgemeinen, realistischen Fällen besteht das Überschneidungswissen aus verschiedenen Merkmalen mit komplexen Abhängigkeiten, und die Fehlerverteilungen sind kaum bekannt. Hier ist eine mathematische Behandlung nicht mehr möglich; man kann aber Simulationen („Identifikationsexperimente“) durchführen, um zu praktisch brauchbaren Ergebnissen über das Identifikationsrisiko in einer bestimmten Datenbank zu kommen. Solche Experimente wurden im AIMIPH-Projekt durchgeführt und werden im nächsten Abschnitt 2.7 behandelt.

In einer etwas veränderten Situation befindet sich der Angreifer, wenn er nicht weiß, ob sein Zielindividuum überhaupt in der Datenbank vertreten ist, wenn die Datenbank also die Stichprobeneigenschaft besitzt. Auch dieser Fall läßt sich ähnlich mathematisch modellieren, in einfachen Fällen explizit berechnen und in komplizierten Fällen simulieren.

2.7 Das AIMIPH-Projekt

Das Projekt AIMIPH (Konstruktion und Erprobung eines **a**nonymisierten **i**ntegrierten **Mi**krodatenfiles der bundesdeutschen **P**rivat**h**aushalte) wurde

1983-1985 bei der Gesellschaft für Mathematik und Datenverarbeitung (GMD) durchgeführt. Der vollständige Bericht ist als Buch [89] erschienen; der Artikel [88] gibt einen Überblick über die Ergebnisse.

Das Ziel waren operationalisierbare Kriterien zur Beurteilung der Anonymität von Daten. Zwei Ansätze wurden verfolgt:

1. Mathematische Verfahren zur expliziten Berechnung des Identifikationsrisikos unter Annahmen über die Verteilung der Fehler in den Daten.

2. Identifikationsexperimente zur empirischen Bestimmung der Identifikationsquote durch Simulation verschiedener Szenarien.

Die Grundlagen für die mathematische Modellierung sind im Abschnitt 2.6 skizziert. Die explizite Berechnung war nur in ganz einfachen Fällen durchzuführen, die aber durchaus schon einige Phänomene deutlich machten, vor allem, da sie zu einprägsamen graphischen Darstellungen führten, siehe [88]. Deutlich wurde, wie hoch das Identifikationsrisiko für einzelne „isoliert gelegene“ Datensätze ist; das sind Datensätze, für die es nicht viele ähnliche, also in der graphischen Darstellung nahebei gelegene gibt. In großen Haufen ähnlicher Datensätze ist dagegen das Identifikationsrisiko gering – natürlich nur unter der Annahme, daß die Streuung der Datenfehler nicht Null ist. Deutlich wurde auch, daß durch die Stichprobeneigenschaft der Datenbank das Identifikationsrisiko erheblich vermindert wird.

Der Hauptteil des Projekts war ein simulierter Angriff auf zwei Datenbestände, die aus realen Daten synthetisch zusammengestellt wurden:

EVS – die Einkommens- und Verbrauchsstichprobe 1978, aus der etwa 50000 Privathaushalte synthetisiert wurden; jeder Datensatz umfaßte 370 Haushaltsvariablen plus 35 Variablen für jede Einzelperson.

MZ – der Mikrozensus 1978; synthetisiert wurden ungefähr 230000 Privathaushalte mit jeweils 50 Haushaltsvariablen und zusätzlich 65 Variablen für jede Person.

Über die Verteilung der Erhebungsfehler wurden nach Beratung durch Experten des Statistischen Bundesamtes plausible Annahmen gemacht. Die Datensätze wurden formal anonymisiert, und extreme Merkmalsausprägungen wurden durch Zusammenfassen oder Weglassen verschleiert.

Für die Angriffsversuche wurden verschiedene realistische Szenarien konstruiert und mit einprägsamen Namen versehen:

Staatsanwalt – das Überschneidungswissen enthielt 68 Variablen, und zwar demographische Variablen und Steuermerkmale.

Steuerfahndung – das Überschneidungswissen enthielt eine Teilmenge von 45 Variablen des Szenarios „Staatsanwalt“.

Kriminalpolizei – hier wurde das Überschneidungswissen weiter auf 15 demographische Variablen eingeengt.

Adreßverlag – hier bestand das Überschneidungswissen aus 7 öffentlich zugänglichen Merkmalen wie Geschlecht und PKW-Besitz.

Der Angriffsversuch bestand jeweils in der gezielten Identifikation eines Datensatzes, wobei als Sicherheitsschwelle 90% angenommen wurde, wenn die Existenz des Datensatzes in der Datenbank als bekannt vorausgesetzt wurde, und 99.9%, wenn das nicht der Fall war, man also die Stichprobeneigenschaft der Datenbank ins Spiel brachte. Die niedrigere Sicherheitsschwelle bei bekannter Existenz ist ausreichend, da alle anderen Datensätze dann zusammen nur noch eine Wahrscheinlichkeit von höchstens 10% haben. Bei unbekannter Existenz muß man die Sicherheitsschwelle erheblich höher setzen, da jeder der 20 Millionen Haushalte der **Grundgesamtheit** als Konkurrenz zu betrachten ist. Die Ergebnisse sind in der Tabelle III-2 zusammengefaßt:

Tabelle III-2: Gezielte Identifikation eines Datensatzes

gemeinsame Variablen	Anteil der identifizierbaren Datensätze in %		Anteil der erfolgreichen Identifikationsversuche in % bei Stichprobeneigenschaft
	Existenz bekannt	Existenz unbekannt	
68	86	56	0.11
45	82	63	0.11
15	44	7	0.01
7	0	0	0.00

Die letzte Spalte zeigt das Identifikationsrisiko für Individuen (Haushalte) der Grundgesamtheit, die ja meistens nicht in der Datenbank enthalten sind. Es wird sehr deutlich, daß die Stichprobeneigenschaft der Datenbank der wirksamste Schutz vor Identifizierung ist. Ein wirksamer Schutz besteht auch, wenn das Überschneidungswissen gering ist. Ist jedoch das Überschneidungswissen groß und die Existenz des Datensatzes bekannt, so ist der einzelne Datensatz praktisch ungeschützt. Eine große Gefahr geht also von Datensammlungen aus, die viele Merkmale für einen genau abgrenzbaren Teil der Bevölkerung speichern.

Es wurde noch ein Zusatzexperiment ausgeführt, um den Einfluß der Fehlerverteilung (Erhebungsfehler oder absichtliche Zusatzfehler als Anonymisierungsmaßnahme) zu erkunden. Es stellte sich heraus, daß eine Vergrößerung der Fehler natürlich das Identifikationsrisiko mindert, aber nicht entscheidend; eine Vergrößerung der Fehler um den Faktor 4 reduzierte die Identifikationsquote auf etwa 1/10 – in kritischen Fällen ist das immer noch viel zu hoch. Ferner stellte sich heraus, daß es ziemlich egal ist, ob der Angreifer Kenntnisse über

Struktur und Ausmaß der Fehler hat. Als Folgerung wurde daraus gezogen, daß die Überlagerung der Daten mit Zufallsfehlern als Anonymisierungsmaßnahme nicht ausreichend wirksam ist; die dadurch entstehenden Schäden für die Brauchbarkeit der Daten sind höher einzuschätzen.

Alles dies bezog sich auf die gezielte Identifikation eines vorgegebenen Datensatzes. Viel kritischer ist das Sicherheitsproblem, wenn der Angreifer gar nicht an einem bestimmten Datensatz interessiert ist, sondern einen Fischzug startet. Die Arbeitsrichtung des Angreifers kehrt sich hierbei sozusagen um – er sucht „extreme“ Datensätze in der Datenbank und versucht dann, diese mit dem Überschneidungswissen abzugleichen. Als zweites Modell eines Angriffsversuch wurde ein solcher Fischzug durchgeführt; als Szenario wurde angenommen:

Journalist – sucht einen auffälligen Datensatz und versucht diesen zu identifizieren; als Überschneidungswissen werden 10 öffentlich zugängliche Merkmale unterstellt.

Das Ziel eines solchen Fischzugs könnte etwa die Diskreditierung von Anonymisierungsmaßnahmen sein. Besonders gefährdet sind dabei natürlich die Datensätze mit seltenen Wertekombinationen.

Dieser Angriff brachte trotz des geringen Überschneidungswissens eine ziemlich hohe Erfolgsquote. Die Schutzwirkung des geringen Überschneidungswissens hat sich also als illusorisch herausgestellt.

Als dritter und gefährlichster Angriffsversuch wurde noch ein Massenfischzug durchgeführt. Zugrundelegt wurden die Szenarien „Steuerfahndung“ und „Kriminalpolizei“, jeweils unter der Annahme daß die Werte der 45 beziehungsweise 15 Variablen des Überschneidungswissens für 25% der Individuen (also Haushalte) der Grundgesamtheit vorliegen; das könnte etwa die Sammlung aller Steuererklärungen für ein Jahr in einem Bundesland sein. Die Datenbank enthalte eine unbekannte Stichprobe. Tabelle III-3 gibt die Identifikationsquote in % wieder; die Werte der Spalte „gezielte Suche“ sind die der letzten Spalte von Tabelle III-2.

Tabelle III-3: Massenfischzug

gemeinsame Variablen	gezielte Suche	Massenfischzug
45	0.11	17.0
15	0.01	2.5

Maximal möglich wäre eine Identifikationsquote von 25%, denn nur von diesem Anteil aller Datensätze kann man überhaupt erwarten, daß ein entsprechender Datensatz im externen Wissen existiert. Vergleicht man insbesondere den Wert 17.0% mit dem maximal möglichen von 25%, so sieht man, daß das Identifikationsrisiko ausgesprochen groß ist.

Die Folgerung ist, daß die Verfügbarkeit von Massen-Datenbanken (man denke etwa an Volkszählungsdaten und Behördendateien) ein hohes Identifikationsrisiko für *alle anderen* Datensammlungen bewirkt. Die Massen-Datenbank braucht dabei selbst gar keine sensitiven Daten zu enthalten; gerade bei der Volkszählung wurde ja argumentiert, daß nur ohnehin leicht zugängliche Daten erhoben würden. Hat diese Massen-Datenbank ein hohes Identifikationsrisiko, sei es wegen fehlender Anonymität, sei es wegen ihrer Vollständigkeit und des Umfangs des möglichen Überschneidungswissens, so kann sie ihrerseits für Angriffe auf andere Datenbanken dienen.

Die Ergebnisse des AIMIPH-Projekts werden in folgenden Aussagen zusammengefaßt:

- Datenbänke mit Stichprobeneigenschaft und wenigen potentiellen Überschneidungsvariablen können – gegebenenfalls nach zusätzlicher Anonymisierung „extremer“ Datensätze – durchaus freigegeben werden.
- Datenbänke mit vielen Variablen sind nicht wirksam zu schützen. Anonymisierungsmaßnahmen müßten so drastisch sein, daß der Wert der Daten entscheidend verdorben wird.
- Der Datenbedarf der Wissenschaft ist nur auf der Grundlage einer „Forschungsklausel“ in den Datenschutzgesetzen zu befriedigen, die eine Verpflichtung auf den Datenschutz vorsieht und Auflagen macht, deren Einhaltung durch Sanktionen abgesichert wird.

Kapitel IV

Datensicherheit in Netzen

Für Großrechner mit überschaubarer Peripherie, die örtlich konzentriert ist, sind die Sicherheitsprobleme durch organisatorische und technische Maßnahmen in den Griff zu bekommen. Solche idyllischen Inseln existieren aber nur noch in seltenen Fällen. Typisch ist das Rechnernetz und der Fernanschluß an Großrechner. Und da sieht die Situation gleich ganz anders aus. Bezeichnend ist, daß ein nach dem Kriterien des 'Orange Book' klassifiziertes System seine offizielle Klassifikation sofort verliert, wenn es in ein Netz mit Protokollen wie TCP/IP und NFS eingebunden wird.

Netze sind verteilte Systeme mit Knoten, also angeschlossenen Rechnern, und einem Nachrichtentransportsystem, also Datenleitungen. Die Knoten lassen sich in der Regel hinlänglich schützen; die Datenleitungen sind nicht notwendig geschützt und lassen sich oft kaum schützen. Nach dem momentanen Stand der Technik sind Netze grundsätzlich als physisch angreifbar zu betrachten. Wirksame Sicherheitsfunktionen sind nur mit kryptographischen Methoden zu verwirklichen. Die Kryptographie bietet Lösungsansätze für alle wesentlichen Probleme, diese sind aber noch längst nicht in genügendem Umfang technisch realisiert; oft gibt es Schwierigkeiten bei der Realisierung, zum Beispiel in Hinsicht auf Verarbeitungsgeschwindigkeit. Noch schlechter sieht es aus, wenn man in die tatsächliche Praxis sieht. Beim Betrieb von Netzen klaffen erhebliche Sicherheitslücken, und existierende Sicherheitsprodukte werden aus Mangel an Kenntnis oder aus Kostengründen nicht eingesetzt. Schließlich ist es auch ohne Sicherheitsvorkehrungen schon kompliziert genug, ein Netz überhaupt ans Laufen zu bekommen.

Manche Datenschützer fordern, daß Rechner, auf denen sensitive Daten liegen, überhaupt nicht in Netze eingebunden werden dürfen. Selbst bei den halbwegs wirksamen Schutzmaßnahmen, die der Gegenstand dieses Kapitels sind, schleichen sich immer wieder Fehler ein, die prompt ausgenützt werden, wie viele der Vorfälle mit Hackern gezeigt haben. Trotzdem scheint dieser rigorose Standpunkt nicht zwingend notwendig; es muß nur verhindert werden, daß

die sensitiven Daten über das Netz unkontrolliert zugänglich sind oder unverschlüsselt gesendet werden. Dazu braucht man möglichst sichere Netzbetriebssysteme, ein wirksames Netzmanagement und zusätzliche Sicherheitseinrichtungen.

1 Typen von Netzen

Netze lassen sich nach verschiedenen Gesichtspunkten unterscheiden; hier soll unter dem Aspekt der Datensicherheit in erster Linie zwischen **öffentlichen** und **lokalen Netzen** unterschieden werden. Diese Unterscheidung ist nicht scharf; zum Beispiel gibt es ja auch viele private Netze auf öffentlichen Trägern wie etwa das SWIFT-Netz für den Zahlungsverkehr von Banken und Sparkassen. Ähnlich ist die Unterscheidung in **Fernverkehrsnetze** ('Wide Area Network', WAN) und lokale Netze ('Local Area Network', LAN), die aufgrund der Entfernung und der verwendeten Technik getroffen wird. Zur Beurteilung der Datensicherheit besonders bedeutsam ist die Unterscheidung zwischen:

Diffusionsnetz, in dem alle Stationen fest am Netz hängen, jeweils den gesamten Datenverkehr mithören und die für sie bestimmten Nachrichten ausfiltern; gesendet wird per 'broadcasting'.

Vermittlungsnetz, in dem bei Bedarf zwischen zwei Stationen aus vorhandenen Teilstrecken eine direkte Verbindung geschaltet wird (wie beim Telefon).

Wichtig ist auch, ob eine Verbindung fest geschaltet oder ob sie eine Wählverbindung ist. Im ersten Fall kommunizieren die Rechner über fest definierte Adressen, die man eventuell zur Definition eines besonderen Zugangsschutzes verwenden kann. Im zweiten Fall handelt es sich dagegen oft um logische Adressen, die bei Bedarf zugeteilt werden und bei denen man vorher nicht weiß, welche Netzstationen wann verbunden sind.

Die verschieden Arten von Netzen werden hier nur ganz kurz beschrieben, soweit es zum Verständnis der Sicherheitsprobleme notwendig ist. Für eine ausführliche Behandlung sei auf [59] und andere Spezialliteratur verwiesen.

1.1 Öffentliche Netze

Die typischen Merkmale eines öffentlichen Netzes sind:

- Der Zugang steht im Prinzip jedem frei; es ist ein offenes System.
- Es ist ein Vermittlungsnetz.
- Die Daten werden einem fremden Netzbetreiber anvertraut (Post, Telekom), auf dessen Sicherheitsmaßnahmen man angewiesen ist.

- Die Übertragungsgeschwindigkeit ist niedrig; die 64 Kbit/sec von ISDN sind hier schon als sehr schnell anzusehen.
- Die Laufzeit ist lang; Echtzeitanwendungen sind kaum realisierbar.
- Die Fehlerrate ist vergleichsweise hoch; die verwendeten Protokolle müssen gute Fehlererkennungs- und Wiederaufsetzmechanismen bieten.

Wichtige Beispiele für öffentliche Netze sind das Datex-P-Netz, das auf dem X.25-Protokoll beruht, das EARN ('European Academic Research Network'), das deutsche Wissenschaftsnetz (WIN) oder Deutsche Forschungsnetz (DFN), das Btx-Netz und die amerikanischen Netze Arpanet, Internet, Bitnet, und viele andere.

Das ISDN befindet sich gerade im Anfangsstadium, obwohl die Sicherheitsbedenken noch längst nicht ausgeräumt sind. Es soll der Übermittlung von Sprache, Daten, Text und Bildern dienen. Weiteres Ziel für die 90er Jahre ist ein integriertes Breitbandfernmeldenetz, das sogar zur Übertragung von bewegten Bildern geeignet ist und auch Rundfunk und Fernsehen umfaßt. Der zentrale Betreiber, die Telekom, ist auch eine große Sicherheitsschwachstelle; schon allein die Sammlung der Verbindungsdaten ermöglicht umfassende Teilnehmerprofile.

In Zukunft soll die Sicherheit des ISDN durch die sogenannten Diensterweiterungen erhöht werden:

- Verschlüsselung,
- sichere Identifizierung,
- Datenkontrolle gegen Manipulation,
- elektronische Unterschrift,
- Empfangsbestätigung.

Hiergegen gibt es aber auch durchaus ernstzunehmende Einwände, denn diese Maßnahmen erschweren das Abhören und behindern die Verbrechensbekämpfung.

1.2 Lokale Netze

Die typischen Merkmale eines lokalen Netzes sind:

- Es befindet sich auf Privatgelände; der Zugang ist physisch beschränkt.
- Es ist ein Diffusionsnetz.
- Dem Netzbetreiber gehören auch die Knotenpunkte und die übermittelten Daten.

- Die Übertragungsgeschwindigkeit ist hoch (zum Beispiel 10 Mbit/sec bei Ethernet; man muß aber wissen, daß solche Angaben Bruttoraten sind – der Endanwender sieht viel geringere Übertragungsraten).
- Die Laufzeit der Daten ist kurz, die Zugriffsgeschwindigkeit ist mit einem Festplattenzugriff auf einem PC vergleichbar.
- Die Fehlerrate ist gering.

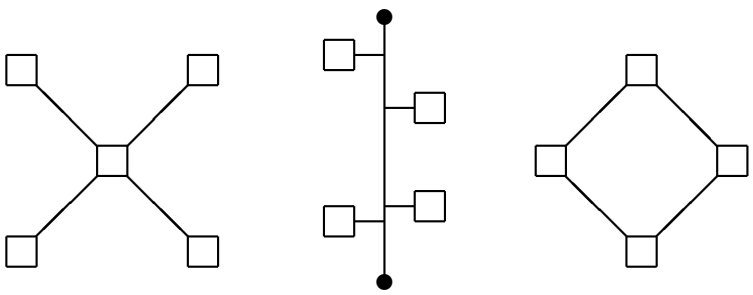
Der physische Aufbau des Netzes wird durch einen Graphen (im mathematischen Sinne) beschrieben und als Topologie bezeichnet. Man unterscheidet hauptsächlich (siehe Abbildung IV-1)

Stern – alle Knoten sind direkt mit einer Zentrale verbunden und kommunizieren stets über diese.

Bus – alle Knoten hängen linear angeordnet an einem Kabel, das von Abschlußwiderständen beendet wird.

Ring – die Knoten sind zyklisch an einem Kabel aufgereiht, Nachrichten laufen immer im Kreis herum. Physisch wird ein solcher Ring meistens auch als Stern realisiert: In der Mitte steht ein Ringleitungsverteiler, von dem aus jeder Knoten mit je einer Sende- und Empfangsleitung angefahren wird, und in dem alle diese Leitungen zu einem Kreis geschlossen werden.

Abbildung IV-1: Topologien lokaler Netze



Die gängigsten Protokolle auf niedriger Ebene sind

- CSMA/CD ('Carrier Sense Multiple Access with Collision Detection'); eine Station darf senden, wenn die Leitung frei ist – das erkennt sie durch

Abhören des Mediums. Probleme (Kollisionen) treten auf, wenn zwei Stationen quasi gleichzeitig zu senden beginnen. Solche Konflikte werden durch das Protokoll gelöst. Bei hoher Belastung sinkt die Durchsatzrate wegen der ständigen Kollisionen oft drastisch, denn keine Station kommt mehr zum Senden.

- Token-Protokolle, die meist im Ring verwendet werden. Wird nicht gesendet, kreist ein Freizeichen ('Token') im Ring. Wer senden will, nimmt das Freizeichen vom Ring, sendet seine Nachricht und erzeugt dann wieder das Freizeichen.

Das erste Protokoll wird beim **Ethernet** benützt, das zweite beim **Token-Ring**. Mit einem Marktanteil von über 50% ist Ethernet mit CSMA/CD das am weitesten verbreitete Protokoll auf dieser Ebene.

1.3 Protokoll-Welten

Es gibt eine ganze Reihe von „Welten", also von Protokoll-Gruppen, die meist wenig miteinander verträglich sind und jeweils ihre eigenen Begriffsbildungen haben. Die wichtigsten davon sind:

- TCP/IP ('Transmission Control Protocol/Internet Protocol'), der herstellerunabhängige de-facto-Standard: Alle wichtigen Computersysteme bieten TCP/IP-Implementierungen.

- SNA ('Systems Network Architecture'), das Netz-Protokoll für IBM-Großrechner, mit dem sich aber auch viele andere Systeme in die „IBM-Welt" einbinden lassen.

- DECnet, das herstellerspezifische Protokoll von DEC, das die Großrechner dieser Firma und PCs verbindet.

- PC-LANs, vor allem Novell,

- OSI, die zukünftige einheitliche Netzumgebung, die in internationalen Normen festgelegt wird. Etwas genauer wird dies in Abschnitt 1.4 behandelt.

Am weitesten verbreitet und universell einsetzbar ist TCP/IP, sowohl im Fernverkehr als auch im lokalen Netz als Minimalstandard. OSI ist dagegen bisher kaum verbreitet, einerseits, weil wesentliche Dienste noch nicht genormt sind, andererseits, weil existierende Implementierungen sehr zu wünschen übrig lassen. Noch ist die Ablösung von TCP/IP durch OSI nicht in Sicht.

1.4 Die OSI-Schichten

Im zukünftigen Standard OSI werden 7 Schichten unterschieden:

7 Anwendungsschicht ('Application Layer')

6 Darstellungsschicht (Präsentationsschicht, 'Presentation Layer')

5 Kommunikationssteuerungsschicht (Sitzungsschicht, 'Session Layer')

4 Transportschicht ('Transport Layer')

3 Vermittlungsschicht (Netzschicht, 'Network Layer')

2 Sicherungsschicht (Verbindungsschicht, 'Link Layer')

1 Bitübertragungsschicht (physikalische Schicht, 'Physical Layer')

Darunter liegt, quasi als Schicht 0, noch das Medium. Siehe auch [43, S.266]. Diese Schichteinteilung ist auch in den anderen Protokollwelten mehr oder weniger erkennbar und wird daher oft für theoretische oder vergleichende Betrachtungen benützt.

Insgesamt dienen die unteren Schichten bis 3 der Vermittlung von Leitungen, Nachrichten und Paketen, der optimalen Wegfindung, der Fluß- und Lastkontrolle und -steuerung.

Schicht 3, die Netzschicht, enthält 'Routing Tables' und die Algorithmen für die Wegfindung (zum Beispiel 'Spanning Tree Algorithmus'). Hier ist auch die eindeutige Adressierung im Fernverkehrsnetz angesiedelt.

Schicht 2, die Verbindungsschicht enthält die eindeutige Adressierung im lokalen Netz und auch eine elementare Fehlerkontrolle.

Schicht 4, die Transportschicht, sorgt für Verbindungsaufbau, Datentransport und Verbindungsabbau.

Schon auf den unteren Ebenen sind Schutzmaßnahmen nötig; allerdings ist der Datenschutz hier nur unvollkommen zu realisieren; er ist im wesentlichen Aufgabe der höheren Schichten und liegt letztlich in der Verantwortung des Netzteilnehmers.

1.5 Netzkomponenten

Die Knotenpunkte im Netz, die durch das Nachrichten-Transportsystem verbunden werden, können unterschiedliche Funktionen haben:

Netzstationen sind die an das Netz angeschlossenen Rechner, die miteinander kommunizieren wollen.

Server sind Knoten, an denen Dienste bereitgestellt werden, die von Benutzern an Netzstationen in Anspruch genommen werden können. Typisch sind Drucker, File-Server, Datenbank-Server, Mailboxen.

Hosts sind im Grunde das gleiche wie Server. Meistens versteht man unter einem Host allerdings einen Großrechner, der eine ganze Reihe von Server-Funktionen gleichzeitig bietet und dazu auch noch die Möglichkeit des direkten Dialogs ('Remote Login', Terminalemulation).

Gateways sind spezielle Server, die den Übergang zwischen Teilnetzen, auch Teilnetzen mit unterschiedlichen Protokollen, vermitteln, im Idealfall so, daß der Benutzer nichts davon merkt. In der Regel sind dies PCs oder sogar größere Rechner mit der entsprechenden Software.

Bridges (Brücken) sind Geräte, die den Übergang zwischen gleichartigen Netzen vermitteln. Sie arbeiten in der Regel auf den unteren Protokollschichten (OSI-Schichten 1 und Teile von 2). Meistens haben sie Filterfunktionen: Sie wissen, welche Adressen auf ihren beiden Seiten liegen, und lassen nur Nachrichten durch, deren Empfänger auf der anderen Seite liegt. Die Konfiguration der Adressen kann vorgegeben oder durch Selbstlernen erworben werden, indem sich die Bridge bei eingehenden Nachrichten die Adresse und die Richtung merkt.

Router haben im Prinzip die gleiche Funktion wie Bridges, arbeiten aber auf der OSI-Schicht 3, das heißt, sie wirken bei der Wegfindung vom Sender zum Empfänger aktiv mit; sie können den Nachrichten selbst Adreß-Informationen hinzufügen, die der Absender nicht zu kennen braucht.

Repeater sind reine Leitungsverstärker.

2 Netzdienste

In diesem Abschnitt werden Netze aus der Sicht des Benutzers betrachtet. Was will er mit Hilfe des Netzes erreichen? Alle solchen Netzdienste werden von Anwenderprozessen zur Verfügung gestellt, die die Dienste der OSI-Schichten (oder vergleichbare „elementare“ Dienste) nutzen, selbst aber darüber angesiedelt sind.

In jedem Fall muß man bei der Datenübertragung unterscheiden zwischen:

Nutzdaten, den eigentlichen Daten, die der Benutzer übertragen will,

Verbindungsdaten: Ziel- und Herkunftsadresse, Zwischenadressen, Zeit, Angaben über den Datenumfang.

2.1 Nachrichten und Post

Nachrichten sind kurze (einzeilige) Texte, die an einen anderen Teilnehmer des Netzes gesendet werden. Ist er empfangsbereit und die Verbindungsstrecke intakt, wird ihm die Nachricht sofort mitgeteilt. In allen anderen Fällen ist

die Nachricht verloren. Das tritt zum Beispiel ein, wenn der Zielknoten oder irgendein Zwischenknoten nicht in Betrieb ist; oder ein Benutzer eines Großrechners nutzt die Möglichkeit, den Nachrichtenempfang zu unterdrücken, etwa damit Bildschirmausgaben nicht unterbrochen werden. An Systeme, die nur einen Prozeß bedienen können (PCs), kann man keine Nachrichten schicken. Auch zwischen unterschiedlichen Rechner- oder Betriebssystemen klappt diese Form der Kommunikation in der Regel nicht.

Dieser Dienst heißt 'Message Transfer'; er bietet die interessante Möglichkeit der direkten Kommunikation über das Netz. Diese Möglichkeit wird auch oft benutzt, um Prozesse fernzusteuern: Sie warten dann auf Nachrichten und setzen diese gegebenenfalls in entsprechende Aktionen um. Das Risiko eines solchen Verfahrens wird in Abschnitt 3.3 analysiert.

Eine andere Form der Kommunikation zwischen Teilnehmern ist die elektronische Post ('electronic mail'; in der OSI-Welt wird hierfür allerdings ebenfalls die Bezeichnung 'Message Transfer' benützt). Hier ist es nicht nötig, daß der Kommunikationspartner empfangsbereit ist; er hat ein elektronisches Postfach ('Mailbox'), in dem die eingehende Post abgelegt wird. In Großrechnern ist das Postfach meist Teil des Betriebssystems. PCs sind auf einen separaten Server angewiesen, bei dem sie ihre Post je nach Laune abholen können. Damit die Post auch wirklich sicher ankommt, wird nach dem 'store-and-forward'-Prinzip gearbeitet: Jeder Zwischenknoten, angefangen beim Ursprungssystem, hält die Sendung so lange zurück, bis sein unmittelbarer Nachfolger auf der Strecke empfangsbereit ist. Dadurch entsteht natürlich ein Sicherheitsproblem, wenn die Durchgangsstationen nicht hinreichend gegen Ausspähen der zwischengelagerten Post geschützt sind.

Die elektronische Post arbeitet mit Dateien, die aus einem Briefkopf ('header') in festem, vorgeschriebenen Format und einem Textteil bestehen. Der Briefkopf ist leicht fälschbar, also kein echter Absendernachweis. Da Post meistens über unsichere und unkontrollierbare Wege übermittelt wird, können Absender- und Empfängerangaben im Zweifelsfall nicht als authentisch angesehen werden.

Mit Hilfe von Mail-Systemen lassen sich leicht elektronische Konferenzen, Anschlagbretter und anonyme Konferenzen ('Chat') verwirklichen, und das wird auf den akademischen Netzen auch gemacht. Im EARN gibt es auch eine Reihe von Datei-Servern ('Listserv', 'Netserv'), die per Post zum Aussenden von Dateien veranlaßt werden können, etwa von elektronischen Zeitschriften oder Public-Domain-Programmen.

2.2 Datentransfer

Während elektronische Post über Systemgrenzen hinweg ohne große Probleme funktioniert (wenn wir mal nicht an die deutschen Sonderzeichen denken wollen), ist ein echter 'File Transfer', also ein Transport beliebiger, auch binärer oder formatierter, Dateien nur zwischen gleichen oder ähnlichen Systemen

möglich und sinnvoll. Ein anderer Unterschied zur Post ist, daß beim File-Transfer die Initiative auch vom Empfänger ausgehen kann. Die Sicherheitsprobleme werden noch behandelt.

2.3 Hintergrund-Prozesse

Großrechnersysteme bieten meistens die Möglichkeit, Programme, die keinen Dialog benötigen („Stapelverarbeitung“, ‘Batch Processing’) auch über ein Netz zu starten (‘Remote Job Entry’); auch Daten und Programmanweisungen sind oft dazu zu übermitteln. Dieses ist der klassische Fall der Datenfernverarbeitung. Auch in PC-Netzen gibt es solche Möglichkeiten in geringerem Umfang, zum Beispiel Drucken, allgemeiner: Inanspruchnahme entfernter Dienstleistungen.

2.4 Dialog

Eine modernere Möglichkeit ist, daß eine Netzstation sich als Terminal für eine andere definiert; man spricht von ‘Remote Login’, virtuellem Terminal oder Terminalemulation. Dabei werden die auf Großrechnern implementierten Sicherheitsvorkehrungen oft aufgeweicht, wenn solche Netzknoten als ‘Logical Devices’ behandelt werden, die nicht festen Adressen zugeordnet sind: Die sehr wirkungsvolle Sperre von Terminal-Adressen bei mißbräuchlichen Login-Versuchen wird erschwert.

Wird der Dialog übers Netz von einem „intelligenten“ Terminal aus geführt, ermöglicht er oft gleichzeitig einen echten Datentransfer (‘download’), meistens aber wenigstens eine primitive Form davon: Man startet einen Mitschnitt (‘console log’) und läßt sich die Daten dann auf den Bildschirm ausgeben; das bekannte Kommunikationsprogramm Kermit bietet zum Beispiel beide Möglichkeiten (zum echten Datentransfer muß der Zielrechner die Kermit-Server-Funktion bieten, für die primitive Form ist das nicht nötig). Auf diese Weise bietet eine Lese-Erlaubnis eine Hintertür zur Kopier-Erlaubnis. Um das zu verhindern, kann man diskettenlose Arbeitsplätze einrichten; aber auch dann gibt es noch Angriffspunkte, etwa die Umleitung der Terminalausgabe auf den Drucker. Schließlich kann ein Angreifer noch den Bildschirm mit den gesuchten Daten abfotografieren, aber das ist keine EDV-typische Gefährdung, denn so wurden und werden auch papierene Akten ausgespäht.

2.5 Verteiltes Dateisystem

Eine höhere Stufe des File-Transfers ist ein ‘Network File System’, ein netzweites einheitliches Dateisystem. Dieses ermöglicht einem Netzknoten, Daten auf einem anderen Knoten so zu behandeln wie eigene Daten, ohne daß der Benutzer es merkt (abgesehen vielleicht von einer zusätzlichen Sicherheitsüberprüfung). Ein Spezialfall ist die Möglichkeit, für einen im Netz befindlichen PC auf einem Server virtuelle Platten anzulegen, die einfach wie ein weiteres

Laufwerk angesprochen werden. Am bekanntesten ist das NFS, das von SUN entwickelt wurde, im UNIX-TCP/IP-Bereich weit verbreitet ist und auch oft für File-Server in PC-Netzen eingesetzt wird. Das ebenfalls von SUN entwickelte YP ('Yellow Pages') stellt Systemparameter-Dateien netzweit zur Verfügung, selbst Paßwort-Verzeichnisse; man bezeichnet solche Dienste auch als NIS ('Network Information Services').

2.6 Verteilte Anwendungen

Die höchste Stufe der Rechnerkopplung im Netz ist eine verteilte Anwendung. Hier arbeiten die Rechner zusammen, als ob sie ein einzelner Rechner wären ('Distributed Processing'); man spricht auch von Programm-zu-Programm-Kommunikation. Ein Beispiel (der unangenehmen Art) sind die Würmer, Programme, die aus mehreren Segmenten auf verschiedenen Rechnern bestehen, die zusammenarbeiten und sich gegebenenfalls auf weitere Rechner ausdehnen. Als neueres Beispiel kann man auch das X-Window-System ansehen, das hauptsächlich im Workstation-Bereich unter UNIX die Möglichkeit einer verteilten Benutzeroberfläche bietet.

2.7 Netzbetriebssystem

PC-Netze werden meistens mit einem Netzbetriebssystem ausgestattet (z. B. „Novell"), das einen einheitlichen Zugang mit Paßwortschutz und eine gemeinsame („transparente") Verwaltung der Ressourcen bietet. Ein solches Netz präsentiert sich dem Benutzer ähnlich wie ein Großrechner-/Mehrbenutzer-System.

3 Gefahren

Datennetze sind vielfältigen Gefahren ausgesetzt. Der Hauptgrund ist, daß sie allein aufgrund ihrer Ausdehnung nicht leicht physisch zu schützen sind. Ein weiterer Grund ist aber auch, daß die technischen Möglichkeiten sehr schnell in die Praxis umgesetzt wurden ohne Rücksicht auf Gefahren. In der hektischen Anstrengung, überhaupt erst einmal funktionsfähige Netze zu schaffen, wurde der Sicherheit zunächst wenig Beachtung geschenkt. Immer wieder haben Hacker gezeigt, wie einfach man die Sicherheitslücken ausnützen kann.

Hier wie auch sonst gilt der Grundsatz, daß Schutz nur gegen bekannte Gefahren möglich ist. Daher ist es wichtig, einen Überblick über diese zu gewinnen, und damit zu einer realistischen Risikoabschätzung für das eigene System zu kommen. Gerade bei Netzen gibt es aber auch wegen der Komplexität der Software viele nicht dokumentierte Sicherheitslücken, gegen die man sich mangels Kenntnis nicht gezielt wappnen kann, sondern wo man sich mit allgemeinen vorbeugenden Maßnahmen begnügen muß.

Als möglicher Angreifer kommt übrigens auch der Hersteller der Netzkomponenten und ihrer Software in Betracht. Wer kann schon garantieren, daß die Netzsoftware keine Trojanischen Pferde enthält!

