Titelblatt

BelWü Spots

+++ in aller Kürze +++

+++ Die Leitungs-Infrastruktur für das BelWü-Kernnetz wird mittlerweile von der Fa. CNS (Communikationsnetze Süd-West, eine Tocherfirma der baden-württembergischen Energieversorger und der Swiss Telekom) bereitgestellt. Derzeit sind die neun Landesuniversitäten mit je 155 MBit/s, sechs Fachhochschulen und das MWK (Ministerium für Wissenschaft, Forschung und Kunst) mit je 34 MBit/s ATM-basierenden Anschlüssen ausgestattet. Fachhochschulen und Berufsakademien sind zunehmend mit 2 MBit/s (i.d.R. über Telekomleitungen) angebunden. +++

+++ Der eine 155 MBit/s und die acht 34 MBit/s BWiN-für die nationale und internationale Versorgung im Rahmen des DFN (Deutsches Forschungsnetz e.V.) werden ab 1.1.98 durch drei 155 MBit/s BWiN-Anschlüsse ersetzt. +++

+++ Zwischen Wissenschafts- und Kultusministerium wurde eine Vereinbarung getroffen, wonach in den nächsten drei Jahren in Baden-Württemberg bis zu 4900 Schulen flächendeckend zum City-Telefontarif an BelWü angeschlossen werden sollen. +++

+++ Inzwischen ist auch der OSS-Zugang zur Firma SAP (Walldorf) über das BelWü möglich, allerdings kostenpflichtig, um die Standleitungskosten dorthin zu decken. +++

+++ Mittlerweile sind an BelWü über 400 Einrichtungen mit über 100.000 Rechnern angeschlossen. +++

Herausgeber:

BelWü-Koordination
Allmandring 30
70550 Stuttgart

Januar 1998
1/ 98

Redaktion:
B. Herrmann
Satz:
B. Herrmann

Auflage: 5300

Druck:
Schwäbische Druckerei,
Stuttgart

Bezug/Leserbriefe:
belwue-koordination@belwue.de

Inhalt

Auswahlkriterien für Firewallsysteme

Karl Gaissmaier / Frank Kargl

Einleitung

Innerhalb der letzten zwei Jahre ist das Angebot der verfügbaren Firewallsysteme stark gewachsen und damit auch sehr unübersichtlich geworden. Einerseits ist damit die Voraussetzung gegeben, für eine spezielle Aufgabe auch ein entsprechendes Produkt zu finden, andererseits wird es immer schwerer, über die am Markt befindlichen Systeme den Überblick zu bewahren und die Vor- und Nachteile der einzelnen Produkte zu erkennen. Ein Firewallsystem für den Anschluß eines Verwaltungsnetzes an das Internet muß z. B. ganz andere Voraussetzungen erfüllen als ein entsprechendes System zur Absicherung eines F&L-Netzes einer Universität oder Forschungseinrichtung. Während im ersten Fall die Abschottung überwiegt, auch unter Inkaufnahme von Einschränkungen in der Benutzerfreundlichkeit, ist im zweiten Fall die Transparenz, Geschwindigkeit und Vollständigkeit der unterstützten Dienste der dominierende Faktor. Da die Entwicklung von neuen Firewallsystemen auch weiterhin mit dem Erfolg des Internet rasant wachsen dürfte, ist man auf die Aussagen von verschiedenen Institutionen angewiesen, die anhand von komplexen Testaufbauten und Testszenarios die Voraussetzungen schaffen, die verschiedenen Produkte vergleichen zu können. Im Rahmen des Deutschen Forschungsnetzes wurde z. B. dafür ein neues Projekt ins Leben gerufen, um Firewalls für Hochgeschwindigkeitsnetze zu testen. Ein weiter gefaßtes Produktspektrum bietet das Zertifikationsprogramm der National Computer Security Association, die anhand bestimmter Kriterien Firewallsysteme zertifiziert. Daneben widmen sich natürlich immer häufiger auch Zeitschriften diesem Thema mit mehr oder weniger detaillierten Vergleichstests, z. B. Data Communications, Network World Fusion, LANline, etc. Einen Überblick über die am Markt befindlichen Produkte verzeichnen mehrere Listen im WWW, z. B. vom DFN, oder auch aus vielen anderen Quellen.

Um die verschiedenen Auswahlkriterien für Firewallsysteme einschätzen zu können werden im nächsten Abschnitt die grundlegenden Begriffe kurz erläutert. Für ein tiefergehendes Verständnis sollte die entsprechende Literatur gelesen werden.

Überblick

Firewall Prinzipien

Der Firewall stellt die Verbindung zwischen geschütztem und ungeschütztem Netz her. Seine Aufgabe ist es, den Verkehr zwischen beiden Netzen zu überwachen und nach vorher festgelegten Regeln durchzulassen oder den Zugang zu verweigern und im Bedarfsfall dies zu protokollieren. Oftmals unterstützt ein Firewall ein drittes Netzwerkinterface, um ein sogenanntes Service Netzwerk (SN) zu unterstützen, in welchem die öffentlich zugänglichen Informationsserver platziert werden, damit ein direkter Durchgriff aus dem Internet ins Intranet überflüssig wird. Müssen verschiedene Standorte über das Internet verbunden werden, spricht man von virtuellen privaten Netzen (VPN‘s). Beim Datenverkehr zwischen den zu schützenden Netzen werden die Daten vor der Überbrükkung offener Netzwerksegmente von den beteiligten Firewallsystemen verschlüsselt.

Basiskomponenten von Firewallsystemen

Paketfilter

Die Filterung des Verkehrs mittels Paketfilter geschieht über die Angabe der Quell- und Ziel-IP-Adresse sowie durch Festlegen der entsprechenden Portnummern. Eventuell kann auch das ein- bzw. ausgehende Interface definiert werden und bei verbindungsorientierten Protokollen die Richtung des Verbindungsaufbaus. Es muß also die Adresse des Rechners, der die Verbindung aufbaut, sowie die IP-Adresse des Rechners, mit dem kommuniziert werden soll, bekannt sein. Die Portnummern legen den

Dienst fest und kennzeichnen die Standard Ports, wie sie unter UNIX in der Datei

/etc/services

definiert sind. Simple Paketfilter benutzen also Informationen aus den OSI Schichten 3 und 4 als Filterkriterien.

Der Vorteil eines Paketfilter-basierenden Firewalls besteht darin, daß die notwendige Hardware (Router) in einer Netzumgebung in der Regel schon vorhanden ist und somit die Investitionskosten gering sind.

Diesem positiven Aspekt stehen allerdings gravierende Nachteile gegenüber:

- Dienste mit Rückkanal, wie z. B. ftp, X11 und die meisten UDP-basierten Dienste lassen sich mit Paketfiltern nicht sicher handhaben
- hohe Anforderungen an das Spezialwissen über die verwendeten Dienste
- der Datenstrom selbst wird nicht analysiert
- keine oder nicht ausreichende Protokollierung der Zugriffe
- keine User-Autorisierung möglich
- kein Aktionsmechanismus bei Mißbrauchsversuchen
- Dynamische Paketfilterung und Stateful Inspection

Paketfilter können auch mittels Software auf Firewallsystemen realisiert werden. Sogenannte dynamische Paketfilter führen Buch (Hash Tabellen) über initiierte Verbindungen und prüfen, ob die Antwortpakete auch tatsächlich zu einer initiierten Verbindung (Socket) gehören. Zusätzlich können Statusinformationen aus Datenpaketen gewonnen werden (Stateful Inspection), die auf der Netzwerk- und Transportebene noch statuslos sind. Die Software kann also Informationen aus der Anwendungsschicht nutzen, um dynamisch Filterregeln zu erstellen, die einen Rückkanal für eine bestimmte Verbindung und innerhalb eines definierten Zeitfensters zulassen. Somit können auch Dienste, die auf UDP oder RPC/UDP basieren, gefiltert werden.

Circuit Level und Application Layer Proxies

Circuit Level Proxies arbeiten auf den OSI Schichten 3 und 4. Bei bestimmten Circuit Level Proxies kann auch User Authentifizierung angewendet werden. Im Gegensatz zu statischen Paketfiltern erlauben diese Proxies zusätzliche Protokoll- und Aktionsmechanismen. Die Filtersprache ist in der Regel wesentlich einfacher und damit weniger fehleranfällig. Zur Konfiguration wird deutlich weniger Spezialwissen benötigt. Sie sind sicherer anzuwenden als statische Paketfilter und haben ihre Berechtigung bei verbindungsorientierten Diensten, deren Sicherheitsanforderungen nicht so hoch sind (z.B. News).

Eine Interpretation des Datenstroms erfolgt hier nicht, z. B. kann nicht unterschieden werden, ob das ftp-Protokoll einen put oder einen get Befehl ausführt. Circuit Level Proxies werden auch als generische Proxies bezeichnet, da ihr Einsatz im Gegensatz zu an Diensten gebundenen Proxies (Application Layer Proxy) für fast jeden beliebigen Dienst möglich ist. Wird eine feinere Steuerung gewünscht, müssen Application Proxies eingesetzt werden.

Ein Application Layer Proxy ist speziell auf einen dedizierten Dienst ausgerichtet. Die Protokollierung ist wie bei den Circuit Level Proxies fester Bestandteil der Funktionalität. Zusätzlich kann der Datenstrom dienstspezifisch analysiert werden. Beim ftp-Protokoll ist es z.B. möglich, get in das geschützte Netz zu unterbinden, aber put zu erlauben. Damit kann verhindert werden, daß Dateien in das geschützte Netz hinein transportiert werden. In Verbindung mit einer Authentifizierungs-/Autorisierungsdatenbank erlaubt das System, die Dienste des Firewalls nur zusammen mit einer Userkennung und einem Paßwort zu nutzen.

Von Nachteil erweist sich allerdings, daß nicht für alle Dienste Proxies vorhanden sind, oder sich bestimmte Protokolle nicht zur User-Authentifizierung eignen. So wird z. B. beim WWW-Dienst mit jedem neuen Hyperlink eine neue tcp-Verbindung aufgebaut. Wenn dabei jedesmal eine User-Authentifizierung durchgeführt werden müßte, wäre der Dienst praktisch unbenutzbar. Weiterhin ist die Transparenz bei vielen Proxies nicht gegeben, d.h. vom Benutzer wird ein an den Firewall angepaßtes Verhalten erwartet (Konfiguration der Client Software, spezielles Vorgehen beim FTP usw.) Es gibt mittlerweile aber auch sog. Transparente Proxies (z.B. bei Gauntlet), die dieses Problem nicht haben.

Beide Proxy-Systeme führen automatisch eine Adreßumsetzung durch, d.h. die interne Netzwerkstruktur tritt nach außen nicht in Erscheinung.

Inhaltliche Filterung

Zum Schutz vor Viren beim Download oder bei Attachments bieten einige Firewallhersteller inhaltlich basierte Filterung an. Zusätzlich können oftmals bestimmte URL‘s gesperrt werden oder bestimmte Scriptsprachen wie Javascript, ActiveX oder auch Java aus einer HTML Seite ausgefiltert werden.

Authentifizierung, Autorisierung und Accounting

In einer Authorisierungsdatenbank werden Benutzer erfaßt, die sich auf dem Firewall zur Nutzung eines Dienstes authentisieren dürfen und deren Autorisierung für verschiedene Dienste nach einem definierten Benutzerprofil geschieht. Der Unterschied zur üblichen Benutzerkennungen besteht darin, daß Benutzer keinen interaktiven Zugriff auf den Firewall erhalten.

Zusätzlich zum üblichen Authentifizierungsverfahren mit Benutzername und Paßwort existieren weitere Verfahren, die gegenüber Abhören von Leitungen unempfindlich sind. Sie basieren entweder auf dem Challenge/Response Prinzip mit sich ständig ändernden Paßwort-Strings oder auf einer Zeitsynchronisation zwischen dem Authentifizierungsserver und einer intelligenten Chipkarte im Besitz des Nutzers.

Diese Verfahren sind sehr sicher, allerdings in der Anwendung unhandlich. Bei Authentifizierungsvorgängen von Nutzern innerhalb des geschützten Netzes kann auf diese Art der Authentifizierung verzichtet werden. Wann immer aber ein Zugang von außen in das geschützte Netz eingerichtet wird (dies sollte weitgehendst vermieden werden), sollte mit einem dieser sicheren Authentifizierungsverfahren gearbeitet werden. Außerdem müssen alle administrativen Arbeiten am Firewall, die nicht direkt an der Konsole vorgenommen werden, auf diese Weise authentisiert und zusätzlich verschlüsselt werden, auch wenn sich die gesamte Kommunikation innerhalb des geschützten Netzes abspielt.

Firewall - Architekturen

Es gibt verschiedene Möglichkeiten, einen Firewall aus den oben beschriebenen Komponenten zu realisieren. Folgende Firewall Systeme werden hier kurz beschrieben:

- Screening Router
- Dual Homed Host
- Multi Homed Host

Auf weitere Systeme, insbesondere auf Kombinationen von Routern und Bastion Hosts (Screened Host, Screened Subnet, Belt and Suspender, etc.), soll hier nicht näher eingegangen werden.

Screening Router

Die einfachste Möglichkeit, mehrere Netze kontrolliert zu verbinden, um bestimmte Dienste oder Zugriffe einschränken zu können, ist der Einsatz eines Routers mit Paketfiltern. Die Nachteile wurden bereits ausführlich beschrieben. Ein derartiger Firewall sollte für die Absicherung eines sensitiven Netzes nicht benutzt werden.

Dual Homed Host

Ein Dual Homed Gateway ist ein Rechner mit zwei Netzwerkinterfaces, wobei IP-forwarding explizit unterbunden ist. Dieser Gateway verhindert das Routing zwischen geschütztem und ungeschütztem Netz, d.h. der einzige erreichbare Rechner des jeweils anderen Netzes ist immer das Gateway. Dienste werden durch dynamische Paketfilterung mit Stateful Inspection und/oder Proxies (Circuit Level Proxy, Application Layer Proxy) kontrolliert.

Größter Nachteil hierbei :

Öffentliche Informationsserver sollten im ungeschützten Netz plaziert werden, da sonst zu viele Dienste vom ungeschützten in das geschützte Netz geöffnet werden müßten. Diese Informationsserver müssen durch zusätzliche Maßnahmen abgesichert werden (z.B. tcp-Wrapper) wobei man dann anstelle der Netzwerksicherheit wieder auf die alleinige Hostsicherheit bei diesen Servern angewiesen ist..

Multi Homed Host

Ein Multi Homed Host verbindet mindestens drei Netze miteinander: ein äußeres, ungeschütztes Netz (z.B. das Internet), ein inneres, zu schützendes Netz und das Service Netzwerk. Im letztgenannten Netz befinden sich alle Rechner, auf die sowohl vom geschützten als auch vom ungeschützten Netz aus zugegriffen werden kann. Für die meisten Dienste sind von beiden Seiten aus jeweils nur die Rechner im Service Netzwerk sichtbar, ein Übergang vom ungeschützten zum geschützten Netz (oder umge-

kehrt) ist nur für wenige, einfache Dienste durch den Firewall zu realisieren.

Auswahlkriterien

Um eine nachvollziehbare Produktauswahl treffen zu können, muß zuallererst eine Sicherheitsrichtlinie aufgestellt werden. Alle Produkte, die diese Security Policy nicht erfüllen können, scheiden von vornherein aus. Anhand weiterer Kriterien können dann die verbleibenden Produkte verglichen werden. Die unterschiedlichen Kriterien lassen sich z. B. in folgende Kategorien zusammengefassen:

- Systemeigenschaften
- Kosten
- Systemsicherheit
- Filter- und Authentifizierungsmöglichkeiten
- Konfiguration und Administration
- Protokollierung
- Leistungsfähigkeit und Erweiterbarkeit
- Sonstiges

Die einzelnen Unterpunkte der Kategorien werden im folgenden aufgelistet, wobei die Wichtigkeit eines jeden einzelnen Punktes für die entsprechende Anforderung abgeschätzt werden muß.

Systemeigenschaften

Ist es ein exotisches Betriebssystem oder hat das Personal bereits ausreichende Erfahrung damit?

Ist die Hardware im Unternehmen verbreitet und existieren eventuell Backup Geräte?

Kann die Firewall-Software getrennt von der Hardware erworben und auf mehreren Hardware Plattformen und Betriebssystemen eingesetzt werden oder ist es ein geschlossenes System? Abhängig von den Anforderungen kann das eine oder andere von Vorteil sein.

Welche Netzwerkschnittstellen und welche Anzahl werden vom Firewall unterstützt?

Kosten

- Was kostet Hardware, Software, Schulung, Installation?
- Was kostet ein Wartungs- und/oder Upgradevertrag? Wie groß ist der administrative Aufwand, d.h. wieviel Personal ist notwendig?

SystemSicherheit

- Ist die Sicherheit auch während des Startvorgangs und nach einem Systemabsturz gewährleistet?
- Werden von der Firewallsoftware periodische Integritätstests der Programme und Dateien unterstützt?
- Ist das Betriebssystem bereits „gehärtet“ oder gibt es dafür eine Anleitung?
- Sind die Routingtabellen des Firewallsystems gegen ICMP Redirects und direkten Einfluß durch Routingprotkolle geschützt?

Filter- und Authentifizierungsmöglichkeiten

- Existiert ein wirksamer Schutz gegen IP Spoofing?
- Können die Filterregeln für die einzelnen Interfaces definiert werden und kann die Richtung angegeben werden?
- Wird nur ein statisches Paketfilter oder auch dynamische Filterung unterstützt? Gibt es Stateful Inspection?
- Können die Filterregeln auf Widerspruchsfreiheit überprüft werden?
- Werden bei der Defaulteinstellung alle Verbindungen blockiert, die nicht explizit erlaubt sind?
- Können Benutzer zu Gruppen zusammengefaßt werden?
- Ist die Angabe der Netzwerkobjekte mit variablen Netzmasken möglich?
- Für welche Dienste werden Proxies geliefert (Application Layer oder Circuit Level)?
- Können UDP-basierte Protokolle z. B. mittels Stateful Inspection sicher benutzt werden?
- Können Dienste auch nur zu bestimmten Zeiten für bestimmte Benutzer freigegeben werden?
- Wird inhaltliche Filterung für importierte Daten (ftp, smtp/MIME und http) angeboten?
- Wird NAT unterstützt?
- Für welche Dienste kann eine Authentifizierung stattfinden? Wie transparent können diese Dienste dann benutzt werden?

- Welche Techniken zur Authentifizierung werden unterstützt? Werden Authentifizierungsserver von Drittherstellern unterstützt?
- Kann ein VPN aufgebaut werden? Wie stark ist die Verschlüsselung?

Konfiguration und Administration

Sind die Rechte in Administrator und Revisor aufteilbar? Wird dies vom Produkt defaultmäßig unterstützt? Wird das Produkt mit sicheren Ausgangsparametern für die Zugriffsrechte von Administrator und Revisor geliefert?

Kann die Administration nur an der Console oder von einem beliebigen Rechner im LAN vorgenommen werden? Die Identifikation und Authentifizierung für Administrator und Revisor darf im zweiten Fall nur über einen verschlüsselten Kanal erfolgen. Steht ein Mechanismus für starke Authentifizierung von Administrator und Revisor zur Verfügung?