3.1 Lauschangriffe

Datenverkehr abzuhören ist bei allen Arten von Netzen für einen Angreifer ein lohnendes Ziel. Der am leichtesten zugängliche Angriffspunkt ist die Leitung, also das physische Übertragungsmedium. Zunächst ist die Lage der Leitung zu beachten:

- Überlandleitung,
- unterirdische Leitung,
- Kabelschächte, -kanäle,
- Richtfunkstrecken,
- Satellitenübertragung.

Funkstrecken sind natürlich besonders leicht abhörbar, zumal sie in der Regel über unbewachtes Gelände laufen oder genügend weit streuen. Aber auch bei Kabelstrecken kann die elektromagnetische Abstrahlung interessante Abhörmöglichkeiten bieten („Nebensprecheffekt“), die eventuell sogar unabsichtlich ausgenützt werden.

Außer der Abstrahlung bieten die Übertragungsmedien die folgenden Möglichkeiten:

- Anzapfen der Leitung (‘wire tapping’),
- Angriff auf spezielle Kommunikationseinrichtungen wie Modems, Knotenrechner, Brücken,
- Schnittstellen (Stecker raus, Schnittstellentester dazwischen, Stecker wieder rein),
- Besetzung unbenutzter Anschlußpunkte.

Dazu kommen auf den höheren Protokollschichten noch die Möglichkeiten, einen regulären Anschluß zu manipulieren, etwa durch Fälschen der eigenen Adresse im IP-Protokoll, oder Systemfehler und -lücken zu mißbrauchen. Mit diesen Möglichkeiten kann man nicht nur lauschen, sondern auch Daten fälschen.

Lichtwellenleiter (Glasfaserkabel) bieten den Vorteil, nicht elektromagnetisch zu strahlen. Auch sind sie vergleichsweise schwer anzuzapfen. Wird dazu die Verbindung eine Zeitlang unterbrochen, führt das vielleicht schon zur Entdeckung. Es gibt aber auch die Möglichkeit, durch starkes Biegen den Totalreflexionswinkel zu unterschreiten, so daß seitlich am Kabel ein Teil des Lichtstroms

austritt. Es gibt sogar Geräte zur Dämpfungsmessung, die nach diesem Prinzip funktionieren. Auf jeden Fall wird die Dämpfung im betroffenen Abschnitt des Lichtwellenleiters während dieses Anzapfversuchs erhöht, was bei geeigneter Ausstattung der Nachbarknoten entdeckt wird.

Im Gegensatz dazu sind Metallkabel leicht und unbemerkt anzuzapfen; bei den Koaxialkabeln für das Ethernet wird sogar damit geworben: Das Einfügen neuer Stationen geschieht unterbrechungsfrei durch einen solchen Anzapfvorgang. Dazu sind im Handel „Vampirkrallen“ erhältlich, mit denen man einen ‘TAP’ (‘Terminal Access Point’) herstellt. Zitat aus der Werbung: „Easy Tap Kits . . . machen Netzwerk-Stops unnötig. Sie zapfen einfach mit dem Easy Tool das Kabel an, schieben den Tap über das Kabel und ziehen fest an. Kein Quetschen oder Löten nötig!“

Natürlich spielt beim Abhören eine wesentliche Rolle, in welcher Weise die Nachrichten über das Netz gehen. Modulations- und Multiplexverfahren erzeugen ein Signal- oder Nachrichtengemisch, das für den Lauscher eventuell schwer zu interpretieren ist. Stark strukturierte kurze Nachrichtenblöcke, etwa Logon-Sequenzen mit Paßwörtern, sind aber leicht aus dem „Datenmüll“ auszufiltern. In Vermittlungsnetzen ist oft nicht vorhersagbar, über welchen Weg eine Nachricht tatsächlich läuft; es können sogar (bei Paketvermittlung) verschiedene Teile einer Nachricht über verschiedene Wege laufen. Der Lauscher kann hier nur mit Zufallstreffern rechnen. Aber um Datenschutzmaßnahmen zu diskreditieren, reicht das allemal (Fischzug). Besonders geeignete Angriffsziele sind dann aber Start und Ziel oder Leitungsabschnitte in deren Nähe.

In Fernverkehrsnetzen entstehen auch durch das Abhören von Verbindungsdaten Datenschutzprobleme – die Nutzdaten kann der Benutzer in eigener Verantwortung durch Verschlüsselung schützen, auf die Verbindungsdaten hat er keinen Einfluß. Sie ermöglichen eine Verkehrsflußanalyse und die Erstellung von Teilnehmerprofilen; oft ist schon die Tatsache einer Kommunikation aufschlußreich, etwa bei Börsen-Aktivitäten oder militärischen Aktionen.

In lokalen Netzen ist das Abhören von Verbindungsdaten wohl nur selten als Problem anzusehen. Dafür ist hier, durch die Diffusionseigenschaft des Netzes, das Abhören insgesamt recht leicht. Natürlich erhält der Angreifer, zum Beispiel wenn er einen Schnittstellentester verwendet, zunächst einmal einen riesigen Berg Datenmüll, der wie ein großes Puzzle erst richtig zusammengesetzt werden muß. Aber zum Filtern und Verarbeiten hilft ein kleines PC-Programm schon sehr, so daß letzten Endes die Interpretation der abgehörten Datenmassen doch leicht ist.

Dataskope oder Schnittstellen-Analysatoren braucht man zur Fehlersuche im Netz. Es gibt sie für Verbindungen aller Art. Man zieht einen Verbindungsstecker ab und stöpselt das Gerät dazwischen. Werbespruch: „. . . nutzen Sie sowohl für die üblichen ‚Kleinigkeiten’, also Pin-Belegung, Brücken oder Baudrate auskundschaften, wie auch für die schwierigeren Fälle

- Datenströme analysieren
- Protokolle und Handshake offenlegen
- Dialoge beobachten
- Leitungen testen
- ...“

Wer keinen Schnittstellentester hat, kann zumindest bei einem Ethernet mit einem TAP genausogut abhören; dabei entfällt sogar das entdeckungsgefährdete kurzzeitige Unterbrechen der Leitung. Bei Token-Ring-Protokollen ist dagegen das Einklinken unberechtigter „echter Netzstationen“ vergleichsweise schwer. Erstens muß jede Station aktiv am Datenverkehr teilnehmen, so daß der Eindringling wissen muß, wie man sich unauffällig verhält. Zum andern müssen neue Stationen aber auch angemeldet werden, wenn sie am Datenverkehr teilnehmen wollen, und das geht nicht unbemerkt.

Ein zusätzliches Abhörproblem entsteht in lokalen Netzen dadurch, daß oft Teile des Betriebssystems über das Netz verschickt werden (‘download’). Das dient zur Vereinfachung der zentralen Netzverwaltung, ermöglicht einem geschickten Angreifer aber auch, Trojanische Pferde einzuschmuggeln.

Verschlüsselung ist letzten Endes der beste Schutz für Datenübertragung auf Kommunikationsleitungen. Aber die Verfügung über ein gutes Verschlüsselungsverfahren löst noch längst nicht alle Probleme. Wie soll etwa das ‘Remote Login’ gestaltet werden? Abhörsichere Erkennungsprozeduren und Lösungen ähnlicher Probleme werden im Kapitel V behandelt.

3.2 Datenverfälschung

Datenverfälschung ist in angezapften Leitungen oder Funkstrecken nur mit großem Aufwand möglich. Vergleichsweise leicht ist sie dagegen, wenn man sich in den Besitz eines Knotens gebracht hat. Protokolle, insbesondere auf der Basis von Ethernet und CSMA/CD, die erlauben, jederzeit neue Stationen ins Netz einzuklinken und ihre Adresse selbst zu definieren, sind besonders gefährdet. Sie erleichtern die Maskerade. Aber auch sonst ist über jeden unbenutzten und unbewachten Anschlußpunkt ein aktiver Angriff möglich. Solche Angriffe auf höheren Verbindungsschichten befreien den Angreifer von der mühsamen Imitation der nötigen Protokoll-Informationen auf den unteren Schichten. Andererseits reicht ein Abhören auf der untersten Schicht aus, um, wenn auch mit Mühe, die nötigen Informationen über den Netzaufbau und die Protokolle der höheren Ebenen zu gewinnen.

Gefahren sind:

- Wiederholen von Nachrichten – auch diese einfache Fälschung kann schon schaden, man denke etwa an Geldüberweisungen; selbst Verschlüsselung bietet hier keinen vollständigen Schutz.

- Einfügen zusätzlicher Daten.
- Modifikation der Daten, zum Beispiel Änderung einer Kontonummer bei einer finanziellen Transaktion.
- Zusammenstückeln von Nachrichten: Aus mehreren abgefangenen authentischen Nachrichten wird eine gefälschte konstruiert. Als analoges Beispiel kann man sich vorstellen, daß eine Unterschrift ausgeschnitten und auf ein anderes Dokument geklebt wird, das anschließend über Telefax gesendet wird.
- Löschen von Nachrichten – abfangen, ohne weiterzuleiten.
- Einschleusen nicht erlaubter Nachrichten, etwa Verbreitung von Viren.
- Manipulation der Verbindungsdaten an einem Gateway – hier lassen sich oft Absenderadressen ändern, so daß Nachrichten maskiert sind und sich auch nicht auf den wahren Absender zurückverfolgen lassen.

3.3 Fernzugriffe

Fernzugriffe, also Zugriffe von einem Rechnersystem auf ein anderes, entstehen bei allen Arten von Netzdiensten mit Ausnahme der gewöhnlichen elektronischen Post (es sei denn, ein Server antwortet automatisch auf Post, wie etwa File-Server im EARN). Das können Zugriffe auf Daten oder sonstige Ressourcen wie Peripheriegeräte sein, auch die Durchwahl in ein am Zielrechner angeschlossenes weiteres Netz.

Die Probleme sind im wesentlichen die bereits behandelten: Identifikation und Authentisierung sowie Zugriffsrechte, wobei durch den Zugriff über das Netz die Gefahren verschärft sind:

- Paßwörter können abgehört und dann mißbraucht werden.
- Die Paßwortverwaltung läßt sich in einem verteilten System nicht so leicht regeln wie in einem geschlossenen.
- Arbeitsplatzrechner als Netzstationen haben eigene Datenverarbeitungskapazität („Intelligenz“), die sie zum Ausprobieren von Paßwörtern, zur Usurpation von Netzverwaltungsfunktionen oder zum Ausfiltern interessanter Datenpakete nutzen können. Als Netzstationen sind sie selbst meistens wenig geschützt, also Manipulationen ausgesetzt.
- Verschlüsselungen sind angreifbar, da viele konstante Texte übermittelt werden und oft der zugehörige Klartext mit einiger Sicherheit erraten werden kann, was Rückschlüsse auf die Verschlüsselungsparameter erleichtert.

- Frei zugängliche Gastbenutzer-Identitäten bieten Einfallspforten für Angreifer, die System-Fehler kennen. In Netzen mit Datenschutzanspruch sind sie unbedingt zu vermeiden.
- In der UNIX-TCP/IP-Welt weit verbreitete Netzdienste stellen meist keine sicheren Autorisierungsmechanismen zur Verfügung und übertragen Daten zudem unverschlüsselt. Berüchtigt ist vor allem YP, dessen Unsicherheit in [7] drastisch vorgeführt wird.
- Ein Netzbetriebssystem ist notwendigerweise viel komplexer als ein Rechnerbetriebssystem. Dadurch bieten sich mehr Fehlermöglichkeiten.
- In einem heterogenen Netz ist es schwer, Zugriffsrechte netzweit konsistent zu definieren. Die Realisierungsmöglichkeit der Zugriffsmatrix kann in jedem Netzknoten anders aussehen. Fehler schleichen sich leicht ein und erleichtern das Unterlaufen der Zugriffsrechte. Auch die Verteilung der Berechtigungstabellen von einer zentralen Verwaltungsstelle aus über das Netz ist wie alle anderen Daten abhörbar und eventuell manipulierbar.
- Angreifer sind schon allein durch ihre Entfernung physisch geschützt.
- Auch einfache Nachrichten- und Postdienste ohne direkte Fernwirkungsmöglichkeiten können Systemzugriffe ermöglichen. Oft aufgetreten ist der Fall, daß ein Hacker ein ausführbares Programm an einen zugelassenen Benutzer sendet, der es leichtsinnigerweise ausführt. Manche Betriebssysteme bieten auch die Möglichkeit, ein solches in einem Mehrfachfile zu verstecken, der bei der Annahme der Postsendung in seine Bestandteile zerlegt wird, ohne daß der Benutzer es bei flüchtigem Hinsehen merkt. Hat das Programm den Namen einer oft benutzten Systemprozedur, wird es vielleicht sogar unabsichtlich aufgerufen. Schließlich können auch Prozesse, die auf Nachrichten warten, durch den einfachen Nachrichtendienst beeinflußt werden. Auch Würmern genügt das, um zu funktionieren. Besser ist es, wenn ein Betriebssystem für die Prozeßkommunikation verschiedene Klassen von Nachrichten mit spezieller Autorisierung vorsieht (‘Special Message’).

3.4 Sabotage

Besonders häßlich, aber auch besonders leicht auszuführen ist die reine Destruktion:

- Zerstörung der Übertragungsleitungen oder Knoten.
- Störung der Übertragungswege durch elektromagnetische Einwirkung (Metallkabel oder Funkstrecken).
- Störung des Datenverkehrs durch gezielte Überlastung des Netzes.

- Unterbrechung des Netzes durch (absichtliches oder versehentliches) Abziehen eines Anschlusses. Beim Thinwire-Ethernet ist dies besonders leicht möglich, da PCs mit einem T-Stück direkt am Kabel hängen.

Dagegen helfen nur organisatorische Maßnahmen, vor allem ein durchdachter Katastrophenplan, und physischer Schutz.

3.5 Undefinierte Zustände

Wegen der Komplexität von Netzen ist die Wahrscheinlichkeit des Auftretens undefinierter Zustände besonders groß.

- Das Hochfahren des Netzes ist viel schwerer abzusichern als das Hochfahren eines geschlossenen Systems: Adressenverwaltung, Stationsanmeldung, Laden von Systemtabellen.

- Momentan unbesetzte legale Adressen können unberechtigt usurpiert werden.

- Einseitig „hängende“ Verbindungen laden zum Einklinken ein.

- Viele Protokolle enthalten „verdeckte Datenkanäle“ – undefinierte Bits in Protokollfeldern, die von Trojanischen Pferden und Würmern ausgenützt werden können [5].

4 Schutzmaßnahmen

Bei öffentlichen Netzen muß man grundsätzlich davon ausgehen, daß Nachrichten abgehört werden können; dazu kommt noch die Fehlleitungsgefahr – besonders auf Gateways zwischen verschieden strukturierten Netzen kam es in der Vergangenheit oft zu falschen Adreßumsetzungen. Aktive und passive Angriffe sind niemals auszuschließen. Auf der Ebene des Benutzers helfen nur kryptographische Maßnahmen. Bevor man die Verantwortung für die Datensicherheit auf irgendwelche anonymen Institutionen („gelber Riese“) abwälzt, greift man besser zur Selbsthilfe. Dennoch sind auch politische Forderungen an die Betreiber der Netze zu stellen:

- Bereitstellung von Sicherheitsdiensten,

- Schutz von Verbindungsdaten,

- Einführung neuer Techniken erst, wenn ein angemessener Sicherheitsstandard erreicht ist.

Bei lokalen Netzen ist ebenfalls immer von der Möglichkeit des Abhörens auszugehen, wobei hier zusätzlich die Möglichkeit des unerlaubten Netzzugangs verhindert werden muß. Da lokale Netze in Eigenverantwortung betrieben werden, ist allerdings die Palette der in der eigenen Institution durchführbaren Sicherheitsmaßnahmen erheblich breiter.

4.1 Physischer Schutz

Auf der physischen Ebene muß vor Sabotage und unbefugtem Zugang geschützt werden. Geeignete Maßnahmen sind:

- Verlegung von Kabeln in Schächten und Kanälen, die gegen unbefugtes Eindringen gesichert sind.
- Geschützte Aufstellung von Netzkomponenten, die nicht unmittelbaren Benutzerzugang brauchen: Server, Gateways, Bridges, ...
- Abschließbare Installationsschränke für Verteilereinrichtungen wie Spleißboxen, Patchfelder, Repeater, Multiplexer usw.
- Keine frei zugänglichen unbenutzten Anschlußpunkte.

Die elektromagnetische Abstrahlung kann durch physische Schutzmaßnahmen verhindert werden:

- Elektromagnetische Abschirmung von Metallkabeln,
- elektromagnetische Abschirmung von Bildschirmen,
- Verwendung von Lichtwellenleitern.

Das Anzapfen von Kabeln läßt sich zusätzlich erschweren durch:

- Ummantelung von Lichtwellenleitern, die nicht ohne Beschädigung des Kabels aufgetrennt werden kann;
- Verlegung von Kabeln in Gasdruckrohren.

4.2 Schutz auf höheren Protokollschichten

Auf den höheren Protokollschichten, aber noch unterhalb der Eigenverantwortung des Anwenders, sind ebenfalls Schutzmaßnahmen erforderlich. Die wichtigste ist ein sicheres Identifikationsprotokoll beim Verbindungsaufbau. Auch Rechner untereinander müssen sich identifizieren und authentisieren, damit sich kein Knoten maskiert ins Netz hängen kann und auch, um die Paßwortfalle zu vermeiden.

Bestehende logische Verbindungen müssen gegen „Aufbrechen“ geschützt werden; es soll sich also kein anderer Netzteilnehmer an die Stelle eines der

Kommunikationspartner setzen können. Zu diesem Zweck dienen Prüfcodes und eine gewisse Redundanz beim Datentransport; diese Verfahren sind etwa in der OSI-Schicht 4 anzusiedeln.

Dem Schutz vor unbefugter Netzbenutzung dienen auch Zeitfallen ('Timeouts'), insbesondere um einseitig hängende Verbindungen kontrolliert zu beenden.

Von besonderer Bedeutung ist in jedem Fall die Überwachung des Verkehrs im Netz durch ein Netzmanagement und die logische Verbindungskontrolle (LLC, 'Logical Link Control'). Dabei ist aber zu beachten, daß Management-Daten auch wieder abhörbar sind und vielleicht besonders interessante Informationen über den Aufbau des Netzes enthalten.

4.3 Netzmanagement

Die Hauptfunktionen des Netzmanagements sind:

- Netzbedienung und deren Automatisierung ('Operations Management'),
- Problemerkennung und -behandlung im Hard- und Software-Bereich, auch automatisiert ('Fault Management'),
- Planung und Durchführung von Änderungen mit Kontrolle der Nebenwirkungen ('Change Management'),
- Verwaltung der Konfiguration, Änderungen und (wenn möglich graphische) Anzeige ('Configuration Management'),
- Beobachtung und Verbesserung der Leistungsfähigkeit des Netzes ('Performance Management'),
- Sammlung und Auswertung von Benutzungsdaten ('Accounting Management'),
- Überwachung von Netzzugang und Verbindungen ('Security Management').

Das Netzmanagement muß Fehler automatisch zugeschickt bekommen und die Konfiguration selbständig laufend abfragen. Es sollte erkennen und melden:

- Leitungsunterbrechungen, sogar Änderungen der Dämpfungsbilanz,
- Einsatz von Dataskopen (Schnittstellentestern),
- Besetzung oder Freigabe von Anschlußpunkten,
- Einfügen neuer Anschlußpunkte,
- Konfigurationsänderungen,
- Adressenänderungen.

4.4 Schutz in Diffusionsnetzen

Der völlig unkontrollierte Datenfluß in Diffusionsnetzen läßt sich durch Filterfunktionen einschränken. Auf niedriger Ebene (in Bridges) werten diese die folgenden Informationen aus:

- Knoten, Adressen,
- Zeit,

auf höherer Ebene:

- Benutzer,
- Anwendungsprogramm,
- Art der Daten.

Brücken ('Bridges') haben Selbstlernfunktion: Sie merken sich bei jeder Adresse, auf welcher Seite sie liegt, und lassen von da an Nachrichten an diese Adresse nicht mehr auf die andere Seite. Dadurch wird einerseits die Netzbelastung reduziert, andererseits aber auch der Diffusionseffekt mit seinen erhöhten Abhörmöglichkeiten abgeschwächt. Um unberechtigte Verbindungen zu verhindern sollten Brücken aber auch konfigurierbar sein, das heißt, eine vom Netzverwalter erstellbare Tabelle erlaubter oder unerlaubter Kommunikationsbeziehungen enthalten.

4.5 Schutz in Vermittlungsnetzen

Für den Schutz der Daten in Vermittlungsnetzen gibt es folgende Methoden und Ansätze:

- Rückrufmethode. Diese kann auf verschiedenen Protokollschichten ansetzen; zum Beispiel gibt es 'Call Back Modems', die aber im Bereich der Deutschen Bundespost bisher nicht zugelassen sind.
- Verschlüsselung.
- Adressenverschlüsselung, um Verbindungsdaten zu verschleiern. Das ist aber für das Netzmanagement hinderlich und erzeugt redundanten Verkehr, es sei denn, alle Zwischenstationen entschlüsseln vor der Weiterleitung, was wiederum die Übertragungsrate und die Sicherheit verringert.
- Streusendungen ('broadcasting'), wobei der Empfänger die für ihn bestimmten Sendungen unbemerkt empfängt. Es gibt Vorschläge für gemischte Vermittlungs- und Verteilnetze mit Ringmechanismen zum Zugriff, um zu einem tragbaren Kompromiß zwischen Anonymität und Netzbelastung zu kommen.

- Erzeugung elektronischen Rauschens (Datenmüll), um die wirklichen Sendungen zu verschleiern.
- Abrechnung nach Pauschalen (wie beim Fernsehen) oder nach dezentral installierten Gebührenzählern, um die Notwendigkeit des Speicherns von Verbindungsdaten zu umgehen.

Für fast alle diese Zwecke braucht man komplizierte, aber effiziente kryptographische Protokolle.

Beim ISDN ist die Möglichkeit der Rufnummern-Anzeige vorgesehen – dem Angerufenen wird die Nummer des Anrufers angezeigt; dieser muß allerdings seine Rufnummer zur Anzeige freigeben. Ein Zielrechner kann dies als ersten Schritt zur Benutzer-Identifizierung nützen und bei unbekannten Nummern den Anschluß verweigern oder ein besonderes Authentisierungsverfahren einleiten.

5 Standardisierungs-Aktivitäten

Zur Zeit gibt es eine Reihe von internationalen Aktivitäten, Sicherheitsstandards für Netze zu definieren, wobei hauptsächlich die Schichten 1 bis 3 des OSI-Standards gemeint sind. Ziele und Anforderungen sind dabei:

- Zugangskontrolle,
- Integrität,
- Authentisierung,
- Vertraulichkeit (Schutz vor unberechtigtem Lesen),
- keine Behinderung der Kommunikation zwischen geschützten und ungeschützten Systemen oder zwischen ungeschützten Systemen untereinander,
- transparente (das heißt, für den Anwender unbemerkte) Operation bei Kommunikation zwischen geschützten Systemen.

Einige Organisationen und Gremien, die sich mit der Standardisierung im Bereich der Netze befassen, sind

CCITT – Comité Consultatif International Télégraphique et Téléphonique, die Organisation der öffentlichen Netzbetreiber und Postverwaltungen, in Genf.

ISO – International Standardization Organization, eine Vereinigung der Computerindustrie, in Genf.

CEPT – Conférence Européenne des Administrations des Postes et Télécommunications in Bern.

DIN – Deutsches Institut für Normung in Berlin.

ANSI – American National Standards Institute, New York.

IEEE – Institute of Electrical and Electronic Engineers in den USA.

NBS – National Bureau of Standards in Washington.

Der Standard IEEE 802.10 betrifft die Sicherheit von lokalen Netzen. Seit Juli 1988 läuft das Projekt SILS ('Standard for Interoperable LAN Security'). Es soll Mindestanforderungen für Verschlüsselung und Schlüsselverwaltung erarbeiten, logische und physikalische Mechanismen definieren und für maximale Interoperabilität im OSI-Umfeld sorgen.

'Secure Data Network System' ist ein Projekt des ANSI ('American National Standards Institute'). Es soll ein SP3-Protokoll ('Security Protocol Layer 3') erarbeitet werden für

- CNLP = ISO Connectionless Network Layer Protocol,
- IP = Internet Protocol, Teil von TCP/IP.

ISO 7498/2 ('Open Systems Interconnection Reference Model Security Architecture') soll Sicherheitsrichlinien im OSI-Modell standardisieren [5]. Es definiert fünf Sicherheitsdienste

- Authentisierung,
- Zugangskontrolle,
- Vertraulichkeit von Daten,
- Datenintegrität,
- Anerkennung von Daten,

vgl. die IT-Sicherheitskriterien. Dafür sind fünf allgemeine Sicherheitsmechanismen vorgesehen:

- vertrauenswürdige Funktionalität,
- Sicherheitskennzeichen,
- Entdecken von Ereignissen,
- Sicherheitsaufzeichnungen,
- sicherer Wiederanlauf,

und acht spezifische:

- Verschlüsselung,
- Signatur,
- Zugangskontrolle,
- Datenintegrität,
- wechselseitige Authentisierung,
- 'traffic padding',
- Verbindungskontrolle,
- Beglaubigung ('notarization').

Es wird auch spezifiziert, welche Mechanismen auf welchen OSI-Schichten sinnvoll sind. Vieles davon ist nur mit kryptographischen Protokollen zu verwirklichen und wird daher im Kapitel V behandelt.

Die CCITT-Empfehlung X.509 ('Authentication Framework') [43] beschäftigt sich mit einem sicheren Teilnehmer-Verzeichnis ('Directory System') und sieht unter anderem die „starke Dreiwege-Authentisierung" vor. Dieser Authentisierungsdienst garantiert durch Zertifikate die Authentizität von Kommunikationspartnern. Diese Empfehlung wird schon für Sicherheitsfunktionen im Teletex verwendet.

Die Aufzählung ist nicht vollständig. Es bleibt zu hoffen, daß alle diese Bemühungen möglichst bald in sichere Kommunikationssysteme umgesetzt werden, die zu erschwinglichen Preisen auf dem Markt erhältlich sind.

Die Nachteile der ganzen Standardisierungs-Aktivitäten sind aber auch zu bedenken. Es zeichnet sich in letzter Zeit ab, daß die Hersteller, anstatt mutig die Innovation voranzutreiben, sich fürchten, vorzupreschen und am künftigen Standard vorbeizuentwickeln. Statt dessen sitzen sie in den Standardisierungsgremien, palavern und warten ab; die bereits Jahre währende Entwicklung des OSI-Standards ist ein Beispiel dafür. Auch führt der in Aussicht stehende Standard zur Zurückhaltung der Käufer bei bereits existierenden Produkten; ein Beispiel hierfür ist FDDI, das trotz seiner hervorragenden Eignung für Glasfaser-Verbindungen noch keinen echten Markt hat und somit noch verhältnismäßig teuer ist. Standardisierung führt somit zu Schwerfälligkeit in den Produkten und in der Weiterentwicklung, obwohl sie genau das Gegenteil bewirken soll.

Kapitel V

Verschlüsselung

Wenn man die Wörter „Verschlüsselung“ oder „Kryptographie“, auf deutsch „Geheimschrift“, hört, denkt man zunächst an Geheimdienste, Verschwörungen, Militär, vielleicht auch an Edgar Allen Poe und Schatzsucher. Diese Vorstellungen sind auch nicht falsch – vor allem aber ist Kryptographie heute die wohl wichtigste Grundlage für die Sicherheit der Informationsverarbeitung. Alle bisherigen Überlegungen zum Datenschutz beruhten auf dem Prinzip des geschlossenen Systems, also auf der Regel: Auf einer höheren Ebene (zum Beispiel in der Software) kann die Sicherheit nicht besser sein als auf einer darunter liegenden (zum Beispiel der Hardware). Anders ausgedrückt: In einer unsicheren Umgebung gibt es keine Sicherheit.

Dies gilt für viele Maßnahmen tatsächlich, aber es gilt nicht immer: Die Kryptologie ist die Lehre von der *Datensicherheit in einer* **unsicheren** *Umgebung.* Oft wird sie etwas enger gesehen als Lehre von der sicheren Kommunikation über unsichere Kanäle. Datenverarbeitungssysteme, die sich nicht physisch sichern lassen, wie etwa große Datennetze, müssen mit kryptographischen Methoden gesichert werden, also mit Verschlüsselung. (**Kryptographie** ist die Lehre von der Verschlüsselung, **Kryptoanalyse** die von der Entschlüsselung durch Unbefugte. Beides wird in der **Kryptologie** zusammengefaßt.) Verschlüsselung ist auch angebracht auf Datenträgern, die nicht ständig physisch geschützt sind oder deren Diebstahl man fürchten muß.

Die primäre Aufgabe der Kryptographie als Wissenschaft ist, Verschlüsselungsmethoden zu entwickeln und ihre Sicherheit gegen unberechtigte Entschlüsselung mathematisch abzusichern. Für den Datenschutz ebenso wichtig ist die weiterführende Aufgabe, für jedes Anwendungsfeld eine geeignete kryptographische Methode zu finden unter der Voraussetzung, daß sichere Verschlüsselungsverfahren existieren. Solche Methoden nennt man **kryptographische Protokolle**.

Unter einem kryptographischen Protokoll versteht man also ein praktisches Verfahren, das ein bestimmtes Sicherheitsproblem löst. Die drei Grundprobleme

dieser Art sind:

Vertraulichkeit – Chiffrierung (Verschlüsselung), Schutz der Daten vor unberechtigtem Einblick (Leseschutz).

Echtheitsnachweis – Authentisierung, Signatur (elektronische Unterschrift, der Urheber der Daten soll seine Urheberschaft nicht bestreiten können), Schutz der Daten vor unbemerkter Verfälschung (wobei das Lesen gestattet sein kann oder auch nicht).

Anonymität – der Sender oder Empfänger einer Nachricht soll nicht bekannt werden (gegenüber dritten oder gegenseitig); dieses Problem betrifft etwa die Verbindungsdaten in Netzen oder den elektronischen Geldverkehr. (Läßt sich die Anonymität eines Markstücks elektronisch simulieren?)

Spezielle Ausprägungen dieser Probleme sind: Gegenzeichnung von Dokumenten, Vieraugen- (Mehrschlüssel-) Prinzip, Einschreiben mit Rückschein (= Sende- und Empfangsbeweis), gegenseitige Identifizierung. Alle diese Probleme lassen sich durch geschickte Kombination von Verschlüsselungsschritten lösen, also durch geeignete kryptographische Protokolle. Es geht hier allerdings keineswegs um eine mathematisch fundierte Darstellung der einzelnen Protokolle. Vielmehr soll nur gezeigt werden, welche Möglichkeiten und Chancen für den Datenschutz solche Protokolle bieten können.

Ich betone aber, daß die Umsetzung der Lösungen in die Praxis erst beginnt, einerseits, weil der Weg von der wissenschaftlichen Grundlagenforschung zur alltäglichen Anwendung oft weit ist, andererseits, weil das Sicherheitsbewußtsein der Hersteller und Käufer von Hard- und Software erst langsam erwacht. Findige Hacker haben zur Zeit noch ein weites Betätigungsfeld.

1 Chiffriermethoden

Gegenstand dieses Abschnitts sind einige ausgewählte Chiffriermethoden, zunächst einige einfache, in der Praxis nicht brauchbare, an denen man aber die grundsätzlichen Probleme gut erkennen kann, anschließend kompliziertere, aber aktuelle und praktisch brauchbare Verfahren.

1.1 Monoalphabetische Chiffrierung

Jeder Chiffrierung liegt ein Alphabet zu Grunde, in dem Klartexte und Geheimtexte abgefaßt sind. Man kann für die Geheimtexte auch ein anderes Alphabet verwenden, aber das spielt hier keine Rolle. In der „klassischen“ Kryptologie verwendet man meist das Alphabet aus den 26 Buchstaben, wobei Groß- und Kleinschreibung nicht unterschieden werden. Will man elektronische Dateien verschlüsseln, ist es zweckmäßig, als Alphabet die Menge der 256 Bytes von (hexadezimal) 00 bis FF zu betrachten.

Das Alphabet werde mit Σ bezeichnet. Die Menge aller Texte (klar oder geheim) ist die Menge $\Sigma^\star$ der endlichen Folgen aus Σ. Oft betrachtet man nur eine Teilmenge $M \subseteq \Sigma^\star$ als Menge der möglichen Klartexte ('messages') und eine Teilmenge $C \subseteq \Sigma^\star$ als Menge der möglichen Geheimtexte ('ciphertexts'); diese nimmt man oft als endlich an, beispielsweise kann man sich M und C vorstellen als die Menge aller Folgen, die höchstens 100 mal so lang wie die Bibel sind.

Eine Chiffrierung (Verschlüsselung) ist eine Abbildung

$$f\colon M \longrightarrow C;$$

damit die Dechiffrierung (Entschlüsselung) eindeutig möglich ist, verlangt man von f die Injektivität, d. h., verschiedene Klartexte werden in verschiedene Geheimtexte umgewandelt. Die Dechiffrierung ist dann eine Abbildung

$$g\colon M \longrightarrow C \quad \text{mit } gf(m) = m \text{ für alle } m \in M.$$

Im Normalfall ist f sogar bijektiv und g dann einfach die Umkehrabbildung f^{-1}. Der Sender einer Nachricht muß die Abbildung f kennen (und berechnen können), der Empfänger g.

Eine monoalphabetische Chiffrierung wird nun einfach von einer Permutation

$$\pi\colon \Sigma \longrightarrow \Sigma$$

induziert. Der einfachste Fall ist eine Verschiebechiffre. Dabei denkt man sich das Alphabet angeordnet; die Permutation ist dann einfach eine Verschiebung um einige Stellen. Beispiel:

$$\pi A = D, \pi B = E, \ldots,$$

Hier wird das gewöhnliche Alphabet um 3 Stellen verschoben. Dieses Verfahren hat Caesar verwendet. Deutlich wird die Unterscheidung zwischen:

- Algorithmus (Verschlüsselungsverfahren), im Beispiel „Verschiebechiffre“, und
- Schlüssel (Parameter des Verfahrens), im Beispiel „Zahl der Stellen“.

Bei der Diskussion der Sicherheit eines Verschlüsselungsverfahrens sollte man immer davon ausgehen, daß der Angreifer das Verfahren kennt, aber nicht den Schlüssel. (Goldene Regel der Kryptographie: Unterschätze niemals den Kryptoanalytiker!) Ein Verfahren, für dessen Sicherheit man auf die Geheimhaltung des Algorithmus angewiesen ist, hat schwere Mängel:

- Der Beweis der Sicherheit des Verfahrens ist nicht öffentlich möglich, ohne das Verfahren zu kompromittieren. Das beunruhigt die Benutzer, die dann einem undokumentierten Verfahren trauen müssen. Wenn das Verfahren Schwächen hat, so sollte der legale Anwender das wissen, damit er sich darauf einstellen kann.

- Die Information über die Art des Verfahrens, also den Algorithmus, ist schwerer geheim zu halten als ein zufällig gewählter Schlüssel. Man muß etwa beim Einsatz in Netzen davon ausgehen, daß viele andere Teilnehmer das Verfahren verwenden und es daher möglicherweise kennen oder erraten können. Man weiß nie, wie geheim das „Geheimnis“ noch ist; auf keinen Fall ist man vor dem Entwickler des Verfahrens sicher. Im Grunde ist die Geheimhaltung des Vefahrens nur bei einer festen Kommunikationsbeziehung ohne Partnerwechsel eine sichere Grundlage, bei Massenanwendung dagegen unpraktikabel. Man sollte grundsätzlich davon ausgehen, daß jede Schutzmethode langfristig bekannt wird, die in größerem Umfang eingesetzt wird. Die Kryptoanalytiker der Vergangenheit konnten oft Verschlüsselungsgeräte und somit den Algorithmus in ihre Gewalt bekommen.

- Bei einem Geheimnisbruch sind viele Anwender betroffen. Der Wechsel eines Verfahrens ist sehr aufwendig, der Wechsel eines Schlüssels dagegen leicht.

- Asymmetrische Verschlüsselungsverfahren setzen sowieso voraus, daß der Algorithmus öffentlich bekannt ist.

Entscheidend für die Sicherheit muß daher ein Stück Information sein, das leicht geheimzuhalten ist und dessen Mitteilung an andere nur unter der Kontrolle des Besitzers geschieht, eben ein „Schlüssel“ oder „Paßwort“.

Bei der Entwicklung und Verwendung von Verschlüsselungsverfahren sind folgende Regeln zu beachten:

- Es muß genügend Möglichkeiten für die Auswahl des Schlüssels geben, um ein systematisches Durchprobieren aller Schlüssel aussichtslos zu machen.

- Schlüssel sollten vergleichsweise kurz oder einfach zu beschreiben sein, damit nicht die Übermittlung des Schlüssels zum Kommunikationspartner ein ähnlich großes Problem darstellt wie die Übermittlung des eigentlichen Textes, und damit sie sich einfach merken lassen und niemand in die Versuchung gerät, sie aufzuschreiben.