Kann man sowohl über ein graphisches als auch ein ASCII-Interface administrieren? Graphische Interfaces sind intuitiver und weniger fehleranfällig, wobei ASCII-Interfaces oftmals flexibler sind und bei großen Listen auch Geschwindigkeitsvorteile bieten.

Können mehrere Komponenten eines Firewallsystems von einem zentralen Standpunkt aus verwaltet werden?

Protokollierung

Können die Protokolle an einen zentralen Standort weitergeleitet werden? Ist die Integrität der übermittelten Daten gewährleistet?

Kann die Protokollierung auf bestimmte Ereignisse beschränkt werden? Gibt es verschiedene Detailstufen der Protokollierung?

Können durch bestimmte Ereignisse Aktionen gestartet werden? Welche Arten von Aktionen können veranlaßt werden?

Was passiert wenn der Protokollmechanismus ausfällt (z. B. Platte/Band voll), werden dann die Dienste gesperrt?

Können nur die Verbindungsdaten oder auch die Daten an sich protokolliert werden?

Stehen geeignete Werkzeuge zur Datenreduktion und zum Erkennen von Einbruchsmustern zur Verfügung?

In welchen Formaten können die Protokolldaten exportiert werden?

Leistungsfähigkeit und Erweiterbarkeit

Können auch Geschwindigkeiten von 100Mbit/s (FDDI, Fast Ethernet) oder noch mehr (ATM) nachprüfbar gefiltert werden?

Ist die Hardware überhaupt in der Lage, diese schnellen Interfaces mit der Nominalgeschwinigkeit zu bedienen?

Ist das System in der Lage, auch andere Interfaces zu betreiben oder ist es eine Black Box mit definierten Schnittstellen?

Kann die Hardware unabhängig von der Software gegen eine leistungsfähigere ausgetauscht werden?

Ist das System hinsichtlich neuer Dienste und Protokolle erweiterbar?

Sonstiges

Ist das Produkt schon länger am Markt etabliert?

Wurde das Produkt trotz korrekter Installation in der aktuellen Version bereits einmal durchbrochen?

Wird das Produkt weiterentwickelt, und wie reagiert der Hersteller auf neue Angriffsmethoden?

Hat der Hersteller eine funktionierende Hotline? Wie lang ist die durchschnittliche Wartungszeit? Funktioniert die Eskalation bei schwierigen Problemen bis zum Entwickler?

Gibt es eine gute Referenzliste? Haben Sie sich bei einem Anwender dieses Produkts unabhängig informiert?

Zusammenfassung

Die Produktpalette der am Markt befindlichen Firewallsysteme ist sehr unübersichtlich und ohne eine vorher definierte Sicherheitsrichtlinie ist es sehr schwierig,sich für ein passendes Produkt nachvollziehbar zu entscheiden. Die Erfahrung zeigt, daß viele Produkte bereits bei der ersten Hürde, die Sicherheitsrichtlinie ausreichend erfüllen zu können, ausscheiden. Die Leistungsfähigkeit und Erweiterbarkeit sollte man bei der Entscheidung nicht zu gering bewerten, denn nichts ist so schnellebig wie die Netzwerk- und Sicherheitstechnik.

Quellen:

[1] Firewall-Labor für Hochgeschwindigkeitsnetze, erst im Aufbau (http://www.cert.dfn.de/fwl/)

[2] NCSA Certified Firewall Products (http://www.ncsa.com/fpfs/)

[3] NCSA FWPD Criteria (http://www.ncsa.com/fpfs/fwct20.html)

[4] Data Communications, Firewalls: Don‘t Get Burned (http://www.data.com/lab_tests/firewalls97.html)

[5] Network World Fusion, Asbestos for the network (http://www.nwfusion.com/ nur nach vorheriger Registrierung/

[6] LANline, Schwerpunkt: das sichere Netz 6/97 und 7/97

[7] Liste deutscher Firewall-Anbieter (http://www.fwl.dfn.de/fwl/fw/fw-prod.html)

[8] Liste internationaler Firewall-Anbieter (http://www.fwl.dfn.de/eng/fwl/fw/fw-prod.html)

[9] Firewall Product Overview (http://www.greatcircle.com/firewalls/vendors.html)

[10] Einrichten von Internet Firewalls, Brent Chapman & Elizabeth D. Zwickey, ISBN 3-93067-31-2

[11] Firewalls und Sicherheit im Internet, William R. Cheswick, Steven M. Belovin, ISBN 3-89319-875-X

[12] Practical UNIX & Internet Security, Simson Garfinkel & Gene Spafford, ISBN 1-56592-148-8

Karl Gaissmaier
karl.gaissmaier@rz.uni-ulm.de

Frank Kargl
frank.kargl@rz.uni-ulm.de

Universitätsrechenzentrum Ulm

Sicherheitsaspekte in AMBIX

Dr. Ralf Schneider, Karl-Peter Gietz, Dr. Kurt Spanier

Ein Directory-Service des DFN-Vereins

Das DFN-Projekt AMBIX[1] ermöglicht Organisationen aus dem Bereich der Forschung in Deutschland, bequem und datenschutzkonform Personendaten im X.500 zu veröffentlichen und aktuell zu halten. Die Einhaltung datenschutzrechtlicher Vorgaben erfordert die Berücksichtigung verschiedener Sicherheitsaspekte.

AMBIX und X.500

Die Speicherung und Veröffentlichung von Personen- und Organisationsdaten bei AMBIX erfolgt mit Hilfe der X.500-Technologie [1], einer von internationalen Normierungsgremien (ISO und ITU) standardisierten, weltweit verteilbaren Datenbank.

Im X.500 können beliebige, hierarchisch darstellbare Informationen gespeichert werden. Alle Daten werden in Servern, den *Directory System Agents* (DSAs), lokal abgelegt und verwaltet. Mittels *Directory User Agents* (DUAs) können über spezielle Protokolle Informationen gesucht, abgefragt und angezeigt sowie verändert oder gelöscht werden. Eines der wichtigsten DUA/DSA-Protokolle, das *Lightweigth Directory Access Protocol* (LDAP) [2], hat sich mittlerweile als Internet-Standard etabliert und wird von vielen führenden Software-Herstellern unterstützt.

Die vom DSAs gespeicherten Informationen sind im *Directory Information Tree* (DIT) hierarchisch strukturiert angeordnet, wobei jeweils ein sog. Master-DSA für die Verwaltung eines definierten Teilbereiches dieses Baumes zuständig ist.

Alle Informationen eines Eintrags werden in Attributen abgelegt, welche sich aus einem Attribut-Typ und einem oder mehreren Attribut-Werten zusammensetzen. Bestimmte Attribut-Werte bilden den innerhalb einer Hierarchieebene eindeutigen *Relative Distinguished Name* (RDN) eines Eintrags. Zusammen mit den im DIT hierarchisch übergeordneten RDNs ergibt sich daraus der im gesamten DIT eindeutige *Distinguished Name* (DN). Als eine Hauptanwendung des X.500 ist die Speicherung von Organisations- und Personendaten zu nennen. Alle hierfür benötigten Attribute werden vom Standard zur Verfügung gestellt.

Bereits bei der Konzeption des Projekts AMBIX wurden die Aspekte der deutschen Datenschutzgesetzgebung berücksichtigt. Grundlage hierfür bildete eine vom DFN-Verein in Auftrag gegebene Datenschutzexpertise [3]. Kernpunkte der Expertise besagen: Werden die erfaßten Personendaten auf ein Minimalset aus Name, Vorname, akademischem Titel, Telefon- und Faxnummer, E-Mail-Adresse sowie Organisationszugehörigkeit beschränkt, genügt vor deren Veröffentlichung im AMBIX-Verzeichnis die Einhaltung einer angemessenen Widerspruchsfrist (6 Wochen), nachdem die Betroffenen über die Datenlieferung informiert und ihre Rechte aufgeklärt worden sind. Dafür dürfen diese Personendaten nur an Länder mit adäquaten Datenschutzbestimmungen weitergegeben werden.

Zukünftig werden auch öffentliche Schlüssel von asymmetrischen Verschlüsselungsverfahren im AMBIX-Verzeichnis abgelegt. Bei der hierfür notwendig gewordenen Erweiterung des bestehenden Minimalsets wurde dieses um öffentliche Schlüssel und zusätzliche, freiwillige Angaben wie Arbeitsgebiet und URL ergänzt.

Mit diesem erweiterten Minimalset können nun auch die Daten der Teilnehmer von WiNShuttle aufgenommen werden, einem Internet-Provider-Dienst des DFN-Vereins für Angehörige von Einrichtungen aus dem öffentlichen bzw. bildungs- und forschungsorientierten Bereich. Da WiNShuttle-Benutzer nur nach einer expliziten Zustimmung zur Veröffentlichung Ihrer Daten im Verzeichnis aufgenommen werden, unterliegen diese Daten nicht den oben genannten Beschränkungen der Datenschutzexpertise.

Organisationsdaten unterliegen keinen datenschutzrechtlichen Beschränkungen und können mit allen relevanten Informationen in der hierarchischen Struktur des X.500 abgebildet und frei zur Verfügung gestellt werden.

---

1 (A)ufnahme von (M)ail-(B)enutzern (i)n das (X).500 Directory. Dieses DFN-Projekt wird gefördert aus Mitteln des BMBF.

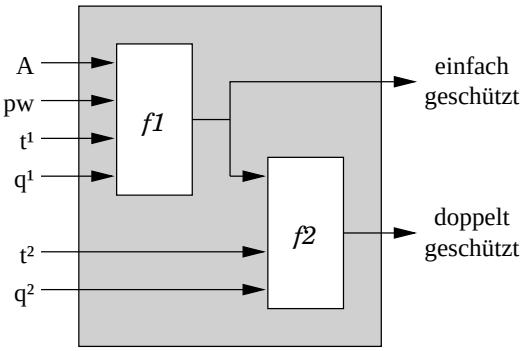
Für die Verarbeitung der Datenlieferungen von Organisationen und den Reaktionen der Betroffenen wurde eine netzwerkfähige *remote procedure call* (RPC) basierende Client-Server Maschine entwickelt [4]. Um alle Vorgaben der Datenschutzexpertise zu erfüllen, werden die Daten zuerst in einer projektinternen Datenbank gespeichert und alle Vorgänge protokolliert, wodurch die Chronologie eines Eintrags jederzeit nachvollzogen werden kann.

TWEB (Tübinger Web-Gateway), ein WWW-X.500-Gateway, ermöglicht einen komfortablen Zugang zu den Daten. Hierin wurden speziell für AMBIX ein Mechanismus zur Realisierung der datenschutzrechtlich vorgegebenen Zugriffsbeschränkung sowie ein Mechanismus zum Selbsteintrag ins AMBIX-Verzeichnis implementiert.

Sicherheit in AMBIX und im X.500

Die Einhaltung der datenschutzrechtlichen Vorgaben erfordert die Berücksichtigung verschiedener Sicherheitsmaßnahmen. So kommt der Authentifizierung der Benutzer und der Bestimmung des Landes, aus dem der Zugriff erfolgt, eine wichtige Rolle zu.

Da nicht davon ausgegangen werden kann, daß alle Benutzer des Dienstes auch im Verzeichnis eingetragen sind, können die im X.509 [5] definierten Authentifizierunsmechanismen nicht für die Autorisierung des Anfragenden herangezogen werden.



A = DN des Benutzers
pw = Paßwort von A
t = Zeitangabe
q = Zufallszahl (optional)
f = gerichtete Funktion

Abb. 1: Einfache geschützte Authentifizierung

Beim AMBIX-DSA sind daher die Zugriffsrechte so eingestellt, daß Personendaten prinzipiell nur an das AMBIX-TWEB weitergegeben werden, welches sich beim DSA durch Angabe von Paßwort und DN mittels *protected simple bind* authentifiziert (siehe Abb. 1).

Hierbei werden Paßwort und DN eines für TWEB eingerichteten Eintrags zusammen mit einer Zufallszahl (q^1) und/oder einer Zeitangabe (t^1) mittels einer gerichteten Funktion (*f1*) geschützt. Die so geschützte Information kann sofort übertragen werden (einfach geschützt) oder vor der Übertragung mittels einer weiteren gerichteten Funktion (*f2*) sowie einer weiteren Zeitangabe (t^2) und/oder Zufallszahl (q^2) geschützt werden (doppelt geschützt).

Mit einer Anfrage beim *Domain Name Service* (DNS) ermittelt das TWEB die Top-Level Domain des abfragenden Rechners. Ist diese nicht in der Liste der erlaubten Top-Level Domains aufgeführt bzw. existiert kein DNS Eintrag, werden keine Personendaten angezeigt. In diesem Fall werden, wie auch bei der Anfrage anderer DUAs oder DSAs, nur Organisationsdaten herausgegeben und angezeigt.

Neben der Möglichkeit *robots exclusion protocol* (REP), konforme Suchroboter vom Zugriff auf die Daten auszuschließen, werden von TWEB weitere Mechanismen zur Zugriffskontrolle bereitgestellt. So lassen sich für ein beliebiges Zeitintervall die Anzahl der Zugriffe aus einem definierbaren IP-Adressbereich aufzeichnen. Werden dabei regelmäßige oder besonders häufige Zugriffe registriert, wird dieser Bereich automatisch für eine gewisse Zeit gesperrt, hierbei sind IP-Adressbereich, Zahl der Zugriffe und die Zeitintervalle frei konfigurierbar.

Schließlich können durch einen entsprechenden Eintrag in der Konfigurationsdatei einzelne Rechner oder Domains vom Zugriff auf Personendaten oder gänzlich vom Zugriff auf die AMBIX-Daten ausgeschlossen werden.

Bei Datenlieferungen von Organisationen und Benutzern muß deren Authentizität sichergestellt sein. Für alle am Projekt beteiligten Organisationen wurde mindestens ein Administrator bestimmt und diesem ein Paßwort übermittelt. E-Mail-Adresse und Paßwort sind im DS-Manager-Eintrag der betreffenden Organisation im X.500 abgelegt. Datenlieferungen werden nur dann akzeptiert, wenn am Anfang der Datei die E-Mail-Adresse und das Paßwort des Administrators angegeben wurden und diese Angaben

mit den im X.500 gespeicherten übereinstimmen.

Im Verzeichnis eingetragene Personen erhalten an die registrierte E-Mail-Adresse ein Datenänderungsformular zugesandt, welches eine mit einer komplexen Prüfsumme versehene Personenidentifikationsnummer (PIN) enthält. Datenänderungen sind für die Benutzer nur mit einem durch diese PIN authentifizierten Formular möglich. Um einen Mißbrauch des Formulars zu verhindern, werden die Daten soweit möglich auf formaler Konsistenz geprüft sowie bestimmte Einschränkungen realisiert, z.B. ist eine gleichzeitige Änderung von Vor- und Nachname nur über den Administrator möglich. Durch zurückschicken einer Mail mit den aktuellen Daten an alle gespeicherten E-Mail-Adressen sowie der automatischen Auswertung der Fehlerrückmeldungen wird die Funktionsfähigkeit der Adressen verifiziert. Aufgrund der Transparenz der Vorgänge und der Information der einzelnen Benutzer wird eine zusätzliche Sicherheit gewährleistet.

Sowohl bei den Organisationslieferungen als auch bei den Benutzer-Formularen ist für die Zukunft eine sicherere Kommunikation durch Verschlüsselung und Authentifizierung der Daten mittels eines asymmetrischen Verfahrens (vgl. nächstes Kapitel) vorgesehen. Die Einrichtung der dafür erforderlichen Infrastruktur ist in Vorbereitung.

Einen gewissen Schwachpunkt bildet die Möglichkeit des Selbsteintrags, die erst geschaffen wurde, nachdem sich herausgestellt hatte, daß nur ein kleiner Teil der angeschriebenen Organisationen zur aktiven Lieferung von Personendaten bereit war. Die Selbsteinträgerdaten können vom Projekt nur formal geprüft werden, werden jedoch vor dem Eintrag ins Verzeichnis den zuständigen Administratoren zur inhaltlichen Überprüfung geschickt.

Nicht zuletzt müssen direkte Angriffe auf die AMBIX-Rechner als unberechtigter Zugriff auf die Personendaten in Betracht gezogen und abgewehrt werden. Da für die verwendete Plattform (HP-UX) kein *Secure-RPC* zur Verfügung steht, wurde für die RPC-Schnittstelle ein eigener Authentifizierungsmechanismus implementiert. Alle Zugriffe auf TCP- und RPC-Dienste werden überwacht, protokolliert (*TCPwrapper* bzw. *portmapper*) und ausschließlich den projekteigenen Rechnern gewährt. Um ein sicheres Login übers Netz zu gewährleisten, wird *Secure Shell* (ssh) eingesetzt, welches eine verschlüsselte Datenübertagung und eine Authentifizierung mit dem RSA-Algorithmus [6] ermöglicht.

Die Quellen der oben erwähnten Programme können z.B. unter ftp://ftp.cert.dfn.de/pub/ gefunden werden.

AMBIX und Verschlüsselungstechnik

In der elektronischen Kommunikation ist ein zunehmender Einsatz von kryptographischen Verfahren zu verzeichnen. Hier sind vor allem *public key cryprosystems* (PKCS) zu nennen.

Bei PKCS wird für die Verschlüsselung der Daten ein Paar von Schlüsseln benötigt, welche einem Benutzer (X) zugeordnet sind. Diese sind ein geheimer Schlüssel (X_S) und ein öffentlich zugänglicher Schlüssel (X_P) der aus X_S berechnet wird. Aus der Kenntnis von X_P kann jedoch nicht auf X_S zurückgeschlossen werden. Den öffentlichen Schlüssel können andere Benutzer zur Verschlüsselung von Daten heranziehen, die sich dann nur noch mit dem komplementären geheimen Schlüssel entschlüsseln lassen.

Solche Verfahren eignen sich auch zum Erstellen digitaler Signaturen, mit denen sowohl die Unversehrtheit des Inhalts wie auch die Urheberschaft einer digitalen Nachricht geprüft und bestätigt werden können. Auch die im X.509 definierte strenge Authentifizierung (*strong bind*) basiert auf der Anwendung eines PKCS.

Für eine sinnvolle Anwendung von PKCS ist eine vertrauenswürdige Quelle erforderlich, von der zuverlässig der öffentliche Schlüssel einer Person bezogen werden kann.

In Zusammenarbeit mit dem Projekt DFN-PCA [7] (PCA - *policy certification authority*) an der Universität Hamburg wird von AMBIX eine Infrastruktur aufgebaut mit dem Ziel eine solche Quelle zur Verfügung zustellen.

Eine Zertifizierungs-Instanz (CA - *certification authority*) stellt die Identität eines Benutzers fest, signiert dessen öffentlichen Schlüssel und erzeugt so ein mit einer Gültigkeitsdauer versehenes Zertifikat. Die für die Abbildung einer CA im X.500 benötigten Attribute werden im X.509 definiert. Ist eine CA in einer im Verzeichnis abgebildeten Organisation angesiedelt, werden diese Attribute zusätzlich zu den bereits vorhandenen Attributen eines Organisationseintrags im Verzeichnis gespeichert, ansonsten wird ein neuer Organisationseintrag für diese CA angelegt.