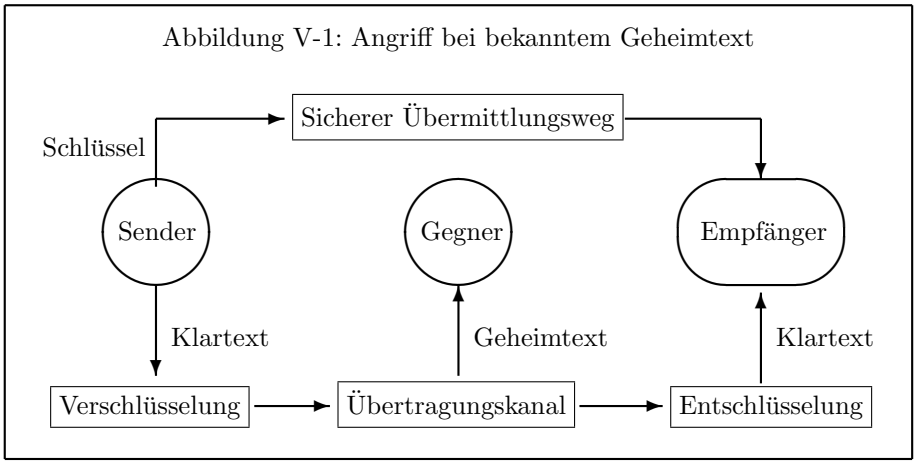
- Die gängigen kryptoanalytischen Attacken sollen keinen Anhaltspunkt finden.

1.2 Kryptoanalytische Attacken

Die Kryptoanalyse ist, entgegen dem ersten Anschein, nicht die Anleitung zu Untaten, sondern Teil der mathematischen Systemanalyse. Sie dient dazu, Schwachstellen von Kryptosystemen aufzudecken, und damit der Absicherung von kryptographischen Verfahren.

Das Ziel des Kryptoanalytikers ist in erster Linie, aus einem Geheimtext den zugehörigen Klartext zu ermitteln. Der Angriff kann aber auch zunächst dem Schlüssel gelten. Im einfachsten Fall nimmt man an, daß der Angreifer einen (ziemlich langen) Geheimtext zur Verfügung hat („Angriff mit bekanntem Geheimtext“, „Nur-Geheimtext-Attacke“, ‘known ciphertext attack’). Diese Situation ist in Abbildung V-1 graphisch dargestellt.

Abbildung V-1: Angriff bei bekanntem Geheimtext



Der einfachste Angriff ist das Ausprobieren aller möglichen Schlüssel („Schlüssel-Durchprobier-Attacke“). Bei der Verschiebechiffre auf dem Alphabet aus 26 Buchstaben gibt es nur 26 verschiedene Schlüssel, die schnell durchprobiert sind. Besser sieht es hier mit der allgemeinen monoalphabetischen Chiffrierung aus; hier ist der Schlüssel eine von 26! möglichen Permutationen des Alphabets, und das sind etwa $4 \cdot 10^{26}$ Stück, viel zu viele, selbst mit dem größtem Computer als Hilfsmittel.

Dennoch ist auch die allgemeine monoalphabetische Chiffre leicht zu brechen. Man geht von der vermuteten Sprache des Klartexts aus; eventuell muß man mehrere Vermutungen durchtesten. Von dieser Sprache sind die durchschnittlichen Häufigkeiten der einzelnen Buchstaben bekannt oder aus der Analyse typischer Klartexte leicht zu ermitteln. Zum Beispiel tritt in deutschen Texten der Buchstabe „e“ zu 17%, der Buchstabe „n“ als zweithäufigster zu 10% auf. Hilfreich sind auch Häufigkeiten von Buchstabenpaaren; so sind etwa im Deutschen „c“ und „h“ als Einzelbuchstaben eher selten, die Kombination „ch“ ist allerdings vergleichsweise häufig.

Eine Variante ist das gemeinsame Verschlüsseln von Buchstabenpaaren; das kann man als monoalphabetische Chiffrierung auffassen, bei der das Alphabet eben aus Paaren von „gewöhnlichen“ Buchstaben besteht. Analog sieht es aus, wenn man Textblöcke fester Länge gemeinsam verschlüsselt. Die Angriffsmethode bleibt die gleiche; allerdings nimmt die dazu benötigte Zeit zu.

Besteht das Alphabet aus den 256 Bytes und ist der Klartext ein Maschinenprogramm, so geht man ähnlich vor. Bei allen derartigen „natürlichen“ Sprachen führt der Angriff per Häufigkeitsanalyse zum Erfolg, außer bei ganz kurzen Texten. Auswege für den Kryptographen sind:

- die Zerstörung der natürlichen Struktur der Sprache,
- polyalphabetische Chiffrierung.

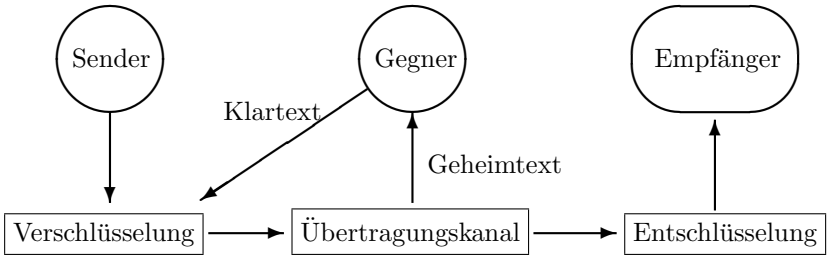
Daß der Angreifer einen Geheimtext (und den Chiffrieralgorithmus) kennt, ist nur die Mindestannahme für alle Angriffsversuche. Oft kennt er mehr: Wenn er weiß, wovon der gesuchte Klartext handelt, kann er vielleicht einige Wörter erraten. Zum Beispiel weiß er, daß ein Brief mit „Sehr geehrter Herr“ beginnt. Oder er hört einen Erkennungsdialog ab und weiß, daß ein bestimmtes Stück Geheimtext der Aufforderung ‘please enter password’ entspricht. Oder er weiß, daß der abgefangene Geheimtext zuvor mit einem bekannten Kompressionsprogramm verkleinert wurde; dann weiß er vielleicht, wo die Kompressionsinformationen stehen und wie sie strukturiert sind. Besonders tückisch ist, daß im ASCII-Zeichensatz alle Kleinbuchstaben mit der Bitfolge 011 beginnen; da jeder gewöhnliche Text längere Sequenzen aus Kleinbuchstaben hat, ergibt sich ein Ansatz zum Knacken. Ähnliches gilt für ausführbare Programme, die meist viele Nullbytes enthalten (5 - 10% bei EXE-Dateien in MS-DOS). In jedem dieser Fälle kann der Gegner einen „Angriff mit bekanntem Klartext“ („Klartextstück-Attacke“, ‘known plaintext attack’) starten, der neue Möglichkeiten eröffnet. Man darf allerdings im allgemeinen davon ausgehen, daß das bekannte Stück Klartext klein ist. Als die Engländer unter Mitarbeit von Turing im zweiten Weltkrieg das deutsche Chiffriergerät „Enigma“ knackten, hatten sie das Gerät, also den Algorithmus, zur Verfügung. Entscheidend war dann der Angriff mit bekanntem Klartext, wobei vor allem die Entsprechungen der Buchstabengruppen „U-Boot“ und „Sehr geehrter“ gesucht wurden.

Bei asymmetrischen Veschlüsselungsverfahren muß man beim Angreifer noch mehr Kenntnisse voraussetzen. Da hier der Algorithmus samt Schlüssel in der Regel für jedermann frei zugänglich ist, kann der Angreifer sich beliebig lange Stücke aus (selbstgewähltem) Klartext und zugehörigem Geheimtext verschaffen. Man spricht dann von „Angriff mit ausgewähltem Klartext“ („Probeverschlüsselungs-Attacke“, ‘chosen plaintext attack’). Der Angreifer kann dann aus besonders geeigneten Klartexten Aufschluß über den Schlüssel für die Dechiffrierung zu gewinnen versuchen, oder er kann Vermutungen über den Klartext beweisen, indem er probeweise verschlüsselt. In diesem Rahmen kann auch ein Fischzug-Angriff erfolgreich sein, etwa bei Einweg-verschlüsselten Paßwörtern. Die Situation dieses Angriffs ist in Abbildung V-2 dargestellt.

1.3 Polyalphabetische Chiffrierung

Bei der polyalphabetischen Chiffrierung wird ein Klartextbuchstabe *nicht* stets auf den gleichen Geheimtextbuchstaben abgebildet; man kann das auch so

Abbildung V-2: Angriff mit ausgewähltem Klartext



deuten, daß man monoalphabetisch mit ständig wechselndem Schlüssel chiffriert. Das Ziel dabei ist, die Häufigkeiten der einzelnen Buchstaben zu verschleiern, um die gängige Attacke durch Häufigkeitsanalysen hinfällig zu machen.

Als Musterbeispiel dient die VIGENÈRE-Chiffre. Sie ist eine Verschiebechiffre mit ständig wechselnder Stellenzahl. Diese wechselnden Stellenzahlen werden durch ein Schlüsselwort gegeben. Ist beispielsweise das Schlüsselwort MAINZ, so wird der erste Klartextbuchstabe um 12 Stellen verschoben (A $\mapsto$ M), der zweite um 0 Stellen (A $\mapsto$ A), der dritte um 8 (A $\mapsto$ I) und so weiter; beim sechsten geht es wieder von vorne los. Ein Beispiel zeigt Abbildung V-3

Abbildung V-3: VIGENÈRE-Chiffre

Klartext:	K	R	Y	P	T	O	G	R	A	P	H	I	E
Schlüssel:	M	A	I	N	Z	M	A	I	N	Z	M	A	I
Geheimtext:	W	R	G	C	S	A	G	Z	N	O	T	I	M

Stellt man die Buchstaben durch die Zahlen von 0 bis 25 dar, so ist der Algorithmus einfach die Addition von Klartext und zyklisch wiederholtem Schlüssel modulo 26.

Die Kryptoanalyse dieses Verfahrens ist schon recht interessant. Sie zielt darauf, die Länge l des Schlüssels zu bestimmen; hat man diese, so kann man den Geheimtext in l Spalten nebeneinander schreiben, die jeweils monoalphabetisch chiffriert und mit der bekannten Häufigkeitsanalyse zu entschlüsseln sind; da sie mit einfachen Verschiebungen chiffriert sind, reicht sogar ein Probieren sämtlicher Verschiebedistanzen.

Zur Bestimmung der Schlüssellänge stehen zwei Verfahren zur Verfügung, die sich ergänzen:

- der KASISKI-Test, mit dem man die Länge bis eventuell auf ein ganzzahliges Vielfaches bestimmt,
- der FRIEDMAN-Test, mit dem man die Größenordnung der Länge bestimmt.

Kombiniert man beide Verfahren, so hat man meist die genaue Länge des Schlüssels. Man kann sich aber auch auf eines der Verfahren beschränken und dann einige wenige mögliche Schlüssellängen durchprobieren.

Beim KASISKI-Test sucht man im Geheimtext Folgen von mindestens 3 Buchstaben, die sich irgendwo wiederholen. Die Wahrscheinlichkeit, daß sie gleichen Klartextstellen entsprechen, ist ziemlich groß, und in diesem Fall ist ihr Abstand ein Vielfaches der Schlüssellänge. Nimmt man die Abstände der auffälligsten Wiederholungen und bildet deren größten gemeinsamen Teiler, so hat man die Schlüssellänge oder wenigstens ein Vielfaches von ihr.

Die theoretische Grundlage des FRIEDMAN-Tests ist komplizierter. Der zentrale Begriff ist der FRIEDMANsche **Koinzidenzindex** einer Sprache: Gibt es k Buchstaben und tritt der i-te davon mit der Wahrscheinlichkeit p_i auf, so ist die Wahrscheinlichkeit, daß an zwei zufällig gewählten Stellen eines Textes der gleiche Buchstabe steht, durch die Größe

$$I_0 = \sum_{i=1}^{k} p_i^2$$

gegeben. Für die deutsche Sprache (mit 26 Buchstaben) ist der Koinzidenzindex zum Beispiel $I_0 = 0.0762$ (näherungsweise), für englisch 0.0661. Analog bildet man den empirischen Koinzidenzindex eines Textes m der Länge n, in dem der i-te Buchstabe n_i mal vorkommt: Es gibt $n_i(n_i - 1)/2$ Paare des i-ten Buchstabens bei einer Gesamtauswahlmöglichkeit von $n(n-1)/2$ Buchstabenpaaren; daraus erhält man die Größe

$$I(m) = \frac{\sum n_i(n_i - 1)}{n(n-1)}.$$

Der Koinzidenzindex eines Zufallstexts z (26 Buchstaben) ist näherungsweise

$$I(z) = \sum_{i=1}^{26} (\frac{1}{26})^2 = 0.0385.$$

Ergibt ein Geheimtext (aus deutschem Klartext) einen Koinzidenzindex von etwa 0.0762, so kann man mit ziemlicher Sicherheit auf eine monoalphabetische Chiffrierung schließen.

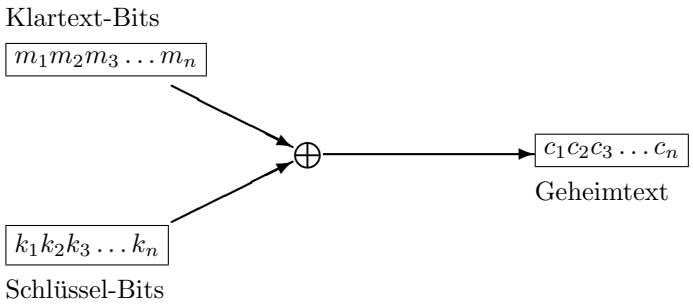
Hat man nun einen VIGENÈRE-chiffrierten (deutschen) Geheimtext c der Länge n, so wird die Schlüssellänge l durch die Formel

$$l \approx \frac{0.0377 \cdot n}{(n-1) \cdot I(c) - 0.0385 \cdot n + 0.0762}$$

approximiert. Diese Formel soll hier nicht hergeleitet werden, siehe [6].

Trotz dieser leicht durchzuführenden Kryptoanalyse wird die binäre Version der VIGENÈRE-Chiffre, die **Bitstrom-Chiffre**, oft in kommerziellen Verschlüsselungssystemen verwendet. Sie wird in Abbildung V-4 beschrieben.

Abbildung V-4: Bitstrom-Verschlüsselung



Der Schlüssel wird modulo 2 zum Klartext addiert; man kann das auch als bitweises XOR beschreiben. Gebräuchlich ist etwa eine Schlüssellänge von 512 Bytes, was einem Sektor auf einer PC-Diskette entspricht. Solch einen Schlüssel kann sich natürlich niemand merken; er muß also aus einem kürzeren Schlüssel berechnet werden.

Eine Verbesserung der VIGENÈRE-Chiffre, die die Analyse der Schlüssellänge verhindert, erhält man, wenn man den Schlüssel (mindestens) so lang wählt wie den Klartext. Hier könnte man etwa den Schlüssel an einer bestimmten Stelle eines bestimmten Buches beginnen lassen; repräsentiert wird er durch den Titel des Buches und die Stelle (etwa Seitenzahl plus Nummer des ersten Wortes); dieses Verfahren nennt man auch **Lauftext-Verschlüsselung**. Da der Schlüssel dann ebenfalls Text einer natürlichen Sprache ist, ist er für Sprachanalysen anfällig; zum Beispiel wird am häufigsten „e“ mit „e“ verschlüsselt. Auch eine Klartextstück-Attacke verspricht Erfolg: Man subtrahiert ein im Klar- oder Schlüsseltext wahrscheinlich vorkommendes Wort von jeder Stelle des Geheimtexts, bis sinnvoller Text erscheint, und arbeitet dann von dieser Stelle aus weiter. Durch mehrfache Verschlüsselung ist allerdings eine gute Sicherheit erreichbar [49].

Eine oft verwendete Variante besteht darin, den eigentlichen, langen Schlüssel mit Hilfe einer „Chiffriermaschine“ oder eines Algorithmus („Pseudozufallsgenerator“) aus einem kurzen Schlüssel zu erzeugen.

Damit kommen wir zu einer weiteren Verbesserung: Als Schlüsselwort wird eine Zufallsfolge von Buchstaben gewählt, so lang wie der Klartext. Falls ein

solcher Schlüssel nicht wiederholt benützt wird, findet der Kryptoanalytiker keinen Anhaltspunkt für statistische Analysen mehr. Diese Situation wird im nächsten Abschnitt noch genauer betrachtet. Das Hauptproblem bei dieser Methode ist die Übermittlung des Schlüssels. Da er ja genauso lang wie der Klartext ist, ist nur das eine Problem durch ein gleich großes anderes ersetzt worden. Die einzig denkbare Anwendung ist in einer Situation gegeben, wo man weiß oder erwartet, daß man irgendwann einmal eine einzige wichtige, aber eher kurze Nachricht übermitteln muß, zum Beispiel eine Entscheidung. Dann kann man den Schlüssel zu einem weit früheren, geeigneten Zeitpunkt auf einem günstigeren Weg übertragen („reitender Bote“ oder „konspiratives Treffen“).

1.4 Theoretische Sicherheit

Die Menge M der Klartexte und die Menge C der Geheimtexte werden jetzt als endlich vorausgesetzt; diese Voraussetzung dient der Bequemlichkeit, weil sie es ermöglicht, „naiv“ mit Wahrscheinlichkeiten umzugehen, und bedeutet in der Praxis keine Einschränkung. Außerdem soll auch nur eine endliche Menge F von Verschlüsselungsfunktionen $f : M \longrightarrow C$ betrachtet werden. Diese sei bijektiv mit einer endlichen Menge K von Schlüsseln parametrisiert; für die mathematische Betrachtung könnte man genausogut F selbst statt K betrachten, für die praktische Anwendung ist aber die Unterscheidung zwischen „Algorithmus“ $f \in F$ und „Schlüssel“ $k \in K$ von Bedeutung. Also ist

$$F = \{f_k \mid k \in K\}.$$

Es sei daran erinnert, daß jedes $f \in F$ injektiv ist; insbesondere ist $\#M \leq \#C$.

Der zugehörige Chiffrieralgorithmus oder kurz die **Chiffre** ist die Abbildung

$$\Phi\colon M \times K \longrightarrow C,$$

$$(m, k) \mapsto f_k(m).$$

Insbesondere ist $f_k = \Phi(\bullet, k)$.

Das Ziel ist die Definition von „perfekter Sicherheit“ nach SHANNON, die hier aber nur informell gegeben werden soll. Zur Illustration wird zunächst ein Beispiel vorgestellt (nach BRASSARD): Ein Geheimtext sei aus einem englischen Klartext mit einer monoalphabetischen Chiffrierung erstellt. Ohne ihn anzusehen, kann der Kryptoanalytiker nur etwa sagen, daß folgende „a-priori-Wahrscheinlichkeiten“ für die ersten fünf Buchstaben des Klartextes bestehen:

Klartext	Wahrscheinlichkeit
hello	$p > 0$
peace	$q > 0$
xykph	0

Dabei kennt er möglicherweise ziemlich genaue Werte für p und q, aber darauf kommt es hier nicht an. Sieht der Kryptoanalytiker, daß der Geheimtext mit 'xtjja' beginnt, so weiß er mehr über den Klartext: Er kann den ersten fünf Buchstaben folgende „a-posteriori-Wahrscheinlichkeiten" zuordnen:

Klartext	Wahrscheinlichkeit
hello	$p_1 \gg p$
peace	0
xykph	0

Anders ausgedrückt: Die Betrachtung des Geheimtextes erlaubt es, gewisse Klartexte auszuschließen, anderen erhöhte Wahrscheinlichkeit zuzuerkennen.

Allgemein seien auf den Mengen M, C und K Wahrscheinlichkeitsverteilungen gegeben. Diese gehören eigentlich mit zur Definition der Chiffre, das heißt, die drei Mengen sind mathematisch gesehen von vorneherein Wahrscheinlichkeitsräume. Auf M sei jedem Klartext $m \in M$ eine a-priori-Wahrscheinlichkeit $P(m) > 0$ zugeordnet mit

$$\sum_{m \in M} P(m) = 1.$$

Diese Wahrscheinlichkeiten sind zum Beispiel aus einer Sprachanalyse bekannt; als einfaches Beispiel stelle man sich M als die Menge aller 5-buchstabigen englischen Wörter vor.

Für jeden Schlüssel $k \in K$ sei eine Auswahlwahrscheinlichkeit $P(k)$ gegeben (eigentlich sollte man einen anderen Buchstaben als wieder P verwenden, aber Verwechslungen werden im folgenden kaum möglich sein). Auch hier ist wieder

$$\sum_{k \in K} P(k) = 1.$$

Im allgemeinen wird man alle Schlüssel als gleich wahrscheinlich annehmen — das bedeutet nämlich, daß die Kryptoanalyse optimal erschwert ist; also $P(k) = 1/\#K$. Im folgenden sei die Wahrscheinlichkeitsverteilung auf K aber beliebig.

Die Wahrscheinlichkeiten $P(c)$ für die Geheimtexte $c \in C$ sind dann unter der Annahme, daß der Schlüssel k stets unabhängig vom Klartext m gewählt wird, festgelegt durch

$$P(c) = \sum_{m \in M} \sum_{k \in K_{mc}} P(m) \cdot P(k);$$

dabei ist

$$K_{mc} = \{k \in K \mid f_k(m) = c\}$$

die Menge aller Schlüssel, die den Klartext m in c umwandeln. Von Bedeutung sind auch die bedingten Wahrscheinlichkeiten $P(c|m)$; die Größe $P(c|m)$ ist

die Wahrscheinlichkeit für den Geheimtext c unter der Annahme, daß m der Klartext ist. Eine solche bedingte Wahrscheinlichkeit ist durch die Formel

$$P(c|m) = \sum_{k \in K_{mc}} P(k)$$

gegeben.

Der Kryptoanalytiker arbeitet an den umgekehrten bedingten Wahrscheinlichkeiten $P(m|c)$, also an der Wahrscheinlichkeit dafür, daß er vom Geheimtext c auf den Klartext m schließen kann. Sein Ziel ist, diese Wahrscheinlichkeit bei gegebenem c für einen Klartext m sehr groß und für alle anderen Klartexte sehr klein zu machen. An diese Größen kommt man über die Formel von BAYES: Die Wahrscheinlichkeit $P(m, c)$ für das gemeinsame Auftreten eines Klartexts m und eines Geheimtexts c bei beliebiger Wahl des Schlüssels k läßt sich auf zwei Weisen ausdrücken:

$$P(m) \cdot P(c|m) = P(m, c) = P(c) \cdot P(m|c).$$

Daraus lassen sich die Wahrscheinlichkeiten $P(m|c)$ bestimmen:

$$P(m|c) = \frac{P(m) \cdot P(c|m)}{P(c)}.$$

Die Chiffre Φ heißt nun **perfekt sicher**, wenn die Kenntnis eines Geheimtextes c die Wahrscheinlichkeiten für die Klartexte nicht verändert, also

$$P(m|c) = P(m) \text{ für alle } m \in M \text{ und } c \in C.$$

Das bedeutet, daß der Kryptoanalytiker aus der Kenntnis eines Geheimtexts keinerlei zusätzliche Information über den Klartext herleiten kann – er ist genauso schlau, wie wenn er gar keinen Geheimtext hat, der Angriff mit bekanntem Geheimtext stößt ins Leere.

Die monoalphabetische Chiffre ist nicht perfekt sicher: Setzt man etwa $M = C = \Sigma^n$, also gleich der Menge aller Texte aus n Buchstaben, so ist $K = S(\Sigma)$ die Gruppe der Permutationen des Alphabets Σ. In dem Beispiel am Anfang dieses Paragraphen war $n = 5$,

$$P(\text{peace}|\text{xtjja}) = 0 < q = P(\text{peace}),$$

im Widerspruch zur perfekten Sicherheit.

Es ist aber überraschend leicht, perfekt sichere Chiffren zu konstruieren, wie sich gleich zeigen wird.

Zunächst seien

$$M_0 := \{m \in M \mid P(m) > 0\}$$

die Menge aller *möglichen* Klartexte und

$$C_0 := \{c \in C \mid P(c) > 0\}$$

die Menge aller möglichen Geheimtexte. Es ist $\#M_0 \leq \#C_0$. Zum Beweis wählt man einen Schlüssel $l \in K$ mit $P(l) > 0$. Für einen Geheimtext $c \in f_l(M_0)$, etwa $c = f_l(n)$, gilt dann

$$P(c) = \sum_{m \in M_0} P(m) \cdot \sum_{k \in K_{mc}} P(k) \geq P(n) \cdot P(l) > 0.$$

Also ist $c \in C_0$ und somit $f_l(M_0) \subseteq C_0$. Die Behauptung folgt aus der Injektivität von f_l.

Ist nun Φ perfekt sicher, so ist $K_{mc} \neq \emptyset$ für alle $m \in M_0$ und $c \in C_0$, das heißt, jeder mögliche Klartext ist in jeden möglichen Geheimtext überführbar. Andernfalls wäre nämlich

$$P(c|m) = \sum_{k \in K_{mc}} P(k) = 0 \neq P(m),$$

Widerspruch. Daran sieht man, daß es bei einer perfekt sicheren Chiffre sehr viele Schlüssel geben muß; in Zahlen ausgedrückt: $\#K \geq \#C_0$. Zum Beweis davon wählt man einen möglichen Klartext $m \in M_0$. Wäre $\#K < \#C_0$, so gäbe es einen Geheimtext $c \in C_0$ mit $f_k(m) \neq c$ für alle Schlüssel $k \in K$, also $K_{mc} = \emptyset$, im Widerspruch zum eben gesagten. Die Zusammenfassung dieser Bemerkungen ergibt ein notwendiges Kriterium von SHANNON:

Satz 1 *Sei $\Phi\colon M \times K \longrightarrow C$ eine perfekt sichere Chiffre. Dann ist*

$$\#K \geq \#M_0,$$

das heißt, es muß mindestens so viele Schlüssel geben wie mögliche Klartexte.

Der folgende Satz, ebenfalls von SHANNON gibt ein hinreichendes Kriterium:

Satz 2 *Sei $\Phi\colon M \times K \longrightarrow C$ eine Chiffre mit den Eigenschaften:*

(i) *Alle Schlüssel sind gleich wahrscheinlich, also*

$$P(k) = \frac{1}{\#K} \quad \textit{für alle } k \in K.$$

(ii) *Für jedes Paar $(m, c) \in M \times C$ gibt es genau einen Schlüssel $k \in K$, der m in c überführt, also*

$$\#K_{mc} = 1 \quad \textit{für alle } m \in M \textit{ und } c \in C.$$

Dann ist Φ perfekt sicher.

Beweis. Für beliebiges $c \in C$ und $m \in M$ gilt

$$P(c|m) = \sum_{k \in K_{mc}} \frac{1}{\#K} = \frac{\#K_{mc}}{\#K} = \frac{1}{\#K},$$

$$P(c) = \sum_{m \in M} P(m) \cdot P(c|m) = \frac{1}{\#K} \cdot \sum_{m \in M} P(m) = \frac{1}{\#K},$$

$$P(m|c) = \frac{P(c|m)}{P(c)} \cdot P(m) = P(m),$$

was zu beweisen war. $\diamondsuit$

Daraus ergibt sich sofort eine Klasse von perfekt sicheren Chiffren, die man mathematisch als **Verschiebechiffren auf Gruppen** beschreiben kann: Sei $M = K = C$ eine Gruppe und $\Phi : M \times K \longrightarrow C$ einfach die zugehörige Verknüpfung. Die Bedingung (i) sei erfüllt; (ii) folgt automatisch, da

$$K_{mc} = \{k \in K \mid mk = c\} = \{m^{-1}c\}$$

stets einelementig ist. Also ist Φ perfekt sicher. Damit ist insbesondere der Fall der gewöhnlichen Verschiebechiffren erfaßt: Sie sind perfekt sicher, aber nur für Klartexte der Länge 1 (!).

Ein weiterer Spezialfall ist die VERNAM-Chiffre, auch 'one time pad' genannt. (Man stellt sich einen Abreißkalender voller Zufallsbuchstaben vor, von denen jeder nur einmal verwendet wird). Das Alphabet Σ hat die Struktur einer zyklischen Gruppe der gleichen Ordnung, und $M = K = C = \Sigma^n$ für eine feste, aber beliebige Textlänge n. Diese Chiffre wurde schon am Ende des vorigen Abschnitts kurz behandelt. Von besonderem Interesse ist die binäre Version, wo ein Strom $m_1m_2m_3 \ldots m_n$ von Klartext-Bits mit einem Strom $k_1k_2k_3 \ldots k_n$ von Zufallsbits zu einem Strom $c_1c_2c_3 \ldots c_n$ von Geheimtext-Bits durch binäre Addition (also die logische Verknüpfung XOR) überlagert wird, also eine Version der Bitstrom-Verschlüsselung, wie in Abbildung V-4 beschrieben, wobei die Schlüssel-Bits durch reine Zufallsbits repräsentiert werden.

Dieses Verfahren hat Vorteile:

- Perfekte Sicherheit.
- Hohe Effizienz; bei einer Hardware-Implementierung erfolgt die Verschlüsselung sogar *ohne jeden Zeitverzug* bei der Übertragung — wenn die Zufallsbits auf Abruf parat stehen.

Es besitzt auch Nachteile:

- Die Schlüsselübermittlung ist genauso problematisch wie die Übermittlung des eigentlichen Klartexts.
- Die Erzeugung der Zufallsbits muß schnell und sicher sein.

Im Abschnitt 1.8 wird eine neue Entwicklung vorgestellt, die diese beiden Nachteile drastisch mindert.

Allgemein weist der Begriff der perfekten Sicherheit die Richtung, in der man gute Chiffren zu suchen hat. Zur Konstruktion gibt es zwei allgemeine Prinzipien:

Konfusion – der Zusammenhang zwischen Geheimtext und Schlüssel wird verkompliziert.

Diffusion – die im Klartext enthaltene Information wird über die ganze Länge des Texts verschmiert.

Als konkrete Methode zur Vergrößerung der Diffusion bietet sich eine Transposition an, also eine Permutation des Klartexts. Diese alleine bietet noch keine Sicherheit; der Kryptoanalytiker kommt mit der Häufigkeitsanalyse von Paaren und Tripeln schnell zur Entschlüsselung. Eine Kombination von Transposition mit Substitutionen (monoalphabetische Verschlüsselung von Blöcken) führt aber zu den besten bekannten „klassischen“ Chiffren, wie dem im nächsten Abschnitt behandelten ‘Data Encryption Standard’.

Neuere Entwicklungen in der Kryprographie gehen von einem komplexitätstheoretischen Ansatz aus: „Praktische Sicherheit“ ist eine Approximation an die perfekte Sicherheit und bedeutet, daß die Kryptoanalyse zwar im Prinzip durchführbar ist, dabei aber wegen ihrer Komplexität mit verfügbaren Ressourcen nicht in einer Zeit durchzuführen ist, in der der Angreifer an der Entschlüsselung noch interessiert sein könnte. Solche Verfahren werden in den Abschnitten 1.7 und 1.8 vorgestellt.

1.5 Data Encryption Standard

Der ‘Data Encryption Standard’ (DES) wurde im wesentlichen bei der IBM entwickelt und 1977 vom NBS (‘National Bureau of Standards’) in den USA genormt. Das Ziel der Entwicklung war, für 10 bis 15 Jahre ein zuverlässiges Verschlüsselungssystem für sensitive (aber nicht hochgeheime) Daten der Regierung zur Verfügung zu haben. Die Norm verlangt eine Hardware-Implementation des Algorithmus; seit 1989 unterliegen DES-Chips der Ausfuhrbeschränkung.

Verschlüsselt werden 64-Bit-Blöcke, wobei ein 56-Bit-Schlüssel verwendet wird. Die Verschlüsselung eines Blocks beginnt mit einer festen (bekannten) Permutation und endet mit der Umkehrpermutation. Obwohl diese Permutationen bekannt sind, wird dadurch schon eine gewisse Diffusion erreicht. Dazwischen werden 16 „Runden“ durchgeführt, in denen sowohl Diffusion als auch Konfusion erhöht werden. Die einzelnen Runden unterscheiden sich nur dadurch, daß

jeweils eine andere 48-Bit-Gruppe aus dem Schlüssel gewählt wird. Die Entschlüsselung unterscheidet sich von der Verschlüsselung nur dadurch, daß die Runden in umgekehrter Reihenfolge durchlaufen werden.

Im folgenden wird der Algorithmus stufenweise „von innen nach außen“ beschrieben. Sei $\mathbf{F}_2 = \{0, 1\}$ die Menge der Bits, also mathematisch gesprochen der Körper mit 2 Elementen. $\oplus$ ist immer die bitweise Addition modulo 2 (XOR).

Im Innern des DES steckt die „Kern-Abbildung“

$$f\colon \mathbf{F}_2^{32} \times \mathbf{F}_2^{48} \longrightarrow \mathbf{F}_2^{32},$$

die als Input 32 Textbits und einen 48-Bit-Teilschlüssel hat. Zuerst werden die 32 Textbits durch teilweise Wiederholung zu 48 Bits aufgebläht; die „Expansionsabbildung“

$$E\colon \mathbf{F}_2^{32} \longrightarrow \mathbf{F}_2^{48}$$

wird durch die Tabelle V-1 beschrieben:

Tabelle V-1: Die Expansionsabbildung im DES

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Die Tabelle ist so zu interpretieren daß

$$E(b_1 b_2 \dots b_{32}) = b_{32} b_1 b_2 b_3 \dots b_{31} b_{32} b_1.$$

Die expandierten 48 Bits werden mit dem 48-Bit-Teilschlüssel per $\oplus$ überlagert. Die resultierenden 48 Bits werden in 8 Gruppen zu je 6 Bits zerteilt und auf diese die 1. bis 8. S(ubstitutions)-Box

$$S_j\colon \mathbf{F}_2^6 \longrightarrow \mathbf{F}_2^4 \quad (j = 1, \dots, 8)$$

angewendet. Jede der acht S-Boxen S_j wird durch eine 4×16-Matrix beschrieben, siehe Tabelle V-2

Jede Zeile ist eine Permutation der Zahlen $0, \dots, 15$. Um $S_j(b_1 \dots b_6)$ zu bestimmen, deutet man $b_1 b_6$ als Binärdarstellung einer Zahl in $\{0, 3\}$ und $b_2 b_3 b_4 b_5$ als Binärdarstellung einer Zahl in $\{0, 15\}$, liest in der Matrix zu S_j die Zahl in Zeile $b_1 b_6$ und Spalte $b_2 b_3 b_4 b_5$ ab und stellt sie binär dar. Beispiel:

$$S_3(101100) = 0011 \quad \rightarrow \quad \text{Zeile 2, Spalte 6.}$$

Tabelle V-2: Die S-Boxen im DES

S_1	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Insgesamt erhält man die Substitution

$$S\colon \mathbf{F}_2^{48} \longrightarrow \mathbf{F}_2^{32}.$$

Schließlich wird noch die P(ermutations)-Box

$$P\colon \mathbf{F}_2^{32} \longrightarrow \mathbf{F}_2^{32}$$

ausgeführt, die durch die Tabelle V-3 beschrieben wird; das heißt,

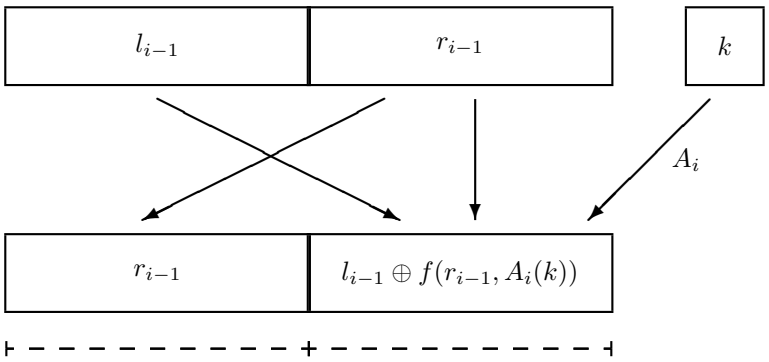
$$P(b_1 b_2 \dots b_{32}) = b_{16} b_7 \dots b_4 b_{25}.$$

Tabelle V-3: Die P-Box im DES

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Zusammengefaßt wird die Kernabbildung in Abbildung V-6.

Abbildung V-5: Eine Runde des DES



Die 16 Runden im DES bestehen aus je einer Abbildung

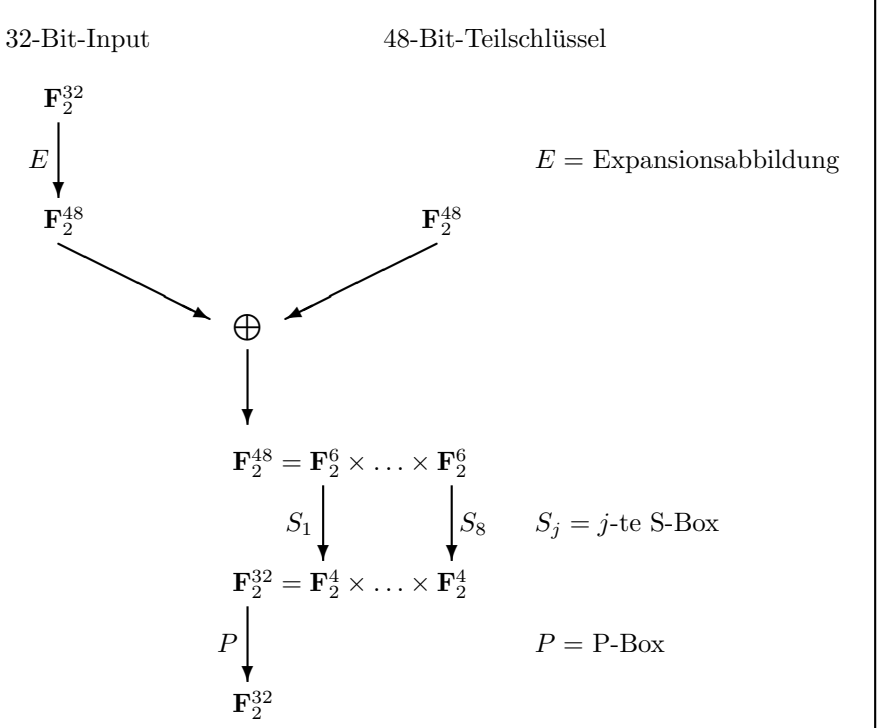
$$R_i\colon \mathbf{F}_2^{64} \times \mathbf{F}_2^{56} \longrightarrow \mathbf{F}_2^{64} \quad (i = 1, \dots, 16),$$

die mit Hilfe der i-ten Schlüsselauswahl

$$A_i \colon \mathbf{F}_2^{56} \longrightarrow \mathbf{F}_2^{48} \quad (i = 1, \ldots, 16),$$

wie in Abbildung V-5 beschrieben wird.

Abbildung V-6: Die Kernabbildung des DES



Die Runden unterscheiden sich also nur durch den verwendeten Teilschlüssel $A_i(k)$. Zur Beschreibung der Runden gehört noch die Beschreibung der Schlüsselauswahl. Zunächst wird der 56-Bit-Schlüssel auf 64 Bit aufgebläht, indem nach je 7 Bits ein Paritätsbit eingefügt wird; welches, ist egal, man kann sogar beliebige Bits einfügen, da die zusätzlichen Bits nicht weiter verwendet werden. Jedenfalls ist der erste Schritt eine Abbildung

$$Par \colon \mathbf{F}_2^{56} \longrightarrow \mathbf{F}_2^{64}.$$

Im zweiten Schritt werden die ursprünglichen 56 Bits wieder extrahiert, allerdings in der folgenden Reihenfolge von Tabelle V-4.

Tabelle V-4: Die ‘Permuted Choice 1’ im DES

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Das ist eine Abbildung

$$PC_1 \colon \mathbf{F}_2^{64} \longrightarrow \mathbf{F}_2^{56}$$

(‘Permuted Choice 1’). Nun werden die 56 Bits in zwei 28-Bit-Hälften geteilt und diese jeweils zyklisch nach links geschoben, insgesamt 16 mal. Das sind also 16 Abbildungen

$$LS_i \colon \mathbf{F}_2^{28} \longrightarrow \mathbf{F}_2^{28} \quad (i = 1, \dots 16);$$

wie weit geschoben wird, zeigt die Tabelle V-5.

Tabelle V-5: Die Verschiebung der Teilschlüssel im DES

1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Die ersten beiden Male wird also um ein Bit geschoben, dann 6 mal um zwei Bits usw. Für die i-te Schlüsselauswahl A_i wird nach der i-ten Verschiebung noch die ‘Permuted Choice 2’,

$$PC_2 \colon \mathbf{F}_2^{56} \longrightarrow \mathbf{F}_2^{48}$$

ausgeführt, wobei die Auswahl in der Reihenfolge von Tabelle V-6 geschieht (die Bits 9, 18, 22, 25, 35, 38, 43, 54 entfallen dabei).

Insgesamt ist

$$A_i = PC_2 \circ LS_i \circ \dots \circ LS_1 \circ PC_1 \circ Par.$$

Diese Konstruktion wird noch einmal in Abbildung V-7 zusammengefaßt.

Nun ist noch die Initial-Permutation

$$IP \colon \mathbf{F}_2^{64} \longrightarrow \mathbf{F}_2^{64}$$

zu beschreiben; das geschieht durch die Tabelle V-7.

Tabelle V-6: Die ‘Permuted Choice 2’ im DES

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Abbildung V-7: Die Schlüsselauswahl beim DES

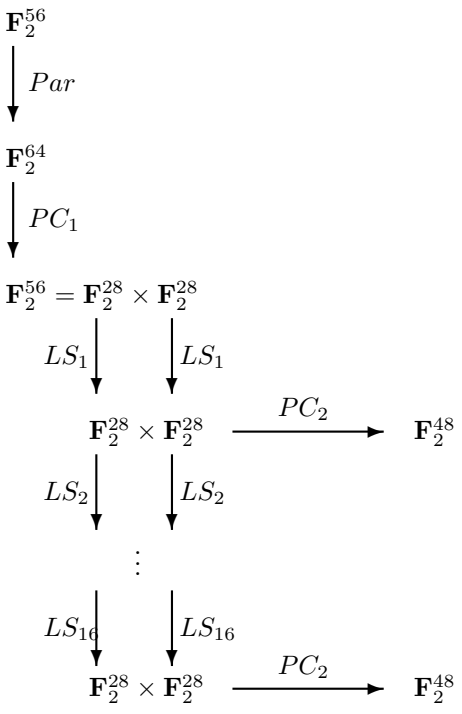


Tabelle V-7: Die Initial-Permutation im DES

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Invers zu IP ist die Ausgabe-Permutation IP^{-1}; der Bequemlichkeit halber wird die zugehörige Tabelle ebenfalls angegeben, siehe Tabelle V-8.

Tabelle V-8: Die Ausgabe-Permutation im DES

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Der gesamte DES-Algorithmus DES_k zum Schlüssel $k \in \mathbf{F}_2^{56}$ ist nun die Zusammensetzung

$$\mathbf{F}_2^{64} \xrightarrow{IP} \mathbf{F}_2^{64} \xrightarrow{R_1(\bullet,k)} \dots \xrightarrow{R_{16}(\bullet,k)} \mathbf{F}_2^{64} \xrightarrow{T} \mathbf{F}_2^{64} \xrightarrow{IP^{-1}} \mathbf{F}_2^{64}.$$

Dabei ist T die Vertauschung der linken und der rechten 32-Bit-Hälften, die man einschiebt, damit DES_k^{-1} bis auf die umgekehrte Reihenfolge der Runden wie DES_k aussieht.

Vorteile des DES-Algorithmus sind, daß es bisher noch niemand gelungen ist, seine Sicherheit ernsthaft in Frage zu stellen, und seine Geschwindigkeit. Gängige DES-Chips verschlüsseln etwa 500 Kbit pro Sekunde; der Rekord liegt bei 20 Mbit/sec. Dagegen sind Software-Implementationen auf einem PC mit etwa 3 bis maximal 20 Kbit/sec für die ernsthafte Anwendung bei umfangreichen Verschlüsselungsaufgaben zu langsam. Allerdings zum Vergleich: RSA-Chips schaffen etwa 10 Kbit/sec, der Rekord liegt bei 30 Kbit/sec. Insgesamt hat sich der DES-Algorithmus in den Jahren seiner Existenz bewährt und darf auch heute noch im Rahmen seiner Zielsetzung als sicher gelten.

Es wurde aber auch einiges an Kritik geäußert:

- Der Schlüsselraum ist zu klein; es gibt nur 2^{56}, also etwa $72 \cdot 10^{15}$ verschiedene Schlüssel. Ein systematisches Durchprobieren aller Schlüssel ist nicht allzuweit vom technisch Machbaren entfernt. Eine Maschine mit 10^6 parallelen Prozessoren, von denen jeder pro Sekunde 10^6 Schlüssel ausprobiert, würde insgesamt im schlechtesten Fall 20 Stunden brauchen. Die Kosten für eine solche Maschine wurden 1982 auf 50 Millionen US-Dollar geschätzt. Aber selbst der Stasi scheint keine solche Maschine gebaut zu haben. Die Aufwandsabschätzung gilt natürlich nur, wenn auch wirklich zufällige Bitfolgen als Schlüssel verwendet werden, nicht etwa nur 7-Buchstaben-Wörter.
- Die Entwurfskriterien und die Beweise für die Sicherheit sind nicht veröffentlicht worden.
- Ebenso sind die Ergebnisse der Sicherheitstests bei der IBM und der NSA ('National Security Agency') nicht veröffentlicht worden.
- Ein Standard ist ein besonders lohnendes Angriffsziel.
- Die Ausfuhr von DES-Chips ist beschränkt. Anscheinend sind aber auch Software-Implementationen (auf leistungsfähigen Großrechnern durchaus brauchbar) betroffen.

1.6 Betriebsarten bei Blockverschlüsselung

Viele Verschlüsselungs-Algorithmen wie der DES wirken nur auf Klartexte fester Länge; beim DES sind das 64 Bits. Kürzere Klartexte kann man auffüllen, etwa mit Leerzeichen oder, besser, mit zufälligen Zeichen. Längere Texte wird man in Stücke der vorgeschriebenen Größe zerhacken; das letzte Stück wird bei Bedarf wieder aufgefüllt. Der zerhackte längere Text wird Stück für Stück oder „Block für Block“ verschlüsselt, man spricht daher von Blockverschlüsselung. Dabei kann man verschiedene Methoden oder „Betriebsarten“ anwenden. Diese Betriebsarten gibt es bei jeder Form von Blockverschlüsselung, sie werden jedoch meist im Zusammenhang mit dem DES behandelt.

Gegeben sei jetzt eine Blockchiffre, also eine Verschlüsselungsfunktion

$$f\colon \mathbf{F}_2^l \longrightarrow \mathbf{F}_2^l,$$

die bijektiv ist (auf die Indizierung mit den Schlüsseln $k \in K$ können wir in diesem Zusammenhang verzichten). Dabei ist $\mathbf{F}_2 = \{0, 1\}$ die Menge der Bits, Klar- und Geheimtexte sind Blöcke von l Bits. Der zu verschlüsselnde Klartext m wird in l-Bit-Blöcke $m_1, \ldots, m_n$ zerlegt; der letzte Block sei dabei auf irgendeine Weise auf l Bits aufgefüllt. Mathematisch gesprochen geht es darum, die Funktion f von $\mathbf{F}_2^l$ auf $(\mathbf{F}_2^l)^n$ fortzusetzen.

Das einfachste und nächstliegende Verfahren heißt 'Electronic Code Book' (ECB): Jeder der Blöcke wird für sich allein mit f verschlüsselt, im Diagramm:

$$\begin{array}{ccc} m_1 & \xrightarrow{f} & c_1 \\ \vdots & & \vdots \\ m_n & \xrightarrow{f} & c_n \end{array}$$

Die Schwäche dieses Verfahrens liegt auf der Hand: Es ist einfach eine monoalphabetische Chiffrierung, wobei das Alphabet aus l-Bit-Blöcken besteht. Die Entschlüsselung ist nur dann hinreichend schwer, wenn die Blocklänge ziemlich groß ist, so daß der Klartext keine identischen Blöcke enthält. Aber auch dann kann ein Gegner noch, wenn er die Möglichkeit zu einem aktiven Angriff hat, früher abgefangene Blöcke in eine Nachricht einschleusen, selbst wenn er sie nicht dechiffrieren kann. Jedenfalls ist die Betriebsart ECB für die meisten Anwendungen zu vermeiden. Besser sind andere Betriebsarten, die eine Diffusion über die Blöcke hinweg erreichen.

Eine solche Diffusion wird beim 'Cipher Block Chaining' (CBC) erreicht. Hierbei wird der i-te Klartextblock m_i mit dem vorhergehenden Geheimtextblock c_{i-1} vor der Anwendung von f per $\oplus$ binär addiert. Die Vorschrift ist also

$$c_i = f(m_i \oplus c_{i-1}).$$

Damit das Verfahren starten kann, wird ein beliebiger Block c_0 gewählt („Initialisierungsvektor"). Diesen muß man nicht unbedingt geheim halten. Da der Empfänger ihn kennen muß, kann man ihn mit der Nachricht übersenden. Als Diagramm ist das ganze Verfahren in Abbildung V-8 dargestellt.

Jeder Geheimtextblock hängt also von *allen vorhergehenden* Klartextblöcken ab. Insbesondere werden gleiche Klartextblöcke mit genügender Sicherheit verschieden chiffriert.

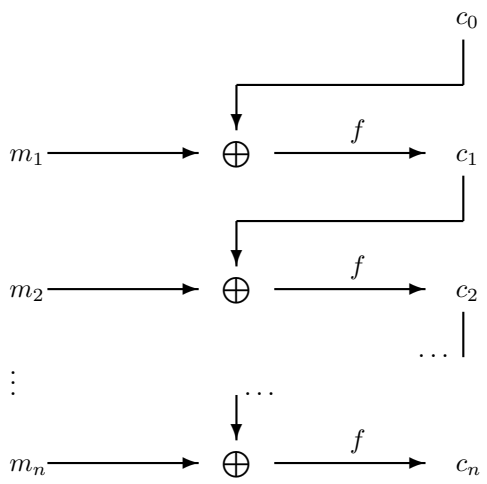
Die Vorschrift für die Dechiffrierung ist

$$m_i = c_{i-1} \oplus f^{-1}(c_i).$$

Jeder Klartextblock hängt dabei von zwei Geheimtextblöcken ab. Bei fehlerhafter Übermittlung eines Geheimtextblocks werden nur zwei Klartextblöcke unleserlich — beim ECB-Betrieb war es sogar nur einer. Man spricht daher in beiden Fällen von einem selbstsynchronisierenden Verfahren. Ein Angreifer wird es aber jetzt sehr schwer haben, Textblöcke zu ersetzen oder einzufügen; um das zu können, muß er schon den ganzen Text entziffern können.

Die dritte Betriebsart heißt 'Cipher Feedback' (CFB). Hier wird als Hilfe ein l-Bit-Schieberegister verwendet. Ferner werden Klartextblöcke der Länge t mit $1 \leq t \leq l$ in ebensolange Geheimtextblöcke umgewandelt, indem sie mit den t ersten Bits des aktuellen Schieberegisterinhalts per $\oplus$ verknüpft werden. Das Schieberegister wird zu Beginn mit einem nicht notwendig geheimen Startwert

Abbildung V-8: Der 'Cipher Block Chaining Mode'



s_0 geladen. Dieser wird mit f verschlüsselt und dann wie beschrieben mit dem ersten Klartextblock m_1 zum ersten Geheimtextblock c_1 verknüpft. Danach wird das Schieberegister um t Bits nach links geschoben; die verwendeten Bits fallen dabei heraus, die übrigen $l - t$ Bits rücken nach links, und von rechts wird das Register mit den t Bits c_i nachgefüllt. Dieser Wert s_1 wird wieder mit f verschlüsselt usw. In Abbildung V-9 ist das entsprechende Diagramm zu sehen.

Die Vorschrift für die Chiffrierung ist also

$$c_i = m_i \oplus p_i.$$

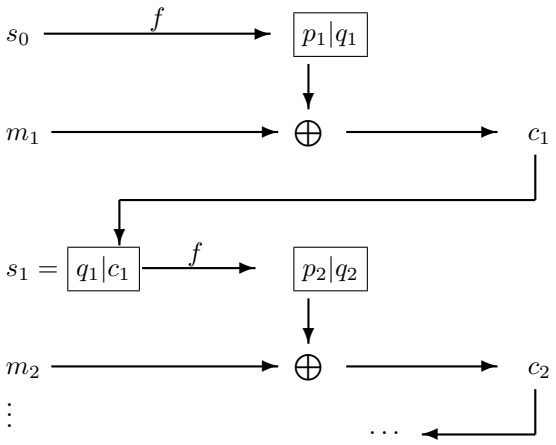
Der Empfänger muß zur Dechiffrierung die s_i und p_i genauso berechnen und verfährt dann nach der Vorschrift

$$m_i = c_i \oplus p_i.$$

Auch in dieser Betriebsart hängt jeder Geheimtextblock von jedem früheren Klartextblock ab, und gleiche Klartextblöcke werden mit ziemlicher Sicherheit verschieden verschlüsselt. Bei fehlerhafter Übertragung sind wenige Blöcke verdorben, je nachdem, wie groß t ist; auch dieses Verfahren ist also selbstsynchronisierend. Bemerkenswert ist allerdings, daß auch der Empfänger nur die Funktion f, nicht etwa f^{-1} braucht. Für asymmetrische Verschlüsselungsverfahren ist die Betriebsart CFB daher nicht geeignet.

Eine ziemlich ähnliche Betriebsart ist 'Output Feedback' (OFB). Der einzige Unterschied ist, daß in das Schieberegister von rechts nicht die t Geheimtext-

Abbildung V-9: Der 'Cipher Feedback Mode'



Bits nachgeschoben werden, sondern die t nach links herausgeschobenen Bits. Es wird also zyklisch geschoben. Bei diesem Verfahren sind Schwächen bekannt geworden, so daß es nicht mehr zur Verwendung empfohlen wird. Es gibt aber eine Verbesserung, von der keine Schwächen bekannt sind. Hierbei wird das Hilfsregister gar nicht mehr als Schieberegister gebraucht, sondern es wird einfach $s_i = f(s_{i-1})$ mit m_i binär addiert (wobei man wieder volle l Bits verwenden kann, also $t = l$). Das ist genau das Verfahren der VERNAM-Chiffre, wobei f als Zufallsgenerator „mißbraucht“ wird. Speziell beim DES gibt es umfangreiche Untersuchungen, die die Qualität der dabei erzeugten Pseudozufallszahlen absichern.

Beim OFB findet keine Diffusion statt, trotzdem werden gleiche Klartextblöcke so gut wie sicher verschieden verschlüsselt. Auch hier braucht der Empfänger wieder nur f und nicht die Umkehrfunktion. Selbstsynchronisation ist nicht gegeben.

1.7 Asymmetrische Chiffrierung

Das mathematische Modell für die Kryptologie besteht wie bisher aus den drei endlichen Mengen der Klartexte, M, der Geheimtexte, C, und der Schlüssel, K. Diese letztere dient als Indexmenge für eine Menge von Paaren (E, D) von Funktionen

$$E\colon M \longrightarrow C, \quad D\colon C \longrightarrow M,$$

mit $D \circ E = \mathrm{id}_M$. Die Schlüssel werden jetzt der Einfachheit halber mit diesen Paaren von Verschlüsselungs- und Entschlüsselungsfunktion identifiziert, haben dann also die Gestalt

$$k = (D, E) \in K \subseteq M^C \times C^M.$$

Die bisher behandelten Verschlüsselungsverfahren waren symmetrisch: Wer verschlüsseln kann, kann auch entschlüsseln, wer E kennt, kennt auch D.

Beispiele.

1. Beim Data Encryption Standard ist $E = DES_k$ und $D = DES_k^{-1}$; die Umkehrfunktion verwendet den gleichen Schlüsselparameter k, durchläuft nur die Runden in umgekehrter Reihenfolge.

2. Bei der Bitstrom-Verschlüsselung ist sogar $D = E$.

Bei solchen symmetrischen Verfahren kennen Sender und Empfänger den gleichen Schlüssel k, und niemand sonst darf ihn kennen. Bestehen Kommunikationsbeziehungen zwischen N Parteien, so sind insgesamt $N(N-1)/2$ geheimzuhaltende Schlüssel nötig. Ihre Anzahl wächst also quadratisch.

Eine völlig andere Idee brachten DIFFIE und HELLMAN 1976 in die Kryptographie ein: die asymmetrische Chiffrierung oder „Chiffrierung mit öffentlichen Schlüsseln“. Dabei besitzt jeder Teilnehmer an der Kommunikation einen Schlüssel (E, D), von dem er den Teil D geheim für sich allein behält, den Teil E dagegen *öffentlich* bekannt macht, etwa zusammen mit seiner Adresse, wie in einem Telefonbuch.

Zwei Vorteile eines solchen Systems springen sofort ins Auge:

- Es ist keine geheime Schlüsselübermittlung nötig; eine Kommunikationsbeziehung kann jederzeit spontan aufgebaut werden.

- Für N Parteien braucht man nur N Schlüssel. Diese Anzahl wächst linear.

Allerdings ist das Problem der Schlüsselverwaltung damit nicht endgültig gelöst, es nimmt eine neue Form an: Ein Schuft kann einen Schlüssel unter falschem Namen veröffentlichen und sich dadurch Nachrichten erschleichen. Um diese Maskerade zu erschweren oder unmöglich zu machen, sind „Zertifikationsverfahren“ nötig. Allgemein gesprochen: Wer einen Schlüssel veröffentlicht, muß sich ausweisen.

Ein solches asymmetrisches Verschlüsselungsverfahren muß neuen Anforderungen genügen:

- Die Entschlüsselungsfunktion D darf aus der Verschlüsselungsfunktion E nicht bestimmbar sein.
- Das Verfahren muß einem Angriff mit ausgewähltem Klartext widerstehen; der Angreifer kann ja selbst beliebige Klartexte verschlüsseln und mit einem aufgefangenen Geheimtext vergleichen.

In den meisten Fällen ist D die Umkehrfunktion von E. Hoffnung, die erste Anforderung erfüllen zu können, erhält man aus der mathematischen Erfahrung, daß Umkehrprobleme oft sehr viel schwieriger sind. Auch die Alltagserfahrung liefert solche Beispiele: Es ist leicht, im Telefonbuch zu einem Namen die Nummer zu finden, schwer dagegen, zu einer Nummer den passenden Namen zu finden.

Als erstes, noch nicht ausreichendes, Beispiel diene eine lineare Abbildung

$$A\colon \mathbf{F}_2^n \longrightarrow \mathbf{F}_2^n.$$

Der Aufwand zu ihrer Auswertung ist von der Größenordnung n^2, für ihre Umkehrung (mit dem bekannten Algorithmus) n^3 — er läßt sich auf n^ω senken, wobei ω in den letzten Jahren immer näher an 2 gerückt worden ist; die zugehörigen Algorithmen sind allerdings praktisch ohne Bedeutung. Wie auch immer, die Umkehrung zu bestimmen, ist etwas aufwendig, aber hinreichend effizient machbar, um dieses Beispiel für asymmetrische Chiffrierung völlig unbrauchbar zu machen.

Ein besseres Beispiel ist die Exponentiation in endlichen Körpern. Sei p eine Primzahl. Die multiplikative Gruppe $\mathbf{F}_p^\times$ ist zyklisch, ihre erzeugenden Elemente heißen „Primitivwurzeln“. Sei a eine solche. Dann ist die Abbildung

$$E\colon \mathbf{F}_p^\times \longrightarrow \mathbf{F}_p^\times, \quad c = E(m) = a^m \bmod p,$$

mit dem binären Potenzalgorithmus effizient auswertbar. Für ihre Umkehrung $m = D(c)$, den „diskreten Logarithmus mod p zur Basis a“ ist dagegen kein effizientes Verfahren bekannt; es wird sogar vermutet, daß es keines geben kann. In diesem Sinne wäre E sogar eine *Einweg-Funktion* und damit für Einweg-Verschlüsselung geeignet.

Allgemein ist es ein offenes mathematisches Problem, ob es überhaupt (in einem präzise zu definierenden Sinne) Einweg-Funktionen geben kann. Dieses Problem ist eng mit dem berühmten ungelösten „$P = NP$-Problem“ verwandt.

Für das Verfahren nach Diffie und Hellman sind solche Einweg-Funktionen aber gar nicht brauchbar: Wenigstens der Empfänger muß ja dechiffrieren können. Hierzu ist eine weitere Idee nötig — die „Falltür-Einweg-Funktion“. Wer eine geheime Zusatzinformation besitzt, kann die Unkehrfunktion effizient berechnen. Diese Zusatzinformation ist dann der geheime private Teil des Schlüssels.

Das ursprünglich von Diffie und Hellman vorgeschlagene konkrete Verfahren beruht auf der Lösung bestimmter linearer diophantischer Gleichungen.

Am bekanntesten ist aber das Verfahren RIVEST, SHAMIR und ADLEMAN — das RSA-Verfahren.

Zur Beschreibung sei an die EULERsche φ-Funktion erinnert: Für eine natürliche Zahl n ist $\varphi(n)$ die Anzahl der zu n teilerfremden Zahlen a mit $1 \leq a < n$, anders ausgedrückt, die Ordnung der multiplikativen Gruppe $(\mathbf{Z}/n\mathbf{Z})^\times$. Ist p eine Primzahl, so

$$\varphi(p) = p - 1.$$

Ist q eine weitere Primzahl, so

$$\varphi(pq) = (p-1)(q-1).$$

Sind a und n teilerfremd, so ist nach dem Satz von EULER

$$a^{\varphi(n)} \equiv 1 \pmod{n}.$$

Die *Schlüsselerzeugung* verläuft jetzt so:

1. Man wählt zwei große Primzahlen p und q zufällig, bildet ihr Produkt $n = pq$ und berechnet $\varphi(n) = (p-1)(q-1)$.
2. Man wählt eine zu $\varphi(n)$ teilerfremde Zahl d in $[1 \ldots n]$ zufällig und bildet ihr Inverses e mod $\varphi(n)$, also $de \equiv 1 \pmod{\varphi(n)}$. Die Inversion geht mit dem Euklidischen Algorithmus effizient.

Das Paar (n, e) ist der öffentliche, die Zahl d der private Teil des Schlüssels. Die Zahlen p, q und $\varphi(n)$ werden nicht mehr benötigt und daher nach der Schlüsselerzeugung am besten vernichtet.

Für die Chiffrierung bildet man die Nachrichtenblöcke, etwa durch ihre Binärdarstellung im ASCII-System, auf die Zahlen zwischen 0 und $n-1$ ab. Man kann also $M = C = [0 \ldots n-1]$ setzen, wenn man sich nicht daran stört, daß der Klartextraum M und der Geheimtextraum C vom Schlüssel abhängen. Ansonsten nimmt man für M die Zahlen bis zu einer festen Schranke und läßt für n nur Zahlen zu, die darüber liegen. Man hat dann stets $M, C \subseteq [0 \ldots n-1]$ auf kanonische Weise. Die Verschlüsselungsfunktion ist dann

$$E\colon M \longrightarrow C, \quad c = E(m) = m^e \bmod n;$$

sie ist wieder mit dem binären Potenzalgorithmus effizient berechenbar. Die Entschlüsselungsfunktion

$$D\colon C \longrightarrow M, \quad m = E(c) = c^d \bmod n$$

kennt nur, wer d kennt, also der Besitzer. (Der Beweis von $DE(m) = m$ bleibt dem Leser als Übungsaufgabe überlassen.)

Damit das RSA-Verfahren den Anforderungen genügt, sind zwei Fragen zu beantworten:

- Wie kann man den geheimen Schlüssel d aus dem öffentlichen (n, e) herleiten?
- Kann man einen Geheimtext dechiffrieren, ohne den Schlüssel zu kennen?

Für die erste Frage ist ein ganz einfaches Verfahren bekannt: Kann man n faktorisieren, so kennt man $\varphi(n)$ und kann d genauso berechnen, wie sein Besitzer bei der Schlüsselerzeugung. Das ist auch im wesentlichen die einzige Methode: Kennt man nämlich d, so kann man n faktorisieren, wenn auch nur „probabilistisch“. Man kennt zunächst das Vielfache $ed - 1$ von $\varphi(n)$. Daraus kann man $\varphi(n)$ durch Probieren oder einen „probabilistischen Algorithmus“ bestimmen. Da

$$\varphi(n) = (p-1)(q-1) = n - (p+q) + 1,$$

entdeckt man auch p und q, die Primfaktoren von n. Das Entdecken des geheimen Schlüssels und die Primzerlegung von n sind damit zwar nicht mathematisch, aber doch vom praktischen Standpunkt aus äquivalent.

Für die Sicherheit des RSA-Verfahrens kommt es also darauf an, wie schwer es ist, große Zahlen in Primfaktoren zu zerlegen. Der derzeit beste bekannte Algorithmus hat einen Zeitaufwand der Größenordnung

$$e^{\sqrt{\log n \log\log n}} \qquad [\log \text{zur Basis } e].$$

Der aktuelle Weltrekord ist die Faktorisierung von Zahlen mit 100 Dezimalstellen in mehreren Tagen auf mehreren weltweit verteilten Rechnern parallel. Der Aufwand für eine 150-stellige Zahl wäre dann das

$$e^{\sqrt{345 \log 345} - \sqrt{230 \log 230}} \approx 13900$$

-fache davon, also in der Größenordnung von 100 Jahren, für eine 200-stellige Zahl das

$$e^{\sqrt{460 \log 460} - \sqrt{230 \log 230}} \approx 51000000$$

-fache, also in der Größenordnung von 400000 Jahren, so daß 200-stellige Zahlen zur Zeit als sicher unfaktorisierbar anzusehen sind.

Für das RSA-Verfahren werden daher Produkte von zwei mindestens 100-stelligen Primzahlen empfohlen. Um schnelleren Faktorisierungs-Algorithmen für spezielle Zahlen zu entgehen, soll man außerdem p und q mit etwa gleich vielen Stellen und (trotzdem) großer Differenz wählen und darauf achten, daß jede der Zahlen $p - 1$ und $q - 1$ einen großen Primfaktor hat.

Um einen Begriff von der Größe solcher Zahlen zu bekommen, sollte man sich klar machen, daß 10^{100} um viele Größenordnungen größer ist als die Zahl aller Atome im Universum (und nochmal viel viel größer als die Zahl der Sandkörner in der Sahara). Aber mit Bleistift ausgeschrieben als 1 mit 100 Nullen schafft man das leicht in zwei Minuten. Primzahlen dieser Größe kann man nicht zufällig erraten: Es gibt nach dem Primzahlsatz über 10^{98} Stück davon. Allgemein

können Probleme exponentieller Komplexität leicht so groß gemacht werden, daß alle Ressourcen des Universums (einschließlich der Zeit) zur Lösung nicht reichen.

Nun zur zweiten Frage: Braucht man den Schlüssel zum Dechiffrieren gar nicht? In bestimmten Fällen ist das tatsächlich so: Iteriert man die Verschlüsselung, so erhält man

$$E^r(c) = c^{e^r} \bmod n = m^{e^{r+1}} \bmod n.$$

Für ein passendes r tritt einmal der Fall $E^r(c) = c$ ein — dann ist $D(E^r(c)) = E^{r-1}(c) = m$ (!!). Daß das eintritt, liegt daran, daß das Potenzieren mit e auf $\mathbf{Z}/n\mathbf{Z}$ eine Permutation ist, und die Permutationsgruppe dieser endlichen Menge ist endlich. Für manche Nachrichten ist r sehr klein, aber diese sind zum Glück ein sehr winziger Teil aller Nachrichten. Es ist hinreichend unwahrscheinlich, eine solche zu erwischen. Für alle anderen ist r so groß, daß man ruhig zusehen kann, wie der Kryptoanalytiker sich tot rechnet.

Beide Fragen führen also nicht auf ernsthafte Einwände gegen das RSA-Verfahren. Es gilt heutzutage als sicher. Sein Hauptnachteil ist die Geschwindigkeit, die schon im Abschnitt 1.5 diskutiert wurde. Dennoch hat es große praktische Bedeutung, zum ersten natürlich in Situationen, wo die Geschwindigkeit keine große Rolle spielt, wie etwa bei elektronischer Post. Zum anderen dient es aber auch dazu, die Schwachstellen symmetrischer Verfahren zu umgehen: Zum Beispiel kann man mit RSA den geheimen Schlüssel für eine anschließende DES-Chiffrierung eines langen und eiligen Textes übermitteln und sogar während einer symmetrischen Verschlüsselung den Schlüssel mehrmals wechseln. Bei solchem gemischten Vorgehen spricht man von „hybrider Chiffrierung“.

1.8 Sichere Zufallsgeneratoren

Bestechend an der Bitstrom-Verschlüsselung ist neben der Geschwindigkeit auch ihre mögliche theoretische Sicherheit. Das Hauptproblem ist die Übermittlung des Schlüssels. Die entscheidende Idee, die es ermöglicht, den Vorteil zu nutzen, ohne den Nachteil in Kauf nehmen zu müssen, ist die Verwendung vom Pseudozufallsfolgen. Das sind Folgen von Zahlen oder Bits, die von einem Algorithmus erzeugt werden und trotzdem „alle“ statistischen Tests auf Unabhängigkeit und Gleichverteilung bestehen. Kein Gegner, der den Ursprung der Folge nicht kennt, findet an ihr irgendeinen Hinweis, der sie von einer „echten“ Zufallsfolge unterscheidet.

Das klassische Verfahren zur Erzeugung von Pseudozufallsfolgen ist die Methode der linearen Kongruenzen. Hierzu werden fest, aber geeignet, gewählt:

- ein **Modul** m mit $m \geq 2$,
- ein **Multiplikator** $a \in [0 \ldots m-1]$,
- ein **Inkrement** $b \in [0 \ldots m-1]$,
- ein **Startwert** $x_0 \in [0 \ldots m-1]$.

Die Zufallserzeugung mit linearen Kongruenzen geht dann so:

$$x_i = ax_{i-1} + b \bmod m.$$

Der Startwert x_0 wird als der geheime Schlüssel angesehen, und nur dieser muß dem Empfänger der Nachricht übermittelt werden; den Rest der Folge kann er dann selbst rekonstruieren. Man wählt meistens $m = 2^{32}$ oder (die Primzahl) $m = 2^{31} - 1$. Von der geschickten Wahl des Multiplikators a hängt die Qualität der Folge in erster Linie ab.

Eine andere klassische, im wesentlichen gleich gute, Methode zur Erzeugung von Pseudozufallsfolgen ist die Schieberegister-Methode. Sie ist besonders leicht in Hardware zu realisieren. Eine lineare Abbildung

$$A\colon \mathbf{F}_2^l \longrightarrow \mathbf{F}_2$$

ist nichts anderes als eine Vorschrift, aus einem l-Bit-Block eine Teilsumme zu bilden:

$$Au = \sum_{i=1}^{l} a_i u_i,$$

und alle Koeffizienten a_i sind 0 oder 1. Man braucht als Parameter des Verfahrens

- die **Registerlänge** l mit $l \geq 2$,
- eine **Rückkopplungsvorschrift** A, die eine Folge $(a_1, \ldots, a_l) \in \mathbf{F}_2^l$ ist,
- einen **Startwert** $u_{l-1} \ldots u_0$ aus l Bits.

Damit wird eine Folge von Bits nach der Vorschrift

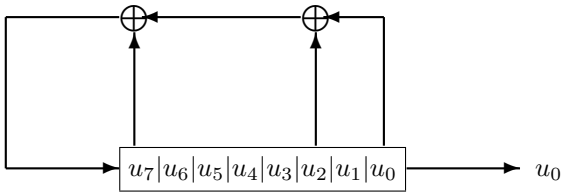
$$u_i = a_1 u_{i-1} + \cdots + a_l u_{i-l}$$

konstruiert. Die Hardware-Realisierung stellt man sich so vor, daß das rechte Bit des Schieberegisters ausgelesen wird, die übrigen $l-1$ Bits nach rechts nachrücken und auf der linken Seite als „Rückkopplung“ die Summe der durch A angegebenen Bits nachgeschoben wird, siehe Abbildung V-10.

Als gemeinsame Verallgemeinerung dieser beiden Methoden kann man die Erzeugung von Folgen mit Hilfe von affinen Abbildungen

$$f\colon (\mathbf{Z}/m\mathbf{Z})^l \longrightarrow (\mathbf{Z}/m\mathbf{Z})^r$$

Abbildung V-10: Ein lineares Schieberegister



ansehen. Alle diese linear erzeugten Pseudozufallsfolgen sind trotz ihrer oft wirklich guten statistischen Zufallseigenschaften kryptographisch nicht sicher. Ein Angriff mit bekanntem Klartext hebelt sie ganz leicht aus. Kennt der Kryptoanalytiker einige Klartext-Bits und die zugehörigen Geheimtextbits, so kennt er einen entsprechend langes Stück der Zufallsfolge:

Klartext	b_i	...	b_{i+r}
Schlüssel	k_i	...	k_{i+r}
Geheimtext	c_i	...	c_{i+r}

Er kann nämlich dann $k_j = b_j + c_j \bmod 2$ ausrechnen. Bei linearen Kongruenzfolgen wie all den oben erwähnten kann man aber aus einem kurzen Abschnitt der Folge alle übrigen Glieder vorwärts und rückwärts berechnen. Falls die Parameter außer dem Startwert (als Schlüssel) bekannt sind, ist das sogar besonders einfach. Aber selbst wenn alle Parameter als Schlüssel geheim gehalten werden, ist es einem Mathematiker möglich, aus wenigen Folgegliedern alle übrigen effizient zu berechnen. Wesentlich schwerer, aber noch längst nicht unmöglich, wird das Knacken, wenn von den erzeugten Zufallszahlen jeweils nur wenige Bits verwendet werden: Dann muß der Angreifer schon einiges über Gitter und Reduktionstheorie, ein sehr schwieriges Gebiet der Mathematik, wissen, in dem selbst professionelle Mathematiker sich in der Regel nicht auskennen; die nötigen Algorithmen existieren erst in der Theorie und sind alles andere als leicht zu implementieren.