CAs sind hierarchisch im DIT organisiert, so daß eine weiter unten angesiedelte CA von der darüberliegenden CA zertifiziert werden kann. An der Spitze eines solchen Zertifizierungsbaumes steht eine PCA (hier DFN-PCA), welche auch die gemeinsame Sicherheitspolitik dieser CAs definiert.

Eine weitere Aufgabe der CAs ist es, ungültige oder zurückgezogene Zertifikate in regelmäßigen Abständen in sog. *certificate revocation lists* (CRLs) herauszugeben.

Zertifikate und CRLs müssen jederzeit öffentlich zugänglich sein. Dem X.500 kommt dabei eine zentrale Bedeutung zu, da anhand des DNs eine eindeutige Zugehörigkeit eines Zertifikats zu einer Person gewährleistet werden kann. Hierfür übermitteln PCA und CAs die von ihnen erstellten Zertifikate und herausgegebenen CRLs über einen sicheren Kommunikationskanal an AMBIX, wonach diese im Verzeichnis veröffentlicht werden.

Ausblick

Die DFN-PCA hat vor kurzem mit der Zertifizierung von CAs begonnen, so daß in nächster Zeit mit der Übermittlung der ersten Zertifikate zu rechnen ist.

Das Ablegen nach X.509 zertifizierter Schlüssel im Verzeichnis ist mit den dort definierten Attributen problemlos möglich. Für die Schlüssel eines weiteren augenblicklich gängigen PKCS, *pretty good privancy* (PGP), existiert derzeit noch kein standardisierter Attribut-Set, sondern nur proprietäre Definitionen, wie sie z.B. im Tübinger Telefon- und Mitarbeiterverzeichnis [8] verwendet werden.

Notwendige Änderungen und Ergänzungen der AMBIX-Software zur Verarbeitung und Darstellung der Zertifikate sind teilweise abgeschlossen bzw. in Arbeit. Hierbei wurde darauf geachtet, daß auch von der DFN-PCA für die nahe Zukunft geplante Zertifikate für *SSL*, zur Absicherung des WWW, und *S/MIME* aufgenommen werden können.

Im Augenblick wird an einem Gesetzesentwurf zur rechtlichen Regelung digitaler Signaturen [9] gearbeitet. Auch ist in Deutschland eine Diskussion über eine Kryptoregulierung im Gange. Inwieweit diese beiden Entwicklungen Auswirkungen auf die weitere Projektarbeit haben werden, bleibt abzuwarten.

Literatur

[1] The Directory: Overview of Concepts, Models and Services - CCITT Recommendation X.500 ISO/ICE 9594-1 / Information Technology; Open System Interconnection (1993)

[2] W. Yeong, T. Howes und S. Kille, Lightweight Directory Access Protocol, RFC 1777, Performance Systems International, Universität Michigan, ISODE Consortium, März 1995

[3] Dr. Jürgen W. Goebel und Jürgen Scheller, Datenschutzrechtliche Probleme bei der Einrichtung und dem Betrieb von X.500-Directories im Rahmen des Deutschen Forschungsnetzes, Frankfurt 1993

[4] K.-P.Gietz, Dr. R. Schneider, Dr. K. Spanier, X.500 für Alle - Das DFN-Projekt AMBIX, DFN-Mitteilungen, Heft 42, Nov. 1996, S. 23 ff.

[5] The Directory: Authentication Framework - CCITT Recommendation X.509 ISO/ICE 9594-8 / Information Technology; Open System Interconnection (1993)

[6] R.L. Riverst, A. Shamir und L.A. Adleman, Method of obtaining Digital Signatures and Public-key Cryptosystems, Communications of the ACM, Vol. 21, No. 2 (February 1978)

[7] Projekt DFN-PCA, http://www.cert.dfn.de/dfnpca/

[8] Telefon- und Mitarbeiterverzeichnis der Universität Tübingen, http://x500.uni-tuebingen.de:8889/

[9] SiG, Artikel 3, IuKDG, Gesetz zur Regelung der Rahmenbedingungen für Informations- und Kommunikationsdienste (Informations- und Kommunikationsdienste-Gesetz) BT-Drs. 13/7934 vom 11.06.1997

Kontakt:

Universität Tübingen
Zentrum für Datenverarbeitung
DFN-Projekt AMBIX
Brunnenstraße 27
72074 Tübingen
Tel. 07071 29-77539
Fax. 07071 26-5912
E-Mail: ambix-d@mail500.uni-tuebingen.de
URL: http://ambix.uni-tuebingen.de/

Pretty Good Privacy

Bernd Lehle / Oliver Reutter

Beim Umgang mit E-Mail (elektronischer Post) mag bei vielen Benutzern das Bild von der Gelben Post im Hinterkopf auftauchen: Man schreibt einen Brief, steckt ihn in einen Umschlag, übergibt ihn dem verantwortlichen System und das sorgt dann dafür, daß er unversehrt beim Empfänger ankommt, der ihn öffnet und liest. In der Bundesrepublik Deutschland ist das Postgeheimnis von der Verfassung garantiert, und darf nur in speziellen Fällen und nach richterlichem Beschluß gebrochen werden. In der Welt der elektronischen Postzustellung sieht es leider etwas anders aus: Eine E-Mail Message ist nicht geheimer als eine Urlaubspostkarte und der eigene Briefkasten nicht sicherer als der eigene Mülleimer. Was man tun kann, um sein Postgeheimnis hier soweit zu sichern, daß nicht einmal der Staatsanwalt herankommt, beschreibt dieser Artikel.

Jeder Benutzer, der sich in dem Metier auskennt, wird sofort wissen, daß hier nur Verschlüsselung hilft. Wer heute im Internet Daten transportieren will, die nicht jeder lesen soll, macht sich erst gar keine Gedanken, ob die Leitungen sicher sind oder nicht: Er geht von unsicheren Leitungen aus und verschlüsselt seine Daten.

Diese Disziplin war bis vor einigen Jahren noch den hehren Kreisen der Mathematiker und Geheimdienste vorbehalten. Mit dem Auftauchen von Public Key-Verfahren und einfacher Software, die sie schnell, portabel und bedienerfreundlich implementiert, kann heute allerdings jeder seine Daten so verschlüsseln, daß sich selbst die Supercomputer der amerikanischen Oberschnüffler NSA (National Security Agency) daran die Zähne ausbeißen würden.

Der Prophet dieser neuen Privatsphäre im Internet ist der Amerikaner Phil Zimmermann, der das Programm PGP (**P**retty **G**ood **P**rivacy) entwikkelte, das heute auf allen gängigen Plattformen public domain erhältlich ist.

Wir wollen hier erst einen kleinen Einblick in die Grundlagen der Public Key-Verschlüsselung geben und dann am Beispiel von PGP die Benutzung detailliert darstellen.

Das von PGP verwendete Public Key-Verfahren heißt RSA, benannt nach seinen Entwicklern Ron Rivest, Adi Shamir und Len Adleman (MIT, 1977).

Die Grundlage von RSA bilden zwei sehr großen Primzahlen *p* und *q*. Zudem benötigt man noch den Encryption Key *e,* der eine beliebige Zahl sein kann, die kein Teiler von

$$(p-1)\cdot(q-1)$$

ist. Typischerweise ist es eine kleine Primzahl, aus der dann der Decryption Key *d* wie folgt über den Euklidischen Algorithmus berechnet wird:

$$d\cdot e = 1 \bmod ((p-1)\cdot(q-1))$$

So hängt also *d* mit *e* über *p* und *q* zusammen. Aus diesen Zahlen berechnet man dann das Produkt *n* = *pq*; *p* und *q* werden dann vergessen.

Verschlüsselt wird eine Zahl (oder ein Buchstabe) *m* nun mit: $c = m^e \bmod n$

Entschlüsselt wird mit: $m = c^d \bmod n$

Dies heißt, was mit *e* verschlüsselt wurde, kann nur mit *d* wieder entschlüsselt werden und umgekehrt. Man spricht - im Gegensatz zu den symmetrischen, die für Ver-/Entschlüsselung denselben Schlüssel benutzen - von einem asymmetrischen Verfahren. Die Kombinationen (*e,n*) und (*d,n*) sind die Schlüssel. Der Einwand, daß *d* jederzeit aus *e* wieder berechnet werden kann greift nur, wenn *p* und *q* bekannt sind. Die Schlüssel enthalten aber nur deren Produkt *n* und damit ist das Knacken des Schlüssels nur über die Faktorisierung des Produkts *n* möglich, was bei genügend großen Primzahlen selbst auf Supercomputern Monate bis Jahre dauern kann.

Beim Public Key-Verfahren wird einer der beiden Schlüssel (*e,n)* oder (*d,n)* veröffentlicht und allen Kommunikationspartnern zur Verfügung gestellt. Der andere wird als geheimer Schlüssel sicher verwahrt. Hierbei wird die Tatsache ausgenutzt, daß beide Schlüssel nur kombiniert ver-/entschlüsseln können. Um jemandem eine geheime Botschaft zukommen zu lassen, verschlüsselt man sie mit dem öffentlichen Schlüssel des Partners. Der einzige Schlüssel, der dies nun wieder entschlüsseln kann ist der private des Empfän-

gers. Die einmal verschlüsselte Nachricht kann man selbst nicht wieder entschlüsseln.

Neben dem Verschlüsseln bietet dieses Verfahren auch die Möglichkeit der digitalen Unterschrift. Dazu wird aus dem zu unterschreibenden Textdokument eine Prüfsumme gebildet und mit dem privaten Schlüssel verschlüsselt; sie wird an den Text gehängt und mitverschickt. Der Empfänger kann nun mit dem öffentlichen Schlüssel des Absenders die Unterschrift entschlüsseln und die erhaltene Prüfsumme mit einer selbst berechneten vergleichen. Wurde die Nachricht verändert, stimmen die Prüfsummen nicht mehr überein. Will ein Fälscher die Nachricht *und* die Unterschrift fälschen, braucht er dazu den privaten Schlüssel des Absenders, da ohne ihn aus der gefälschten Prüfsumme keine gültige Unterschrift erzeugt werden kann. Die Prüfsummen werden mit dem sogenannten MD 5-Verfahren erstellt, das es nicht zuläßt, die Prüfsumme gezielt zu verändern.

Wenn man davon ausgeht, daß private Schlüssel wirklich geheim und bewacht sind, sind die beschriebenen Verfahren bei genügend großen Schlüsseln gegen alle Knack-Verfahren in menschlichen Zeiträumen immun. Man spricht daher von starker Kryptographie.

Löcher, die gleich am Beispiel von PGP noch etwas genauer beschrieben werden, sind in diesem Verfahren natürlich auch vorhanden. Schwierig ist es beispielsweise auf einem Mehrbenutzersystem wie UNIX, den privaten Schlüssel auf dem Dateisystem wirklich geheim zu halten, vor allem vor dem Systembetreuer. Deshalb werden diese Schlüssel auch noch durch ein verbessertes Paßwort, eine sogenannte Paßwortphrase beliebiger Länge geschützt. Sie ist jedoch bei der Tastatureingabe wieder den Spähversuchen böswilliger Spione ausgesetzt. Eine Alternative wäre, den Schlüssel nur auf einem nicht vernetzten Einplatz-Rechner, auf den Nachrichten aber per Diskette übertragen werden müssen, zu halten.

Ein anderes Problem ist der sogenannte Man in the Middle Attack. Hier spiegelt jemand zwischen den beiden Kommunikationspartnern an beiden Enden vor, er wäre das vermutete Gegenüber. Er gibt beiden seinen eigenen öffentlichen Schlüssel und behauptet, es wäre der des Gegenübers. So kann er alle oben beschriebenen Verfahren aushebeln, indem er beide Partner täuscht. Hier hilft nur, daß die beiden Kommunikationspartner Mittel und Wege finden, ihre öffentlichen Schlüssel ohne Veränderung auszutauschen. Dies kann durch zentrale Schlüssel-Verwaltungsstellen geschehen oder durch die Signatur öffentlicher Schlüssel wie Dokumente mit einem privaten Schlüssel. Trägt nun ein öffentlicher Schlüssel die Unterschrift einer Person, der man traut, kann auch dem Schlüssel vertraut werden, da die Unterschrift nur mit dem privaten Schlüssel dieser Person erzeugt worden sein kann.

Ein letztes Problem ist die Langsamkeit des Verfahrens. Da es sich hier um Festkomma-Arithmetik jenseits aller Registerbreiten handelt, müssen die Zahlen symbolisch verarbeitet werden, was dazu führt, daß diese Verfahren bis zu 1000 mal langsamer sind als symmetrische, wie z.B. DES oder IDEA, die in Registern arbeiten.

PGP umgeht diese Probleme, indem es nicht die Nachricht mit RSA verschlüsselt, sondern einen 128 Bit langen Session Key. Dieser Schlüssel wird dann für ein schnelles, symmetrisches Verfahren benutzt, das die Nachricht verschlüsselt. Hier kommt allerdings nicht das in den USA entwickelte DES zum Einsatz, sondern das schweizerische IDEA.

Noch ein paar Worte zu den rechtlichen Problemen, die ständig im Zusammenhang mit PGP auftreten: Zum einen gibt es patentrechtliche Probleme, weil das IDEA-Verfahren von der schweizerischen Firma Ascom in Solothurn patentiert ist und bei kommerzieller Verwendung von PGP lizenziert werden muß. An Universitäten und von Privatpersonen darf es allerdings frei benutzt werden.

Ein weiteres Problem sind nationalstaatliche Beschränkungen von starken Kryptographie-Verfahren. So darf Kryptographie-Software mit Schlüsseln von über 64 Bit Länge beispielsweise nicht aus den USA exportiert werden. Wer es dennoch tut, macht sich des Schmuggelns von Munition schuldig. Würde PGP in den USA auf einen ftp-Server gelegt, läge derselbe Tatbestand vor wie bei der Mitführung einer Kiste Patronen im Fluggepäck!

Um niemanden in diese Gefahr zu bringen, gibt es eine amerikanische und eine internationale Version von PGP. Das RSA-Verfahren an sich ist publiziertes mathematisches Grundlagenwissen und daher nicht einschränkbar, d.h. jeder mathematisch versierte Programmierer kann es nachbauen. So z.B. Stale Schumacher in Norwegen, der die internationale Version von PGP erstellt. Beiden Versionen sind völlig identisch in der Leistungsfähigkeit. Die internationale Version darf allerdings nicht in den USA verwendet werden, weil sie RSA-Algorithmen enthält, die dort patentiert, aber außerhalb der USA nicht anerkannt sind. Damit das amerikanische PGP frei verteilt

werden kann, wird eine spezielle lizenzfreie Version der RSA-Algorithmen namens RSAREF verwendet. Alles klar?

Manche Länder, wie Frankreich oder Rußland, verbieten ihrem Volk grundsätzlich die Verwendung starker Kryptographie, wodurch die Verwendung von PGP dort illegal ist, außer man holt sich bei einer staatlichen Stelle eine Genehmigung. Da es aber derzeit keine staatlichen Stellen mit genügend Personal und Know-how gibt, um die Sache zu beurteilen, ist diese Angelegenheit momentan eher als Witz zu betrachten.

Auf europäischer Ebene laufen ständige Verhandlungen zu einer einheitlichen Regelung der Kryptographie in der EU. Die Ergebnisse sind allerdings ziemlich diffus, da die beauftragten Kommissionen es oft am Blick für die Realität missen lassen. In Deutschland ist die Verwendung aller PGP-Versionen legal und auch schon relativ weit verbreitet, so daß viele Partner zur vertraulichen Kommunikation bereitstehen.

Technische Details der PGP-Verwendung

PGP kann man z.B. über `ftp.uni-stuttgart.de:/pub/unix/security` bekommen.

Obwohl im UNIX-Pfad, ist es dort für alle gängigen Plattformen - aktuelle Version (Stand April 96) 2.6.3i - verfügbar. Sämtliche 2.6-Versionen sind untereinander kompatibel. Die Version 2.3 sollte allerdings nicht mehr verwendet werden. In `/sw` ist die Version 2.6i bzw. 2.6.2i für einige Plattformen installiert. Die Endung i kennzeichnet die internationale Version.

Wer PGP für eine exotische Plattform haben will, muß u.U. den Schlüssel etwas patchen, hat aber meist Erfolg, da das Programm einfach geschrieben ist und eigentlich nur rechnet sowie Dateien liest bzw. anlegt. Wir empfehlen die englischsprachige Version, weil die deutsche leicht bizarr übersetzt ist. Da das Programm ursprünglich auf MS-DOS entwickelt wurde, benimmt es sich unter UNIX manchmal etwas daneben (Anlegen von Dateien namens `stdin`).

Hat man PGP nun auf seinem System installiert, sollte die erste Aktion das Erstellen eines Schlüsselpaares sein, ohne das man in der PGP-Welt ein Niemand ist.

Wer auf die Devise *Real Hackers don't read Manuals* vertraut, dem seien hier noch schnell zwei Optionen nahegelegt:

1. `-h` bietet allgemeine Hilfe
2. `-k` bietet Hilfe zur Schlüsselverwaltung.

Ansonsten viel Spaß!

Für alle Nicht-Hacker folgt hier eine detailliertere Erklärung.

Zuerst muß ein Schlüsselpaar erzeugt werden:

```
# pgp -kg

Pretty Good Privacy(tm)2.6.2i-Public-key encryption for the masses. (c) 1990-1995
Philip Zimmermann, Phil's Pretty Good Software. 7 May 95 International version - not
for use in the USA.Does not use RSAREF . Current time: 1996/03/25 15:14 GMT
Pick your RSA key size:
1) 512 bits- Low commercial grade, fast but less secure
2) 768 bits- High commercial grade, medium speed, good security
3) 1024 bits- „Military" grade, slow, highest security Choose 1, 2, or 3, or enter desired number of bits:
```

Ohne als Militarist zu gelten, kann ruhig ein 1024 Bit-Schlüssel gewählt werden. `slow` ist unerheblich, weil, wie bereits erwähnt, mit RSA nur minimale Datenmengen verschlüsselt werden. 512 Bit sind zu kurz, da bereits ein 429 Bit Key durch einen Workstation-Verbund von einigen tausend Rechnern geknackt wurde. Falls Sie den begründeten Verdacht haben, daß Sie von einer Institution ausgehorcht werden, die über starke Supercomputer verfügt, können Sie an dieser Stelle je nach Version auch höhere Schlüssellängen (bis 2048 Bit) angeben. Die Knack-Dauer eines Schlüssels steigt exponentiell mit dessen Länge. Genauere Informationen dazu stehen in den FAQs am Artikelende. Mit Schlüssellänge ist hier die Zahl *n* gemeint.

```
Choose 1, 2, or 3, or enter desired number of bits: 3
Generating an RSA key with a 1024-bit modulus.
You need a user ID for your public key. The desired form for this user ID is your name, followed by
your E-mail address enclosed in <angle brackets>,  if you have an E-mail address.
```

```
For example:

John Q. Smith <12345.6789@compuserve.com>
Enter a user ID for your public key:
```

Es bietet sich der eigene Name samt E-Mail-Adresse an:
Otto B. Nutzer <otto@uni-stuttgart.de>

```
You need a pass phrase to protect your RSA secret key. Your pass phrase can be any sentence or phrase
and may have many words, spaces, punctuation, or any other printable characters.

Enter pass phrase: Meine Oma faehrt im Huehnerstall Motorrad
Enter same pass phrase again: Meine Oma faehrt im Huehnerstall Motorrad
```

Man sollte hier unbedingt ausnutzen, daß mehr als acht Buchstaben zur Verfügung stehen. Das Eingegebene wird nicht angezeigt. Bitte verwenden Sie keine Zeichen, die Sie nicht auf jeder Tastatur wiederfinden! Insbesondere sei hier vor Umlauten gewarnt.

```
Note that key generation is a lengthy process. We need to generate 968 random bits. This is done by measuring the time intervals between your keystrokes. Please enter some random text on your keyboard until you hear the beep:
```

Geben Sie nun einen beliebigen Text ein. Dabei werden die Abstandszeiten zwischen dem Drükken der Tasten gemessen und als Zufallszahlen verwendet. Hier empfiehlt es sich, die Paßwortphrase abzutippen, damit sie sich besser einprägt. Sollte sie Ihnen nämlich entfallen, ist Ihr Schlüsselpaar wertlos und Sie müssen allen Partnern einen neuen Schlüssel geben.

```
.....................................
.....................**** ........****
Key generation completed.
```

Nun haben Sie ein Schlüsselpaar, das typischerweise in einem Verzeichnis namens `pgp` oder `.pgp` steht:

```
1 drwxr-xr-x 2 zrzv0111 zr0111 512 Mar 25 16:30 .
4 drwxr-xr-x 20 zrzv0111 zr0111 2048 Mar 25 16:30 ..
1 -rw------- 1 zrzv0111 zr0111 190 Mar 25 16:30 pubring.pgp
1 -rw------- 1 zrzv0111 zr0111 408 Mar 25 16:30 randseed.bin
2 -rw------- 1 zrzv0111 zr0111 523 Mar 25 16:30 secring.pgp
```

Hier sehen Sie den öffentlichen Schlüsselring, einen Satz Zufallszahlen und den privaten Schlüsselring. Ihr privater Schlüsselring steht allerdings nicht im Klartext auf der Platte, sondern ist mit IDEA verschlüsselt.