Als Verbesserung bietet sich eine nichtlineare Erzeugung von Zufallszahlen an. Bei der Schieberegister-Methode kann man etwa eine nichtlineare Rückkopplungsfunktion

$$f\colon \mathbf{F}_2^l \longrightarrow \mathbf{F}_2$$

wählen. Solch eine Funktion ist automatisch ein Polynom, wie man mit einer Interpolationsformel leicht zeigt; $\mathbf{F}_2$ ist ja ein endlicher Körper. Auch in diesem

Fall ist die Vor- und Rückwärtsberechnung von Folgegliedern immer noch effizient möglich, wenn die Algorithmen dazu auch so kompliziert sind, daß sie zur Zeit nur von einigen wenigen Mathematikern auf der ganzen Welt implementiert werden könnten. Für private Zwecke wäre ein solches Verschlüsselungsverfahren durchaus brauchbar, wenn man darauf achtet, eine Funktion mit möglichst großer „linearer Komplexität“ zu wählen (also eine, die sich sehr von einer linearen unterscheidet) und keine Ansatzpunkte für einen Angriff mit bekanntem Klartext zu bieten. Als offizielles, standardisiertes Verfahren scheidet dieses Vorgehen natürlich aus.

Ein weiteres, gut brauchbares Beispiel ist der DES im ‘Output Feedback Mode’, wie im vorigen Abschnitt erwähnt.

Eine wirklich gute Methode ist in den letzten Jahren entwickelt worden, wobei die Entwicklung durchaus noch nicht abgeschlossen ist: kryptographisch sichere (oder „perfekte“) Zufallsgeneratoren. Das Kriterium für Perfektheit ist, daß sich aus einem beliebigen Teil der Pseudozufallsfolge *kein weiteres* Bit durch einen effizienten Algorithmus mit einer Wahrscheinlichkeit schätzen läßt, die signifikant größer als $\frac{1}{2}$ ist. Die Existenz perfekter Zufallsgeneratoren ist genauso wenig bewiesen wie die Existenz von Einweg-Funktionen. Sie läßt sich aber genauso auf plausible zahlentheoretische Vermutungen wie die Nichteffizienz der Primzerlegung gründen.

Ein erstes Beispiel für einen (vermutlich) perfekten Zufallsgenerator stammt von Blum, Blum und Shub. Hier wählt man zwei große Primzahlen p und q, aus technischen Gründen beide $\equiv 3 \pmod 4$, und bildet ihr Produkt n. Die Faktoren p und q bleiben geheim. Mit einem geheimen Startwert x_0 bildet man die Folge

$$x_i = x_{i-1}^2 \bmod n$$

und nimmt als i-tes Zufallsbit das letzte von x_i,

$$u_i = x_i \bmod 2.$$

Der Beweis, daß die Perfektheit äquivalent zu einer zahlentheoretischen Vermutung ist, ist ziemlich kompliziert; die Nichteffizienz des Quadratwurzelziehens liefert zunächst nur, daß das Rückwärtsberechnen der Pseudozufallsfolge nicht effizient möglich ist. Man darf übrigens nicht nur ein Bit sondern etwa $\log_2 \log_2 n$ Bits von den Zahlen x_i verwenden, ohne die kryptographische Sicherheit zu verletzen. Wenn n also etwa 200 Dezimalstellen hat, darf man stets 9 bis 10 Bits verwenden. Dennoch beträgt der Rechenaufwand selbst dann ungefähr 60 multiplikative Operationen mit 32-Bit-Ganzzahlen, um ein einziges Zufallsbit zu gewinnen. Das ist definitiv zu langsam.

Zur Zeit werden wesentlich schnellere (vermutlich) perfekte Zufallsgeneratoren entwickelt. So haben Micali und Schnorr einen Algorithmus gefunden, bei dem ein Bit nur etwa $\frac{2}{3}$ Ganzzahl-Multiplikationen kostet. Der Hauptvorteil des Algorithmus ist aber seine vollständige Parallelisierbarkeit; mit 16 parallelen Prozessoren arbeitet er schon schneller als die klassischen Zufallsgeneratoren.

Ein Chip, der den DES schlägt (in der Geschwindigkeit, nicht im Preis), steht also in Aussicht.

Werden auf einer Kommunikationsleitung sehr viele Daten übertragen, so könnte man sich zur Beschleunigung der Bitstrom-Verschlüsselung mit Pseudozufallszahlen auch vorstellen, daß Sender und Empfänger je aus einem identischen Vorrat von Zufallsbits schöpfen, die von je einem besonderen Chip auf Vorrat erzeugt werden. Die beiden Chips müßten natürlich synchron arbeiten (logisch, nicht unbedingt zeitlich, das heißt, sie müßten für jede Nachricht ihren Zufallszahlenvorrat an der gleichen Stelle verwenden). Ein solcher Chip müßte einen Prozessor und genügend viel Speicherplatz haben. Wenn der Prozessor sonst nichts zu tun hat, füllt er den Speicher zyklisch mit Pseudozufallsbits auf. Der Prozessor kennt also drei Zustände:

senden: Eine zu sendende oder empfangene Botschaft wird bitweise mit den Pseudozufallsbits überlagert.

Speicher auffüllen: Verbrauchte Pseudozufallsbits werden ersetzt.

warten: Wenn der Speicher voll ist und keine Nachricht ansteht.

Der Prozessor braucht ferner zwei Zeiger-Register, von denen eines auf das nächste unverbrauchte und einer auf das erste noch vorhandene bereits verbrauchte Bit zeigt. Außerdem sollte im Protokoll eine regelmäßige Überprüfung der Synchronisation der Zeiger bei Sender und Empfänger vorgesehen sein.

Mit einer solchen Einrichtung ist die Verschlüsselung in der Regel mit keinerlei Geschwindigkeitsverlust verbunden.

Eine Variante der Bitstrom-Verschlüsselung mit Pseudozufallsbits ist die „probabilistische Chiffrierung“. Sie ist wieder ein asymmetrisches Verfahren. Verwendet man etwa den Blum-Blum-Shub-Generator, so bilden p und q den geheimen, n den öffentlichen Teil des Schlüssels. Der Sender wählt einen Startwert x_0, bildet die Folge so weit, wie er sie benötigt, sagen wir bis x_t, verschlüsselt damit und hängt dann x_{t+1} an seine Botschaft an. Der Empfänger, und nur dieser, kann mit seiner Kenntnis der Primfaktoren p und q den Startwert x_0 aus x_{t+1} effizient berechnen und damit die benötigten Pseudozufallsbits selbst erzeugen. (Wie diese Rechnung geht, wird hier nicht erörtert.) Ein Vorteil vor dem RSA-Verfahren ist, daß gleiche Botschaften verschieden verschlüsselt werden.

1.9 Spezielle Aspekte der Anwendung

Allein die Existenz eines guten Verschlüsselungsverfahrens ist noch nicht die Lösung aller Geheimhaltungsprobleme. Bei der Anwendung muß man sich oft noch weitere Gedanken machen, wie und wann man am besten verschlüsselt. Als Beispiel werden hier zwei Anwendungen aufgeführt: Verschlüsselungen in Datenbanken und in Netzen.

Bei der Verschlüsselung einer Datenbank ist es sicher unzweckmäßig, die ganze Datenbank als Einheit zu verschlüsseln — bei jedem Zugriff müßte die ganze Datenbank entschlüsselt werden; Suchoperationen dauern dann besonders lang. Aber auch die feldweise Verschlüsselung ist nicht besonders gut. Zum einen muß man sehr darauf achten, daß gleiche Felder nicht immer gleich verschlüsselt werden, und auf jeden Fall muß das Verfahren Widerstand gegen einen Angriff mit bekanntem Klartext bieten, da sich der Inhalt einzelner Felder oft erraten läßt. Das Problem wird in [134] im Zusammenhang mit den Betriebsarten des DES diskutiert. Besser scheint es zu sein, die Datenbank dem Server für seine Arbeit unverschlüsselt zur Verfügung zu stellen, und dieser verschlüsselt die Kommunikation mit dem jeweiligen Anwender mit einem speziell für diesen definierten geheimen Schlüssel oder mit dem öffentlichen Schlüssel des Anwenders. Natürlich müssen dann auch Sicherungskopien der Datenbank verschlüsselt sein, und der Server sollte bei Beendigung seiner Arbeit die ganze Datenbank verschlüsseln und das Klartextexemplar vernichten. Für Abstürze braucht man besondere Vorkehrungen.

Im Netz gibt es drei Ebenen, auf denen eine Verschlüsselung sinnvoll ist. Zunächst kann der Anwender seine Nutzdaten verschlüsseln; das liegt dann in seiner Eigenverantwortung, oder ein Anwendungsprogramm, etwa ein Datenbanksystem, besorgt das automatisch. Die Darstellungsschicht (Schicht 6 des OSI-Modells) ist der geeignete Ort für eine vom Netzbetriebssystem vorgegebene Datenverschlüsselung. Eine Verschlüsselung zum Schutz des physischen Datenverkehrs hat ihren Platz in der Verbindungsschicht (OSI-Schicht 2), wo die Daten auch byte- oder blockweise gesichert werden; eventuell ist auch die Vermittlungsschicht (Schicht 3) geeignet. Auf den Schichten 2 oder 3 stellt man sich eine Verbindungsverschlüsselung vor — jeder Zwischenknoten entschlüsselt die ankommenden Daten, liest die Routing-Informationen, sucht sich als Ziel den nächsten passenden Zwischenknoten aus, verschlüsselt wieder und sendet. Auf den höheren Schichten handelt es sich dagegen um eine Endpunktverschlüsselung; sie wird erst wieder aufgelöst, wenn die Nachricht an ihrem physischen Ziel ist.

Für ein verteiltes System mit unsicheren Übertragungswegen, etwa ein Abteilungssystem in einem Krankenhaus, kann man das Sicherheitsmodell aus II-2.5 so modifizieren: Zur Speicherung der sensitiven Daten dient ein Server, der starken physischen Schutz genießt. Die Arbeitsplätze, wo die Daten eingegeben und verändert werden, sind mindestens mittelstark geschützt (abschließbare Räume) und während der Arbeitszeiten, wo der Datenzugriff möglich ist, unter Aufsicht. Verteilerschränke sind so geschützt, daß Manipulationen an ihnen auszuschließen sind. Dann sind die Kommunikationsleitungen, also die unsicheren Übertragungswege, durch Verbindungsverschlüsselung wirksam zu schützen. Ein solches Modell bietet eine hohe Sicherheit bei minimaler Belästigung der Benutzer und ist auch nachträglich durch geeignete Hardware im Server, den Verteilern und den Arbeitsplatzrechnern auf ein bestehendes System aufzupfropfen, ohne daß man bestehende Arbeitsabläufe und Organisationsstrukturen allzusehr

umkrempeln muß.

Jede automatische Verschlüsselung durch ein Netzbetriebssystem oder die Netz-Hardware muß einem Angriff mit ausgewähltem Klartext widerstehen. Unterstellt wird dabei, daß der Angreifer seinen ausgewählten Klartext abschickt und die verschlüsselte Version unterwegs abhört.

Kleinere Probleme sind, daß nach der Verschlüsselung der Klartext vielleicht noch physikalisch gelöscht werden muß, oder daß MS-DOS als Datenmüll ans Dateiende ein Stück Klartext anhängt. Ein gutes Verschlüsselungsprogramm sollte diese Probleme von selbst lösen. Größer ist das Problem, daß ein Schlüssel ja ein Paßwort und damit den gleichen Gefahren ausgesetzt ist. Angreifbar sind auch alle Stellen, an denen Klartexte zwischengespeichert oder bearbeitet werden, vor allem der Hauptspeicher eines PCs.

2 Identifikation und Authentisierung

Dürfen wir elektronischen Dokumenten trauen? Die Antwort darauf ist zunächst ein klares „Nein“. Einer der unpassendsten Sprüche, die man immer wieder hört oder liest, ist: „Dieses Dokument wurde von einer elektronischen Datenverarbeitungsanlage erstellt und ist daher ohne Unterschrift gültig.“ Richtig müßte es heißen „. . . und ist daher ohne jede Beweiskraft.“ Die Beispiele aus Kapitel I.1 haben das belegt. Auch Bilddaten werden zunehmend digital gespeichert, kinderleicht manipulierbar und verlieren dadurch ihren dokumentarischen Charakter. Geradezu grotesk ist, daß es im §37, Absatz 4, des Verwaltungsverfahrensgesetzes heißt, daß „bei einem schriftlichen Verwaltungsakt, der mit Hilfe automatischer Einrichtungen erlassen wird, die Unterschrift fehlen“ kann.

Damit sind wir beim Problemkreis der Signatur. Kryptographische Methoden dienen nicht nur zur Geheimhaltung von Daten und Nachrichten, sondern auch zur Authentisierung: Weder der Urhebernachweis noch der Inhalt sollen verfälscht werden können, wobei die Geheimhaltung des Inhalts oft nicht notwendig ist, aber durchaus als zusätzlicher Anspruch bestehen kann. Auch der Beweis der Identität einer Person (oder eines Prozesses) gehört in diesen Problemkreis. In vielen Anwendungssituationen muß die Kommunikation vom Prinzip des gegenseitigen Mißtrauens ausgehen.

2.1 Authentisierung

Authentisierung ist der Schutz von Daten vor unbefugter Veränderung, wobei hier auch wieder eine ungesicherte Umgebung angenommen wird. Modell: Der Angreifer C ändert eine Nachricht der Senderin A an den Empfänger B unterwegs ab. (In der Kryptologie ist der Gerechtigkeit halber üblicherweise A = Alice weiblich und B = Bob männlich.) Was in Abschnitt II.4.3 unter Authentisierung verstanden wurde, ist ein Spezialfall hiervon: A hat gar nicht gesendet,

sondern C hat die Nachricht überhaupt erst aufgesetzt, und die Nachricht lautet „Ich bin A“. Das Problem der Maskerade ist hiermit also auch erfaßt. Der Schutz gegen Veränderung kann in der Regel aber gar nicht gewährleistet werden; worauf es ankommt, ist, daß die Veränderung entdeckt wird, daß der Täter also Spuren hinterlassen muß.

Das *Ziel* ist also: B soll, wenn er eine Nachricht mit Absender A empfängt, sicher sein können, daß sie wirklich von A stammt und daß sie auch unverfälscht ist, kurz, daß er eine authentische Nachricht erhalten hat.

Ein erster Lösungsansatz ist, daß A und B ein gemeinsames Geheimnis haben, zum Beispiel ein Paßwort, um sich glaubhaft zu identifizieren; das hat freilich nur einen ganz beschränkten Anwendungsbereich und ist ohne weitere Vorkehrungen in einer unsicheren Umgebung leicht abzuhören. Geeignet ist dagegen ein (symmetrischer) Schlüssel: Wenn B bei Entschlüsselung eine sinnvolle Nachricht erhält, kann er sicher sein daß sie auch von A stammt; außerdem ist die Nachricht gleichzeitig vor unbefugtem Lesen geschützt. Es entsteht das übliche Geheimhaltungsproblem: Ist das Geheimnis erst einmal abgelauscht, kann sich der Angreifer perfekt maskieren. Es gibt aber auch weitere Probleme:

- Der Angreifer C kann die Nachricht wiederholen, auch wenn er sie nicht verstanden hat („Wiederholungsattacke“, ‘replay attack’); Schutz bieten zusätzliche Zeitstempel oder Zähler, die mitverschlüsselt werden.

- Der Empfänger B muß vertrauenswürdig sein; sonst ist nicht ausgeschlossen, daß B selbst Nachrichten herstellt und behauptet, sie seien von A. (Zum Beispiel „Ich, A, schulde B 10000 DM“.)

- Andererseits kann B aus genau diesem Grund vor jemand anders nicht beweisen, daß eine Nachricht wirklich von A stammt; insbesondere sind Rechtsstreitigkeiten zwischen A und B so nicht entscheidbar.

- Mit öffentlichen Verschlüsselungsverfahren ist die Methode so nicht durchführbar.

Aus Geschwindigkeitsgründen wird auch oft vorgeschlagen, die eigentliche Nachricht nicht zu verschlüsseln, sondern im Klartext zu übertragen und nur eine verschlüsselte Prüfsumme anzuhängen. Das ist natürlich nur sinnvoll, wenn die Vertraulichkeit keine Rolle spielt, sondern es nur auf die Authentizität ankommt. Hier steckt aber auch ein nicht auf Anhieb erkennbares Problem: Die gängigen Prüfsummen-Verfahren dienen zum Schutz vor *zufälligen* Fehlern. Der Angreifer verfälscht aber systematisch; kennt er das Prüfsummen-Verfahren, kann er Nachrichten ändern, ohne die für ihn nicht lesbare Prüfsumme zu ändern [4]. Es gibt aber auch Prüfsummen-Verfahren (‘hash functions’), die gegen diesen Angriff sicher sind [43].

Eine elegante Lösung des Authentisierungsproblems erhält man mit asymmetrischer Verschlüsselung; sie wird im nächsten Abschnitt 2.2 behandelt.

2.2 Digitale Unterschrift

Eigenschaften einer Unterschrift sind:

- Nur der Eigner kann sie erzeugen.
- Jeder kann sie auf Echtheit prüfen.
- Sie sichert die Authentizität eines Schriftstücks, auch im Falle eines Rechtsstreits.

DIFFIE und HELLMAN haben hierfür das folgende Verfahren vorgeschlagen, das auf asymmetrischer Verschlüsselung beruht, wobei vorausgesetzt wird, daß Verschlüsselung und Entschlüsselung bijektive Abbildungen sind:

1. A verschlüsselt ihre Nachricht m mit ihrem *privaten* Schlüssel D_A.
2. A sendet $c := D_A(m)$.
3. B oder jeder beliebige andere verifiziert die Nachricht und die Absenderin A, indem er den öffentlichen Schlüssel E_A anwendet: $E_A(c) = E_A D_A(m) = m$. Genau dann stammt die Nachricht von A, wenn sich ein sinnvoller Text ergibt.
4. Zum Nachweis der Authentizität muß B nicht nur m, sondern auch c aufheben.

Nicht einmal B kann jetzt eine Nachricht von A fälschen.

Mit einem solchen Signaturverfahren läßt sich auch ein wirksamer Softwareschutz erreichen:

- Beim 'download' in Netzen wird die Software in signierter Form vom Server auf den Knoten überspielt. Damit kann ein Angreifer nicht eine eigene Version mit einem Trojanischen Pferd einbringen.
- Ebenso sind signierte Systemprogramme nicht durch vom Benutzer selbstgefertigte gleichen Namens zu ersetzen, wenn grundsätzlich nur als echt erkannte ausgeführt werden.

Besteht zusätzlich der Bedarf nach Geheimhaltung, so braucht man eine signierte und vertrauliche Nachricht, eine *versiegelte* Nachricht. Dazu sendet A die Nachricht $c := E_B D_A(m)$, und B bildet $s := D_B(m)$ und $m = E_A(s)$; zu Beweiszwecken hebt er s auf. Es könnte aber auch die Abspeicherung von c in einem öffentlichen Archiv als Authentizitäts- und Empfangsnachweis sinnvoll sein.

Welche Probleme gibt es jetzt noch? Das Problem der Schlüsselverwaltung und der Authentizität eines Schlüssels:

- Der öffentliche Schlüssel von A könnte in Wirklichkeit von jemand anders veröffentlicht worden sein.

- Der Schlüssel von A könnte kompromittiert sein;
- oder seine Eigentümerin A könnte dies behaupten und so die Verantwortung für die Nachricht m ablehnen.

Das bekannteste asymmetrische Verfahren, das RSA-Verfahren, ist schon nicht besonders schnell; wird es zum Versiegeln von Nachrichten verwendet, wird es noch langsamer. Sind die Moduln n_A und n_B der jeweiligen Schlüssel unterschiedlich, so muß eine längere Nachricht zwischendurch noch „umgeblockt“ werden, was zusätzliche Zeit kostet. Als Lösung wird hierfür vorgeschlagen, daß ein Schwellenwert h global gesetzt wird und jeder Benutzer zwei Schlüsselpaare bekommt: eines zum Verschlüsseln mit Modul $> h$ und eines zum Signieren mit Modul $< h$. Dann liegen alle Blöcke der signierten Nachricht in einem Bereich, der für die anschließende Verschlüsselung direkt geeignet ist ($c = E_B D_A(m)$; Entsiegelung $m = E_A D_B(c)$ dann auch ohne Block-Probleme).

Ein Musterbeispiel dafür, daß die blinde Anwendung sicherer Verschlüsselungsverfahren zu unvermuteten Sicherheitslücken führen kann, zeigt der folgende Angriff auf das RSA-Verfahren, wenn es sowohl zum Verschlüsseln als auch zum Signieren verwendet wird. Der Grundgedanke dabei ist, dem Empfänger einer Nachricht die abgefangene (verschlüsselte) Nachricht zur Unterschrift vorzulegen. Dabei ist natürlich die Gefahr sehr groß, daß er die Nachricht wiedererkennt oder am Ergebnis sieht, was er getan hat. Der Angreifer kann diese Attacke aber auch verschleiern: Seien dazu n, e und d wie üblich die Schlüssel von A.

Angriff 1. Der Angreifer C hat einen Geheimtext c der Absenderin A abgefangen und möchte den zugehörigen Klartext m kennen. Dazu geht er so vor:

1. C wählt eine beliebige Zahl s und berechnet
   (a) $x := s^e \bmod n$ — dann ist $s = x^d \bmod n$,
   (b) $y := xc \bmod n$,
   (c) $t := s^{-1} \bmod n = x^{-d} \bmod n$.
2. C veranlaßt A, die Zahl y zu signieren (dies ist der springende Punkt des Angriffs), und erhält $u := y^d \bmod n$.
3. C berechnet $tu \bmod n = x^{-d} y^d \bmod n = x^{-d} x^d c^d \bmod n = m$. (!!!)

Damit hat C die Nachricht entschlüsselt – allerdings keinen Anhaltspunkt über den geheimen Schlüssel gewonnen.

Angriff 2. Der Angreifer C möchte die Unterschrift von A fälschen. Dazu geht er genauso vor, nur wählt er für c die zu signierende Nachricht; am Ende hat er c^d, wie gewünscht.

Der Angriff 1 bricht zusammen, wenn zum Verschlüsseln und Signieren jeweils unterschiedliche Schlüssel verwendet werden, der Angriff 2 bleibt aber selbst dann erfolgreich.

Insgesamt hängt der Erfolg natürlich davon ab, daß A eine ihr vorgelegte Nachricht „blind“ unterschreibt. In der Praxis wird ein solcher Angriff nur in Ausnahmefällen glatt über die Bühne gehen. Aber ein sicheres Protokoll sollte auch vor solchen unbewußten Nebenwirkungen törichter Handlungen schützen und nach Möglichkeit nicht dem Benutzer den Schutz vor solchen Gefahren aufbürden. Im übrigen gibt es eine einfache Version eines abhörsicheren Erkennungsdialogs, die genau das vom Benutzer verlangt: eine zufällige Nachricht zu unterschreiben; sie ist im Abschnitt 2.4 behandelt.

Die mathematische Grundlage des verschleierten Angriffs ist, daß die RSA-Verschlüsselung ein Homomorphismus auf der multiplikativen Gruppe $(\mathbf{Z}/n\mathbf{Z})^\times$ ist, ebenso die Entschlüsselung. Ein Lösungsvorschlag von DENNING ist daher, in die Signatur eine (öffentlich bekannte) Einweg-Funktion h einzubauen, die die Homomorphie zerstört; die Signatur ist dann $D_A(h(m))$, und diese wird zusammen mit dem Klartext m verschickt. Da h ebenfalls so etwas wie eine Prüfsumme ist, muß man aber darauf achten, die am Ende von 2.1 erwähnte Falle zu vermeiden.

2.3 Paßwortverschlüsselung

Das Problem der Paßwortverschlüsselung wurde schon wiederholt behandelt. Typisch ist, daß der Klartext für niemand rekonstruierbar sein soll. Es wird nur ein neu erzeugter Geheimtext mit einem abgelegten verglichen. Die Fähigkeit, einen Klartext einzugeben, der nach Durchlaufen der Chiffrierung ein bestimmtes Resultat ergibt, gilt als Nachweis der echten Identität.

Gebraucht wird dazu eine Einweg-Verschlüsselung. Klartextraum M und Geheimtextraum C sind vergleichsweise klein, etwa jeweils die Menge aller 8-Byte-Folgen. Einen Schlüssel braucht man nicht, sondern nur eine feste Einweg-Funktion

$$E\colon M \longrightarrow C.$$

Im Benutzerverzeichnis abgespeichert werden Identität (= Benutzername) und E (= Paßwort). Im Abschnitt 1.7 wurde als Einweg-Funktion die Exponentialfunktion in endlichen Körpern vorgestellt. In der Praxis wird oft ein Polynom über einem endlichen Körper verwendet:

$$f \in \mathbf{F}_p[X],$$

wobei p eine große Primzahl und der Grad von f ebenfalls groß ist; um die Berechnung schnell zu machen, wählt man „dünn besetzte“ Polynome, also solche, bei denen fast alle Koeffizienten 0 sind. Die Umkehrfunktion, also im wesentlichen die Nullstellenbestimmung, gilt als nicht effizient berechenbar. Es sei aber noch einmal darauf hingewiesen, daß die Existenz von Einweg-Funktionen mathematisch noch nicht bewiesen ist.

Die Einweg-Verschlüsselung von Paßwörtern ist anfällig für eine Probeverschlüsselungs-Attacke (Angriff mit ausgewähltem Klartext) mit Fischzug-Angriff. Auch der Internet-Wurm hat das ausgenützt. Auf jeden Fall sollten die

Paßwörter wirklich zufällig sein. Es gibt 2^{64}, also etwa 10^{19}, 8-Byte-Folgen. Aus den 26 (Groß-)Buchstaben des Alphabets lassen sich dagegen nur etwa 10^{11} Folgen von 8 Buchstaben bilden — das ist bereits ein Verlust von 8 Zehnerpotenzen. Werden sogar nur „sinnvolle“ Wörter als Paßwörter gewählt, so gibt es nochmal erheblich weniger Möglichkeiten, die ein Angreifer durchprobieren muß.

Wie man das systematische Probieren durch Schutz auf der Betriebssystemebene unterbindet, wurde schon behandelt. Auf jeden Fall sollte auch ein Verzeichnis mit verschlüsselten Paßwörtern für Unbefugte nicht lesbar sein; damit ist der mögliche Täterkreis sehr eingeschränkt. Aber auch für Systemverwalter sollte kein Fischzug möglich sein. Das Problem, daß Paßwörter ja auch abgehört werden können, wird im nächsten Abschnitt behandelt.

Eine Möglichkeit, Fischzüge zu erschweren und trotzdem leicht zu merkende Paßwörter zuzulassen, ist, vor die Einweg-Funktion noch eine andere Abbildung zu setzen,

$$f\colon M_0 \longrightarrow M,$$

die surjektiv ist. M_0 stellt man sich als Menge von (mehr oder weniger „sinnvollen“) Zeichenketten vor, die genügend groß ist, im Beispiel also mindestens 2^{64} Elemente enthält. (Ob es wohl so viele sinnvolle deutsche Wortbildungen aus 20 Buchstaben gibt, Satz- und Leerzeichen mitgerechnet, Groß- und Kleinschreibung unterschieden?) Die Funktion f sollte so gewählt sein, daß die angenommene Wahrscheinlichkeitsverteilung auf M_0 der Gleichverteilung auf M entspricht. Ein praktischer Vorschlag von KONHEIM unter der Bezeichnung ‘key crunching’ sieht so aus, (wobei E und f gleich zusammengefaßt sind): Man verschlüsselt die Zeichenketten in M_0 mit DES in der Betriebsart CBC und spaltet dann die letzten 64 Bits als Ergebnis ab.

Eine weitere Möglichkeit, die sich mit der ersten kombinieren läßt, wird in manchen UNIX-Systemen verwendet. Hier wird die Identität um eine 12-Bit-Zufallszahl X (genannt ‘salt’) ergänzt; verschlüsselt wird das Paßwort zusammen mit X. Der Angreifer muß bei seinem Fischzug dann nicht nur jedes mögliche Paßwort ausprobieren, sondern jede mögliche Kombination aus Zufallszahl und Paßwort. Der Angriff auf einen bestimmten Benutzer (dessen Zufallszahl als bekannt unterstellt wird) ist dadurch natürlich nicht erschwert.

2.4 Abhörsicherer Erkennungsdialog

Alle Maßnahmen zur Paßwortverschlüsselung nützen nichts, wenn der Gegner das Paßwort abhören kann, bevor es verschlüsselt wird, sei es durch Anzapfen einer Kommunikationsleitung oder durch Aufbau einer Paßwortfalle. Daher werden Protokolle für einen Erkennungsdialog benötigt, bei denen kein Paßwort im Klartext über eine Kommunikationsleitung verschickt wird, oder besser noch, bei denen ein einmal abgehörtes Paßwort für den nächsten Versuch nicht taugt. Die Lösung sieht im Prinzip so aus, daß das Zielsystem einen Zufallstext schickt,

der vom Benutzer mit seinem Paßwort verschlüsselt zurückgeschickt wird. Dieses einfache Schema ist noch anfällig für die Klartextstück-Attacke. Es läßt sich aber auf verschiedene Weise zu wirklich sicheren Protokollen erweitern.

Das erste Protokoll beruht auf einer asymmetrischen Chiffre; der Benutzer, der sich identifizieren soll, braucht ein „intelligentes“ Terminal oder eine Chipkarte. Das Terminal besitzt einen privaten Schlüssel D, der aber auch nur aktiviert werden sollte, wenn der Benutzer ein „lokales“ Paßwort richtig eingegeben oder sich sonst zweifelsfrei identifiziert hat. Der Computer oder das Netz, allgemeiner das Zielsystem, bei dem der Benutzer sich anmelden will, kennt den passenden öffentlichen Schlüssel E. Dann läuft der Dialog so ab:

1. Das Terminal identifiziert sich beim Zielsystem (diese Identifikation muß im folgenden authentisiert werden).
2. Das Zielsystem erzeugt eine Zufallsnachricht m, verschlüsselt sie zu $c = E(m)$ und sendet c an das Terminal.
3. Das Terminal entschlüsselt $m = D(c)$ — niemand sonst kann das — und sendet m zurück.
4. Das Zielsystem vergleicht die Antwort mit der Originalnachricht m. Bei Übereinstimmung wird das Terminal zugelassen.

Ein Abhören dieses Dialogs nützt dem Angreifer überhaupt nichts. Man kann das Verfahren so ausbauen, daß das Terminal (oder die Chipkarte) auch während der Sitzung in zufälligen Intervallen abgefragt wird (‘continuous challenge mode’). Dadurch wird ein unbefugtes Einschleichen in hängende Verbindungen entdeckt und abgewürgt.

Eine Bedingung für dieses Protokoll ist, daß die Zufallsnachrichten im Zielsystem auf kryptographisch sichere Weise erzeugt werden; sonst hat der Angreifer vielleicht Zeit, eine passende Antwort auf die nächste zu erwartende Zufallsnachricht zu berechnen.

Chipkarten werden immer mehr als Ausweise für solche Dialoge vorgeschlagen. Sie wurden schon im Abschnitt 3.4 beschrieben. Zu bemerken ist, daß DES-Chips für die Anforderungen an die Ausmaße einer solchen Karte noch zu sperrig sind.

Auch mit symmetrischen Chiffren kann man einen abhörsicheren Erkennungsdialog gestalten, auch wenn dies etwas umständlicher ist. Das folgende Verfahren wurde von Feistel, Notz und Smith vorgeschlagen. Das Terminal A und das Zielsystem C haben einen gemeinsamen geheimen Schlüssel E mit Entschlüsselungsfunktion D.

1. Das Terminal identifiziert sich beim Zielsystem (diese Identifikation muß im folgenden authentisiert werden).
2. Das Zielsystem verschlüsselt die aktuelle Zeitangabe t zu $s = E(t)$ und sendet s an das Terminal.

3. Das Terminal entschlüsselt $t = D(s)$ — niemand sonst kann das — und prüft die Zeitangabe. Dadurch ist es vor früher abgefangenen Nachrichten sicher, die ein Gegner einspielt.

4. Das Terminal verschlüsselt die Nachricht $m = (t, \text{Paßwort})$ zu $c = E(m)$ und sendet c.

5. Das Zielsystem entschlüsselt c zu m und vergleicht sowohl Zeitangabe als auch Paßwort (das zusätzlich Einweg-verschlüsselt sein kann). Bei Übereinstimmung wird das Terminal zugelassen.

Das dritte hier vorgestellte Protokoll ist eine asymmetrische Version des zweiten. Das Terminal A hat ein Paar (D_A, E_A) aus privatem und öffentlichem Schlüssel, ebenso das Zielsystem C ein Paar (D_C, E_C).

1. Das Terminal identifiziert sich beim Zielsystem (diese Identifikation muß im folgenden authentisiert werden).

2. Das Zielsystem authentisiert die aktuelle Zeitangabe t zu $s = D_C(t)$ und sendet s an das Terminal.

3. Das Terminal prüft die Authentizität der Zeitangabe durch $t = E_C(s)$ — nur das echte Zielsystem kann die Zeitangabe so unterschrieben haben.

4. Das Terminal unterschreibt die Nachricht $m = (t, \text{Paßwort})$ zu $c = D_A(m)$ und sendet c. Am besten wird das Paßwort zuvor mit E_C verschlüsselt.

5. Das Zielsystem löst c zu t und dem Paßwort auf und kann jetzt sowohl Zeitangabe als auch Paßwort prüfen. Bei Übereinstimmung wird das Terminal zugelassen.

2.5 Münzwurf per Telefon

Wie kann man geschäftliche Transaktionen abwickeln, wenn sich beide Partner gegenseitig mißtrauen? Erst die Ware, dann das Geld — oder umgekehrt? Am besten ist es, wenn jeder der Partner bei einem Betrugsversuch des anderen die ganze Transaktion rückgängig machen kann. Das Thema könnte also auch „elektronische Betrugssicherheit“ heißen.

Als einfachster Fall wird betrachtet: A (= Alice) teilt B (= Bob) den Wert eines Bits so mit, daß B ihn nicht ohne As Hilfe lesen kann, aber A ihn auch nach der Mitteilung nicht mehr ändern kann. Der Anschaulichkeit halber wird dieses Protokoll immer als „Münzwurf per Telefon“ bezeichnet; selbstverständlich kann man das auch wörtlich nehmen und Glücksspiele per Telefon damit veranstalten. Die ernsthafte Anwendung wird dadurch ja nicht ausgeschlossen. Als Modell kann man sich vorstellen, daß A das Bit in eine Kiste packt, zu der nur sie selbst einen Schlüssel hat, und diese Kiste an B gibt, der sie bis zur Öffnung beaufsichtigt. Dann gibt B seinen Tip ab, danach gibt A den Schlüssel an B, der

die Kiste öffnet. Beide wissen dann, ob B richtig getippt hat, und keiner hatte die Möglichkeit zu schummeln.

Eine mögliche Implementation dieses Protokolls sieht so aus:

1. A wählt zufällig eine BLUM-Zahl n, also ein Produkt zweier großer Primzahlen p und q, die $\equiv 3 \pmod 4$ sind. Um einen Quadratrest y für den Modul n zu erzeugen, wählt A eine zu n teilerfremde Zahl x, bildet $y = x^2 \bmod n$ und $z = y^2 \bmod n$. Dann ist y der eindeutige Quadratrest, der Wurzel aus z ist. Das geheime Bit ist die Parität von y.

2. A übermittelt n und z an B (die Kiste mit dem eingeschlossenen Bit).

3. B übermittelt seinen Tip über das geheime Bit an A.

4. A übermittelt y und, zum Beweis, daß y Quadratrest ist, auch x, ferner die Faktorisierung von n zum Beweis, daß n eine BLUM-Zahl ist.