Der Schlüssel dafür ist die 128 Bit MD5-Prüfsumme Ihrer Paßwortphrase.

Den öffentlichen Schlüssel können Sie sich mit der Option `-kv` (key view) oder `-kvv` (key view verbose) anschauen:

```
#pgp -kv
Pretty Good Privacy(tm) 2.6.2i - Public-key encryption for the masses. (c) 1990-1995 Philip Zimmermann,
Phil's Pretty Good Software. 7 May 95 International version - not for use in the USA. Does not use RSAREF.
Current time: 1996/03/25 15:41 GMT
Key ring: /home/otto/.pgp/pubring.pgp'
Type bits/keyID Date User ID
pub 1024/04A5D509 1996/03/25 Otto B. Nutzer <otto@uni-stuttgart.de>
1 matching key found.
```

Momentan ist nur Ihr eigener Schlüssel enthalten, aber es kommen sicher bald mehrere hinzu. Anfügen kann man Schlüssel mit der Option `-ka` (key add). Wenn Sie eine Datei `herbert.key` haben, die einen Schlüssel enthält, geben Sie einfach nur `pgp -ka herbert.key` ein und der öffentliche Schlüssel wird angefügt. Noch leichter wird es, wenn jemand seinen öffentlichen Schlüssel in der `finger`-Information bereit hält; man kann dann die Ausgabe des `finger`-Kommando direkt mit der Option `-f` in `pgp` umleiten. Meinen Public Key bekommen Sie folgendermaßen:

```
# finger bernd@visbl.rus.uni-stuttgart.de | pgp -kaf
Pretty Good Privacy(tm) 2.6.2i - Public-key encryption for the masses. (c) 1990-1995 Philip Zimmermann,
Phil's Pretty Good Software. 7 May 95 International version - not for use in the USA. Does not use RSAREF.
Current time: 1996/03/25 15:46 GMT
```

```
Looking for new keys... pub 768/64EABDF5 1995/07/07 Bernd Lehle
<lehle@rus.uni-stuttgart.de>
Checking signatures...
Keyfile contains: 1 new key(s)
Looking for new keys... No new keys or signatures in keyfile.
```

Schauen wir uns den öffentlichen Schlüsselring nun genau (verbose) an:

```
# pgp -kvv
Pretty Good Privacy(tm) 2.6.2i - Public-key encryption for the masses. (c) 1990-1995 Philip Zimmermann,
Phil's Pretty Good Software. 7 May 95 International version - not for use in the USA. Does not use RSAREF.
Current time: 1996/03/25 15:52 GMT
Key ring: /home/otto/.pgp/pubring.pgp'
Type bits/keyID Date User ID
pub 768/64EABDF5 1995/07/07 Bernd Lehle <lehle@rus.uni-stuttgart.de>
sig 64EABDF5 Bernd Lehle <lehle@rus.uni-stuttgart.de>
sig 8E0A49D1 (Unknown signator, can't be checked) Bernd Lehle
pub 1024/04A5D509 1996/03/25 Otto B. Nutzer <otto@uni-stuttgart.de> 2 matching keys found.
```

Die erste Zeile zeigt, daß nun mein öffentlicher Schlüssel mit einer Länge von 768 Bit eingebunden ist. Somit können Sie mir geheime Nachrichten schicken. Die weiteren Zeilen fangen mit `sig` an, was bedeutet, daß mein öffentlicher Schlüssel Unterschriften trägt. Die erste Signatur ist meine eigene, d.h. der Schlüssel kann nur von der Person kommen, die den zugehörigen privaten Schlüssel hat.

Die zweite können Sie nicht lesen, da Sie den öffentlichen Schlüssel des Unterzeichnenden nicht haben; Sie brauchen ihn aber zwingend, um die Unterschrift zu entziffern, die mit seinem privaten Schlüssel erstellt wurde. Zum Nachprüfen können Sie sich den benötigten Schlüssel mit `#finger ger ley-pgp@ftp.cert.dfn.de | pgp -kaf` holen und ihn anfügen.

Der eigene öffentliche Schlüssel sollte gleich nach der Erzeugung mit `pgp -ks Nutzer` unterschrieben werden. Dies beweist, daß er jemanden gehört, der auch den geheimen Schlüssel hat, also weniger leicht gefälscht sein kann. `Nutzer` muß dabei innerhalb des Rings eindeutig sein, da sonst der erste Schlüssel genommen wird, auf den der Name paßt.

Wie benutze ich den Schlüssel?

Wenn Sie mir beispielsweise vertraulich mitteilen wollen, daß Sie ab heute PGP benutzen, müssen Sie folgende Schritte unternehmen: Sie erstellen eine Datei, die diese Nachricht enthält, nennen wir sie `hallo.txt` „`Hallo, ich habe jetzt pgp! Otto.`“ Sie wird nun mit der Option `-e` (encrypt) verschlüsselt. Damit die Nachricht danach als Mail verschickt werden kann, muß der verschlüsselte Text im ASCII-Format vorliegen, was Sie mit der Option `-a` erreichen:

```
# pgp -ea hallo.txt
```

```
Pretty Good Privacy(tm) 2.6.2i - Public-key encryption for the masses. (c) 1990-1995 Philip Zimmermann,
Phil's Pretty Good Software. 7 May 95 International version - not for use in the USA. Does not use RSAREF.
Current time: 1996/03/25 16:03 GMT
Recipients' public key(s) will be used to encrypt. A user ID is required to select
the recipient's public key.
Enter the recipient's user ID:
```

Hier ist nun ein im Schlüsselring eindeutiger Name gefragt:

```
Enter the recipient's user ID: Bernd
Key for user ID: Bernd Lehle <lehle@rus.uni-stuttgart.de> 768-bit key, Key ID
64EABDF5, created 1995/07/07
Also known as: Bernd Lehle
WARNING: Because this public key is not certified with a trusted signature, it is not
known with high confidence that this public key actually belongs to: „Bernd Lehle
<lehle@rus.uni-stuttgart.de>“.
```

Dies bedeutet, daß mein öffentlicher Schlüssel keine Unterschrift von jemandem trägt, dem Sie vertrauen - wie sollte er auch. Wenn Sie das stört, und Sie mir vertrauen, dann unterschreiben Sie meinen Schlüssel für Ihren Privatgebrauch mit `pgp -ks Bernd` einfach selbst und die Meldung taucht nicht wieder auf.

```
Are you sure you want to use this public key (y/N)? y
.
Transport armor file: hallo.txt.asc
```

Ein `transport armor` ist immer nötig, wenn Daten via Mail verschickt werden. Schauen wir einmal nach, was aus dem Text geworden ist:

```
-----BEGIN PGP MESSAGE----- Version: 2.6.2i
hGwDIEeAeWTqvfUBAv9WLqNOmTkOHPJD9VpMjkRabYLTSrSuMTKZosMcIRJZLbwn2kWD3yqO2cVafQN21mx
xaY5oSQ1cfs2eeq8tRIlDhfb4m1vAxdx91Qj0uZJgbeGlhKbfjai5url5Xx6U9ymAAAAPClWstI+ZOGi7Xb
ryzWn2+J+buKKrLXS5Tr0XhmJ1IZwnjVnNzVuxNIliuOF169bPFGyq4j2mI5YqoEg== =uhlI
-----END PGP MESSAGE-----
```

Nicht wiederzuerkennen. Die BEGIN- und END-Zeilen sowie die Versionsnummer dürfen nicht verändert werden, da sonst eine Entschlüsselung unmöglich wird.

Dieses Ungetüm verschicken Sie mir nun per Mail.

Sobald es bei mir ankommt, muß ich es entweder abspeichern oder es direkt per Pipeline durch pgp ohne Argumente schicken. Das Programm erkennt an der BEGIN-Zeile selbst, was es mit der Nachricht zu tun hat. Angenommen, ich speichere es unter `mail.otto`, geht es folgendermaßen weiter:

```
# pgp mail.otto
Pretty Good Privacy(tm) 2.6.3i - Public-key encryption for the masses. (c) 1990-96 Philip Zimmermann,
Phil's Pretty Good Software. 1996-01-18 International version - not for use in the USA. Does not use RSA-
REF. Current time: 1996/03/25 16:22 GMT
File is encrypted. Secret key is required to read it. Key for user ID: Bernd Lehle
<lehle@rus.uni-stuttgart.de> 768-bit key, key ID 64EABDF5, created 1995/07/07 Also
known as: Bernd Lehle
You need a pass phrase to unlock your RSA secret key. Enter pass phrase:
```

Da mein privater Schlüssel gefragt ist, muß ich ihn durch die Paßwortphrase freigeben:

```
Enter pass phrase: Pass phrase is good. Just a moment......
Plaintext filename: mail.otto Output file ‚mail.otto' already exists. Overwrite (y/N)? y
#
```

Dies ist eine Marotte von PGP. Es könnte den Text einfach anzeigen, aber es will unbedingt einen File schreiben. Nun habe ich die ursprüngliche Botschaft in `mail.otto` und kann sie lesen. Direkt anzeigen kann man den Text mit `pgp -m` (more).

Jetzt will ich auch auf die freudige Botschaft antworten. Dabei reicht es mir aber, daß die Antwort signiert ist, sie muß nicht verschlüsselt sein. Dazu schreibe ich die Antwort direkt in `mail.otto` und signiere sie mit der Option `-as`(ASCII signature):

```
# pgp -as mail.otto
Pretty Good Privacy(tm) 2.6.3i - Public-key encryption for the masses. (c) 1990-96 Philip Zimmermann,
Phil's Pretty Good Software. 1996-01-18 International version - not for use in the USA. Does not use RSA-
REF. Current time: 1996/03/25 16:30 GMT
A secret key is required to make a signature. You specified no user ID to select your secret key, so
the default user ID and key will be the most recently added key on your secret keyring.
You need a pass phrase to unlock your RSA secret key.
Key for user ID: Bernd Lehle <lehle@rus.uni-stuttgart.de> 768-bit key, key ID 64EABDF5, created 1995/07/07
Enter pass phrase:
```

Da zum Unterschreiben der private Schlüssel benötigt wird, muß er auch hier erst mittels Paßwortphrase freigegeben werden:

```
Enter pass phrase: Pass phrase is good. Just a moment....
Output file ‚mail.otto.asc' already exists. Overwrite (y/N)? y
Transport armor file: mail.otto.asc
#
```

Schauen wir nach, was wir erzeugt haben:

```
-----BEGIN PGP SIGNED MESSAGE-----
> Hallo, ich hab jetzt pgp ! Otto.
Freut mich ! Bernd


-----BEGIN PGP SIGNATURE----- Version: 2.6.3i Charset: ascii
iQB1AwUBMVbLuiBHgHlk6r31AQGZmAMAoBdhjemZy3YCWmL6SCSdOe6+kjsdf8J934fFkikhsddkuCQl4FL
HGVFshmjE4VnmZBn5UfGIEPhPClcHlJ3vgCJO87dfh339jhxA5TYXYQI8LnMDZlV7xjRUVDonHGL+bOr0cR
5mC =4e30
-----END PGP SIGNATURE-----
```

Hier tauchen wieder die BEGIN- und END-Zeilen auf. Diesmal ist aber auch die Unterschrift unten angefügt, die die Authentizität des Textes garantiert. Sie besteht aus einer mit dem geheimen Schlüssel verschlüsselten Prüfsumme des Textes zwischen den Markierungszeilen.

Anmerkung:
Normalerweise wird bei pgp -as der Text auch noch Base 64-codiert, um einen Verlust von Umlauten zu verhindern, der den Text verfälschen würde. Dies ist nicht immer erwünscht; man kann es abwenden, indem man pgp -as +clearsig=on explizit angibt oder in der Konfigurationsdatei config.txt die Zeile ClearSig = on einträgt.

Wenn nun diese Antwort empfangen wird, kann man sie wieder abspeichern und ebenfalls einfach an pgp übergeben:

```
# pgp answer.bernd

Pretty Good Privacy(tm) 2.6.2i - Public-key encryption for the masses. (c) 1990-1995 Philip Zimmermann,
Phil's Pretty Good Software. 7 May 95 International version - not for use in the USA. Does not use RSAREF.
Current time: 1996/03/25 16:47 GMT
Warning: Unrecognized ASCII armor header label „Charset:“ ignored.
File has signature. Public key is required to check signature. . Good signature from
user „Bernd Lehle <lehle@rus.uni-stuttgart.de>“.Signature made 1996/03/25 16:44 GMT
```

Die Unterschrift ist also in Ordnung. Dies kann PGP allerdings nur feststellen, wenn es meinen öffentlichen Schlüssel hat. Wenn Sie eine Nachricht von jemand erhalten, dessen öffentlichen Schlüssel Sie nicht haben, müssen Sie sich bemühen, diesen auf möglichst vertrauenswürdigem Weg zu erhalten (möglichst mit Unterschriften von Leuten, denen Sie trauen). Erst wenn er an Ihrem öffentlichen Schlüsselring hängt, können Sie die Unterschrift prüfen.

```
Plaintext filename: answer.bernd Output file
‚answer.bernd' already exists. Overwrite (y/N)? y
```

answer.bernd enthält nun den Text ohne Steuerzeilen.

Ich hoffe, daß das Prinzip und die Anwendung jetzt verständlicher geworden sind.

Zum Abwenden weiterer Unklarheiten nachfolgend einige Frequently Asked Questions.

Q: Was mache ich, wenn ich an mehrere Empfänger dieselbe Mail verschicken will?
A: pgp -ea text.file User1 User2 User3 ...

Q: Kann ich Dateien auch zum Eigengebrauch verschlüsseln?
A: Ja, mit pgp -e file Otto durch den eigenen öffentlichen Schlüssel oder pgp -c file. Bei pgp -c wird eine neue Paßwortphrase abgefragt und die Datei dann mit IDEA verschlüsselt.

Q: Muß ich immer alles abspeichern und dann die Dateien bearbeiten? Geht das nicht einfacher?
A: Das kommt auf die benutzte Mail Software an. Wer unter UNIX zum Schreiben von Mails einen Editor benutzt, der Makros zuläßt (z.B. vi und emacs), kann PGP-Makros auf Funktionstasten legen. Bei manchen Programmen mit graphischer Oberfläche (zmail für UNIX, einige POP Mail Clients für Windows und Mac) ist ein PGP-Knopf vorhanden, der dann PGP entsprechend aufruft. Eine Version von elm, die PGP verwenden kann, ist in Arbeit. Bis dahin kann man bei elm zum Entschlüsseln und Unterschrift prüfen einfach | pgp eingeben. Pine bietet diese Funktionalität leider nicht.

Q: Woher kommt config.txt?
A: Es ist Bestandteil des Paketes. Wer /sw verwendet findet es unter:
/sw/<platform>/pgp-<Version>/doc/config.txt

Q: Woher bekomme ich öffentliche Schlüssel? Wie verteile ich meinen?
A: Viele Benutzer bieten sie per finger in ihrem .plan an. Es gibt aber auch sogenannte Public Key Server, in denen man Schlüssel suchen oder deponieren kann. Der deutsche liegt unter http://www.de.pgp.net/pgp/ im Web.
Per Mail ist er unter pgp-public-keys@keys.de.pgp.net erreichbar. Einfach als Subject der Mail HELP angeben.

Q: Was kann ich falsch machen?
A: Die Paßwortphrase vergessen, den privaten Schlüssel verlieren, einen zu kurzen Schlüssel (<512 Bit) nehmen, Verschlüsselung übertreiben, Verschlüsselung nicht ernst nehmen. Es macht zum Beispiel keinen Sinn, sich einen 2048 Bit-Schlüssel zu machen und ihn dann lesbar auf der Platte liegen zu lassen. Auch das Anhängen eines 2048 Bit-Schlüssels mit 20 Unterschriften zur besseren Verteilung an jede Mail ist nicht gerade fachmännisch, da es die Mail um 100 Zeilen länger macht.

Q: Was soll ich überhaupt verschlüsseln?

A: Jede Kommunikation, die auf dem Weg zum Empfänger abgefangen und gelesen werden kann, bei der man genau das verhindern will. Die Gründe dafür sollte jeder selbst wissen. Die Arbeitsgruppe Systemsicherheit verschlüsselt nur im Einsatzfall gegen Eindringlinge.

Q: Wie kann ich meinen privaten Schlüssel schützen?
A: Auf UNIX die Permission richtig setzen (am besten `-r--------`, `mode 400`). Die Paßwortphrase nicht aufschreiben und sicherstellen, daß sie nicht vom Terminal abgehört wird.

Q: Was ist eine Key-ID?
A: Eine Art Nummer, um einen Schlüssel innerhalb eines -rings zu identifizieren. Die Wahrscheinlichkeit, daß zwei Schlüssel dieselbe Key-ID haben ist zwar sehr gering, aber nicht gleich Null. Außerdem kann man einen Key mit einer bestimmten Key-ID absichtlich erzeugen, daher darf man sie nicht als eindeutig ansehen.

Q: Was ist ein Key Finger Print?
A: Der Finger Print ist eine MD 5-Prüfsumme des Schlüssels. Daß zwei Schlüssel den selben Finger Print haben, ist äußerst unwahrscheinlich, zumal ein Finger Print auch nicht gezielt beeinflußbar ist. Der Finger Print eignet sich daher als Ausweis für einen Schlüssel und kann anstelle des Schlüssels zur Kontrolle weitergegeben werden. Ein Finger Print ist wesentlich kürzer als ein Schlüssel. Berechnen kann man den Finger Print eines Schlüssels mit `pgp -kvc <Name>`

Q: Kann ich meine Paßwortphrase ändern, wenn sie abgehört wurde? Oder meine User-ID, wenn ich eine andere Mail-Adresse bekomme?

A: Natürlich geht das: Mit `pgp -ke User [Schlüsselring]` können Sie die Paßwortphrase eines Schlüssels in dem betreffenden -ring (z.B. `secring.pgp`) ändern.
Eine neue User-ID kann nur angefügt werden. Dadurch wird verhindert, daß Unterschriften übertragen werden oder verloren gehen.

Q: Wie sicher sind die Schlüssel? Warum ist der Session Key nur so kurz?
A: Die Sicherheit eines Schlüssels hängt von den Angriffsmethoden ab, die das Verschlüsselungsverfahren bietet. RSA ist durch Faktorisierung des Produkts der beiden Primzahlen angreifbar, IDEA bisher nur durch Probieren aller Möglichkeiten. Da dieses aber ungleich mehr Rechenaufwand erfordert, können IDEA-Schlüssel bei gleicher Sicherheit wesentlich kürzer sein.
Zum Brechen eines 512 Bit RSA-Schlüssels werden etwa 30 000 MIPS-Jahre veranschlagt (ein Prozessor, der 30 Milliarden Instruktionen pro Sekunde leistet, rechnet daran ein Jahr). Ein vergleichbar sicherer IDEA-Schlüssel muß dafür nur 64 Bit lang sein. Bei einem doppelt so langen 1024 Bit RSA-Schlüssel würde dieselbe Maschine bereits 10 Millionen Jahre rechnen. Der entsprechende IDEA-Schlüssel käme auf etwa 100 Bit. Der größte RSA-Schlüssel, den PGP anbietet, hat 2048 Bit. Um ihn zu brechen kann man einen weiteren Faktor von einer Milliarde hinzurechnen. Etwa so sicher ist der von PGP verwendete 128 Bit IDEA Session Key. Diese Zahlen können sich aber bei Entdeckung neuer Verfahren drastisch ändern.Der bereits geknackte 429 Bit RSA-Schlüssel benötigte etwa 5000 MIPS-Jahre.

Q: Wie komme ich an mehr Informationen?
A: Im O'Reilly-Verlag ist ein gutes Buch von Simpson Garfinkel erschienen: *PGP*.
Wer Online-Information bevorzugt, sollte die dem PGP beigefügten Dokumentation lesen oder sich unter
`http://www.uni-mannheim.de/studorg/gahg/PGP/HANDBUCH/`
informieren. Weitere Web Pages können Sie auf den großen Suchmaschinen finden, wenn Sie nach PGP suchen.
Es gibt natürlich auch News Groups zu dem Thema, z.B. `alt.security.pgp` und `sci.crypt`

Fingerprints einiger Ansprechpartner am RUS

- Sascha Buhr 2E B9 91 AE 04 8C 16 E8 67 A5 14 7F CB 7A C6 93
- Dr. Lothar Ehnis C1 BF DC 39 91 CE 40 5C 96 50 21 36 9F E5 4E 0A
- Herbert Franz 75 77 BA FE 72 A1 66 95 8E 15 8B 15 C3 0B D3 23
- Dr. Lisa Golka 05 6F 43 E2 E8 95 4B DB 7D 97 CF 68 3B DA 6E 2E
- Bernd Lehle 3E B0 35 8D 59 D5 AE AA 5A F9 60 80 9E E0 55 48
- Helmut Springer AE 42 C3 2C A1 3E 55 6D B3 AC 3C D2 F3 CF FF E7

Bernd Lehle
Lehle@rus.uni-stuttgart.de

Oliver Reutter
Oliver.Reutter@rus.uni-stuttgart.de

Universität Stuttgart

RTCA - Die RUS Test Certification Authority

Oliver Göbel

Zertifizierung von öffentlichen PGP-Schlüsseln

Zur Erprobung einer Zertifizierung von öffentlichen kryptographischen Schlüsseln hat sich am Rechenzentrum der Universität Stuttgart eine Arbeitsgruppe zum Aufbau einer zertifizierenden Instanz (CA) gebildet. Diese soll im RUS-internen Betrieb getestet werden, wobei zunächst ausschließlich öffentliche PGP-Schlüssel von Mitarbeitern des Rechenzentrums zertifiziert werden. Dieser Artikel gibt einen Überblick über die verwendeten kryptographischen Protokolle und Verfahren, beschreibt den Aufbau der RTCA und informiert über die notwendigen Schritte, die zur Erlangung eines Zertifikates durch die RTCA notwendig sind.

Einleitung

Die Benutzung elektronischer Post (E-Mail) zum Versenden von Nachrichten, stellt heute neben WWW und FTP den am häufigsten genutzten Dienst innerhalb des Internets dar. Die Menge der Nachrichten und Dokumente, die elektronisch auf Netzwerken transportiert werden, steigt ständig. Vor allem steigt der Bedarf, auch Dokumente, die nicht nur informativen, sondern vertraulichen oder gar juristisch bindendenden Charakter besitzen, über diesen Kanal zu transportieren.

Zur Zeit bieten die dazu verwendeten Kommunikationssysteme und Protokolle allerdings keine Mechanismen zum Schutz der zu übermittelnden Nachrichten. Bei der Gelben Post und der Telekom sind die Briefe und Telefonverbindungen durch das Fernmeldegeheimnis gesetzlich geschützt. Im Gegensatz dazu läßt sich bei der Verwendung der gängigen Transportprotokolle des Internets die unberechtigte Einsichtnahme oder gar Veränderung von Nachrichten durch Dritte nicht verhindern. Ohne zusätzliche Sicherheitsmaßnahmen sind solche Dienste für den Transport sensibler Daten ungeeignet, da sie den Kommunikationspartnern abverlangen, dieses Risiko einzugehen.

Seit einiger Zeit finden Verschlüsselungsprogramme wie PGP im Internet eine zunehmende Verbreitung. Sie bieten durch Verschlüsselung und digitale Unterschrift die Möglichkeit, die geforderte Sicherheit zu implementieren. Wie nachfolgend beschrieben, basieren sie auf der sogenannten Public Key-Kyptographie.

Problematisch bei der Anwendung von Public-Key-Verfahren ist jedoch, daß a priori keine eindeutige Bindung zwischen einem Public Key und dem jeweiligen Benutzer besteht: es gibt keine technische Möglichkeit, festzustellen, ob ein bestimmter Public Key wirklich zu einem Benutzer gehört oder nicht. Soll dennoch die Authentizität eines Schlüssels nachgewiesen werden, werden digitale Zertifikate benötigt, die - von Zertifizierungsinstanzen erteilt - die eindeutige Zuordnung zwischen einem kryptographischen Schlüssel und einer bestimmten Person bestätigen.

Öffentliche Schlüssel können von Benutzern mißbraucht werden, da ein Schlüssel unter falschem Namen erzeugt werden kann. Durch die Erteilung eines Zertifikates ist sichergestellt, daß der Benutzer sein Eigentum am Schlüssel bewiesen hat. Ein Mißbrauch ist dadurch ausgeschlossen.

Kryptographie

Begriffserklärungen

Um die Verständlichkeit zu erhöhen, sollen zunächst einige Begriffe erklärt werden.

Kryptographie ist die Wissenschaft, die sich mit der Absicherung vor Veränderung und unbefugter Einsichtnahme von Nachrichten und Daten befaßt. Eine Nachricht besteht aus **Klartext**. Das Verfahren, eine Nachricht unverständlich zu machen, um ihren Inhalt geheim zu halten, wird als **Verschlüsselung** (auch Chiffrierung) bezeichnet. Eine verschlüsselte Nachricht heißt **Kryptogramm** oder **Chiffretext**. Das Verfahren, das die Umwandlung des Kryptogrammes in den Klartext bewerkstelligt, wird **Entschlüsselung** (auch Dechiffrierung) genannt.

Historisch können Kryptographieverfahren unterschieden werden nach Verfahren, deren Sicherheit auf die Geheimhaltung des Verschüsselungsprinzips angewiesen ist und - vornehmlich neueren - Algorithmen, deren Sicherheit auf der Geheimhaltung eines Schlüssels beruht. Bei den letzteren wird meist der Algorithmus veröffentlicht, was die Sicherheit durch eine öffentliche Überprüfung sogar noch erhöht.

Ein Beispiel für die erste Kategorie ist die sogenannte Caesar-Chiffrierung, bei der jedes Zeichen im Alphabet drei Positionen weiter nach rechts modulo 26 verschoben wird (d. h., daß A auf D, B auf E, ..., X auf A, Y auf B und Z auf C abgebildet wird). Es ist ein sehr einfacher Algorithmus, den schon die Römer benutzten. Er ist allerdings wertlos, sobald jemand über das Prinzip Bescheid weiß. Man kann die Vorschrift der Verschlüsselung beliebig verkomplizieren, solange man aber auf ihre Geheimhaltung angewiesen ist, erscheint diese Art der Kryptographie im allgemeinen für den heutigen Einsatz weitgehend uninteressant. Für einfache Verschlüsselungsanwendungen werden solche Verfahren allerdings auch heute noch verwendet.

Hier setzt nun die schlüsselbasierte Kryptographie an, die Berechnungsvorschriften mit einer Variablen, dem Schlüssel, benutzt. Wird der **Schlüsselraum** (die Anzahl der möglichen Schlüssel) groß genug gewählt, kann der Algorithmus bedenkenlos bekanntgegeben werden.

Verwendet man z.B. einen Schlüssel, der eine Länge von 128 Bit hat und läßt alle Permutationen dieses Musters (alle Zahlen zwischen 2^0 und 2^{128}-1) als gültigen Schlüssel zu, so wäre der Schlüssel (Zufallstreffer und eventuelle Schwächen des Algorithmus ausgenommen) nicht in einer Zeit zu brechen, in der die Kenntnis der verschlüsselten Daten von Nutzen wäre. Wenn man das Entschlüsseln des Textes mit einem der möglichen Schlüssel durch eine Operation ausführen könnte und eine Million Operationen pro Sekunde und Prozessor annähme und dann eine Million Prozessoren verwendete, um das Kryptogramm mit allen möglichen Schlüsseln probeweise zu entschlüsseln (sog. Brute-Force-Angriff), bräuchte man immer noch über 10^{19} Jahre, um diese Aufgabe zu lösen. Selbst wenn man den Schlüssel innerhalb eines Tausenstels dieser Zeit (statistisch am wahrscheinlichsten wäre die Hälfte) finden würde, wäre immer noch eine Zeitspanne nötig, die ungefähr dem millionenfachen des Alters des Universums entspricht[1] [1].

Im folgenden werden nur noch schlüsselbasierte Kryptographieverfahren betrachtet.

Starke Kryptographie

Als starke Kryptographie bezeichnet man Algorithmen, die Kryptogramme erzeugen, deren Brechung ohne Kenntnis des Schlüssels eine Zeit in Anspruch nehmen würde, die um Größenordnungen länger ist als die Zeit, in der die verschlüsselten Daten von Nutzen sind. Wenn z. B. eine Nachricht verschickt wird, die ein Initialpaßwort enthält, so ist die Verschlüsselung dieser Nachricht nur solange notwendig, bis dieses Paßwort geändert wurde. Geht man davon aus, daß es innerhalb einer Woche geändert wird, wäre starke Kryptographie in diesem Falle ein Algorithmus, der ein Kryptogramm erzeugt, das der Brechung mindestens hundert Wochen (dies entspräche zwei Größenordnungen) standhält.

Kryptographieverfahren zerfallen im wesentlichen in zwei Hauptgruppen: symmetrische Verfahren und asymmetrische Verfahren. Bei **symmetrischen Kryptographieverfahren** existiert nur ein Schlüssel, der sowohl zum Ver- als auch zum Entschlüsseln verwendet wird.

Wenn ein Sender S einem Empfänger R (recipient) eine verschlüsselte Nachricht schicken möchte, so müssen entweder beide schon den Schlüssel besitzen oder vor der geheimen Kommunikation vereinbaren. Dazu ist dann entweder ein persönliches Treffen oder ein - wohlgemerkt unverschlüsselter - Erstkontakt nötig. Dies ist die Stelle, an der die symmetrischen Verfahren verwundbar sind. Fällt bei diesem Erstkontakt der Schlüssel in die falschen Hände, ist später die verschlüsselte Kommunikation nicht mehr sicher. Wird beispielsweise E-mail als Kanal zur Übertragung des Schlüssels benutzt, ist es für den Administrator eines Mail-Relay-Rechners, der auf dem Weg liegt, den die Nachricht nimmt, ein Leichtes, die Nachricht mit dem Schlüssel zu kopieren und die nachfolgende Kommunikation abzuhören.

Asymmetrische Verschlüsselungsverfahren basieren auf einem mathematisch zusammenhängenden Schlüsselpaar, wobei der eine zum Verschlüsseln und der andere zum Entschlüsseln verwendet wird. Hat man nur einen der bei-

[1] Laut gängiger Lehrmeinung wird das Alter des Universums auf ca. 10^{10} Jahre geschätzt.

den Schlüssel, kann man aus ihm bei ausreichender Schlüssellänge nicht den anderen berechnen. Beide Schlüssel eignen sich sowohl zum Ver- als auch zum Entschlüsseln, allerdings kann man ein Kryptogramm, das man mit einem der beiden erzeugt, nur mit dem jeweils anderen wieder entschlüsseln. Nun wird einer der beiden Schlüssel als Chiffrierschlüssel deklariert und öffentlich bekannt gegeben. Er wird daher als „**Public Key**“ (**PK**) bezeichnet. Der andere, der zur Entschlüsselung einer Nachricht verwendet wird, wird von seinem Besitzer geheim gehalten und deshalb auch „**Secret Key**“ (**SK**) genannt.

Möchte ein Sender S einem Empfänger R eine geheime Nachricht schicken, so muß er sich zunächst den Public Key des Empfängers verschaffen. Dies kann geschehen, indem entweder der Empfänger dem Sender in einem Erstkontakt seinen PK_R schickt (z. B. über E-mail) oder der Sender auf einem der zahlreichen Public-Key-Server [2,13] nach diesem Schlüssel sucht und ihn, falls vorhanden, von dort bezieht. Dann verschlüsselt S die Nachricht und schickt das Kryptogramm an R. Da nur R den zu seinem öffentlichen Schlüssel passenden geheimen Schlüssel SK_R besitzt, kann nur er mit diesem die Nachricht wieder entschlüsseln. Eine Entschlüsselung durch Dritte, die PK_R beim Erstkontakt von S und R abgefangen haben, ist nicht möglich, denn eine mit PK_R chiffrierte Nachricht kann nicht mit demselben Schlüssel wieder dechiffriert werden. Die Schwachstelle des Erstkontaktes, die symmetrische Kryptographieverfahren kennzeichnet, vermeiden asymmetrische Verfahren durch die Deklarierung des öffentlichen Schlüssels. Fällt allerdings SK_R in die falschen Hände, ist es vorbei mit der Vertraulichkeit und der Schlüssel ist **kompromittiert**.

Digitale Signaturen

Einige Public-Key-Kryptographieverfahren eignen sich sowohl zur Ver-/Entschlüsselung als auch zur Erzeugung digitaler Signaturen (z. B. RSA). Eine digitale Signatur ist eine Art Siegel, das einer Nachricht oder Datei angehängt wird und deren Unveränderheit (Integrität) sowie die Identität des Senders garantiert (letztere zumindest theoretisch, wie man im nächsten Abschnitt noch sehen wird).

Möchte ein Sender S eine Nachricht signieren, wird von dieser Nachricht M (message) mittels einer Einweg-Hash-Funktion (z. B. MD-5) [1] eine Prüfsumme erzeugt, die charakteristisch für genau diese Nachricht ist. Sie kann als eine Art Kurzfassung oder **Fingerabdruck** (**Fingerprint**) von M verstanden werden. Diese Prüfsumme ist nicht durch einfaches bit-kippen zu manipulieren, wie das z. B. bei einer UNIX-Prüfsumme möglich ist. Wird ein bit der Nachricht verändert, ändert sich die Prüfsumme gravierend und nicht ebenfalls nur um ein bit. Verändert jemand die Nachricht, paßt die ursprüngliche Prüfsumme nicht mehr mit dem veränderten Text zusammen, und dieser jemand wird nicht in der Lage sein, den Text so weiterzuverändern, daß die Prüfsumme wieder stimmt. Die Prüfsumme kann als ein sicheres Siegel der Unverfälschtheit einer Nachricht angenommen werden. Als nächstes wird diese Prüfsumme mit dem geheimen Schlüssel SK_S des Senders S verschlüsselt. Dies stellt sicher, daß M auch wirklich von S signiert ist; der Empfänger kann die Signatur nur mit dem zu SK_S passenden öffentlichen Schlüssel von S, nämlich PK_S , wieder entschlüsseln. Da aber nur S SK_S besitzt, stellt dies sicher, daß M auch nur von S unterzeichnet worden sein kann.

Probleme der Public-Key-Kryptographie

Zwar vermeiden Kryptographieverfahren mit öffentlichen Schlüsseln die Probleme des Erstkontaktes, wie sie bei symmetrischen Verfahren auftreten, doch decken sie nicht die Sicherstellung der Identität von S und R ab. Gesetzt den Fall, daß R und S sich nicht persönlich kennen, bieten diese Verfahren keinen Mechanismus an, der R die Identität von S und S die von R überprüfen läßt.

Die augenblicklich verbreitetste Implementierung, die RSA zur Verschlüsselung von Daten benützt - PGP [3] - erlaubt eine lokale Schlüsselgenerierung bei S und R. Die Identifizierung erfolgt über die Angabe des Namens sowie der E-mail-Adresse. Für einen Bösewicht B stellt es kein Problem dar, ein Schlüsselpaar $SK_{R(B)}$ und $PK_{R(B)}$ mit dem Namen von R zu erzeugen (der Index R(B) soll andeuten, daß der Schlüssel aussieht, als gehöre er R, in Wirklichkeit aber von B erzeugt wurde). Nimmt nun S mit R Kontakt auf (E-Mail), um R´s öffentlichen Schlüssel zu erhalten, so kann B die an S gerichtete Nachricht, die R‘s echten Public Key PK_R enthält, abfangen und durch eine Nachricht ersetzen, die nun stattdessen $PK_{R(B)}$ enthält (s. Fig. 1). S wird von dieser Fälschung nichts bemerken, denn selbst wenn die Nachricht signiert ist bedeutet das nur, daß R(B) (B, der sich für R ausgibt) den passenden Secret Key $SK_{R(B)}$ zu $PK_{R(B)}$ besitzt.