5. B verifiziert den Zusammenhang zwischen x, y und z (schließt die Kiste auf und überzeugt sich von der richtigen Lösung).

Falls es keinen effizienten Algorithmus zur Faktorisierung gibt, gibt es auch keinen, der B erlaubt, das geheime Bit mit einer Wahrscheinlichkeit zu schätzen, die signifikant größer als $\frac{1}{2}$ ist.

Bei der ernsthaften Anwendung will A in der Regel nicht eine Münze werfen, also ein zufälliges Bit wählen, sondern ein bestimmtes. Dann muß sie eventuell ein paar Zahlen x durchprobieren, bis y die gewünschte Parität hat. Die Antwort, die die Schlüsselübergabe auslöst, kann etwas völlig anderes als das Raten des geheimen Bits sein.

Auch mit dem diskreten Logarithmus kann man ein solches Protokoll implementieren.

2.6 Das elektronische Vieraugenprinzip

Hier geht es um die gegenseitige Überwachung mehrerer Mitglieder einer Gruppe. Ein „Schloß“ kann beispielsweise geöffnet werden

- von jedem einzelnen Mitglied der Gruppe; das erreicht man durch einen gemeinsamen Schlüssel, den jedes Mitglied kennt;

- oder, im entgegengesetzten Extremfall, nur von allen Mitgliedern der Gruppe gemeinsam; dazu muß man den Schlüssel in entsprechend viele Teile teilen und jedem Mitglied einen Teil geben;

- im allgemeinen Fall von mindestens r beliebigen Mitgliedern der Gruppe; die Lösung dieses Problems ist schon eine Denksportaufgabe.

Die Verwirklichung eines solchen Protokolls heißt „Schwellwertschema“ (‘threshold scheme’) oder „Mehrschlüsselprinzip“, im Falle $r = 2$ auch „Vieraugenprinzip“. Die Größe der Gruppe sei n, und $1 \leq r \leq n$. Der Schlüssel k muß dazu so in n Teile geteilt werden, daß k aus je r Stücken eindeutig und leicht bestimmbar ist, während man aus weniger als r Stücken keinerlei Information über den Schlüssel herleiten kann.

Folgende Anwendungen sind möglich:

- Ein Geheimnis kann auf mehrere Orte verteilt werden; die Entdeckung eines Teilgeheimnisses schadet nichts. Die Zerstörung einiger der Teilgeheimnisse schadet auch nichts, solange wenigstens noch k Stück übrig sind.

- Ganz analog kann man eine geheime Nachricht beim Senden auf n Kanäle verteilen.

- Das Gegenzeichnen bei finanziellen Transaktionen wird ermöglicht, so wie es etwa auf der Bank heißt: „Dokumente sind gültig, wenn sie von zwei der folgenden Berechtigten unterschrieben sind.“

Eine mathematische Lösung für das Schwellwertschema wurde 1979 von Shamir vorgeschlagen. Sie beruht auf der Polynom-Interpolation; analog kann man auch den „chinesischen Restsatz“ verwenden. Man bildet ein Polynom

$$q = a_0 + a_1 X + \cdots + a_{r-1} X^{r-1} \in \mathbf{Z}[X]$$

mit $a_0 = k = q(0)$. Als Teilschlüssel werden die Werte $k_i = q(i)$ für $i = 1, \ldots, n$ verteilt. Weniger als r davon lassen für a_0 *jede* Möglichkeit offen, r Stück dagegen legen q und damit $a_0 = k$ eindeutig fest. Die Berechnung erfolgt mit der Newtonschen Interpolationsformel.

Als weitere Vorteile dieses Schemas zeigen sich:

- Die Zahl n der zugelassenen Personen ist jederzeit leicht erweiterbar, ohne die anderen Teilschlüssel zu ändern.

- Die Teilschlüssel sind durch Ersatz von q änderbar, ohne daß der eigentliche Schlüssel k sich ändert.

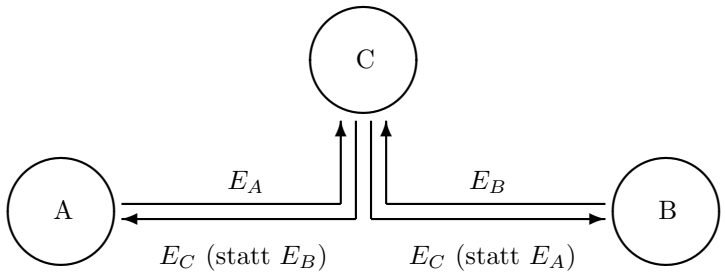
- Man kann leicht eine Hierarchie einführen, indem man besonderen Personen mehrere Teilschlüssel gibt.

2.7 Entlarvung von Lauschern

In diesem Abschnitt wird ein Kommunikationssystem mit asymmetrischer Chiffrierung, aber ohne zentrale Schlüsselverwaltung betrachtet. Das Problem ist dann, daß ein Angreifer seinen Schlüssel in die Kommunikation einschleusen kann. Genauer:

A und B wollen miteinander kommunizieren. Dazu sendet A an B ihren öffentlichen Schlüssel E_A und B an A seinen öffentlichen Schlüssel E_B. Der Angreifer C fängt diese Sendungen ab und ersetzt beide Male den abgefangenen öffentlichen Schlüssel durch seinen eigenen E_C. Dann kann C unbemerkt den ganzen Nachrichtenverkehr zwischen A und B abhören und sogar verfälschen; diese Situation ist in Abbildung V-11 dargestellt.

Abbildung V-11: Maskerade mit öffentlichen Schlüsseln



Mit einer zentralen Schlüsselverwaltung wie im Abschnitt 2.8 läßt sich diese Gefahr abwehren. Es ist aber bei weltweiter Kommunikation nicht unbedingt eine zuverlässige zentrale Schlüsselverwaltung zu verwirklichen. Eine andere Möglichkeit bildet das „Interlock-Protokoll“, das Rivest und Shamir 1984 vorgestellt haben.

In diesem Protokoll werden die Nachrichten immer paarweise verschickt — A sendet m_A an B, und B sendet m_B an A. Diese Situation kann man auch künstlich herbeiführen, indem man jede Nachricht mit einer Empfangsbestätigung quittiert, die aber nicht jedesmal gleich aussehen darf. Das Protokoll verfährt dann so:

1. A verschlüsselt $c_A = E_B(m_A)$ und sendet die erste Hälfte der Bits von c_A an B.
2. B verschlüsselt $c_B = E_A(m_B)$ und sendet die erste Hälfte der Bits von c_B an A.
3. A sendet die zweite Hälfte von c_A.
4. B sendet die zweite Hälfte von c_B.
5. A und B setzen ihre Hälften zusammen und entschlüsseln.

Das ist also ganz einfach. Wie sieht es nun aus, wenn sich der Angreifer C wie beschrieben in die Beziehung eingeschmuggelt hat? Nach dem ersten Schritt hat er die Hälfte von $c_A = E_C(m_A)$, die A ja mit dem untergeschobenen Schlüssel chiffriert hat. Diese Hälfte reicht ihm nicht, um m_A zu entschlüsseln. C muß aber irgendetwas an B senden, sonst bricht die Kommunikation ab. Also bleibt ihm nichts anderes übrig, als eine Nachricht m'_A zu erfinden und die erste Hälfte von $c'_A = E_B(m'_A)$ zu senden. Analog muß er im Schritt 2 verfahren. Nach dem dritten Schritt kann C dann die ganze Nachricht m_A lesen, aber dann ist es zu spät, um m'_A noch passend zu ändern. C hat zwar die Nachricht m_A unberechtigt gelesen, hat sich dabei aber verraten — A und B entdecken, daß sich jemand in ihre Beziehung eingemischt hat, wenn sie auch nicht wissen, wer der Lauscher war.

Wenn man sich auf dieses Protokoll verläßt, wird man als erste Nachricht natürlich eine harmlose Testbotschaft senden; C kann ja nur Erfolg haben, wenn er *alle* Nachrichten mit seiner Methode abfängt. Daher wird durch dieses Protokoll ein solcher maskierter Lauschangriff zuverlässig verhindert.

2.8 Schlüsselverwaltung

Bei symmetrischen Chiffrierverfahren besteht das Problem, die geheimen Schlüssel so zu verteilen, daß nur die jeweiligen Partner einer Kommunikationsbeziehung den passenden Schlüssel kennen. Bei den asymmetrischen Verfahren ist dagegen das Hauptproblem, daß nicht notwendig sicher ist, wer hinter einem öffentlich bekanntgegebenen Schlüssel wirklich steckt. Beide Probleme lassen sich unter dem Thema Schlüsselverwaltung und -verteilung zusammenfassen.

Der einfachste Fall ist natürlich, daß Sender und Empfänger identisch sind; er tritt zum Beispiel ein, wenn ein Benutzer seine privaten Daten auf einer Festplatte verschlüsselt. Der Schlüssel muß dann nur bei einer Person sein, die auch allein für ihn verantwortlich ist.

Der einfachste nichttriviale Fall ist eine Zweierbeziehung. Übertragen werden soll ein Schlüssel für eine symmetrische Chiffrierung. Dazu haben Diffie und Hellman das folgende Verfahren vorgeschlagen, das auf der Exponentialfunktion in endlichen Körpern, also einer (mutmaßlichen) Einweg-Funktion beruht:

1. A und B einigen sich (öffentlich) über eine Primzahl p und eine zugehörige Primitivwurzel a.
2. A erzeugt eine Zufallszahl x, bildet $u = a^x \bmod p$ und sendet u an B.
3. B erzeugt eine Zufallszahl y, bildet $v = a^y \bmod p$ und sendet v an A.
4. A berechnet $k = v^x \bmod p$, und B berechnet $k = u^y \bmod p$.

Die Zahl k ist der gemeinsame geheime Schlüssel (oder dient zu dessen Bestimmung nach einem öffentlich bekannten Verfahren). Daß sowohl A als auch B den gleichen Schlüssel haben, liegt an der Gleichung

$$v^x \equiv a^{xy} \equiv u^y \pmod{p}.$$

Ein Lauscher kann nur die Zahlen u und v abfangen, die ihm nicht gestatten, k oder x oder y effizient zu berechnen.

Für komplizierte Kommunikationsbeziehungen ist für die Schlüsselverwaltung und -verteilung eine zuverlässige Zentralstelle von Vorteil (KDC = 'Key Distribution Center'). Diese entwickelt und verteilt die Schlüssel. Der Empfänger des Schlüssels muß sicher sein können, daß dieser authentisch ist.

Ein Modell für ein symmetrisches Chiffriersystem sieht etwa so aus:

1. Die Zentrale bestimmt für jeden Teilnehmer A einen individuellen Hauptschlüssel H_A. Dieser wird auf einem sicheren Wege übermittelt, etwa durch einen Kurier — dieser kann auch A selbst sein, die ihren Schlüssel in der Zentrale abholt. Da das ein seltener Vorgang ist, ist dieses Übermittlungsverfahren tragbar.
2. Will A mit B kommunizieren, beantragt sie das bei der Zentrale. Diese erzeugt dann einen zufälligen (Einmal-)Schlüssel k und schickt diesen an A mit H_A verschlüsselt und an B mit H_B verschlüsselt. Dann können A und B den Schlüssel k zur Kommunikation verwenden und sonst niemand.

Etwas unwohl kann einem bei diesem Protokoll schon werden, weil die Zentrale die Rolle des Großen Bruders spielt und alle Kommunikationsbeziehungen registrieren kann. Hier sollten die Protokolle auch noch die Anonymität gewährleisten; dieser Problemkreis wird in 3 behandelt.

Bei öffentlichen Schlüsseln ist ein Zertifikat einer zentralen Stelle die geeignete Absicherung. Diese signiert den öffentlichen Schlüssel, den ein genügend ausgewiesener Benutzer vorlegt, mit ihrem eigenen geheimen Schlüssel (oder besser den Schlüssel zusammen mit einem bestätigenden Text). Sie selbst ist dadurch authentisiert, daß sie allgemein bekannt ist. Sie spielt also eine Rolle, die mit der eines Notars vergleichbar ist. Aus organisatorischen Gründen kann diese Zentralinstanz auch aus mehreren Hierarchiestufen bestehen, von denen jede durch die nächsthöhere zertifiziert wird und die oberste eine allgemein bekannte Institution ist. (Vergleichbar: Der Bundespräsident stellt eine Urkunde

aus, mit der die nachgeordneten Instanzen sich gegenüber dem Bürger ausweisen können.)

Ein wichtiger Aspekt der Schlüsselverwaltung kann auch das Ungültigmachen eines Schlüssels sein, vergleichbar der Sperre einer Scheckkarte, wenn der Besitzer sie als verloren gemeldet hat. Seien D_A und E_A der private und der öffentliche Schlüssel der Benutzerin A. Dieser soll ab dem Zeitpunkt t ungültig sein. Die folgende Methode verlangt, daß alle Transaktionen über den Schlüsselverwalter S abgewickelt werden (wieder nach dem Modell eines Grundstückskaufs unter den Augen eines Notars). A will B eine Nachricht m schicken und signiert sie als $c = D_A(m)$. S versieht sie mit dem Zeitstempel und signiert seinerseits: $s = D_S(c, t)$. B erhält dann die Nachricht s und kann sowohl den Zeitstempel des „Notars“ als auch die Authentizität der eigentlichen Nachricht m verifizieren.

2.9 Das Identifikationsschema von FIAT und SHAMIR

Hier geht es darum, wie man sich zweifelsfrei ausweisen kann. Die Situation sei wie folgt: Eine Benutzerin A hat einen Ausweis in Form einer Chipkarte (plus eventuell Paßwort oder PIN), mit dem sie sich vor einem Zielsystem B ausweisen muß. Das Zielsystem hat Terminals oder Kartenleser, die keine Geheimnisse enthalten sollen (damit sie keinen Anreiz zum Klauen bieten). Ausgegeben werden die Karten von einer vertrauenswürdigen Zentrale C, etwa einer Regierung, einer Kreditkartenorganisation, einem Rechenzentrum oder einem militärischen Hauptquartier, je nach Anwendungsfall. Die Zentrale soll *nur* die Aufgabe der Kartenausgabe haben; insbesondere soll sie nichts mit den Identifikationsprozeduren zu tun haben und auch die Benutzer nicht speichern.

Ein Schema dafür wurde 1986 von FIAT und SHAMIR vorgeschlagen. Darin hat die Zentrale C ein großes „Zentralgeheimnis“, das aus zwei BLUM-Primzahlen p und q besteht. Das Produkt $n = pq$ ist öffentlich bekannt und ebenso eine Pseudozufallsfunktion

$$f\colon [0 \ldots n-1] \times \mathbf{N} \longrightarrow [0 \ldots n-1].$$

Alle Karten und Terminals kennen n und f.

Wenn A nun eine Karte haben will, muß sie sich zunächst zweifelsfrei vor C ausweisen; sei I eine Zahl, die die Identität und Kartenmerkmale je nach Zweck, etwa Verfallsdatum oder Befugnisse, repräsentiert. Daraus berechnet C einige „Varianten“ von I:

$$v_1 = f(I, 1), \quad v_2 = f(I, 2), \quad \ldots,$$

und wählt k Stück

$$v_{i_1}, \ldots, v_{i_k}$$

davon aus, die Quadratreste modulo n sind; da C die Faktoren p und q kennt, kann C das leicht entscheiden und genauso leicht Zahlen s_j berechnen mit

$$s_j^2 v_{i_j} \equiv 1 \pmod{n}.$$

Auf die Karte werden dann die Paare $(s_1, i_1), \ldots, (s_k, i_k)$ als *Geheimnis* geschrieben — sie dürfen also nicht auslesbar sein. Die Kennzahl I wird offen auf die Karte geschrieben, ebenso die Indizes $i_1, \ldots, i_k$.

Jetzt zum Identifizierungsvorgang: A identifiziert sich vor B, indem sie beweist, daß ihre Karte die Zahlen $s_1, \ldots, s_n$ kennt, ohne daß sie diese preisgeben muß.

1. Die Karte sendet I und $i_1, \ldots, i_k$ ans Terminal.
2. Das Terminal berechnet die $v_{i_j} = f(I, i_j)$.
3. Jetzt beginnt eine Schleife über $\nu = 1, \ldots, t$ mit genügend großem t.
   (a) Die Karte erzeugt eine Zufallszahl $r_\nu \in [0 \ldots n-1]$, bildet $x_\nu = r_\nu^2 \bmod n$ und sendet dies ans Terminal.
   (b) Das Terminal sendet eine zufällige Folge $(e_{\nu 1}, \ldots, e_{\nu k}) \in \mathbf{F}_2^k$ von k Bits an die Karte.
   (c) Die Karte berechnet
   $$y_\nu = r_\nu \cdot \prod_{e_{\nu j}=1} s_j \bmod n$$
   und sendet dies ans Terminal.
   (d) Das Terminal prüft, ob
   $$x_\nu = y_\nu^2 \cdot \prod_{e_{\nu j}=1} v_{i_j} \bmod n.$$

Nur die rechtmäßige Karte, die die s_j kennt, kann solche Zahlen zuverlässig produzieren. Die Wahrscheinlichkeit, daß B eine falsche Karte akzeptiert, ist $\leq 1/2^{kt}$. Die Parameter $k = 5$, $t = 4$ liefern die „Irrtumswahrscheinlichkeit“ $1/2^{20}$, die in der Praxis noch zu groß sein dürfte. Aber eine leichte Verschärfung reicht schon aus. Eigenschaften des Verfahrens:

- Das Identifikationsverfahren ist effizient, also hinreichend schnell.
- Die Interaktion zwischen Karte und Terminal führt nicht zu einer Kopiermöglichkeit der Karte, da sie die geheimen Zahlen s_j nicht enthüllt. Das ist besser als bei den gegenwärtigen elektronischen Zahlungssystemen, die dem Händler das Kopieren der Karte ermöglichen.

- Wird das Geheimnis einer Karte enträtselt, so ist es dennoch nicht möglich, diese Karte zu ändern oder irgendeine neue Karte zu erstellen.

Zur Sicherheit der Karte sollte natürlich auch eine Identifikation des Benutzers gegenüber seiner Karte erfolgen, mindestens durch eine PIN.

Bei diesem System ist das „Zentralgeheimnis“ natürlich der lohnendste Angriffspunkt.

3 Anonymität

Die Möglichkeit zur anonymen Kommunikation ist Teil des informationellen Selbstbestimmungsrechts und dient der Verhinderung von Kommunikationsprofilen. Sie ist sicher auch ein wichtiges Kriterium für die Akzeptanz neuer technischer Entwicklungen. Andererseits ist die Frage, wie weit die Netzbetreiber verpflichtet sein sollten, anonyme Kommunikation zu ermöglichen — schließlich könnten sich die Kommunikationspartner auch treffen, einen richtigen Brief aus Papier schreiben oder aus einer Telefonzelle anrufen. Auch das Argument, daß Verbrecher durch anonyme Kommunikationsmöglichkeiten aufregende neue Möglichkeiten geschenkt bekommen, darf nicht aus den Augen verloren werden.

Anonymität kann den Sender, den Empfänger oder die Kommunikationsbeziehung betreffen, je nach Problemfall. Als mögliche Angreifer betrachten muß man je nach Lage

- den Netzbetreiber oder -verwalter,
- Fremde, zum Beispiel Spione,
- den Kommunikationspartner,
- Trojanische Pferde in der Netz-Software.

Für alle diese Forderungen gibt es Protokolle, die theoretisch machbar sind. In den meisten Fällen sind diese Verfahren in der Praxis allerdings quälend langsam oder völlig indiskutabel.

3.1 Empfängeranonymität

Die Anonymität des Empfängers wird durch Streusendung (‘broadcasting’) erreicht. Das bedeutet, daß die Nachricht an alle geschickt wird, der Empfänger muß erkennen, daß sie für ihn bestimmt ist. Man stellt sich etwa eine Anzeige auf einem elektronischen Anschlagbrett vor, das jeder liest; natürlich ist eine Verschlüsselung möglich. Das Modell hierfür ist die Zeitungsannonce

> „Mausi, ich liebe Dich — komm zurück.
> Dein Bussibär.“

Das Problem ist der riesige Filteraufwand bei allen potentiellen Empfängern.

3.2 Senderanonymität

Hier besteht die Lösung in der Existenz von „Pseudonymen“, also einem Satz von Identitäten, die außer vom Besitzer selbst von niemandem in Beziehung gebracht werden können, und die die eine universelle Identität ersetzen, die den universellen Datenabgleich ermöglicht.

Detaillierte Vorschläge zur Verwirklichung digitaler Pseudonyme stammen von Chaum, siehe etwa [22]. Ein digitales Pseudonym ist im wesentlichen ein öffentlicher Schlüssel, von denen jeder Teilnehmer genügend viele besitzt. Zur Erzeugung gibt es im Chaumschen Modell Chipkarten im Taschenrechner-Format, die keine Geheimnisse enthalten und auf dem freien Markt erhältlich sind.

Probleme:

- Die Pseudonymverwaltung analog zur Schlüsselverwaltung.
- Die Betrugssicherheit.
- Das Zusammenspiel der verschiedenen Pseudonyme einer Person, zum Beispiel der Wertetransfer zwischen ihnen.
- Die Erreichbarkeit einer bestimmten Person.

3.3 Anonymität von Verbindungsdaten

Die nächstliegende, aber unrealistische Lösung ist das elektronische Rauschen: Jeder Teilnehmer sendet ständig nach der „Streusendungsmethode“. Der richtige Empfänger erkennt echte Nachrichten an einem speziellen Signal, das er mit dem Sender vereinbart hat oder an der Verschlüsselung.

Problem: Damit ist jedes Netz überlastet.

Eine gemäßigte Abart dieser Methode, Kommunikationsbeziehungen zu verschleiern, ist die Einrichtung von Sammelstellen; die Teilnehmer sind bei Sammelstellen nur unter Pseudonymen bekannt, Nachrichten werden gesammelt und in zufälliger Reihenfolge an den Empfänger oder eine andere Sammelstelle weitergeleitet.

Nachteile: Zeitkritische Nachrichten werden verzögert; Vertrauenswürdigkeit der Sammelstellen.

Auch ringförmige Netztopologien können zur Verschleierung von Kommunikationsbeziehungen beitragen.

3.4 Elektronische Münzen

Geld ist anonym — sein Weg läßt sich nicht verfolgen. Zumindest gilt das für Münzen, aber auch Geldscheine anhand ihrer Nummer zu verfolgen, ist nicht ohne weiteres möglich. Chaum schlug 1985 das folgende Protokoll für elektronische Münzen vor; selbstverständlich ist es noch nicht realisiert. Es beruht auf

der RSA-Chiffre. Vorausgesetzt ist, daß die zur Münzausgabe berechtigte Bank einen Schlüssel mit öffentlichem Teil (n, e) und geheimem Teil d hat.

Nehmen wir nun zum Beispiel an, daß die Kundin A von der Bank B ein elektronisches 1000-DM-Stück möchte. Später möchte sie damit beim Händler H einkaufen, ohne daß sie dabei dem Händler ihre Identität offenbaren muß, wie etwa bei der Ausstellung eines Schecks. Dazu dient das folgende Protokoll:

1. A erzeugt folgendermaßen Rohmaterial für die Münze:
   - A wählt eine große Zahl v (als „Banknoten-Nummer“); diese Zahl stellt man sich etwa 100-stellig vor, wobei n mindestens 200 Stellen haben sollte.
   - A wählt eine große Zahl c (als „Camouflage“ = Verschleierung) kleiner als n und teilerfremd zu n.
   - A bildet eine Zahl w, indem sie v zweimal hintereinander schreibt.
   - A bildet die Zahl $s = c^e \cdot w \bmod n$ (als anonymisierte Banknoten-Nummer), reicht diese bei der Bank ein und bittet, ein 1000-DM-Stück daraus zu prägen.
2. Die Bank bildet die „Rohmünze“ $t = s^d \bmod n$ und übergibt sie an A.
3. A macht daraus die endgültige Münze $m = tc^{-1} \bmod n$. Dadurch wird die Camouflage entfernt, denn
   $$m = s^d c^{-1} = c^{ed} w^d c^{-1} = w^d \bmod n.$$
   Geschieht dies erst im Moment des Bezahlens und wird c als Paßwort behandelt, so schützt das Verfahren auch vor Diebstahl.
4. Jetzt geht's ans Bezahlen. Der Händler H prüft die Echtheit der Münze, indem er den öffentlichen Schlüssel der Bank anwendet. Das Ergebnis $w = m^e \bmod n$ beweist die Echtheit, wenn es aus einer sich wiederholenden Ziffernfolge besteht.
5. H reicht die Münze bei der Bank zur Gutschrift ein; diese prüft die Echtheit ebenso und merkt sich die Banknoten-Nummer v, damit die Münze nicht zweimal eingereicht wird. Die beiden letzten Schritte sollten praktisch gleichzeitig erfolgen, damit auch H vor Bezahlung mit einer schon verbrauchten Münze sicher sein kann.

Etwas allgemeiner braucht man, daß Klartext- und Geheimtextraum Halbgruppen mit 1 und die Verschlüsselungsfunktionen Isomorphismen sind. Die Camouflage c muß als invertierbares Element gewählt werden. Dann wird analog der Reihe nach gebildet:

$$s = E(c) \cdot w, \;\; t = D(s), \;\; m = c^{-1} \cdot t.$$

Bei diesem Protokoll bleibt A nicht nur vor dem Händler, sondern auch vor der Bank anonym: diese kann die Münze nämlich nicht wiedererkennen, da sie bei der Prägung ja mit c verschleiert war. Dieses Verfahren garantiert also die vollständige *Käuferanonymität*.

Wie sieht es bei Diebstahl (trotz der erwähnten Sicherung) oder Erpressung (gegen die es keine direkte Sicherung gibt) aus? Dann verzichtet A sofort auf ihre Anonymität und meldet die Banknotennummer v an ihre Bank. Der Dieb oder Erpresser kann die Münze dann nicht mehr unbemerkt einreichen oder wird nachträglich identifiziert.

Insgesamt beruht das Protokoll also auf drei Säulen:

Kennzeichnung durch Numerierung schützt vor Mißbrauch.

Anonymisierung durch Camouflage schützt den rechtmäßigen Besitzer vor Ausspähung seiner Geschäfte.

Verschlüsselung schützt die Bank (letztlich die Volkswirtschaft) vor Falschmünzerei.

3.5 Elektronische Bescheinigungen

Oft kommt es vor, daß ein Individuum einer Organisation eine Bescheinigung einer anderen Organisation vorlegen muß. Zum Beispiel könnte das Individuum eine Autofahrerin, die Bescheinigung der Führerschein, die fordernde Organisation die Verkehrspolizei und die ausstellende Organisation die Kreisverwaltung sein. In der heutigen Praxis werden zu diesem Zweck Ausweispapiere eingesetzt. Dabei gibt es Probleme:

- Die Ausweise enthalten Informationen, die im Einzelfall nicht alle relevant sind.
- Mißbrauch und Fälschung sind vergleichsweise leicht (besonders bei Magnetkarten).

Ein Beispiel für irrelevante Informationen tritt bei der Verkehrskontrolle auf. Hier ist vielleicht im Moment nur die Ein-Bit-Information, ob die Fahrerlaubnis vorliegt, von Interesse. Alle anderen Informationen wie Name und Wohnort können zu einer unerwünschten Datenspeicherung, zum Datenabgleich und langfristig zu einer Profilbildung mißbraucht werden.

Der aktuelle Trend ist, maschinenlesbare und fälschungssichere Ausweise einzuführen. Dabei wird noch mehr auf zentrale Speicherung gesetzt; der Umfang der leicht abrufbaren irrelevanten Informationen nimmt zu. Auf jeden Fall gibt die Organisation die Ausweise an die Individuen aus und schützt sich selbst dabei so gut wie möglich vor Mißbrauch. Der Schutz der Individuen vor Datenspeicherung und Datenabgleich wird zu wenig beachtet.

Dagegen verwaltet beim Ansatz von CHAUM jedes Individuum seine Urkunden selbst. Es besitzt zu diesem Zweck mehrere Pseudonyme, zwischen denen niemand sonst eine Verbindung herstellen kann. Auf diese Weise läßt sich mit dem informationellen Selbstbestimmungsrecht in der Tat ernst machen. Ein kryptographisches Protokoll verhindert dabei Fälschung, Änderung, Weitergabe und Diebstahl und ermöglicht den periodischen Wechsel der Pseudonyme, um Profilbildung weiter zu erschweren.

Voraussetzung für diesen Ansatz ist, daß jeder Bürger eine Art Chipkarte besitzt; sie könnte vielleicht auch die Größe eines Taschenrechners haben. Sie enthält keine inneren Geheimnisse und dient nur zur Unterstützung des Besitzers, indem sie ihm die für die kryptographischen Protokolle nötigen Rechnungen abnimmt. Diese Karten sind auf dem freien Markt erhältlich; jeder kann sich unter den Konkurrenzprodukten das aussuchen, welches ihm am geeignetsten erscheint. Eine solche Karte kann auch jederzeit durch ein schickes neues Modell ersetzt werden. Zu diesem Zweck und auch aus anderen Gründen sind diese Karten Backup-fähig, das heißt, die in ihnen gespeicherten Daten können als Sicherungskopie abgezogen und etwa auf einer Diskette oder sogar auf Papier aufbewahrt werden — vom rechtmäßigen Besitzer. Auf jeden Fall muß sich der Besitzer der Karte gegenüber ausweisen, vielleicht durch ein Paßwort oder durch einen Fingerabdruck. Die Karte kann auch als elektronische Brieftasche dienen, indem sie Informationen über das Vermögen enthält, etwa in Form von elektronischen Münzen.

Als Ausweis gegenüber einer Organisation dient diese Karte, nachdem der Besitzer sich für diese Organisation ein Pseudonym erzeugt hat. Gegenüber jeder Organisation benutzt er ein anderes Pseudonym; zum bargeldlosen Einkauf kann er sogar Einmal-Pseudonyme erzeugen.

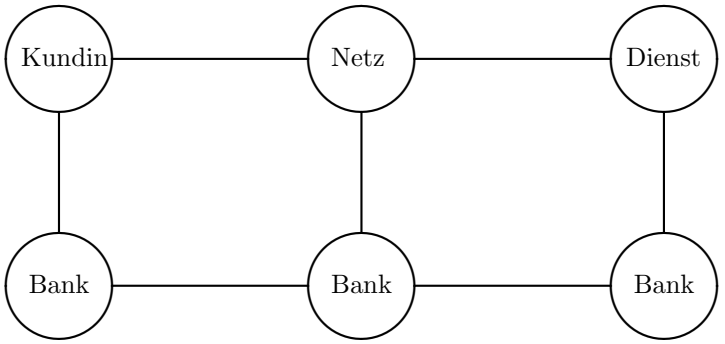
Trotz dieser völlig dezentralisierten Verwaltung der Chipkarte können die Organisationen vor Mißbrauch sicher sein. Dazu dient vor allem ein Zertifikat von einer entsprechenden Behörde, eine Art elektronische Geburtsurkunde. Diese wird pro Person nach einwandfreier Identifizierung nur einmal vergeben. Da diese Behörde keine weiteren Funktionen hat und auf keine weiteren Daten zugreifen kann (insbesondere kennt sie die Pseudonyme eines Individuums nicht), macht es nichts, wenn sie eindeutige Merkmale wie etwa den genetischen Code speichert.

Braucht der Besitzer der Karte eine Bescheinigung von einer Organisation, so erzeugt er sich für diese Organisation ein Pseudonym und erhält unter diesem die signierte Bescheinigung. Und genau diese wird bei Vorlage der Chipkarte und Angabe des entsprechenden Pseudonyms auch nur enthüllt. Muß die Bescheinigung ganz oder teilweise einer anderen Organisation vorgelegt werden, so hat der Besitzer, und nur er, die Möglichkeit, die entsprechende Information auf sein zugehöriges anderes Pseudonym zu übertragen.

3.6 Anonymität bei elektronischen Diensten

Eine weitere Stufe der Anonymität ist die völlig anonyme Geschäftsbeziehung. Stellen wir uns einen Dienstanbieter, ein öffentliches Netz und eine Kundin vor; sowohl Anbieter als auch Netzbetreiber und Kundin haben je eine Bank. Diese Konstellation ist in Abbildung V-12 dargestellt. Alle Benutzer, nämlich Anbieter und Kundin und deren Banken sollen gegenüber dem Netzbetreiber anonym bleiben. Ferner soll die Kundin gegenüber dem Dienstanbieter und den Banken anonym bleiben.

Abbildung V-12: Elektronisches Dienstangebot



Dazu schlagen Pfitzmann und Waidner ein Protokoll vor. Man braucht dafür elektronische Pseudonyme, insbesondere Nummernkonten, und elektronisches Geld oder anonyme Schecks. Der Einfachheit halber werden die drei Banken als identisch angenommen. Die Bank B hat dann ein Schlüsselpaar (E_B, D_B), dessen öffentlichen Teil E_B sie allen ihren Kundinnen bekannt gibt (sozusagen die Bankleitzahl). Der Dienstanbieter C hat ein Schlüsselpaar (E_C, D_C), dessen öffentlichen Teil E_C die Bank und alle potentiellen Kundinnen kennen. Der Bank gegenüber dient E_C als Kontonummer im Sinne eines Pseudonyms. Ebenso hat die Kundin A ein Schlüsselpaar (E_A, D_A), dessen öffentlicher Teil als Kontonummer bei der Bank dient; für die anonyme Dienstanforderung verwendet sie einmalig ein weiteres Schlüsselpaar (E'_A, D'_A).

Die Kundin A bestellt nun einen Dienst bei C, indem sie eine Botschaft m schickt; man kann sich darunter etwa eine Datenbankabfrage vorstellen. Gleichzeitig schickt sie einen Scheck vom Betrag b zur Bezahlung mit. Als Antwort sendet C eine Botschaft a und einen Rechnungsbetrag r, der auch an die Bank zur Verrechnung mitgeteilt wird. Die Bank ist in die Kommunikationsbeziehung zwischengeschaltet, garantiert die Deckung des Schecks und sorgt für die korrekte Verbuchung. Dieses Verfahren wird jetzt im Detail beschrieben. Es ist schon

ziemlich kompliziert; da es aber weitgehend selbständig in Chips abgewickelt wird, ist es trotzdem schnell genug.

1. Die Kundin A
   (a) verschlüsselt die Bestellung m zusammen mit dem Antwortschlüssel E'_A zu
   $$m_1 = E_C(m, E'_A),$$
   (b) signiert die Bestellung, die Kontonummer (Pseudonym) des Anbieters und den Scheck,
   $$m_2 = D_A(m_1, E_C, b),$$
   (c) verschlüsselt dies zusammen mit ihrer Kontonummer,
   $$m_3 = E_B(m_2, E_A),$$
   (d) und sendet m_3 an die Bank.
2. Die Bank B empfängt die Nachricht m_3, entschlüsselt sie und kennt dann den Absender (unter seinem Pseudonym E_A) und die signierte Bestellung m_2. Daher kennt sie auch die verschlüsselte Bestellung m_1, die Bestelladresse E_C und den Betrag b des Schecks.
3. Die Bank B
   (a) signiert die Bestellung m_1 und eine Deckungszusage, bildet also
   $$m_4 = D_B(m_1, b),$$
   (b) verschlüsselt dies zusammen mit ihrer Absenderangabe,
   $$m_5 = E_C(m_4, E_B),$$
   (c) und sendet m_5 an den Dienstanbieter C.
4. Der Anbieter C empfängt die Nachricht m_5, entschlüsselt sie und kennt dann den Absender B (durch E_B) und die signierte Bestellung m_4. Er erkennt m_1 und die Deckungszusage für b als authentisch, entschlüsselt dann m_1 und kennt somit die Bestellung m im Klartext und das Einmal-Pseudonym E'_A als Rücksendeadresse.
5. Der Anbieter C
   (a) verschlüsselt die Antwort a zu
   $$a_1 = E'_A(a).$$

(b) signiert a_1 und den Rechnungsbetrag r,

$$a_2 = D_C(a_1, r),$$

(c) verschlüsselt dies zusammen mit seiner Kontonummer,

$$a_3 = E_B(a_2, E_C),$$

(d) und sendet a_3 an die Bank B,

natürlich nur, falls der Betrag b für die Bezahlung des Dienstes ausreicht.

6. Die Bank B empfängt a_3, liest a_2 und E_C, erkennt a_1 und r als authentisch und bucht r vom Konto E_A auf das Konto E_C um.

7. Die Bank B

(a) signiert die Antwort a_1 und den Rechnungsbetrag r,

$$a_4 = D_B(a_1, r),$$

(b) verschlüsselt dies zusammen mit ihrer Absenderangabe,

$$a_5 = E_A(a_4, E_B),$$

(c) und sendet a_5 an die Kundin A.