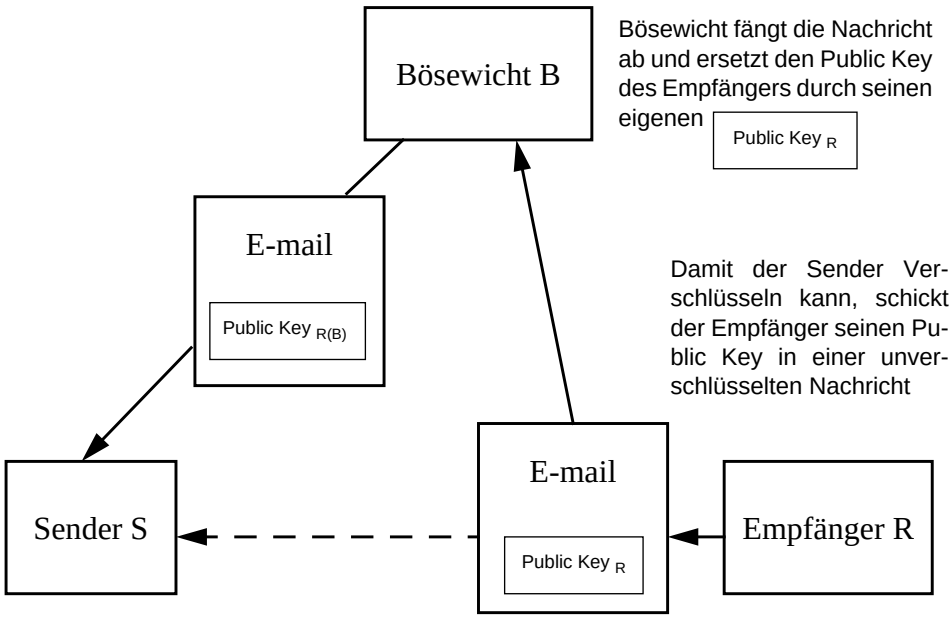


Fig. 1 Der sogenannte „Man-in-the Middle-Attack“

Beginnt nun S seine Kommunikation mit $PK_{R(B)}$ zu verschlüsseln, kann B sie mit $SK_{R(B)}$ entschlüsseln, mit dem korrekten öffentlichen Schlüssel PK_R von R wieder verschlüsseln und an R weitersenden (Fig. 2).

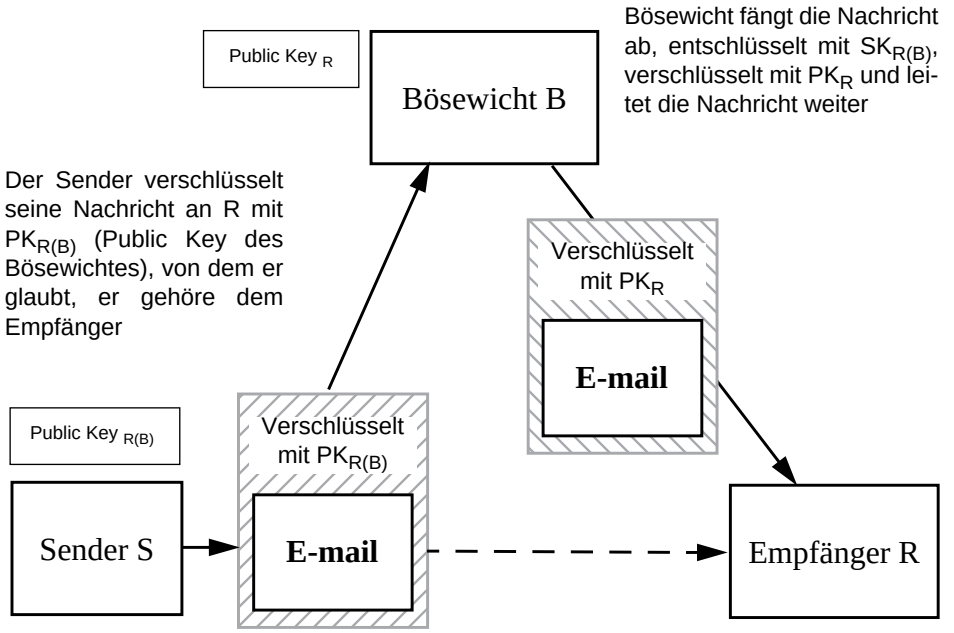


Fig. 2: B liest die vertrauliche Kommunikation zwischen S und R mit.

Noch gravierendere Auswirkungen hat diese Schwäche, wenn sich B nicht nur auf das Mitlesen der Nachrichten beschränkt, sondern den gefälschten Schlüssel dazu benützt, Nachrich-

ten oder Dokumente im Namen von R zu signieren. Sobald also die digitale Signatur dazu benützt werden soll, juristisch relevant zu kommunizieren oder Verträge und andere Dokumente in elektronischer Form der Papierform gleichzustellen, sind die Mechanismen, die asymmetrische Kryptographieverfahren zur Verfügung stellen, nicht ausreichend.

Was ist Zertifizierung?

Um die Schwäche der schweren Nachprüfbarkeit der Identität der beteiligten Parteien zu beseitigen, gibt es die Möglichkeit einer Bestätigung der Zugehörigkeit eines Schlüssels zu einer Person durch einen Dritten T (Trusted Person), dem man vertraut. Sei hier nun vorausgesetzt, daß sowohl S als auch R T kennen und ihm vertrauen, sich gegenseitig aber nicht kennen. Nachdem T ein Schlüsselpaar erzeugt hat, kann er den Public Key von R, den er während eines persönlichen Treffens von ihm erhalten hat, digital signieren. Mit seiner Unterschrift bestätigt T, daß der öffentliche Schlüssel PK_R zu R gehört. Der unterschriebene Schlüssel soll in Zukunft als $PK_{R,sig(T)}$ bezeichnet werden. Gleichzeitig erhält R den öffentlichen Schlüssel PK_T von T. Bei einem anderen persönlichen Treffen erhält S ebenfalls PK_T von T. Wenn nun S eine vertrauliche Nachricht an R schicken möchte, so erhält er beim Erstkontakt von R nicht mehr nur PK_R , sondern $PK_{R,sig(T)}$. Anhand der Unterschrift sig(T) kann S mit Hilfe des öffentlichen Schlüssels von T die Echtheit des öffentlichen Schlüssels PK_R überprüfen (vgl. den Abschnitt über digitale Signaturen). Da kein Bösewicht B den geheimen Schlüssel von T besitzt, kann er die Unterschrift sig(T) nicht fälschen. Die bestätigende Unterschrift von T unter dem Public Key von R heißt **Zertifikat** von PK_R durch T.

Das Netzwerk, das entsteht, wenn ein S einen T_1 kennt, der wiederum einen T_2 kennt, ..., der wiederum einen T_n kennt, der schließlich R kennt, nennt man ein **Web of Trust**. Es ist ein dezentrales Netzwerk, das mit anderen solcher Netzwerke zusammenwachsen kann, aber immer auf die Ehrlichkeit aller T_n angewiesen ist.

Dieses unübersichtliche Gewirr von gegenseitigen Vertrauensbeweisen führt zu sehr langen Zertifikaten, da die Schlüssel immer von einer ganzen Reihe der T_i unterschrieben werden müssen, damit sichergestellt ist, daß alle Teilnehmer mindestens einen der Unterschreibenden kennen und damit dem Schlüssel vertrauen können.

Die Certification Authority (CA)

Ein übersichtlicherer Ansatz als das Web-of-Trust bietet sich, wenn alle Teilnehmer einer einzigen CA vertrauen und sie mit ihrer Unterschrift die Identität der Teilnehmer bestätigt

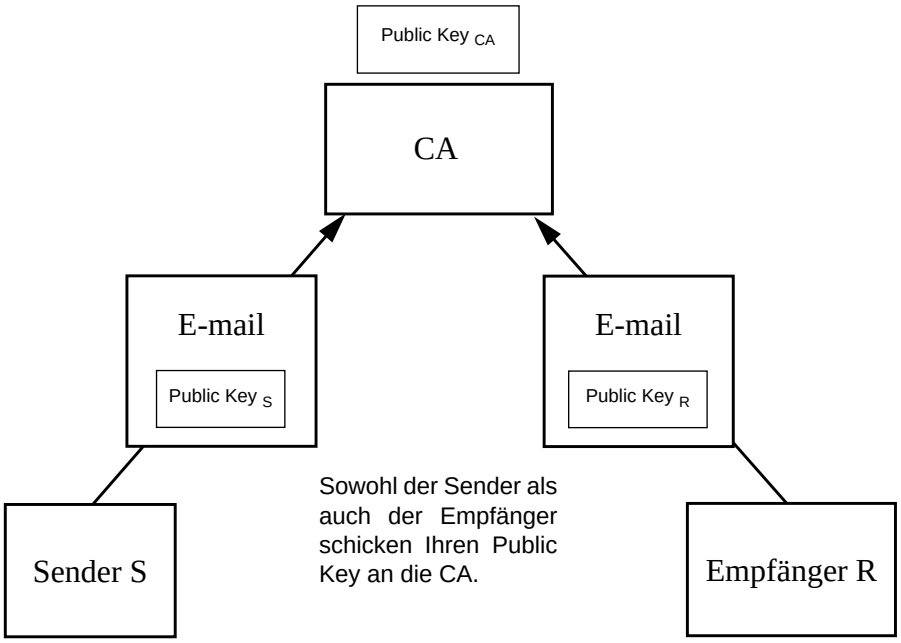


Fig. 3: Übermittlung des öffentlichen Schlüssel des Teilnehmers an die CA

Pro Teilnehmer wäre dann ein einziger Besuch bei der CA nötig, damit diese sich von der Identität des Teilnehmers überzeugen kann (Fig. 4)

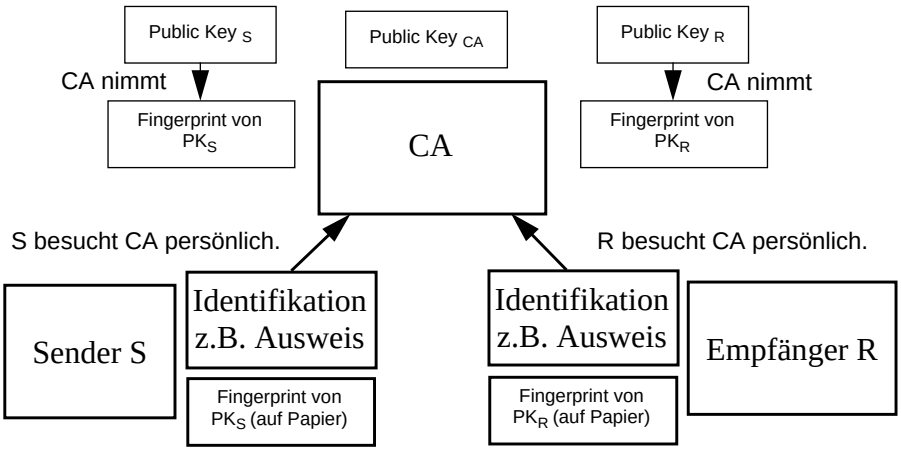


Fig. 4: Der Teilnehmer weist während eines Besuches bei der CA seine Identität sowie durch die Präsentation des Fingerprints den Besitz an seinem öffentlichen Schlüssel nach. Die CA prüft die Identität und vergleicht den Fingerprint, den vom vorher übermittelten PK des Teilnehmers nimmt mit dem, den der Teilnehmer schriftlich vorlegt.

Diese Verfahrensweise hat zusätzlich den Vorteil, daß sie sich nicht mehr auf die Ehrlichkeit aller Teilnehmer verlassen muß, sondern nur von der CA Integrität gegenüber der Teilnehmergemeinde fordert. Dazu ist es sinnvoll, diese Funktion nicht einer Person, sondern einer Institution zu übertragen, die Durchführungsrichtlinien zur Zertifizierung festlegt (sog. **Policy**). Diese Policy muß sowohl von jedem Teilnehmer akzeptiert werden, der mit einem Zertifizierungswunsch an die CA herantritt, als auch die Verpflichtungen der CA genau definieren.

Wenn die CA das erteilte Zertifikat an den Teilnehmer schickt, legt sie ihren eigenen öffentlichen Schlüssel bei, was dem Teilnehmer ermöglicht, zertifizierte Schlüssel, die er von Kommunikationspartnern erhält, auf deren Echtheit zu überprüfen (Fig. 5).

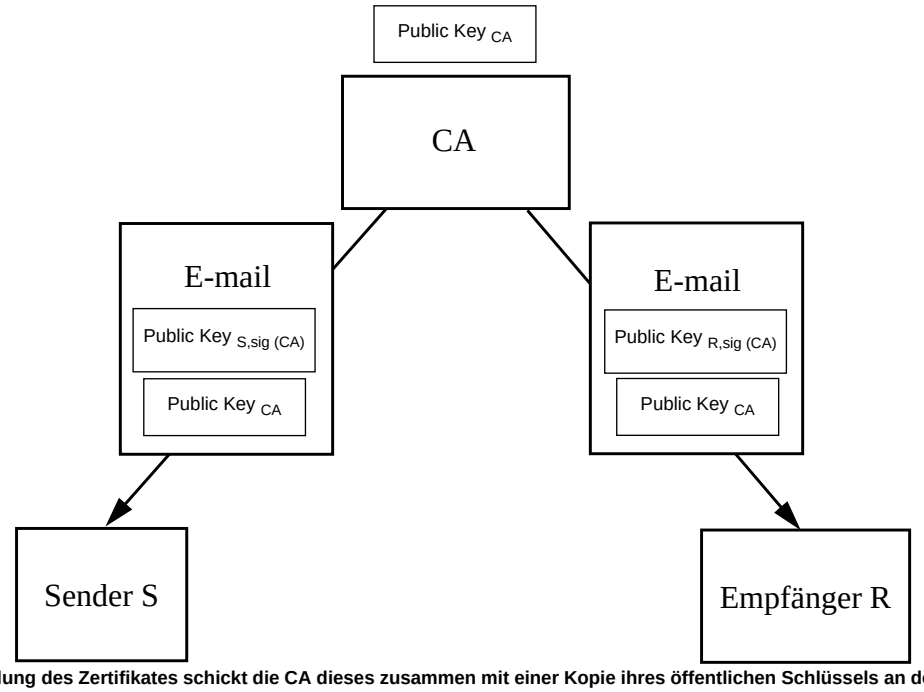


Fig. 5: Nach Erteilung des Zertifikates schickt die CA dieses zusammen mit einer Kopie ihres öffentlichen Schlüssels an den Teilnehmer (hier R und S).

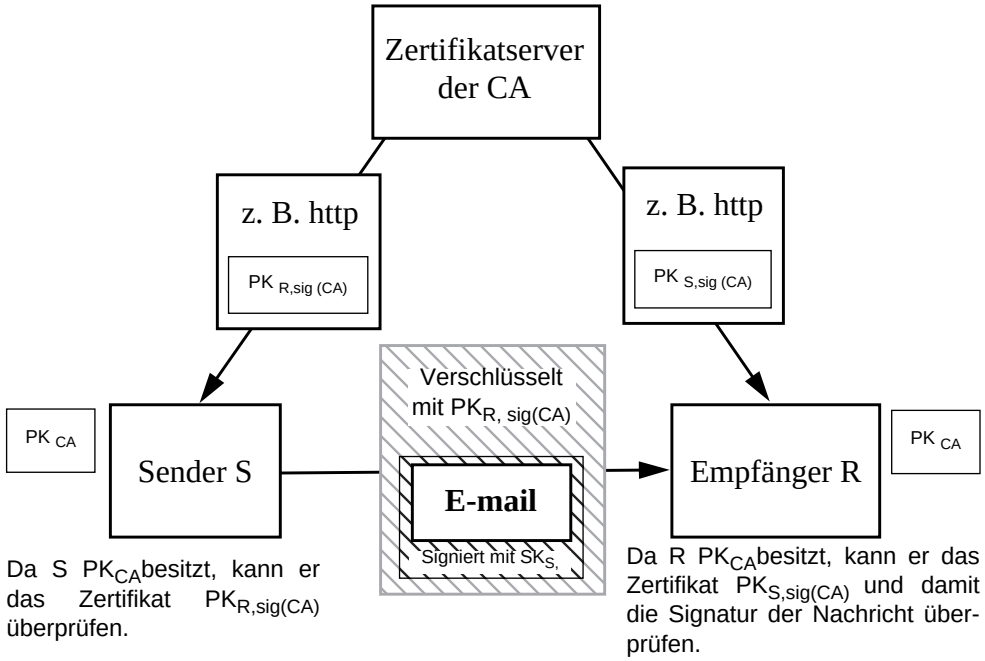


Fig. 6: Die Benutzung der Zertifikate und des Zertifikatservers der CA: S möchte R eine vertrauliche, integre und authentische Nachricht schicken. Dazu holt er sich den Public Key von R nämlich $PK_{R,sig(CA)}$ vom Zertifikatserver der CA. Nach Erhalt kann er mittels PK_{CA} die Echtheit von $PK_{R,sig(CA)}$ überprüfen. Er signiert mit seinem Secret Key, verschlüsselt mit $PK_{R,sig(CA)}$ und sendet an R. R entschlüsselt mit seinem Secret Key und kann mit dem Zertifikat $PK_{R,sig(CA)}$ die Echtheit der Signatur sig(S) überprüfen.

Auch die CA richtet einen Dienst ein, der die Suche und den Abruf erteilter Zertifikate ermöglicht. Dies kann z. B. ein WWW-basierter Zertifikatserver sein, der die Suche nach dem zertifizierten öffentlichen Schlüssel eines bestimmten Teilnehmers erlaubt.

Erst unter den Voraussetzungen, die eine CA durch die Bestätigung der Zugehörigkeit eines öffentlichen Schlüssels zu einer bestimmten Person schafft, sind asymmetrische Verschlüsselungsverfahren geeignet, wirkliche Authentizität, Integrität und Vertraulichkeit von Nachrichten, die mit elektronischer Post transportiert werden, zu gewährleisten.

Die RUS Test Certification Authority - RTCA

Was ist die RTCA?

Die RTCA ist eine Certification Authority, die am RUS zum Zwecke der Prüfung der Durchführbarkeit einer zentralen Zertifizierung von PGP Public Keys aufgebaut wird. Sie beschränkt sich im Testbetrieb ausschließlich auf die Zertifizierung öffentlicher PGP-Schlüssel von Mitarbeitern des RUS. Nach Abschluß der Testphase ist, abhängig von deren Auswertung, geplant, einen uni-weiten CA-Dienst einzurichten.

Auszüge aus der Policy der RTCA [9]

Die Policy der RTCA basiert auf analogen Dokumenten des **DFN-PCA-Projektes** an der Universität Hamburg [11] und des **Individual Network e.V.** [12] Das vorrangige Ziel der RTCA besteht im Aufbau einer RUS-weiten Public-Key-Zertifizierungs-Infrastruktur mit einem Stab von Personen, die als Zertifikatgeber (**RTZG**) der RTCA operieren und dazu von der RTCA zertifiziert werden.

Die RTCA besteht im wesentlichen aus einem Schlüsselpaar (dem RTCA-Wurzelzertifikat) das ausschließlich für die Zertifizierung der Zertifikatgeber (RTZG) benüzt wird.

Mit diesem Wurzelzertifikat werden keine weiteren CA-, RA- (Registration Authority, eine Einrichtung oder Person, die lediglich die Identität eines Benutzers und sein Eigentum an dem zu zertifizierenden Schlüssel überprüft, selbst aber keine Zertifizierung durchführt) oder Benutzerzertifikate ausgestellt.

Als RTZG für den Testbetrieb werden die folgenden Personen fungieren:

- Dr. Lisa Golka, NA-5983
- John Antoniou, NA-5961
- Herbert Franz, NA-5887
- Oliver Göbel, NA-5963

Der öffentliche Schlüssel der RTCA wird über geeignete Medien (i. w. der Zertifikatsserver der RTCA, `http://rtca.rus.uni-stuttgart.de`) veröffentlicht. Zur Zertifizierung von weiteren CA-, RA- oder Benutzerschlüsseln verwenden die RTZG ihre von der RTCA zertifizierten Schlüssel. Die RT-Sub-CAs operieren ihrerseits im Namen der jeweiligen Organisation, die sie gegenüber der RTCA repräsentieren (Bereich, Abteilung). Sie sollen zunächst nur als RTRA (RUS Test Registration Authority) fungieren und keine Zerifikate ausstellen, sondern lediglich die Identitätsüberprüfung der Benutzer mit Zertifizierungswunsch durchführen.

Eine Erzeugung von Schlüsselpaaren für Benutzer wird durch die RTCA sowie durch RT-Sub-CAs **nicht** durchgeführt.

Unterstützt werden im Rahmen der technischen Möglichkeiten:

- PGP in erster Linie,
- weitere Protokolle, je nach Bedarf und Ausbaustufe.

Eine Zertifizierung durch die RTCA zieht keinerlei rechtliche Bedeutung nach sich; ein gesetzlicher Anspruch auf die Erteilung eines Zertifikates durch die RTCA oder deren Organe besteht nicht.

Insbesondere ist die allgemeine rechtliche Relevanz digitaler Signaturen derzeit unklar. Der Bedeutung einer RUS- (bzw. später universitätsweiten) Public-Key-Infrastruktur liegt in der Schaffung der technischen Voraussetzungen für eine gesicherte elektronische Kommunikation. Insbesondere das Rechenzentrum der Universität Stuttgart, die beauftragten Personen sowie die Mitarbeiter der RT-Sub-CAs übernehmen keine Form der Gewährleistung. Alle Aufgaben werden von den Projektmitarbeitern nach bestem Wissen und Gewissen im Rahmen ihrer Dienstobliegenheiten durchgeführt.

Die dem X.509/PEM-Modell [5, 6 ,7 ,8] folgende Zertifizierungshierarchie unterhalb der RTCA (vertreten durch die RTZG) besteht aus drei verschiedenen Einheiten (Zertifikatnehmern):

- untergeornete Zertifizierungsinstanzen (**RT-Sub-CA**s)
- Registrierungsinstanzen (RTRAs)
- Teilnehmer

Die Anbindung der kompletten RTCA-Zertifizierungshierarchie an andere Hierarchien, z. B. die Hierarchie der DFN-PCA, ist prinzipiell möglich, aber im Rahmen des Testbetriebes der RTCA nicht vorgesehen.

Zertifizierungsregeln

Ein Benutzer, welcher zertifiziert werden möchte, generiert zunächst mit einer lokalen Installation des Programmes PGP (2.6.3i oder neuer) ein persönliches asymmetrisches Schlüsselpaar und übermittelt anschließend ein selbst-signiertes Zertifikat oder eine Zertifikatsaufforderung per E-Mail an die für ihn zuständige Stelle. Die aktuell zuständigen Personen bzw. Stellen für die Entgegennahme von Zertifizierungswünschen werden unter

`http://rtca.rus.uni-stuttgart.de/RTCA-CH`

veröffentlicht (die Benutzung von PGP ist in [2, 3, 4] beschrieben; ich verzichte hier auf eine weitere Erläuterung). Jeder Benutzer, der einen Zertifizierungswunsch vorträgt, muß mit dem Key, den er zur Zertifizierung übergibt, der RTCA einen **Revocation Key**, der den Widerruf seines Zertifikates ermöglicht, mit übergeben. Die RTCA verwaltet diesen Revocation Key separat und vor Zugriff von Unberechtigten geschützt. Falls der zertifizierte Schlüssel eines Benutzers kompromittiert wird, ist der Benutzer verpflichtet, dies unverzüglich der RTCA mit der Bitte um Widerruf seines Zertifikates mitzuteilen. Die RTCA wird daraufhin das Zertifikat dieses Benutzers zurückziehen und auf dem Zertifikatserver durch den entsprechenden Revocation Keyt ersetzen. Des weiteren wird die Liste der zurückgezogenen Zertifikate (CRL - Certificate Revocation List; `http://rtca.rus.uni-stuttgart.de/CRL`) aktualisiert. Um auszuschließen, daß Teilnehmer öffentliche Schlüssel benützen, die bereits zurückgezogen sind, ist ein regelmäßiger Abgleich des privaten Public Keyring (.pgp/pubring.pgp) mit der CRL der RTCA erforderlich. Eine Anleitung dazu finden sie unter

`http://rtca.rus.uni-stuttgart.de/CRL`.

Vor der Zertifizierung vergewissert sich die zuständige Stelle von der Identität des Benutzers. Dies muß durch einen persönlichen Besuch des Benutzers bei der zuständigen Stelle erfolgen, wobei sich der Benutzer durch die Vorlage des Personalausweises oder eines gleichwertigen Dokumentes zu identifizieren hat und zur Überprüfung der Authentizität seines Schlüssels dessen Fingerprint in gedruckter Form vorzulegen hat.

Die Gültigkeitsdauer von Zertifikaten für Benutzer ist in der Policy der RTCA festgelegt

(`http://rtca.rus.uni-stuttgart.de/POLICY`).

Anleitung zum Erhalt eines Zertifikates der RTCA

Notwendige Bedingungen

Zertifiziert werden während der Testphase ausschließlich öffentliche PGP-Schlüssel von Mitarbeitern des Rechenzentrums der Universität Stuttgart. Die RTCA erzeugt keine Schlüssel für Benutzer. Deshalb ist die Installation des Programmes PGP beim Benutzer für die Erzeugung eines Schlüsselpaares notwendig (siehe dazu auch BI 5/6 1996, S.11 ff [3]).

Bei der Erzeugung des Schlüsselpaares ist für die UserID das folgende Namensschema bindend:

Die UserID besteht aus zwei Teilen. Der Name (wie im Ausweis) im ersten Teil geht dem formalen Eintrag voraus. Er darf keine Klammern oder andere Unregelmäßigkeiten enthalten. Über den US-ASCII-Zeichensatz hinausgehende Kodierungen sind unzulässig. Umlaute sind in zwei Buchstaben zu zerlegen (ä wird zu ae, ö wird zu oe, ü wird zu ue), ß wird zu ss, Akzente, Cedillen und ähnliches sind wegzulassen.

Zumindest der Rufname ist auszuschreiben; diesem folgen weitere (ausgeschriebene oder abgekürzte) Vornamen sowie der ausgeschriebene Nachnamen. Im Anschluß an den Realnamen können in runden Klammern Kommentare eingefügt werden.

Im formalen Teil der UserID muß eine gültige E-mail-Adresse in spitzen Klammern stehen.

Beispiele für gültige Benutzernamen sind:

```
Erika Mustermann
<Erika.Mustermann@rus.uni-stuttgart.de>
```

```
Hans Mustermann (Hansi)
<Hans.Mustermann@rus.uni-stuttgart.de>
```

Die RTCA wird keinesfalls einen Benutzernamen zertifizieren, für den es bereits ein Zertifikat für eine andere Person gibt. Dies vor der Zertifizierung zu prüfen obliegt der jeweiligen zuständigen Stelle (vgl. `http://rtca.rus.uni-stuttgart.de/RTCA-CH`). Falls die gewählte UserID eines Schlüssels, der zertifiziert werden soll, bereits existiert, muß der Schlüsselinhaber diese ändern, so daß eine eindeutige Zuordnung einer Person zu ihrem Schlüssel möglich ist.

Die RTCA zertifiziert lediglich die Zuordnung einer Person (eines Namens) zu einem Schlüssel. Die Korrektheit der E-mail-Adresse wird nicht überprüft.

Erzeugung des Revocation Keys

Wie in den Auszügen aus der Policy der RTCA bereits genannt, muß jeder Benutzer zu seinem öffentlichen Schlüssel einen sogenannten Revocation Key erzeugen und zusammen mit ersterem an die RTCA schicken. Dies erfolgt durch die folgenden Schritte:

1. Erzeugen Sie ein temporäres Verzeichnis (z. B. temp) im Heimatverzeichnis Ihres Benutzers (falls Ihr Betriebssystem keine Tilde-Expansion zur Verfügung stellt, ersetzen Sie bei allen folgenden Schritten ‚~' durch den kompletten Pfad Ihres Heimatverzeichnisses).

UNIX:

```
$ mkdir ~/temp
```

NT:

```
> mkdir ~\temp
```

2. Ändern sie die Zugriffsrechte für dieses Verzeichnis so, daß ausschließlich Ihr Benutzer schreibenden, lesenden und ausführenden Zugriff hat. Alle anderen Rechte sollten unbedingt gesperrt sein!

UNIX:

```
$ chmod 700 ~/temp
```

NT:

```
> cacls ~\temp /g <Ihr Benutzername>:f
```

3. Kopieren Sie Ihren Public Keyring sowie Ihren Secret-Keyring in das Verzeichnis.

UNIX:

```
$ cp ~/.pgp/pubring.pgp ~/temp/pubring.pgp
$ cp ~/.pgp/secring.pgp ~/temp/secring.pgp
```

NT:

```
> copy ~\pgp\pubring.pgp ~\temp\pubring.pgp
> copy ~\pgp\secring.pgp ~\temp\secring.pgp
```

4. Um Verwechslungen mit anderen Schlüsseln in Ihrem Keyring auszuschliessen, lassen Sie sich die Übersicht aller in Ihrem originalen Keyring enthaltenen Schlüssel zeigen:

UNIX:

```
$ pgp -kv
```

NT:

```
> pgp -kv
```

PGP wird Ihnen die Übersicht in der folgenden Form präsentieren:

```
Pretty Good Privacy(tm) 2.6.3i - Public-key encryption for the masses.
(c) 1990-96 Philip Zimmermann, Phil‘s Pretty Good Software. 1996-01-18
International version - not for use in the USA. Does not use RSAREF.
Current time: 1997/06/12 16:30 GMT
Key ring: ,/user/rus/erika/.pgp/pubring.pgp‘
Type Bits/KeyID    Date       User ID
pub 1024/12345ABC 1997/06/07 Erika Mustermann <Erika.Mustermann@rus.uni-stuttgart.de>
```

Die KeyID (hier: 12345ABC) identifiziert diesen Schlüssel eindeutig und soll daher für die nachfolgenden Operationen benützt werden.

5. Führen Sie die Revocation Ihres öffentlichen Schlüssels auf den Schlüsselringen in ~/temp aus. **ACHTUNG! Führen Sie diesen Schritt nicht auf Ihrem originalen Keyring durch.** Bei der Verwendung der KeyID muß dieser ‚0x‘ als Präfix vorangestellt werden.

UNIX:

```
$ pgp -kd 0x12345ABC ~/temp/pubring.pgp
```

NT:

```
> pgp -kd 0x12345ABC ~\temp\pubring.pgp
```

PGP wird Ihnen mit dem folgenden Dialog antworten:

```
Pretty Good Privacy(tm) 2.6.3i - Public-key encryption for the masses.
(c) 1990-96 Philip Zimmermann, Phil‘s Pretty Good Software. 1996-01-18
International version - not for use in the USA. Does not use RSAREF.
Current time: 1997/06/12 16:39 GMT

Key for user ID: Erika Mustermann <Erika.Mustermann@rus.uni-stuttgart.de>
1024-bit key, key ID 12345ABC, created 1997/06/07

Do you want to permanently revoke your public key
by issuing a secret key compromise certificate
for „Erika Mustermann <Erika.Mustermann@rus.uni-stuttgart.de>“ (y/N)?
```

Tippen Sie ‚y‘ ein.

```
You need a pass phrase to unlock your RSA secret key.
Key for user ID: Erika Mustermann <Erika.Mustermann@rus.uni-stuttgart.de>
1024-bit key, key ID 12345ABC, created 1997/06/07
Enter pass phrase:
```

Geben Sie Ihren Paßsatz ein.

```
Enter pass phrase: Pass phrase is good.  Just a moment....
Key compromise certificate created.
```

6. Extrahieren Sie nun den Revocation Key aus dem temporären Schlüsselring in eine ASCII-Datei (z. B. ~/temp/Erika_revoked.asc):

UNIX:

```
$ pgp -kxa 0x12345ABC ~/temp/Erika_revoked.asc
~/temp/pubring.pgp
```

NT:

```
> pgp -kxa 0x12345ABC ~\temp\Erika_revoked.asc ~\temp\pubring.pgp
```

```
Pretty Good Privacy(tm) 2.6.3i - Public-key encryption for the masses.
(c) 1990-96 Philip Zimmermann, Phil‘s Pretty Good Software. 1996-01-18
International version - not for use in the USA. Does not use RSAREF.
Current time: 1997/06/12 17:02 GMT
```

```
Extracting from key ring: ,/user/rus/erika/temp/pubring.pgp‘, userid „0x12345ABC“.
Key for user ID: Erika Mustermann <Erika.Mustermann@rus.uni-stuttgart.de>
1024-bit key, key ID 12345ABC, created 1997/06/07
Key has been revoked.
Transport armor file: Erika_revoked.asc

Key extracted to file ,Erika_revoked.asc‘.
```

Fügen Sie diesen Key nicht in Ihren originalen Schlüsselring ein. Er würde Ihren normalen, nicht zurückgezogenen Schlüssel überschreiben und damit unbrauchbar machen.

7. Löschen Sie die Schlüsselringe im temporären Verzeichnis.

UNIX:

```
$ rm ~/temp/pubring.pgp
$ rm ~/temp/secring.pgp
```

NT:

```
> del ~\temp\pubring.pgp
> del ~\temp\secring.pgp
```

Extrahieren des gültigen öffentlichen Schlüssels

Extrahieren Sie Ihren öffentlichen Schlüssel analog Ihres Revocation Keys:

UNIX:

```
$ pgp -kxa 0x12345ABC ~/temp/Erika.asc ~/.pgp/pubring.pgp
```

NT:

```
> pgp -kxa 0x12345ABC ~\temp\Erika.asc ~\.pgp\pubring.pgp

Extracting from key ring: ,/user/rus/erika/.pgp/pubring.pgp‘, userid „0x12345ABC“.
Key for user ID: Erika Mustermann <Erika.Mustermann@rus.uni-stuttgart.de>
1024-bit key, key ID 12345ABC, created 1997/06/07
Transport armor file: Erika.asc
Key extracted to file ,Erika.asc‘.
```

Beantragung eines Zertifikates

Zur Beantragung eines Zertifikates ist eine E-Mail an die für Sie zuständige Stelle der RTCA

(http://rtca.rus.uni-stuttgart.de/RTCA-CH)

zu senden, die den Namen, die Telefonnummer, die Abteilung sowie den originalen öffentlichen Schlüssel und den zugehörigen Revocation Key enthält.

```
mit der Bitte um Zertifizierung meines PGP-Public Key:
Name: Erika Mustermann,
Tel: 685-9999
Abteilung: Rechnerbetrieb, RUS

PGP-Public-Key:
Type Bits/KeyID    Date       User ID
pub  1024/12345ABC 1997/06/07 Erika Mustermann <Erika.Mustermann@rus.uni-stuttgart.de>

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.3i

mQCNAzOgLhIAAAEEAMJdSvGUzhGB2l+P+HUWV0dhYPcpyxVmy+hq8oggRkn9jQs3xo9webj4qB5PHhv8j9w
d4TfN4NFgD89WaK5T2D88mctplECSuKCypwjcXBZAbqMyrgrGZAy+MF6My5jMGa5htJWOYpGPamzKXhem67
X4K0fvtRBu2++X9nJAAURtDhFcmlrYSBNdXN0ZXJtYW5uIDxFcmlrYS5NdXN0ZXJtYW5uQHJ1cy51bmktc3
R1dHRnYXJ0LmRlPokAlQMFEDOgLhPUQbtvvlZyQEBuCID0lQWZNr3OlvIYp1hio9QtAuxkDSrWsevrdEodl
GS58hqlYH9SPUe8q5wk9Bw++KJnLJrb9FBGYHtU2LzxH9mzPaIzkH0PWU1uN0GlD85OwzwcfJ1UM6VWjR4q
5AO9+E29yXuZsZ2YYKfNw2KCW72YyFA6dEej1smew01iDT=/zPb
-----END PGP PUBLIC KEY BLOCK-----
```

```
PGP-Revocation-Key:
Type Bits/KeyID    Date       User ID
pub  1024/12345ABC 1997/06/07 *** KEY REVOKED ***
Erika Mustermann <Erika.Mustermann@rus.uni-stuttgart.de>

-----BEGIN PGP PUBLIC KEY BLOCK-----

Version: 2.6.3i

mQCNAzOgLhIAAAEEAMJdSvGUzhGB2l+P+HUWV0dhYPcpyxVmy+hq8oggRkn9jQs3xo9webj4qB5PHhv8j9wd4
TfN4NFgD89WaK5T2D88mctplECSuKCypwjcXBZAbqMyrgrGZAy+MF6My5jMGa5htJWOYpGPamzKXhem67X4K0
fvtRBu2++X9nJAAURiQCVAwUgM6AvDtRBu2++X9nJAQGFFgP9HLVf4KAA1iDXkuibPR8Vcaw9Pgd7cefn8Xx8
w4c5j8eI48K5PLrUW3ovmiHbYFSHrP+nqpK2mAyloB3NFwszZWVaSxk30J+Z+k0IJDrkcwygokseI6qY6y89Z
d72wL77FAxySTFMPNmM0Bci6qLrWf7w1Sd+rm2Prsnbpu0OEVyaWthIE11c3Rlcm1hbm4gPEVyaWthLk11c3R
lcm1hbm5AcnVzLnVuaS1zdHV0dGdhcnQuZGU+iQCVAwUQM6AuE9RBu2++X9nJAQG4IgPSVBZk2vc6W8hinWH+
Kj1C0C7GQNKtax6+t0Sh2UZLnyGqVgf1L89R7yrnCT0HD74omcsmtv0UEZge1TYvPEf2bM9ojOQfQ9ZTW43Qa
UPzk7DPBx8nVQzpVaNHirkA734Tb3Je5mxnZhgp83DYoJbvZjIUDp0R6PWyZ7DTWINM==yo/e

-----END PGP PUBLIC KEY BLOCK-----

Mit freundlichen Gruessen

Erika Mustermann
```

Nach Erhalt dieser E-Mail wird die für Sie zuständige Stelle der RTCA einen Besuchstermin mit Ihnen vereinbaren, bei dem Sie sich gegenüber der RTCA identifizieren sowie einen Fingerprint in gedruckter Form Ihres öffentlichen Schlüssels präsentieren, um Ihr Eigentum an diesem Schlüssel zu dokumentieren. Den Fingerprint Ihres Schlüssels erhalten Sie folgendermaßen:

UNIX/NT:

```
> pgp -kvc 0x12345ABC
Pretty Good Privacy(tm) 2.6.3i - Public-key encryption for the masses.
(c) 1990-96 Philip Zimmermann, Phil's Pretty Good Software. 1996-01-18
International version - not for use in the USA. Does not use RSAREF.
Current time: 1997/06/12 17:46 GMT

Key ring: ,/user/rus/erika/.pgp/pubring.pgp', looking for user ID „0x12345ABC".
Type Bits/KeyID    Date       User ID
pub  1024/12345ABC 1997/06/12 Erika Mustermann <Erika.Mustermann@rus.uni-stuttgart.de>
            Key fingerprint = 8D 23 9D 82 EA 75 47 A7  EC AF 21 89 EE 09 0E 1F
1 matching key found.
```

Nach Erhalt des Zertifikates sollten Sie das temporäre Verzeichnis mitsamt den darin enthaltenen Schlüsseldateien (Erika.asc und Erika_revoked.asc) löschen, um eine unabsichtliche Benutzung auszuschließen.