8. Die Kundin A empfängt a_5, liest a_4 und E_B, erkennt a_1 und r als authentisch und entschlüsselt schließlich die eigentliche Antwort $a = D'_A(a_1)$.

Diese vereinfachte Version enthält noch einige Probleme, die sich aber durch Verkomplizierung des Protokolls lösen lassen:

- Richtige Zuordnung von Fragen und Antworten bei lebhaftem Geschäftsverkehr über die Bank – dies wird durch die Verwendung von Kennummern gelöst.
- Kontoauszüge.
- Zuverlässigkeit der Bank.
- Reklamationen, Stornieren von Bestellungen usw.

Ein Spezialfall des Protokolls ist der Nachrichtenaustausch zwischen A und C unter Pseudonym ohne Wertetransfer über eine Vermittlungs- oder Sammelstelle B (Scheck und Rechnung entfallen).

4 Ausblick

In den vorangegangenen Abschnitten wurde gezeigt, welche vielfältigen Anwendungsmöglichkeiten Verschlüsselungsverfahren und kryptographische Protokolle für Datenschutz und Datensicherheit bieten. Von allen solchen Maßnahmen soll der Endanwender möglichst wenig merken. Daher müssen kryptographische Protokolle, sollen sie routinemäßig angewendet werden, in Hardware, Betriebssystemen und Netz-Einrichtungen verankert sein. Im Alleingang kann man so etwas schon gar nicht implementieren.

Kryptographische Protokolle lassen sich mit symmetrischen und asymmetrischen Verschlüsselungsverfahren verwirklichen, mit asymmetrischen allerdings stets leichter. Die Langsamkeit der asymmetrischen Verfahren stört dabei kaum, da die Protokollsequenzen meist nur aus kurzen Einzelnachrichten bestehen. Umfangreiche Systeme, wie die im Abschnitt über Anonymität vorgestellten, überschreiten aber wohl noch die Grenzen des derzeit technisch Machbaren – abgesehen davon, daß sie eines breiten gesellschaftlichen und politischen Konsenses bedürften, um überhaupt eingeführt zu werden. Es ist zu fürchten, daß eine technikfeindliche Diskussion gerade solche Maßnahmen verhindert: Wer will schon ständig mit einem „kryptographischen Protokoll-Umsetzer“ herumlaufen. Die wirkliche Gefahr dagegen, die zunehmende Verdatung, braut sich dagegen, kaum bemerkt und weitgehend unkontrolliert, im Hintergrund zusammen. Durch die Explosion der technischen Entwicklung werden Fakten geschaffen, die später kaum noch zu revidieren sind.

Die kryptologische Grundlagenforschung schreitet in den letzten Jahren mit imposanter Geschwindigkeit voran. Den besten Eindruck vom aktuellen Stand bekommt man durch die jährlichen Konferenzen ‘CRYPTO’ (in den USA), ‘EUROCRYPT’ (in Europa) und ‘AUSCRYPT’ (in Australien), deren Beiträge unter dem Namen ‘Advances in Cryptology’ vom Springer-Verlag in der Reihe ‘Lecture Notes in Computer Science’ veröffentlicht werden; siehe etwa [26][96].

Für den Betreiber eines Datenverarbeitungssystems ergeben sich aus der Diskussion der kryptographischen Methoden die Leitlinien:

- Soweit möglich sollte man ein geschlossenes System anstreben, insbesondere für physischen Schutz sorgen.
- Daten lassen sich in offenen Systemen wirksam schützen, wenn, aber auch nur wenn, man kryptographische Methoden anwendet.
- Verschlüsselung ohne Hardware-Unterstützung ist in der Regel zu langsam; es gibt aber auf dem Markt brauchbare Sicherheitssysteme, die auf Verschlüsselungs-Chips basieren.
- Kryptographische Protokolle sind nicht vom Endanwender in Eigenregie einführbar; die Hersteller offener Systeme sind in der Pflicht, wirksame Implementierungen anzubieten.

- Für alle Arten von sensitiven Systemen sind geeignete technische Standards wünschenswert, die man den Herstellern gegenüber durchsetzen kann.

Anhang A

Checklisten zum Datenschutz

1 Anforderungen und Bestandsaufnahme

1.1 Checkliste Anforderungsdefinition

Leistungsanforderungen an die Datenverarbeitung

- Welche Anwendungen werden eingesetzt oder sollen eingesetzt werden?
- Was wird von der Arbeitsgeschwindigkeit gefordert oder erwartet?
- Welcher Komfort soll den Benutzern geboten werden? Welche Einschränkungen müssen sie hinnehmen?
- Zu welchen Zwecken werden Daten verarbeitet? (Verwaltung, geschäftliche Transaktionen, Forschung, ...)

Gesetzliche Schutzanforderungen

- Welche Gesetze enthalten für den Betrieb relevante Regelungen?
- Welche vertraglichen Regelungen haben Einfluß auf den Datenschutz? (Datenverarbeitung für andere Firmen, eingesetzte kommerzielle Software, ...)
- Welche Auskunftspflichten bestehen?
- Welche Aufbewahrungspflichten bestehen?

Betriebliche Schutzanforderungen

- Welche Datenschutzregelungen liegen im Interesse des Betriebs?
- Welche betrieblichen Sicherheitsnormen gibt es?
- Welche materiellen Schäden könnten bei Verstößen entstehen?
- Welche Folgen haben Datenverluste?
- Wo ist das Prinzip der Verhältnismäßigkeit zu beachten? Kosten-Nutzen-Abwägungen?

1.2 Checkliste Bestandsaufnahme

Datenverarbeitungskonzept

- Gibt es ein Datenmodell des Betriebs? (Siehe auch Abschnitt „Daten und Ressourcen“)
- Wo und wie sind die Benutzer spezifiziert? (Siehe auch Abschnitt „Benutzer“)
- Wo und wie ist das Datenverarbeitungssystem spezifiziert? (Siehe auch die Abschnitte „Systemkonfiguration“ und „Kommunikation“)
- Welches sind die Perspektiven für die Weiterentwicklung des Datenverarbeitungssystems?

Systemkonfiguration

- Gibt es Zentral- oder Abteilungssysteme? Mit welcher Peripherie sind sie ausgestattet?
- Wie sieht eine Skizze der Systemkonfiguration aus?
- Wird mit Arbeitsplatzrechnern (PCs, Workstations) gearbeitet?
- Wie sind diese ausgestattet? (Liste!)
- Existieren lokale Netze?
- Wie sieht eine Skizze der Netzkonfiguration aus?
- Existieren Anschlüsse an öffentliche Netze? Welche? Wo?

Daten und Ressourcen

- Welche personenbezogenen Daten werden verarbeitet? Welche Anonymisierungsmaßnahmen sind möglich?
- Welche Betriebsgeheimnisse oder schutzbedürftigen betriebsinternen Daten werden verarbeitet? (Dokumente, Pläne, Rechnungen, eigene Software-Entwicklungen, ...)
- Welche Systemdaten sind schutzbedürftig? Welche Systemprogramme sind sicherheitskritisch?
- Welche personenbezogenen Daten fallen bei der Systemüberwachung an? Mitbestimmung durch Betriebs- oder Personalrat?
- Welche Gefahren drohen den Daten? Wo ist die Ausspähung schon kritisch? Wo nur die Änderung?
- Welche fremden Daten werden (im Auftrag) verarbeitet?
- Welche „privaten“ Daten von Mitarbeitern werden im System gespeichert? (Z.B. wissenschaftliche Korrespondenz, Gutachen, Entwürfe)
- Was darf mit den Daten gemacht werden und was nicht?

Kommunikation

- Welcher betriebsinterne Kommunikationsbedarf soll durch Vernetzung befriedigt werden?
- Welcher externe Kommunikationsbedarf soll durch Fernanschlüsse befriedigt werden?

Benutzer

- Wer darf mit welchem System arbeiten?
- Welche Benutzergruppen lassen sich abgrenzen?

Schwachstellenanalyse, Angriffspunkte

- Welche Sicherheitsverstöße sind in der Vergangenheit vorgekommen (Viren oder ähnliches, Hacker, Wirtschaftsspione, eigene Mitarbeiter)? Wie wird ihre Wiederholung verhindert?
- Liegt eine Risikoanalyse für die eingesetzten Verfahren vor?

- Welche potentiellen Angreifer gibt es? Welche Motive könnten sie haben? Welchen Nutzen könnten sie aus Sicherheitsverstößen ziehen? Welchen Aufwand nehmen sie mutmaßlich in Kauf? Welche System- und Schwachstellenkenntnisse muß man ihnen unterstellen?
- Welche Gefahren drohen durch gezielte Ausspähung? Durch Fischzüge?
- Lohnt sich der Einsatz eines Tiger-Teams? Oder kann ein Mitarbeiter des Betriebs kompetent Penetrationstests durchführen?

2 Organisation

2.1 Checkliste Planung von Maßnahmen

- Wie ist die Datensicherheit in das allgemeine Datenverarbeitungskonzept eingebunden?
- Welches Sicherheitsniveau wird angestrebt? Welches Restrisiko wird in Kauf genommen?
- Wie sieht die Berechtigungsmatrix aus? (Benutzer $\times$ Ressourcen, Zuordnung zwischen Benutzergruppen und Anwendungen)
- Wer ist zuständig für Sicherheitsmaßnahmen auf Hardware-Ebene? Wer hat Sachverstand?
- Wer ist zuständig für Sicherheitsmaßnahmen auf Software-Ebene? Wer hat die nötigen Systemkenntnisse?
- Wer plant und definiert organisatorische Maßnahmen?
- Wer ist für die Katastrophenvorsorge zuständig?
- Welche Risiken lassen sich durch Spezialversicherungen abdecken?

2.2 Checkliste Personal

- Wie ist das Personal organisiert? Wer hat welche Funktion? Stellenbeschreibungen? Dienstverträge?
- Wie werden Zuverlässigkeit und Kompetenz geprüft?
- Wie sind die Benutzergruppen organisiert? Wer leitet sie?
- Wie groß ist die Gefahr der Abwerbung durch Konkurrenzunternehmen? Welche Vorkehrungen sind zu treffen?
- Wie wird das Betriebsklima gepflegt? Einstellungs- und Entlassungspolitik? Sind Racheakte zu befürchten?

- Welche Interessenkonflikte bestehen zwischen Management und Personal? Zwischen verschiedenen Mitarbeitergruppen?
- Welche Akzeptanzprobleme oder inneren Widerstände bestehen?
- Wie wird das Personal motiviert? (Vorbildwirkung, Ernstnehmen von Vorschriften, Arbeitserleichterung, Unterstützung, Respekt vor Persönlichkeitsrechten, Schulung, Vertrauen, Kommunikationsverhalten des Managements)
- Wird die Ergonomie genügend berücksichtigt?
- Welche Dienstvorschriften und Arbeitsanweisungen gibt es? Welche sollte es noch geben?
- Wie sind und werden Zuständigkeiten geregelt?
- Wer bedient welche Geräte?
- Wer sorgt für Ordnung im Geräteraum, im Bandarchiv, ...?
- Wie ist der Reinigungsdienst organisiert? Einsatz von Fremdfirmen?
- Wer hat welche Zugangsberechtigungen? Welche Schlüssel?
- Wer ist für die Dokumentation der Datenschutzmaßnahmen zuständig? Wer verfaßt den Datenschutzbericht?
- Wer hält sich über aktuelle Sicherheitsprobleme auf dem laufenden?

2.3 Checkliste Überwachung

- Welche Überwachungssysteme werden eingesetzt?
- Wer ist für die Revision zuständig? Wird diese Funktion genügend objektiv ausgeübt?
- Wo ist das Vier- (oder Mehr) -Augenprinzip einzuführen?
- Wie werden Mitarbeiter von Fremdfirmen überwacht?
- Wie weit greifen Überwachungsmaßnahmen in Persönlichkeitsrechte ein?

2.4 Checkliste Benutzerkontrolle

- Wer darf mit dem System arbeiten?
- Wer darf bestimmte Informationen lesen oder verändern?
- Warum muß eine bestimmte Operation ausgeführt werden?
- Wann darf eine bestimmte Operation ausgeführt werden?
- Wo darf eine bestimmte Operation ausgeführt werden?
- Wer darf einen Auftrag zu einer bestimmten Operation geben?
- Wer verbirgt sich hinter einer Benutzerberechtigung?
  - Eine eindeutige Person,
  - ein Stellvertreter („der Sekretär im Auftrag der Chefin“),
  - ein Funktionsträger („der diensthabende Operator“),
  - eine Rolle („Materialausgabe“)?

2.5 Checkliste Auftragskontrolle

- Welche Datenverarbeitungsaufträge werden ausgeführt oder sollen ausgeführt werden? Für welche Firmen oder Institutionen?
- Welche vertraglichen Regelungen gelten für die Aufträge?
- Welche besonderen Datenschutzmaßnahmen sind für die Aufträge nötig?
- Wer darf Aufträge annehmen?
- Wie werden Aufträge ausgeführt?
- Wie gelangen die zugehörigen Fremddaten ins System?

3 Datensicherung und Katastrophenschutz

3.1 Checkliste Katastrophenvorsorge

- Wie werden folgende Risikofaktoren berücksichtigt:
  - Feuer, Sturm, Erdbeben,
  - Wasser (Brauchwasser, Regenwasser, Hochwasser, Löschwasser),
  - Schmutz,
  - Störungen der Infrastruktur (Stromausfall, Klimaanlage),

  - Bedienungsfehler, menschliches Versagen,
  - Hardware- und Softwarefehler,
  - Sabotage, Zerstörung, Vandalismus,
  - Kriminalität, Mißbrauch,
  - Einbruch, Diebstahl?
- Welche Gefahren lauern in der Umgebung der Gebäude?
- Wie sieht Ausstattung und Umfeld der Räume aus?
  - Feuersichere Baumaterialien,
  - Brandschutztüren,
  - feuerhemmende Datentresore,
  - Schutz vor Wasserschäden, etwa Rohrbrüchen in höheren Stockwerken,
  - Sicherheit vor Hochwasser und anderen Naturkatastrophen,
  - Meldesysteme für Rauch, Feuer, Wasser,
  - Sprinkler und andere Feuerlöscheinrichtungen,
  - Notausschalter?
- Welche Brandschutzmaßnahmen sind eingeführt oder einzuführen?
  - Sichere Lagerung brennbarer Stoffe (auch Druckpapier und Datenträger),
  - Rauchverbote,
  - Schutz vor Kabelbränden?
- Wie steht es mit der Ausfallsicherheit der Geräte und der Notstromversorgung?
- Welche Richtlinien für Notfälle existieren?
- Wer ist in Notfällen zuständig? Krisenstab?
- Wer ist in Notfällen für Notmaßnahmen kompetent?

3.2 Checkliste Datensicherung

- Wie soll nach einem Totalausfall ein lauffähiges Betriebssystem wiederhergestellt werden?
- Wie sollen nach einem Totalausfall alle Daten restauriert werden?
- Wie lange sollen gesicherte Daten aufbewahrt werden?

- Wie oft sollen die Daten gesichert werden?
- Wie schnell soll der Zugriff auf gesicherte Daten sein?
- Wann sind vollständige Sicherungen durchzuführen? Wann reichen inkrementelle Sicherungen?
- Welche Daten brauchen überhaupt nicht gesichert zu werden?
- Wo werden die gesicherten Daten aufbewahrt?
- Gibt es eine Möglichkeit, Zwillingskopien der gesicherten Daten in einem anderen Gebäude aufzubewahren? (Mit möglichst unterschiedlichem Gefährdungsprofil bei Katastrophen)
- Wer führt die Datensicherung durch?

4 Physischer Schutz

4.1 Checkliste Baupläne

- Gibt es Baupläne des Gebäudes?
- Wo liegen die Zugänge?
- Gibt es ungesicherte Zugänge? Fenster? Schächte?
- Wo gibt es Doppelböden oder abgehängte Decken? Wie sieht es darüber bzw. darunter aus?
- Gibt es eine Skizze der Kabelwege?
- Wo befinden sich Verteilerschränke und Anschlußpunkte (auch momentan unbenutzte)?
- Wo sind Kabelschächte? Welche Kabel verlaufen in ihnen?
- Wo besteht aktive Brandgefahr? Mögliche Brandursachen?
- Wo besteht passive Brandgefahr?
- Wo liegen elektrische Leitungen?
- Wo liegen Wasserleitungen? Gasleitungen? Sonstige Versorgungsleitungen?

4.2 Checkliste Zugangskontrolle

- Wie sind Gelände und Gebäude geschützt?
- Welche Sicherheitsbereiche gibt es?
  - Maschinenraum?
  - Stromversorgungs-, Hausanschlußraum?
  - Klima-Anlagen-Raum?
  - Datenarchiv?
  - Operatorräume?
  - Räume der Systemabteilungen?
  - Räume für Benutzer und Benutzergruppen?
- Wie ist der Zugang zu den Sicherheitsbereichen geregelt?
  - Schließanlagen und Schleusen für Sicherheitsbereiche?
  - Türsicherung mit Schlüsselregelung oder Zugangskontrollsystem?
  - Personalschleusen mit Ausweis- oder Gesichtskontrolle?
  - Schalter mit Sicherheitsglas, Durchreiche und Gegensprechanlage zur Datenträgerausgabe?
  - Nebeneingänge?
- Welche Maßnahmen zur Objektsicherung sind nötig?
  - Videoüberwachung?
  - Sicherung durch Alarmanlage, besonders außerhalb der Dienstzeit?
  - einbruchsicheres Glas in den Fenstern der Sicherheitsbereiche?
  - Stahltüren zu den Maschinenräumen?
  - Sicherung von Zugangsmöglichkeiten zu Kellerräumen und benachbarten Geschossen?
  - Zugangssicherung zu Mitarbeiterräumen?
- Wer hat Zugang zu den Sicherheitsbereichen?
- Können Zugangssperren in Notfällen von autorisierten Personen abgeschaltet werden?
- Wie ist der Zugang für Betriebsfremde geregelt? (Besucher, Wartungspersonal, Handwerker, Fremdfirmen)
- Wird über den Zugang zu den Sicherheitsbereichen Buch geführt?
- Wo ist das Vieraugenprinzip nötig?
- Wie läßt sich nachträglich ermitteln, wer wann Zugang hatte?
- Wie kann man diese Aufzeichnung umgehen oder fälschen?

4.3 Checkliste Datenträgerkontrolle

- Welche Datenträger werden verwendet?
  - Disketten?
  - Festplatten?
  - Datenbänder?
  - Datenkassetten?
  - Papier?
  - Sonstige?
- Wo werden Datenträger aufbewahrt? Wie ist der Zugang zu Datenträgerarchiven geschützt? Sicherheitsbereiche?
- Wer ist für das Wegschließen der Disketten verantwortlich?
- Wer kann die Datenträger lesen? Wer darf das? Welche Ausrüstung braucht man dazu?
- Wie können die Datenträger kopiert werden? Wer kann das? Wer darf das?
- Bleiben unerwünschte Datenreste auf Datenträgern stehen? Wie werden sie gelöscht oder geschützt?
- Wer ist für die Aufbewahrung und Ausgabe von Datenträgern verantwortlich? Klare Definition der Befugnis zur Datenträgerverwaltung?
- Wer ist für die Bestandskontrolle der Datenträger verantwortlich?
- Wie ist die Abgangskontrolle für Datenträger geregelt?
  - Ausgabe von Datenträgern nur an befugte Personen?
  - Kontrollierte Löschung oder Vernichtung von Datenträgern?
  - Abgabemöglichkeit für zu vernichtende Druckerlisten, Reißwolf?
- Wie wird der Transport von Datenträgern kontrolliert?
  - Verpackungs- und Versandvorschriften, zum Beispiel Verwendung verschlossener Transportkoffer?
  - Transport nur durch befugte Personen?
  - Nutzung eines gesicherten Eingangs und von Schaltern und Schleusen für An- und Ablieferung?
  - Verschlüsselungsvorschriften?

5 Hardware und Betriebssystem

5.1 Checkliste Hardware

- Welche Aussagen macht der Hersteller zu Sicherheitsfragen?
- Wie wird der Hauptspeicher geschützt? Grenzregister? Speicherschutzschlüssel? Virtuelle Adressierung?
- Welche Zustände kennt die CPU? Wie werden die Übergänge kontrolliert? Wie gelangt man in einen privilegierten Zustand?
- Wie sind Ein- und Ausgabemedien geschützt?
  - Schreibschutz auf Bändern und Disketten?
  - Sperre von Diskettenlaufwerken?
  - Schutz von Festplattenlaufwerken?
  - Tastatursperre und Bildschirmverdunklung bei inaktiven Sitzungen? Timeout oder absichtliche Aktivierung der Sperre?
- Welche spezielle Sicherheitshardware auf dem Markt paßt ins System?
  - Zugangskontrollsysteme mit Ausweislesern?
  - Separate Rechner oder Prozessoren?
  - Verschlüsselungs-Chips?

5.2 Checkliste Betriebssystem

- Welche Aussagen macht der Hersteller zu Sicherheitsfragen?
- In welcher Programmiersprache ist das Betriebssystem erstellt?
- Wie vollständig ist das Betriebssystem dokumentiert?
- Gibt es offizielle Zertifikate über die Sicherheit?
- Wie werden die Benutzerbereiche beim Mehrbenutzerbetrieb getrennt?
  - Überwachung des Ressourcenverbrauchs?
  - Erzeugung von deutlich sichtbaren Drucker-Trennseiten, um Fehlleitung von Ausdrucken zu verhindern?
  - Überwachung der Stapelverarbeitung (‘Batch jobs’, ‘Remote Job Entry)?
  - Automatische Löschung von Plattenbereichen und Bändern vor einem Besitzwechsel?

  - Schutz von Datenbändern vor falscher Zuordnung durch interne Markierungen ('Labels') und deren Überprüfung?
  - Ein System zur Datensicherung ('Backup'), das Zuordnungsfehler beim Restaurieren verhindert?
  - Automatische Löschung von temporären Daten, die bei verschiedenen Arbeitsvorgängen erstellt werden, zum Beispiel in „Spool-Bereichen“ (Warteschlangen vor Ausgabegeräten), von Benutzern angelegten temporären Plattendateien, Hauptspeicherbereichen, auch beim 'Paging' auf Platte ausgelagerten Hauptspeicherbereichen?
  - Werden Angriffsversuche auf Daten sowohl dem Systemverwalter als auch dem Besitzer gemeldet?

- Wie funktioniert die Kommunikation zwischen verschiedenen Prozessen?
- Wie werden Serviceprozesse erzeugt und gestartet?
- Welche Befugnisse hat der Systemverwalter ('Super User')?
- Welche Möglichkeiten hat der Systemverwalter ('Super User')? Wie wird er überwacht?
- Welche Befugnisse und Möglichkeiten hat der Operator? Wie wird Bedienungsfehlern vorgebeugt? Ergonomie?
- Welche Sicherheitslücken im Betriebssystem sind bekannt? (Hersteller, Anbieter von Zusatzsoftware, Benutzerkonferenzen fragen, Literatur durchforsten)
- Welche Möglichkeiten gibt es, Schutzmaßnahmen zu umgehen, etwa durch Laden einer anderen Version des Betriebssystems?
- Wie wird die Berechtigungsmatrix vom Betriebssystem unterstützt?
- Wie wird die Paßwortpolitik (siehe Checkliste Identifikation und Paßwörter) vom Betriebssystem unterstützt? Braucht man dafür Zusatzsoftware? Eigene Modifikationen?
- Welche Sicherheitsvorkehrungen gibt es bei besonderen Betriebszuständen? (Wartung, Notfälle, Systemabstürze)
  - Zugangs- und Zugriffssperren bis zum vollständigen Wiederanlauf?
  - Selbstprüfungsmechanismen beim Wiederanlauf?
  - Kontrolle von Dumps?
  - Wartungspersonal unter Aufsicht?

5.3 Checkliste Identifikation und Paßwörter

- Welcher Zugangsschutz ist vorgesehen?
  - Paßwörter?
  - Erkennungsdialog? Abhörsicher?
  - Magnetkarten?
  - Chipkarten?
  - „Harte Schlüssel“?
  - Prüfung persönlicher Merkmale (Fingerabdruck, Netzhautbild)?
  - Bindung von Personen an bestimmte Terminals oder Adressen? Mit zusätzlichem physischen Zugangsschutz?
- Bleiben Paßwörter bei Eingabe automatisch unsichtbar?
- Kann ein Benutzer sein Paßwort selbst wählen und jederzeit ändern?
- Kann der Systemverwalter jedes Paßwort in einem Notfall ändern (z.B. im Falle des Vergessens)? Wird eine solche Änderung manipulationssicher dokumentiert? Kann er jeden Benutzer zu einer Änderung zwingen?
- Ist das Paßwortverzeichnis lesegeschützt? Einweg-verschlüsselt? Werden Lesezugriffe protokolliert?
- Welche Maßnahmen sind bei Eingabe eines falschen Paßworts vorgesehen?
  - Alarm an zentraler Stelle?
  - Aufzeichnung im Sicherheitsprotokoll?
  - Zeitsperre?
  - Meldung an den betroffenen Benutzer bei der nächsten korrekten Anmeldung?
  - Stillegung des Anschlusses und der Benutzer-Identität nach einigen Versuchen?
- Wird bei Verwendung von variablen logischen Adressen oder Wählanschlüssen nach Paßwortverstößen wenigstens die Benutzer-Identität automatisch gesperrt? Gibt es eine Zeitsperre der Leitung? Mit wachsenden Sperrintervallen?
- Welche Vorschriften zur Wahl von Paßwörtern gibt es?
  - Länge des Paßworts?
  - Negativliste von Paßwörtern, die zu einfach sind?
  - Umkehrungen oder Wiederholungen naheliegender Wörter?

  - Verfallsdatum?

  Werden diese vom Betriebssystem automatisch geprüft?

- Müssen sich die Benutzer viele komplizierte Paßwörter gleichzeitig merken?
- Besteht die Notwendigkeit, irgendwelche Paßwörter in einer Benutzergruppe gemeinsam zu verwenden?
- Welche Vorkehrungen gibt es gegen eine Paßwortfalle?
- Welche Möglichkeiten hat ein Benutzer nach Erraten eines privilegierten Paßworts? Welcher zusätzliche Schutz besteht?

5.4 Checkliste Sicherheitsprotokolle

- Wie werden die Anzeigen der Systemkonsole aufgezeichnet?
- Werden außergewöhnliche Betriebszustände entdeckt, gemeldet und aufgezeichnet?
- Welche Möglichkeiten gibt es, einzelne Benutzer gezielt zu überwachen?
- Wie werden Sicherheitsverstöße und privilegierte (sicherheitskritische) Operationen protokolliert?
  - An- und Abmeldevorgänge?
  - Dateizugriffe?
  - Änderungen von Systemparametern?
  - Änderungen von Sicherheitsdefinitionen?
- Wie wird der Betriebsmittelverbrauch protokolliert?
- Welche Prozesse erledigen die Aufzeichnungen? Wer kann sie beeinflussen? Welche Systemprivilegien haben oder brauchen sie?
- Was passiert bei einem Systemabsturz mit noch offenen Protokolldateien?
- Wer hat Zugang zu den Protokollen? Vieraugenprinzip?
- Werden Protokolle manipulationssicher ausgewertet?
- Welche Persönlichkeitsrechte der Mitarbeiter werden durch die Aufzeichnungen berührt? Mitbestimmung des Betriebsrats oder Personalrats?
- Wie lange werden die Aufzeichnungen aufgehoben? Wie werden sie gelöscht?

5.5 Checkliste Viren und andere Schadprogramme

- Welche Zugangsbeschränkungen verhindern das Einbringen unerwünschter Programme?
- Wird die eingeführte Software streng genug kontrolliert?
- Gibt es eine Quarantäne für Software unsicheren Ursprungs? Gibt es eine Möglichkeit, sie auf einem völlig isolierten System zu testen?
- Werden Originaldatenträger vor Installation mit Schreibschutz versehen, danach sicher verwahrt?
- Wird, wo immer möglich, mit Schreibschutz gearbeitet?
- Werden ungewöhnliche Ereignissen aufgezeichnet? Ungewöhnliche Aktivitäten im System sofort verfolgt?
- Sind geeignete Überwachungsprogramme vorhanden? Werden diese auch jeweils vor einer Datensicherung angewendet?
- Werden infizierte Programme sofort entfernt?
- Werden mehrere Generationen von gesicherten Daten aufbewahrt und dabei auch Boot-Sektoren und Systemtabellen nicht vergessen?

6 Anwendungsprogramme

6.1 Checkliste Zugriffsrechte

- Wie sieht die Berechtigungsmatrix aus? Welche Subjekte (Benutzer, Programme) dürfen auf welche Objekte (Programme, Daten) in welcher Weise zugreifen?
- Welche unterschiedlichen Zugriffsmöglichkeiten bietet das Betriebssystem?
- Gibt es einen ‘execute only’-Zugriff? Wie ist er abgesichert?
- Wie läßt sich die Zugriffsmatrix im System implementieren?
- Wo und wie werden Zugriffsrechte abgelegt? Wer hat auf diese Daten Zugriff?
- Erlöschen Zugriffsrechte automatisch, wenn ein Subjekt oder Objekt ausgelöscht wird?
- Werden Daten und Zugriffsrechte von einem Server verwaltet? Welche Privilegien hat dieser, wenn er im Auftrag eines Benutzers arbeitet?

- Sind Sicherheitsstufen eingeführt oder ist ihre Einführung sinnvoll?
- Lassen sich Zugriffsrechte beim Restaurieren von Daten aus der Datensicherung umgehen?

6.2 Checkliste Selbsterstellte Software

- Gibt es Programmierregeln für kritische Anwendungen?
- Welche Programmiersprachen und -werkzeuge werden verwendet? Welche besonderen Sicherheitslücken haben sie?
- Wie wird selbsterstellte Software getestet?
  - Formale Verfahrensprüfung?
  - Sachlogische Programmprüfung?
  - Testdaten?
  - Schnittstellenprüfung zwischen Programmteilen?
  - Spezielle Prüfprogramme?
- Wer gibt selbsterstellte Software zur Anwendung frei?

6.3 Checkliste Fremdsoftware

- Welche Fremdsoftware wird eingesetzt? Von welchen Herstellern oder Vertreibern?
- Wer entscheidet über Anschaffung und Einsatz von Fremdsoftware?
- Wer nimmt Anpassungen der Fremdprogramme vor? ('Customizing')
- Welche Möglichkeiten zur Meldung von Fehlern und Problemen bietet der Hersteller oder Vertreiber? ('Hot Line'?)
- Wie wird die Fremdsoftware gewartet? Wie schnell werden Fehler behoben?

6.4 Checkliste Anwendungskontrolle

- Sind die Verfahrensabläufe für kritische Anwendungen ausreichend dokumentiert?
- Gibt es Prüfregeln für kritische Anwendungen?

6.5 Checkliste Datenbanken

Siehe auch Checkliste Zugriffsrechte.

- Welche Daten werden in einer Datenbank gehalten?
- Wo ist die Datenbank lokalisiert? Großrechner mit virtuellem Server? Server als Station im Netz? ...?
- Welches Datenbanksystem wird eingesetzt?
- Welche eigenen Sicherheitsfunktionen bietet es?
- Welche Benutzer-Oberfläche bietet es? Wie sicher ist diese? Wie ausbruchsicher?
- Welche Anfragen sind erlaubt oder gesperrt? Welche Möglichkeiten zum Datenabgleich gibt es? Wie werden Tracker-Angriffe behindert?

6.6 Checkliste Benutzer-Oberfläche

- Bieten Betriebssystem oder Anwendungsprogramm eine ausbruchsichere Benutzer-Oberfläche?
- Lassen sich Benutzerprofile manipulationssicher implementieren?
- Wie werden Privilegien, Verbrauchsrechte, Kommunikationsmöglichkeiten und Zugang zu Anwendungsprogrammen gesichert?
- Welcher Schutz besteht nach Programmabstürzen oder Programmabbrüchen?
- Welche Auskünfte kann ein Benutzer über das System erfragen?

7 Personal-Computer

7.1 Checkliste Physische Sicherheit

- Wer schließt den Rechner ab? Wer hat Schlüssel?
- Wer schließt die Disketten weg?
- Wer schließt die Räume ab?
- Wie leicht ist ein Gerät wegzutragen?
- Wie leicht ist ein Gerät unbefugt zu öffnen?

7.2 Checkliste Anschlüsse

- An welche Netze ist der Rechner angeschlossen? Welche Fernanschlüsse hat er?
- Welche Kommunikationssoftware wird eingesetzt?
- Welche Programme erleichtern den Zugang zu den Anschlüssen? Enthalten sie Paßwörter im Klartext?

7.3 Checkliste Systemsicherheit

- Welche Sicherheits-Hardware und -Software wird eingesetzt? Was leistet sie?
- Gibt es einen Paßwortschutz für die Festplatte? Wie kann man ihn umgehen?
- Werden die Daten auf der Festplatte verschlüsselt? Nach welchem Verfahren? Wie werden die Schlüsselwörter behandelt? (Siehe auch Checkliste Identifikation und Paßwörter)
- Wann wird verschlüsselt? Laufend oder nach Arbeitsende? Automatisch oder auf Kommando?
- Wie sind Systemdaten und Sicherheitsfestlegungen geschützt? Welche Ausrüstung und welche Kompetenz braucht man, um sie zu manipulieren?
- Kann man von einem Diskettenlaufwerk booten? Welche Möglichkeiten hat man dann?
- Können Drucker, Diskettenlaufwerke und andere Peripheriegeräte gezielt für einzelne Benutzer gesperrt werden?
- Gibt es Zeitsperren (Timeout) bei Inaktivität? Tastatursperren und Abdunkeln des Bildschirms, auch absichtlich aktivierbar?

8 Netze

8.1 Checkliste Kabel

- Gibt es eine Skizze der Kabelwege?
- Wo befinden sich Verteilerschränke und Anschlußpunkte (auch momentan unbenutzte)?
- Wo sind Kabelschächte? Welche Kabel verlaufen in ihnen?

- Welche Kabeltypen werden verwendet?
- Wie sind die vorgesehenen Kabel abhörbar?
- Was müßte ein Angreifer unternehmen, um vorhandene Kabel anzuzapfen?
- Wie würde er dabei entdeckt?
- Welche baulichen Maßnahmen sind zum physischen Schutz der Kabel notwendig?
- Welche elektromagnetische Abschirmung ist zum Schutz der Kabel notwendig?

8.2 Checkliste Knotenpunkte

- Wie kann ein Angreifer Knotenpunkte abhören?
- Welche Geräte braucht er dazu? Was leisten Schnittstellentester?
- Wie kann ein Angreifer den Netzverkehr aktiv verfälschen?
- Wie gut sind die Installationsschränke physisch geschützt?

8.3 Checkliste Netzmanagement

- Welche Daten kann ein Netzverwalter sehen? Welche kann er manipulieren?
- Wie reagiert das Netz auf Unterbrechungen, etwa bei Anzapfversuchen?
- Wie reagiert das Netz auf das (eventuell unbefugte) Einfügen neuer Stationen?
- Welche Möglichkeiten bietet das Abhören von Netzmanagementdaten etwa beim 'download' von Konfigurationsdaten auf Bridges und ähnliche Komponenten?
- Durch welche Manipulationen in den unteren Protokollschichten sind Sicherheitsmaßnahmen der oberen Schichten zu unterlaufen?
- Treten beim Hochfahren des Netzes Sicherheitslücken, etwa in Form von undefinierten Zuständen, auf?
- Enthalten die Übertragungsprotokolle verdeckte Datenkanäle, die vom Netzmanagement nicht erkennbar sind?
- Was passiert mit einseitig hängenden Verbindungen (Absturz eines Kommunikationspartners)?

8.4 Checkliste Subnetze und Subsysteme

- Wie weit ist der Datenverkehr in den Subnetzen abgeschottet?
- Welche Eingriffe sind von einem Subnetz aus ins Backbone-Netz möglich?
- Wie weit sind Anschlußpunkte manipulierbar? Wie kann man freie „Datensteckdosen“ besetzen? Kann man besetzte Anschlußpunkte umstöpseln?
- Können Stationen in den Subnetzen ihre eigene Netzadresse manipulieren?
- Wie werden neue Stationen ins Netz eingebunden?
- Wie läßt sich der Anzapfbarkeit von Koax-Ethernet-Segmenten wirksam begegnen?
- Gibt es berücksichtigenswerte Unterschiede zwischen Token-Ring und Ethernet unter dem Aspekt der Sicherheit?
- Wie ist die Gefahr der elektromagnetischen Abstrahlung von Bildschirmen zu beurteilen?
- Wie läßt sich der Zugriffsschutz in den Subsystemen regeln? Kennt ein Datenserver die Netzadresse, von der eine Anfrage kommt? Kennt ein Host die Netzadresse, von der ein Logon-Versuch kommt?
- Welche Schutzmaßnahmen auf höheren Protokoll-Ebenen sind wünschenswert und machbar?
- Welche Dienste sollen die Subsysteme bieten?
  - Nachrichten (‘Message Transfer’)?
  - Post (‘electronic mail’)?
  - Fernverarbeitung (‘Remote Job Entry’)? Welche Prozesse lassen sich durch einfache Nachrichtenübermittlung anstoßen?
  - Dialog (‘Remote Login’, Terminalemulation)?
  - Verteilte Anwendungen?

8.5 Checkliste Daten im Netz

- Welche Kommunikationsbeziehungen umfassen zu schützende Daten?
- Sind kryptographische Maßnahmen (Verschlüsselung) unumgänglich?
- Sind Verbindungsdaten schützenswert? (Im lokalen Netz wohl nicht)

8.6 Checkliste Fernzugriffe

- Welche Gefahren entstehen durch die Anbindung an Fernverkehrsnetz (Datex-P, ISDN)?
- Werden Zugriffsmöglichkeiten auf Daten innerhalb des Betriebs eröffnet?
- Wer darf elektronische Post von außerhalb empfangen? Nach außerhalb senden?
- Sind Fernwartungsmaßnahmen vorgesehen?

8.7 Checkliste Normen und Standards

Sind Exemplare folgender Schriften vorhanden?

- IT-Sicherheitskriterien?
- IT-Evaluationshandbuch?
- IEEE 802.10?
- ANSI-SP3-Protokoll ('Secure Data Network System')?
- ISO 7498/2?

Wie weit sind die Standards erfüllt?

Anhang B

Sicherheitsprodukte für den PC-Bereich

Produkt	Vertrieb	Preis ca.	Funktionen
Clavis	IBD Frankfurt	680 DM	Verschlüsselung, Benutzerverwaltung
Close Access	Datasoft Eschborn	750 DM	Verschlüsselung, Benutzerverwaltung, Systemüberwachung
Crypton	pc-plus München	430 DM	Online-Chiffrierung, Zugriffsschutz
Elkey	Infosys Bodenheim	1400 DM	Verschlüsselung, Benutzerverwaltung, Systemüberwachung
Chiffriermodem Hetrocrypt	Hetron Gräfelfing	3300 DM	Verschlüsselung für öffentliche Netze
Hetrolock	Hetron Gräfelfing	560 DM	PC-Verriegelung
Oculis Plus	IBD Frankfurt	900 DM	Verschlüsselung, Benutzerverwaltung, Systemüberwachung
Orgasafe	Orgasafe München	1020 DM	Verschlüsselung, Benutzerverwaltung, Zugriffsschutz

PC+		modular kombinierbar	je nach Zusammenstellung
PCSS	Görlitz Koblenz	1100 DM	Zugangskontrolle, Zugriffskontrolle
PC-Vault	Computer Solutions München	340 DM	Zugangskontrolle, Zugriffskontrolle
Proficode	IM Software Leonberg	320 DM	Verschlüsselung, Zugriffskontrolle
Protect	Mema Computer	280 DM	Verschlüsselung
Safeboard mit Safeguard	Utimaco Frankfurt	1000 DM	Verschlüsselung, Zugangskontrolle
Safeguard	Utimaco Frankfurt	340 DM	Zugangskontrolle, Zugriffskontrolle
Savedir	Andreas Müller Berlin	200 DM	Verschlüsselung, Benutzerverwaltung
Sentinelpro	Kontron Eching	140 DM	Kopierschutzstecker
Signum	Mbp Dortmund		Chipkartensystem, Verschlüsselung, Signatur (RSA)
Spirotect-M	Siemens München		Chipkartensystem, Zugangskontrolle, Zugriffskontrolle
Superkey	Borland München	400 DM	Verschlüsselung
Turboclean	Andreas Müller Berlin	100 DM	Physikalisches Löschen von Dateien
Ultracrypt	ultraware München	150 DM	Verschlüsselung

Anhang C

Zwei kleine Sicherheitshilfen

Da die Hersteller des Betriebssystems MS-DOS offenbar nicht gewillt sind, die Probleme des physikalischen Löschens von Dateien und des Datenmülls hinter den Dateienden zu beheben, enthält dieser Anhang zwei kleine Programme in Turbo-Pascal, die helfen, diese Probleme wenigstens nachträglich zu korrigieren, und zeigen, wie einfach das im Prinzip ist.

1 Hilfsprozeduren

In diesem Abschnitt werden Hilfsprozeduren für die beiden Programme zusammengefaßt. Man kann sie je nach Laune in die jeweiligen Programme mit aufnehmen oder zusammengefaßt als 'Unit' stehen lassen.

1.1 Verwendung von Standard-Units

Aus der Turbo-Pascal-Standard-Unit 'Crt' werden die Prozeduren 'ClrScr', 'KeyPressed', 'ReadKey' und 'GotoXY' verwendet. Aus der Unit 'Dos' werden verwendet die Datentypen 'Registers', 'DirStr' und 'SearchRec', die Konstante 'ReadOnly', die globale Variable 'DosError' und die Prozedur 'DiskFree'.

1.2 Die Disk-Parameter

Der Datentyp **DiskParam** faßt für einen Datenträger die folgenden Parameter zusammen (alle vom Typ INTEGER):

- Anzahl der Sektoren pro Cluster,
- Anzahl der Bytes pro Sektor,

- Anzahl der Bytes pro Cluster,
- Anzahl der Cluster,
- Anzahl der freien Cluster.

```
TYPE DiskParam = RECORD
                   SektorenProCluster, BytesProSektor,
                   BytesProCluster, Clusterzahl,
                   freieCluster: INTEGER;
                   END;
```

1.3 Holen der Parameter

Die Prozedur **GetDiskParam** bestimmt die Parameter für einen Datenträger: Mit Hilfe der MS-DOS-Funktion $36 werden die Parameter Clusterzahl, Zahl der freien Cluster, Sektoren pro Cluster, Bytes pro Sektor bestimmt und daraus die Zahl der Bytes pro Cluster berechnet. Eingabe-Parameter ist der Buchstabe des Laufwerks.

```
PROCEDURE GetDiskParam(d: CHAR; VAR ParameterSet: DiskParam);
VAR Lw : BYTE;
    cpu: Registers;
BEGIN
  d:= UpCase(d);
  Lw:= ORD(d) - 64;
  cpu.AH:= $36;
  cpu.DL:= Lw;
  MsDos(cpu);
  ParameterSet.SektorenProCluster:= cpu.AX;
  ParameterSet.BytesProSektor    := cpu.CX;
  ParameterSet.Clusterzahl       := cpu.DX;
  ParameterSet.freieCluster      := cpu.BX;
  ParameterSet.BytesProCluster   := cpu.AX * cpu.CX;
END;
```

1.4 Korrekte Bezeichnung eines Verzeichnisses

Die eingegebene Zeichenkette *s* wird in einen vollständigen Verzeichnisnamen einschließlich Laufwerksangabe und abschließendem „\“ umgewandelt, und zwar in Großbuchstaben. Ist *s* leer, so wird das aktuelle Verzeichnis genommen. Beginnt *s* nicht mit „\“ oder einer Laufwerksangabe, so wird *s* ans aktuelle Verzeichnis angehängt.

```
PROCEDURE CorrectDir(s: STRING; VAR d: DirStr);
VAR aktDir: DirStr;
    ch    : CHAR;
    i     : BYTE;
BEGIN
  GetDir(0,aktDir);
  IF s = '' THEN d:= aktDir
    ELSE BEGIN
      d:= s; FOR i:= 1 TO Length(d) DO d[i]:= UpCase(d[i]);
      IF d[2] <> ':' THEN                       {Laufwerksangabe fehlt}
        IF d[1] <> '\' THEN d:= aktDir + '\' + d {Unterverzeichnis   }
                                                 {des aktuellen      }
          ELSE d:= aktDir[1] + ':' + d;          {Unterverzeichnis des}
                                                 {Wurzelverzeichnisses}
      END; {else}
  ch:= d[Length(d)];
  IF ch <> '\' THEN d:= d + '\';                 {Backslash anhaengen}
END;
```

1.5 Prüfen des Datenträgers

Das Laufwerk *d* wird auf Betriebsbereitschaft getestet. Als Ergebnis wird ein 'Return Code' *rc* zurückgegeben, der folgende Werte annehmen kann:

0 :	In Ordnung.
1 :	Kritischer Fehler (undefinierter Natur).
3 :	Laufwerk existiert nicht ('Path not found').
150 :	Diskette ist schreibgeschützt.
151 :	Laufwerk ist nicht bekannt.
152 :	Laufwerk ist nicht bereit.

```
PROCEDURE ChDisk(d: CHAR; VAR rc: BYTE);
VAR Lw, Result: BYTE;
    Dummy     : FILE;
BEGIN
  d:= UpCase(d);
  Lw:= ORD(d) - 64;
  Assign(Dummy, d + ':\DUMMY.TMP');
  {$I-}
  Rewrite(Dummy,1);
  {$I+}
  Result:= LO(IOResult);
  CASE Result OF
    3       : rc:= Result;
    150..152: rc:= Result;
```

```
    153..162: rc:= 1;
    ELSE rc:= 0;
    END; {case}
  IF rc = 0 THEN BEGIN Close(Dummy); Erase(Dummy) END;
END;
```

2 Physikalisches Löschen

Alle unbenutzten Cluster einer Diskette oder Festplatte werden durch Überschreiben mit einem Füllbyte physikalisch gelöscht. Dazu wird einfach eine Datei angelegt, die so groß ist wie der freie Platz auf dem Datenträger; sie wird mit dem Füllbyte vollgeschrieben und anschließend wieder gelöscht. Aufruf:

CLNDISK [?] [*Laufwerk* [*Füllbyte*]]

Ist der erste Parameter das „?“, so wird die Syntax des Aufrufs angezeigt; sonst geschieht nichts. Als Laufwerk ist das aktuelle Laufwerk voreingestellt, als Füllbyte das Formatierbyte $F6.

```
PROGRAM ClnDisk;

CONST blksize = 61440;

VAR Laufwerk: STRING;
    rc, Lw, fbyte: BYTE;
    code    : INTEGER;
    free    : LONGINT;
    Dummy   : FILE;
    ch      : CHAR;
    Block   : ARRAY[1..blksize] OF BYTE;
    i, WrAnz: WORD;

BEGIN
  IF ParamStr(1) = '?' THEN BEGIN
    WriteLn('Aufruf: CLNDISK [Laufwerk [Fuellbyte]]');
    Exit;
    END;
  ClrScr;
  WriteLn('Physikalisches Loeschen der unbenuetzten Bloecke einer Disk.');
  Laufwerk:= ParamStr(1); IF Laufwerk = '' THEN GetDir(0,Laufwerk);
  Val(ParamStr(2),fbyte,code); IF code <> 0 THEN fbyte:= $F6;
  ChDisk(Laufwerk[1], rc);
  CASE rc OF
      1 : WriteLn('Fehler in Laufwerk ', Laufwerk[1], ':');
```

```
      3 : WriteLn('Laufwerk ', Laufwerk[1], ': existiert nicht.');
    150 : WriteLn('Diskette ist schreibgeschuetzt.');
    151 : WriteLn('Laufwerk ', Laufwerk[1], ': ist nicht bekannt.');
    152 : WriteLn('Laufwerk ', Laufwerk[1], ': ist nicht bereit.');
    ELSE BEGIN
      Lw:= ORD(UpCase(Laufwerk[1])) - 64;
      free:= DiskFree(Lw);
      WriteLn('Noch ', free:12, ' Bytes zu loeschen. Abbruch mit [Esc].');
      Assign(Dummy, Laufwerk[1]+':\xpqarkbv.tmp');
      Rewrite(Dummy,1);
      ch:= ' ';
      FillChar(Block,blksize,fbyte);
      WHILE (free >= blksize) AND (ch <> #27) DO BEGIN
        IF KeyPressed THEN ch:= ReadKey;
        BlockWrite(Dummy, Block, blksize, WrAnz);
        free:= free - blksize;
        GotoXY(6,2); Write(free:12);
        END; {while}
        IF ch <> #27 THEN BEGIN
          BlockWrite(Dummy, Block, free, WrAnz);
          GotoXY(1,4); WriteLn('Disk gereinigt.');
          END;
        Close(Dummy);
        Erase(Dummy);
      END; {else}
    END; {case}
END.
```

3 Müll am Dateiende

Das jeweils letzte Cluster einer Datei wird hinter dem Dateiende durch Überschreiben mit einem Füllbyte physikalisch gelöscht. Behandelt werden alle Dateien eines Verzeichnisses, soweit sie nicht versteckt oder Systemdateien sind. Aufruf:

CLNFILE [?] [*Verzeichnis* [*Füllbyte* [r]]]

Ist der erste Parameter das „?“, so wird die Syntax des Aufrufs angezeigt; sonst geschieht nichts. Als Verzeichnis ist das aktuelle Verzeichnis voreingestellt, als Füllbyte das Nullbyte. Die Option „r“ sagt, daß auch schreibgeschützte Dateien gereinigt werden sollen.

```
PROGRAM ClnFile;

CONST maxsize = 4096;                          {maximale Blockgroesse}

TYPE Block     = ARRAY[1..maxsize] OF BYTE;  {Ein- bzw. AusgabePuffer}

VAR Laufwerk: CHAR;
    par     : DiskParam;
    Dir     : DirStr;
    Info    : SearchRec;
    att, rc, fbyte, att0: BYTE;
    code    : INTEGER;
    fsize, Zeiger, Rest: LONGINT;
    Datei   : File;
    bl      : Block;
    blksize, RAnz, WAnz: WORD;

BEGIN
  IF ParamStr(1) = '?' THEN BEGIN
    WriteLn('Aufruf: CLNFILE [Verzeichnis [Fuellbyte [r]]]');
    WriteLn('Die Option <r> bedeutet, dass auch schreibgeschuetzte');
    WriteLn('Dateien gereinigt werden.');
    Exit;
    END;
  Val(ParamStr(2),fbyte,code); IF code <> 0 THEN fbyte:= 0;
  IF ParamStr(3) = 'r' THEN att0:= $1E ELSE att0:= $1F;
                        {schreibgeschuetzte Dateien beruecksichtigen}
  CorrectDir(ParamStr(1),Dir);  {Parameter oder aktuelles Verzeichnis}
  Laufwerk:= Dir[1];
  ChDisk(Laufwerk, rc);
  CASE rc OF
      1 : WriteLn('Fehler in Laufwerk ', Laufwerk, ':');
      3 : WriteLn('Laufwerk ', Laufwerk, ': existiert nicht.');
    150 : WriteLn('Diskette ist schreibgeschuetzt.');
    151 : WriteLn('Laufwerk ', Laufwerk, ': ist nicht bekannt.');
    152 : WriteLn('Laufwerk ', Laufwerk, ': ist nicht bereit.');
  ELSE BEGIN
    GetDiskParam(Laufwerk, par);       {Clustergroesse wird gebraucht}
    blksize:= par.BytesProCluster;
    FindFirst(Dir+'*.*',AnyFile,Info);
    WHILE DosError = 0 DO BEGIN
      att  := Info.Attr;
      fsize:= Info.Size;
      Rest:= fsize MOD blksize;
```

```
      IF ((att AND att0) = 0) AND (Rest > 0) THEN BEGIN
        {Nicht behandelt werden Datentraeger- und Verzeichnisnamen   }
        {sowie versteckte Dateien, Systemdateien und Dateien, die    }
        {zufaellig an einer Clustergrenze enden. Schreibgeschuetzte  }
        {Dateien werden je nach Option 'r' behandelt.                }
        Zeiger:= fsize - Rest;           {Anfang des letzten Clusters}
        Assign(Datei,Dir+Info.Name);
        IF (att AND ReadOnly) <> 0 THEN SetFAttr(Datei, att AND $FE);
                             {Schreibschutz voruebergehend aufgehoben}
        Reset(Datei,1);                          {Satzgroesse = 1 Byte}
        Seek(Datei, Zeiger);                      {auf letztes Cluster}
        BlockRead(Datei,bl,blksize,RAnz);             {dieses einlesen}
        FillChar(bl[RAnz+1], blksize-RAnz, fbyte); {Puffer auffuellen}
        Seek(Datei, Zeiger);                      {auf letztes Cluster}
        BlockWrite(Datei,bl,blksize,WAnz);
        Seek(Datei,fsize);                          {wahres Dateiende}
        Truncate(Datei);  {hier abschneiden -- physikalisch bleiben  }
                          {die Fuellzeichen erhalten                 }
        SetFTime(Datei,Info.Time); {alte Zeitangabe wieder herstellen}
        Close(Datei);
        SetFAttr(Datei,att);   {ggfs. Schreibschutz wieder herstellen}
        END; {if}
      FindNext(Info);
      END; {while}
    END; {else}
  END; {case}
END.
```

Literaturverzeichnis

[1] ACM: *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing.* Providence, Rhode Island, May 6-8, 1985. The Association for Computing Machinery, Inc., New York 1985.

[2] Rudolf Baer: *Sicherheit in der EDV.* BSG Unternehmensberatung, St. Gallen 1988.

[3] Wulfdieter Bauerfeld: Hier wird gehackt (?). DFN Mitteilungen 16 (1989), 6–10.

[4] Henry Beker, Fred Piper: *Cipher Systems.* Northwood Books, London 1982.

[5] Thomas A. Berson, Thomas Beth (eds.): *Local Area Network Security.* Workshop LANSEC '89, Karlsruhe FRG, April 1989, Lecture Notes in Computer Science, Springer Verlag, Berlin 1989.

[6] Albrecht Beutelspacher: *Kryptologie.* Vieweg, Braunschweig 1987.

[7] Torsten Beyer: Sicherheitsaspekt von Computernetzwerken. In [90], 510–522.

[8] Joachim Bickenbach, Reinhard Keil-Slawik, Michael Löwe, Rudolf Wilhelm (Hrsg.): *Militarisierte Informatik.* Schriftenreihe Wissenschaft und Frieden, Berlin 1985.

[9] Ernst Biersack, Armin Liebl: Die Sicherheit des UNIX-Betriebssystems. In [90], 523–537.

[10] George Robert Blakley, David Chaum (Eds.): *Advances in Cryptology: Proceedings of* CRYPTO 84. Proceedings Univ. of California, Santa Barbara 1984. Lecture Notes in Computer Science 196, Springer-Verlag, Berlin usw. 1985.

[11] Hans-Peter Boell: Gesicherte Informationen. Online 10/89, 26.

[12] T. Börstler, Ch. Fischer: Sabotage vorprogrammiert! Computer-Viren bedrohen die Datenbestände. CAK Universität Karlsruhe 8/1989, 44–53.

[13] Gilles Brassard: *Modern Cryptology.* Lecture Notes in Computer Science 325, Springer-Verlag, Berlin usw. 1988.

[14] Hans H. Bräutigam: Falsch verstandener Datenschutz — Die Erforschung von Krankheitsursachen wird stark behindert. Die Zeit 13.4.1990, 86.

[15] Hans H. Bräutigam: Schwache Leistung – Falsch verstandener Datenschutz behindert die Forschung. Die Zeit 24.8.1990.

[16] Klaus Brunnstein, Simone Fischer-Hübner: Risk Analysis of "Trusted Computer Systems". SEC' 90 Sixth International Conference and Exhibition on Information Security, Helsinki, May 23–25, 1990.

[17] Der Bundesbeauftragte für den Datenschutz (Hrsg.): *Bürgerfibel Datenschutz.* Bonn 1980.

[18] Michael Burrows, Martin Abadi, Roger Needham: A logic of authentication. ACM Transactions on Computer Systems 8 (1990), 18–36.

[19] Dietrich Cerny: Vertrauenswürdige DV-Systeme – Das Bewertungsverfahren des US-Verteidigungsministeriums. In [130], 171–187.

[20] Chaos Computer Club: *Die Hackerbibel.* Grüne Kraft, Löhrbach 1985.

[21] David Chaum: New secret codes can prevent a computerized big brother. In [130], 33–34.

[22] David Chaum: Security without identification: Transaction systems to make Big Brother obsolete. Communications of the ACM 28 (1985), 1030–1044.

[23] David Chaum, Wyn L. Price (Eds.): *Advances in Cryptology* – EUROCRYPT '87. Proceedings Amsterdam 1987. Lecture Notes in Computer Science 304, Springer-Verlag, Berlin usw. 1988.

[24] James Arlin Cooper: *Computer and Communications Security – Strategies for the 1990s.* McGraw-Hill, Hamburg 1989.

[25] Gene Dallaire: Computer matching: Should it be banned? Communications of the ACM 27 (1984), 537.

[26] Ivan Bjerre Damgård (Ed.): *Advances in Cryptology* – EUROCRYPT '90. Proceedings Aarhus 1990. Lecture Notes in Computer Science 473, Springer-Verlag, Berlin usw. 1991.

[27] J. Damschen: Störungssicherheit ist auch Datensicherheit. Computer Magazin 3/91, 22–25.

[28] Datenschutzkommission Rheinland-Pfalz: *Datenschutzrechtliche Anforderungen an wissenschaftliche Forschungsvorhaben.* Informationen zum Datenschutz, Heft 3, Mainz 1987.

[29] D. W. Davis, Wyn L. Price: *Security for Computer Networks.* Wiley, New York 1984.

[30] Dorothy Elizabeth Robling Denning: *Cryptography and Data Security.* Addison-Wesley, Reading Mass. 1982.

[31] Dorothy Elizabeth Robling Denning: Digital signatures with RSA and other public-key cryptosystems. Communications of the ACM 27 (1984), 388–392.

[32] Steven Dickman: Heikler Datenschatz – Das größte Krebsregister der Welt birgt wichtige Informationen und brisante Probleme. Die Zeit 5.4.1991.

[33] Dworatschek, Büllesbach, Koch u. a.: *Personal Computer und Datenschutz.* Datakontext-Verlag, Köln 1990.

[34] Edelgard Eberlein, Stefan Seibold: Sicher fast wie im Tresor. Personal Computer 11/1989, 34–37.

[35] Frank Eckgold: Das Grünbuch macht die DV sicher. Online 8/1989, 25.

[36] Mark W. Eichlin, Jon A. Rochlis: With microscope and tweezers: The worm from MIT's perspective. Communications of the ACM 32 (1989), 689–698.

[37] Angela von Elling, Michael Wunder: *Krebsregister – Erfassung als Politik.* Konkret Literatur Verlag, Hamburg 1986.

[38] Amos Fiat, Adi Shamir: How to prove yourself: Practical solutions to identification and signature problems. In [87], 186–194.

[39] Rainer Gebauer: Sicherheit hat Vorrang. Personal Computer 11/1989, 30–32.

[40] Dahl A. Gerberick: Cryptographic key management or strong network security management. SIGSAC Review, Summer 1990, 12–23.

[41] David K. Gifford: Cryptographic sealing for information secrecy and authentication. Communications of the ACM 25 (1982), 274–286.

[42] Michael Gleißner: *Magnetkartensysteme.* Th. Watter, Lappersdorf 1989.

[43] Winfried Gleißner, Rüdiger Grimm, Siegfried Herda, Hartmut Isselhorst: *Manipulation in Rechnern und Netzen.* Addison-Wesley, Bonn usw. 1989.

[44] Joachim Graf: Computerkriminalität – Die Laus im Pelz. Computer Persönlich 23/1989, 52–57.

[45] Rick Grehan: Cloak and data – Using secret codes to secure your data from prying eyes. BYTE, June 1990, 311–324.

[46] Carla Grüning: Datensicherheit: Alle müssen mitziehen. Online 8/1989, 20–27.

[47] Christoph G. Günther (Ed.): *Advances in Cryptology* – EUROCRYPT '88. Proceedings Davos 1988. Lecture Notes in Computer Science 330, Springer-Verlag, Berlin usw. 1988.

[48] Michael Haller, Kuno Kruse, Thomas von Randow: Spione im Computernetz. Die Zeit 10.3.1989, 17–23.

[49] F. P. Heider, D. Kraus, M. Welschenbach: *Mathematische Methoden der Kryptoanalyse.* Vieweg-Verlag, Braunschweig 1985.

[50] Jürgen Hepe: Computer elektronisch verriegeln. Personal Computer 11/1989, 39–42.

[51] Lance J. Hoffman (ed.): *Rogue Programs: Viruses, Worms, and Trojan Horses.* Van Nostrand Reinhold, New York 1990.

[52] Marie-Luise Hoffmann, Gunhild Lütge: Blind für die Risiken. Ein Zeit-Gespräch mit Herbert Kubicek. Die Zeit 3.2.1989, 24–25.

[53] Patrick Horster: *Kryptologie.* Reihe Informatik/47, BI-Wissenschaftsverlag, Zürich 1985.

[54] IBM: *Virtual Machine/System Product, Introduction to Security, Release 6.* Document Number SC24-5316-01, June 1988.

[55] IBM: *Communications Security: "In-House" Cable and Line Considerations.* Document Number ZZ81-0232, December 1989.

[56] Jon A. Hupp, John F. Shoch: The "worm" programs – early experience with a distributed computation. Communications of the ACM 25 (1982), 172–180.

[57] David R. Johnson, Thomas P. Olson, David G. Post: *White Paper on Computer Viruses.* American Council on Education and United Educators Insurance, 1989.

[58] Franz-Joachim Kauffels: Schwachstellen der Informationssicherheit in lokalen Netzen. In [130], 51–69.

[59] Franz-Joachim Kauffels: *Rechnernetzwerksystemarchitekturen und Datenkommunikation.* Reihe Informatik, Band 54, BI-Wissenschaftsverlag, Mannheim 1987, 1989.

[60] Reinhard Keil-Slawik: „Intelligente“ Kriegsführung ohne Menschen? In [8].

[61] Heinrich Kersten: Sichere Daten – Anforderungen an PC-Sicherheitssysteme. PC Magazin 10/1989, 86–98.

[62] Michael Kienle: Mit Netz und doppeltem Boden. iX 2/1991, 101–107.

[63] Wilfried Köhler: Verwaltungsnetze: Chiffrierung. online 12/89, 80–84.

[64] Evangelos Kranakis: *Primality and Cryptography.* Wiley-Teubner Series in Computer Science, Teubner, Stuttgart 1986

[65] Hartwig Kreutz: Ein deutsches „Orange-Book“. DFN Mitteilungen 16 (1989), 11–12.

[66] Thomas Krivachy: The chipcard – an identification card with cryptographic protection. In [93], 200–207.

[67] Hanno Kühnert: Geheim beäugt. Die Zeit 12.5.1989, 87.

[68] Hanno Kühnert: Kein Knopf fürs Bürgerrecht. Die Zeit 2.11.1990, 85.

[69] Hanno Kühnert: Im Speicherstaat. Die Zeit 18.1.1991, 57.

[70] Richard P. Kusserow: The government needs computer matching to root out waste and fraud. Communications of the ACM 27 (1984), 542–545.

[71] Matthias Leclerc: Datensicherheit im MHS. Datacom 5/89, 90–98.

[72] Matthias Leclerc, Michael Steinacker: Eine Sicherheitsarchitektur für PC-Netze. In [90], 690–703.

[73] Hans-Albert Lennartz: *Datenschutz und Wissenschaftsfreiheit.* DuD-Fachbeiträge 10, Vieweg, Braunschweig 1989.

[74] Michael Löwe, Gerhard Schmidt, Rudolf Wilhelm: *Umdenken in der Informatik.* VAS in der Elefantenpress, Berlin 1987.

[75] Helmut Ludwigs: Sicherheit und UNIX. Online 8/1989, 24.

[76] Gunhild Lütge: Angst vor den Bitnappern. Die Zeit 25.10.1985, 25–26.

[77] Gunhild Lütge: Das gefährliche Hobby der Hacker. Die Zeit 15.4.1988, 23–25.

[78] Gunhild Lütge: Verstrickt im Datennetz. Die Zeit 24.3.1989, 25–26.

[79] Gunhild Lütge: Plastikgeld: Peinliche Panne. Die Zeit 13.10.1989, 40.

[80] Gunhild Lütge: Anstoß zum Fortschritt. Die Zeit 9.3.1990, 24.

[81] Gunhild Lütge: Alles unter Kontrolle? Die Zeit 11.5.1990, 25, 27.

[82] Gunhild Lütge: Rückschlag beim Fortschritt – Mit seiner Datenschutz-Verordnung setzt sich der Postminister über massive Proteste hinweg. Die Zeit 3.5.1991, 29.

[83] Derwent Maude, Tim Maude: Hardware protection against software piracy. Communications of the ACM 27 (1984), 950–959.

[84] Reinhard Merkel: Ist Hacken strafbar? Die Zeit 11.3.1989, 23.

[85] Barton P. Miller, Lars Frederiksen, Bryan So: Fatale Fehlerträchtigkeit. iX 3/91, 104–116.
Original: An empirical study of the reliability of UNIX utilities. Communications of the ACM 33 (1990), 32–44.

[86] Robert Morris, Ken Thompson: Password security: A case history. Communications of the ACM 22 (1979), 594–597.

[87] Andrew M. Odlyzko (Ed.): *Advances in Cryptology:* CRYPTO '86. Proceedings Univ. of California, Santa Barbara 1986. Lecture Notes in Computer Science 263, Springer-Verlag, Berlin usw. 1987.

[88] Gerhard Paaß: Anonymität von Individualdaten in statistischen Datenbanken. In [130], 156–170.

[89] Gerhard Paaß, Udo Wauschkuhn: *Datenzugang, Datenschutz und Anonymisierung.* Oldenbourg, München 1985.

[90] M. Paul (Hrsg.): *GI – 19. Jahrestagung I. Computergestützter Arbeitsplatz.* München, Oktober 1989. Informatik-Fachberichte 222, Springer-Verlag, Berlin 1989.

[91] Sicherheitsmaßnahmen reichen oft nicht aus. PC-Woche, 13.6.1988, 8.

[92] Andreas Pfitzmann: *Diensteintegrierende Kommunikationsnetze mit teilnehmerüberprüfbarem Datenschutz.* Informatik-Fachberichte 234, Springer-Verlag, Berlin usw. 1990.

[93] Franz Pichler (Ed.): *Advances in Cryptology:* EUROCRYPT '85. Proceedings Linz 1985. Lecture Notes in Computer Science 219, Springer-Verlag, Berlin usw. 1986.

[94] Hartmut Pohl: Der Mikro erschwert den Datenschutz. PC-Woche, 26.2.1990, 17-18.

[95] Klemens Polatschek: Achtung, Computer-Viren! Die Zeit 8.12.1989, 84.

[96] Carl Pomerance (Ed.): *Advances in Cryptology:* CRYPTO '87. Proceedings Univ. of California, Santa Barbara 1987. Lecture Notes in Computer Science 293, Springer-Verlag, Berlin usw. 1988.

[97] Anne Preissner-Polte: Datendieben auf der Spur. Die Zeit 28.2.86.

[98] Gero von Randow: Zu Gast in tausend Computersystemen. Die Zeit 10.3.1989, 20.

[99] Gero von Randow: Von Viren und Würmern. Die Zeit 10.3.1989, 21.

[100] Gero von Randow: Neues Bundesamt für Computersicherheit – Harte Zeiten für Bit-Banditen. CHIP, 3/1991, 24–26.

[101] Thomas von Randow, Michael Sontheimer: Die Hacker. Die Zeit 23.10.1987, 13–16.

[102] Karl Rihaczek: *Datenverschlüsselung in Kommunikationssystemen.* Vieweg, Braunschweig 1984.

[103] Rob Rosenberger: Computer virus myths. SIGSAC Review 7/4 (1990), 21–24.

[104] Christoph Ruland: *Datenschutz in Kommunikationssystemen.* Datacom, Pulheim 1987.

[105] Christoph Ruland: Datensicherheit in lokalen Netzen. Datacom, 12/1989, 94–99, und 1/1990, 100–107.

[106] G. Santucci et al.: Rationale for a community strategy in the field of information and communications technologies applied to health care. Methods of Information in Medicine, 29 (1990), 84–91.

[107] Angelika Schrader: ISDN – Ein Abenteuer in Sachen Datenschutz? Datacom 6/89, 34–35.

[108] Donn Seely: Password cracking: A game of wits. Communications of the ACM 32 (1989), 700–703.

[109] Adi Shamir: How to share a secret. Communications of the ACM 22 (1979), 612–613.

[110] John Shattuck: Computer matching is a serious threat to individual rights. Communications of the ACM 27 (1984), 538–541.

[111] Gustavus J. Simmons: Cryptology: The mathematics of secure communication. The Mathematical Intelligencer 1/4 (1979), 233–246.

[112] Gustavus J. Simmons: The practice of authentication. In [93], 261–272.

[113] Eugene H. Spafford: The Internet worm – crisis and aftermath. Communications of the ACM 32 (1989), 678–687.

[114] Computer – Verbogener Befehl. Der Spiegel 47/1984, 262–267.

[115] Die großen Systeme reizten Robert. Der Spiegel 47/1988, 252–258.

[116] Einstieg durch die Hintertür. Der Spiegel 47/1988, 258–265.

[117] Sicherheit – Weiße Elefanten. Der Spiegel 20/1989, 61–63.

[118] Computer – Chaos machbar. Der Spiegel 25/1989, 185–186.

[119] Gesellschaft – Gefährliche Netze. Der Spiegel 30/1989, 65–67.

[120] Telekommunikation – Stärker rüberbringen. Der Spiegel 38/1989, 253–259.

[121] Datenschutz – Magisches Datum. Der Spiegel 41/1989, 44–47.

[122] Bildschirmtext – Voll reingetappt. Der Spiegel 44/1989, 286–289.

[123] Geheimdienste – Stasi West. Der Spiegel 46/1989, 107–109.

[124] Computer – Warnung vom Türhüter. Der Spiegel 11/1990, 248–256.

[125] Datenschutz – Jeder macht es. Der Spiegel 41/1990, 118–121.

[126] Die programmierte Katastrophe. Der Spiegel 42/1990, 80–93.

[127] Datenschutz – Insel der Anonymität. Der Spiegel 52/1990, 64–65.

[128] Schrankenlos gesammelt. Der Spiegel 8/1991, 65–70.

[129] Datenschutz – Alles oder nichts. Der Spiegel 16/1991, 50–53.

[130] Peter Paul Spies (Ed.): *Datenschutz und Datensicherung im Wandel der Informationstechnologien.* 1. GI-Fachtagung, München, Oktober 1985, Proceedings, Informatik-Fachberichte 113, Springer-Verlag, Berlin usw. 1985.

[131] Peter Paul Spies: Datenschutz und Datensicherung im Wandel der Informationstechnologien. In [130], 1–25. Informatik-Fachberichte 113, Springer-Verlag, Berlin usw. 1985.

[132] Clifford Stoll: Stalking the wily hacker. Communications of the ACM 31 (1988), 484–497.

[133] Ken Thompson: Reflections on trusting trust. Communications of the ACM 27 (1984), 761–763.

[134] Roy F. Van Buren: How you can use the Data Encryption Standard to encrypt your files and databases. SIGSAC Review, Summer 1990, 33–39.

[135] Willis H. Ware: Information systems security and privacy. Communications of the ACM 27 (1984), 315–321.

[136] Gerhard Weck: *Datensicherheit.* Leitfäden der angewandten Informatik, B. G. Teubner, Stuttgart 1984.

[137] Gerhard Weck: Wirksamer Schutz auf Betriebssystem-Ebene. Online 10/1989, 58–60.

[138] Jürgen Weimann: Chip-Karten: Realisierungs- und Anwendungsmöglichkeiten. In [130], 26–32.

[139] Horst Wettstein: *Architektur von Betriebssystemen.* Hanser, München 1978, 1987.

[140] Stefan Wichmann: Umwelt – Gläserne Bürokratie. Die Zeit 1.12.1989, 43.

[141] Hugh C. Williams (Ed.): *Advances in Cryptology:* CRYPTO '85. Proceedings Univ. of California, Santa Barbara 1985. Lecture Notes in Computer Science 218, Springer-Verlag, Berlin usw. 1986.

[142] Peter Wollschläger: Vorsicht, tückische Fallen! Computer Persönlich 5/91, 110–111.

[143] Marie A. Wright: Communication security in a distributed network. SIGSAC Review 7/4 (1990), 1–6.

[144] Brigitte Zander: Datenklau statt Tütenkleben. Stern 44/89, 138–140.

[145] Zentralstelle für Sicherheit in der Informationstechnik (Hrsg.): *IT-Sicherheitskriterien – Kriterien für die Bewertung der Sicherheit von Systemen der Informationstechnik (IT).* Bundesanzeiger, Köln 1989.

[146] Zentralstelle für Sicherheit in der Informationstechnik (Hrsg.): *IT-Evaluationshandbuch – Handbuch für die Prüfung der Sicherheit von Systemen der Informationstechnik (IT).* Bundesanzeiger, Köln 1990.

[147] Dieter E. Zimmer: Schüttelfrost im System. Die Zeit 2.9.1988, 63.

Index