UNIX:

```
$ rm -rf ~/temp
```

NT:

```
> rmdir ~\temp /s
```

Benutzung der RTCA

Der Zertifikatserver der RTCA (http://rtca.rus.uni-stuttgart.de) bietet eine komfortable WWW-Schnittstelle, die auf einfache Weise nach dem Public Key Ihres Kommunikationspartners suchen läßt.

Nachdem ein Suchstring eingegeben wurde gibt der Server eine Liste von KeyIDs bzw. UserIDs zurück, die auf diesen String passen. Die KeyID und die UserID jedes gefundenen Schlüssels wird als Hyperlink ausgegeben.

Um den Public Key ihres Kommunikationspartners aus der Datenbasis des Servers zu erhalten, ist einer der Hyperlinks des gewünschten Schlüssels anzuklicken. Der Schlüssel wird im ASCII-Format ausgegeben und kann nun in die lokale PGP-Umgebung kopiert werden.

Fig. 7: Die WWW-Schnittstelle des Zertifikatservers der RTCA: Suche nach „rtca"

Public Key Server -- Index ``rtca''

Type bits/keyID Date User ID

pub 2048/E77783E5 1997/05/23 John Antoniou, Zertifikatgeber der RTCA <RTZG.antoniou@rtca.rus.uni-stuttgart.de> Dieses Zertifikat wird ungueltig 2 Jahre nach der Erzeugung des zertifizierten Schluessels

pub 2048/EFF36C1D 1997/05/23 Dr. Elisabeth Golka, Zertifikatsgeberin der RTCA <RTZG.golka@rtca.rus.uni-stuttgart.de> Dieses Zertifikat wird ungueltig 2 Jahre nach Erzeugung des zertifizierten Schluessels

pub 2048/D8116AB1 1997/05/23 Herbert Franz, Zertifikatgeber der RTCA <RTZG.franz@rtca.rus.uni-stuttgart.de> Dieses Zertifikat wird ungueltig 2 Jahre nach Erzeugung des zertifizierten Schluessels

pub 2048/F105A65D 1997/05/22 RUS Test Certification Authority <rtca@rtca.rus.uni-stuttgart.de> (Rechenzentrum der Universitaet Stuttgart) Dieses Zertifikat ist gueltig 2 Jahre nach der Erzeugung des zertifizierten Schluessels

pub 2048/BEE0B935 1997/05/20 Oliver Goebel, Zertifikatgeber der RTCA <RTZG.goebel@rtca.rus.uni-stuttgart.de> Dieses Zertifikat wird ungueltig 2 Jahre nach Erzeugung des zertifizierten Schluessels

Fig. 8: Ergebnisseite der Suche nach ‚rtca"

Public Key Server -- Get ``rtca@rtca.rus.uni-stuttgart.de''

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.2
Comment: PGP Key Server 0.9.1
mQENAzOEGWwAAAEIANR8skCEuPnMlTFqxesR1Qa+9qtqoEZKniZfzbNyoJjXywWa
Ri0e8Yvht2+dI7jjwvY/D0N+pV0llWRDqLVNgmRL6uhW4/KBxYht/0PoEIhjvSpz
sW1rsf+GE2QFIYM8yz//MHpxUiZ9pDbbU5sitt7ktnEh9FWfv8aJgO7y7/UBqRey
ao26/tdyuoxeibvMOvpymB8EMV+3iDgmlpfc04fsIPuDyH30DMRXyJ8DrJ0f6Njd
jXDCBBq+j2q48mMUJvPaYmOJrP0UfM9Gm6snZ9L9wo19yyKrv/uIe4nAuK7B+oG4
fUqupanvE4J3MYrENVAs7KODfYK7Hdw7IvEFpl0ABRG0xFJVUyBUZXN0IENlcnRp
ZmljYXRpb24gQXV0aG9yaXR5IDxydGNhQHJ0Y2EucnVzLnVuaS1zdHV0dGdhcnQu
ZGU+IChSZWNoZW56ZW50cnVtIGRlciBVbml2ZXJzaXRhZXQgU3R1dHRnYXJ0KSBE
aWVzZXMgWmVydGlmaWthdCBpc3QgZ3VlbHRpZyAyIEphaHJlIG5hY2ggZGVyIEVy
emV1Z3VuZyBkZXMgemVydGlmaXppZXJ0ZW4gU2NobHVlc3NlbHOJARUDBRAzhBls
Hdw7IvEFpl0BAWx7B/0cPcfz3oTCNBgDz5Cmz/qCcasG1bMhvWbCShIcHIXG3Xkb
aSe9/WVxQYkjo7SGI/lNWRJIFCswaA7FGxDMhU7cuSMatBrOoQTrlG3qar8tuKZS
36rUKm4OcvYfiALHk7Ixsg9on6CWieEBvW1W/k5j0fbwPw16ujk5Gus7XrS7Bo8J
25OSk6sc9tF+m7187KXnGzpJdraE+3W0DgN0DDbqH/Ix67MLdiIetav6a7v/C2pb
fmiyEO4reQym2dU+BKWF3KPqanD/nyp8WwewwhKGfxTy4XXAKVx32HWRYbYfCdi0
OSwn2S6MqGMrvNjn3rDYeCnHubR96pg+PtuG2lp4
=ZEv5
-----END PGP PUBLIC KEY BLOCK-----
```

Fig. 9: Das Wurzelzertifikat der RTCA (ASCII-armored Public Key der RTCA)

Nach dem Abspeichern des Schlüssels (z. B. in ~/tmp/rtca.asc) kann der Schlüssel mit dem PGP-Kommando

```
> pgp -ka ~/tmp/rtca.asc
```

in den lokalen Schlüsselring kopiert werden. Danach ist eine Verschlüsselung von Nachrichten an den Schlüsselinhaber, bzw. die Überprüfung seiner Signaturen möglich.

PGP verwaltet für jeden Benutzer einen eigenen Public Keyring und arbeitet mit diesem autonom. PGP ist nicht dafür vorgesehen mit einer Hierarchie von zertifizierenden Stellen zusammenzuarbeiten. Daher ist es nötig, die Abgleichung des persönlichen Keyrings mit der Datenbasis des Zertifikatservers selbst regelmäßig durchzuführen. Dies ist notwendig, um sicherzustellen, daß der einzelne Benutzer nicht mit öffentlichen Schlüsseln von Kommunikationspartnern arbeitet, die bereits ungültig sind.

Eine Anleitung zum Abgleich des persönlichen Schlüsselringes mit der CRL der RTCA ist unter `http://rtca.rus.uni-stuttgart.de/CRL` zu finden

Benutzungsschnittstellen für PGP und PGP-fähige Mail-Programme

Unter `ftp://ftp.uni-stuttgart.de/pub/security/pgp-primary/` ist eine Zusammenstellung von Benutzungsschnittstellen für PGP zu finden. Die Arbeitsgruppe der RTCA ist bemüht, auf diesem Gebiet erscheinende Neuheiten unter `http://rtca.rus.uni-stuttgart.de/NEWS` vorzustellen, soweit dies für den laufenden Betrieb interessant ist.

Fingerprint des öffentlichen Schlüssels der RTCA

```
pub  2048/F105A65D 1997/05/22 RUS Test Certification Authority <rtca@rtca.rus.uni-stuttgart.de>
( Rechenzentrum der Universitaet Stuttgart)
          Key fingerprint = 43 11 37 D2 57 EE 0E 1C  6A 13 F1 25 C8 2F E5 1D
```

Literatur

[1] Bruce Schneier, *Angewandte Kryptographie*, Addison-Wesley, John Wiley & Sons, 1996

[2] http://www.pgp.net/pgpnet/pgp-faq/, *The comp.security.pgp FAQ*, Arnoud Engelfriet.

[3] Bernd Lehle und Oliver Reutter, *Security Tools 4: Pretty Good Privacy*, BI 5/6 1996, S.11 ff.

[4] Simson Garfinkel, *PGP: Pretty Good Privacy*, O'Reilly & Associates, 1994

[5] RFC 1421, *Privacy Enhancement for Internet Electronic Mail (PEM) Part I Message Encryption and Authentication Procedure*; IAB, IRTF, IETF PEM Working Group, February 1993, http://andrew2.andrew.cmu.edu/rfc/rfc1421.html

[6] RFC 1422, *Privacy Enhancement for Internet Electronic Mail (PEM) Part II, Certificate-based Key Management*; IAB, IRTF, IETF PEM Working Group, February 1993, http://andrew2.andrew.cmu.edu/rfc/rfc1422.html

[7] RFC 1423, *Privacy Enhancement for Internet Electronic Mail (PEM) Part III, Algorithms, Modes and Identifiers*; IAB, IRTF, IETF PEM Working Group, February 1993, http://andrew2.andrew.cmu.edu/rfc/rfc1423.html

[8] RFC 1424, *Privacy Enhancement for Internet Electronic Mail (PEM) Part IV, Key Certification and Related Services*; IAB, IRTF, IETF PEM Working Group, February 1993, http://andrew2.andrew.cmu.edu/rfc/rfc1424.html

[9] RTCA-Gruppe des RUS, *RTCA - RUS Test Certification Authority: Certification Policy*, http://rtca.uni-stuttgart.de/POLICY

[10] Stefan Kelm, Britta Liedtke, *Projekt DFN-PCA*, DFN-Mitteilungen, Heft 40, März 1996

[11] DFN-PCA, *DFN - PCA: Low-Level Policy,* http://www.pca.dfn.de/dfnpca/policy/low-level.html, 1996-1997

[12] Arbeitsgruppe IN-CA des Individual Network e.V., *Individual Network e.V.: Certification Policy*, http://www.in-ca.individual.net/policy.html

[13] Oliver Göbel, *IPKS - the InfoWin Public Keyserver, Documentation and Help*, http://keyserver.InfoWin.org/HELP, InfoWin-Projekt, Rechenzentrum Universität Stuttgart

Oliver Göbel
goebel@rus.uni-stuttgart.de
Universität Stuttgart

Secure Shell - die sichere Verbindung

Frank Kargl

Jeder Systemadministrator kennt sicher das Problem. Remote Wartung auf einem anderen Rechner. Bei einem normalen telnet/rlogin geht dabei in der Regel auch das root-Passwort im Klartext über die Leitung. Das sollte jedoch in einer Zeit, wo auf jedem vernetzten PC ein Netzwerk-Monitor laufen kann, tunlichst vermieden werden. Auf Vertrauensbeziehungen wie hosts.equiv oder .rhosts sollte man (zumindest für den Superuser-Zugang) genauso verzichten, können diese doch von gewieften Hackern durch IP-Spoofing mißbraucht werden.

Was bleiben dem Administrator also für Alternativen zur „Birckenstock-Wartung“ ?

Eine gängige Lösung sind One-Time-Passwort-Systeme ala s/key. Diese bieten (richtig angewandt) zwar einem recht hohen Sicherheitsstandard, sind für die tägliche Arbeit aber oftmals zu umständlich. Besser geeignet sind kryptographische Verfahren, die neben dem beschriebenen Authentifizierungsproblem auch alle übertragenen Daten sichern.

Seit längerem existiert mit der Secure Shell (kurz ssh) ein sehr mächtiger Vertreter dieses Ansatzes. ssh kann die unter Unix üblichen r-Tools (rlogin, rsh, rcp, rdist) vollständig durch die entsprechenden ssh-Pendants ersetzen. Weiterhin ist die sichere Weiterleitung von X11- oder ganz generell das Tunneling beliebiger TCP-Verbindungen möglich. Die Verbindung wird dabei durch ein (wählbares) private-key Kryptoverfahren gesichert. Zur Auswahl stehen IDEA, three-key triple-DES, DES, RC4-128, TSS und Blowfish. Der Schlüsselaustausch ist über das RSA public-key Kryptoverfahren abgesichert. Die sog. Session-Keys werden stündlich gewechselt und nirgends auf Platte geschrieben. Jeder Benutzer und jeder Host verfügt über ein RSA-Key-Paar, so daß sich sowohl die Host- als auch die User-Identität über die entsprechenden Schlüssel verifizieren lassen. Wenn gewünscht, ist somit eine komplett RSA-basierte Authentifizierung machbar. Alternativ stehen Passwort- und .rhosts-Authentifizierung zur Verfügung. Moderene Hackertechniken wie IP-Spoofing-, DNS- und Routing-Angriffe können beim Einsatz der ssh weitestgehend ausgeschlossen werden.

Die Einsatzmöglichkeiten der ssh sind vielfältig: so ist das sichere Verbinden zweier Netzwerke über einen verschlüsselten (und komprimierten!) ssh-Kanal möglich, RPC-basierte Dienste wie NIS können mit ssh abgesichert werden und ssh arbeitet auch mit anderen Systemen wie SOCKS oder AFS/Kerberos zusammen.

Alle Möglichkeiten der ssh aufzuführen, würde den Rahmen dieses Artikels sprengen, es sei deshalb auf die weiteren Quellenangaben verwiesen. Die Entwicklung der ssh wird energisch vorangetrieben. An einem Schlüsselmanagement-System wird ebenso gearbeitet, wie an der Standardisierung des ssh Protokolls in einem RFC.

Unter UNIX sind sowohl Server als auch Client frei verfügbar (aktuelle Version 1.2.20). Ebenso existiestieren freie Client-Versionen für OS/2 und Windows (beta). Weiterhin sind freie Versionen für Macintosh und VMS in Arbeit. Kommerzielle Client-Versionen sind verfügbar für UNIX, Windows und Macintosh (beta).

Es gibt praktisch keine, Gründe die gegen den Einsatz der ssh sprechen. Messungen haben gezeigt, daß die Verschlüsselung der Verbindung bei den heute verfügbaren, schnellen Rechnern in der Regel keine merklichen Performance-Einbußen mehr mit sich bringt. Die ssh ist ein vollwertiger Ersatz für die bekannten r-Tools, und viele anderen Netzwerk-Dienste lassen sich einfach auf die Verwendung von ssh umstellen. Bei volumenintensiven Anwendungen und langsamen Leitungen bringt die Umstellung auf einen komprimierten ssh-Tunnel oft sogar Performance- und Lastvorteile. Bleibt also als Resümee, daß die Verwendung der ssh für jeden sicherheitsbewußten Administrator zur Pflichtaufgabe gehören sollte.

Installation und Einsatz unter Unix

Die Installation der ssh (FTP-Archiv s.u.) ist unter den meisten Unix-Derivaten denkbar einfach.

Nach Entpacken der Quellen und Eingabe von

$ ./configure # evtl. mit -prefix='installpath'

$ make

$ make install

steht bereits ein lauffähiges System zur Verfügung. Im Zuge des make install wird dabei gleichzeitig ein Host-Key für den lokalen Rechner erstellt. Soll dies auch auf anderen Rechnern erfolgen, so ist dort ein

$ make hostinstall

auszuführen. Nun sollte noch der Server sshd in ein geeignetes Startup-Skript eingetragen werden. Ein Aufruf über den inetd ist zwar möglich, man sollte jedoch aus Performancegründen davon absehen. Der sshd generiert bei jedem Start einen neuen Schlüssel, was sich bei Aufruf über inetd durch Verzögerungen beim Verbindungsaufbau bemerkbar macht.

Anschließend kann man noch die Konfigurationsfiles /etc/sshd_config und /etc/ssh_config, die das Verhalten des Servers resp. des Clients beeinflussen, überprüfen und gegebenenfalls anpassen. Hier sind speziell die Optionen interessant, mit denen man verschiedene Authentifizierungsarten anbieten bzw. abschalten kann, so daß z.B. User-lokale .rhosts Dateien nicht mit ssh funktionieren. In diesem Zusammenhang ist zu überlegen, inwieweit man einen Zugriff über ssh verbindlich vorschreibt oder ob er alternative Zugänge (telnet, rlogin etc.) weiterhin verfügbar bleiben.

Zur Absicherung gegen Spoofing-Angriffe sollte der Administrator auf jedem Rechner ein aktuelles File /etc/ssh_known_hosts führen, das die öffentlichen Schlüssel aller anderen Rechner enthält, zu denen Vertrauensbeziehungen existieren. Dazu kann das Tool make-ssh-known-hosts eingesetzt werden.

Bei der ersten Nutzung der ssh wird für jeden Benutzer automatisch ein eigener Schlüssel erzeugt. Dieser Schritt kann auch manuell mittels ssh-keygen durchgeführt oder wiederholt werden. Der Schlüssel läßt sich durch eine Passphrase schützen und kann auch zu Authentifizierungszwecken genutzt werden kann. Hierzu ist ein File $HOME/.ssh/authorized_keys auf dem Zielrechner anzulegen, in welches der Inhalt von $HOME/.ssh/identity.pub des Ausgangsrechners zu kopieren (bzw. anzuhängen) ist.

Dieser Artikel gibt nur einen ersten Überblick über die Möglichkeiten der ssh. Deswegen empfehle ich dem interessierten Leser dringend einen Blick auf die FAQ und die anderen angegebenen Quellen.

Quellen und Literaturangaben:

[14] SSH-FAQ http://fg70.rz.uni-karlsruhe.de/~ig25/ssh-faq/ssh-faq.html

[15] SSH-Archiv ftp://ftp.cs.hut.fi/pub/ssh/

[16] Deutscher Mirror ftp://ftp.cert.dfn.de/pub/tools/net/ssh

[17] kom. SSH-Entwicklung http://www.ssh.fi/

[18] http://www.europe.datafellows.com/

[19] Deutscher Distributor http://www.think.de/f-secure/

weitere Quellen siehe SSH-FAQ

Frank Kargl
frank.kargl@rz.uni-ulm.de
Universitätsrechenzentrum Ulm

SichererFiletransfermit SAFT/sendfile

Ulli Horlacher

Für das Internet existierte bisher kein asynchroner Filetransfer-Service, wie es ihn beispielsweise im Bitnet gab. Der bereits vorhandene Standard-Filetransfer, das ftp, ist ein synchroner Dienst: Der Benutzer muß sowohl auf Sender als auch auf Empfängerseite über einen Account verfügen und die Datenverbindung von beiden Seiten aus aufbauen.

Wollte ein Benutzer A einem beliebigen Benutzer B eine Datei zukommen lassen, hatte er bisher nur folgende wenig brauchbare Möglichkeiten:

- ftp zum Empfängeraccount
  Dazu muß das Paßwort des Empfängeraccounts bekannt sein. Wenn A und B nicht physisch identisch sind, ist diese Methode sicherheitstechnisch indiskutabel. Aber auch wenn es sich bei A und B um dieselbe Person handelt, so wandert doch das Paßwort als Klartext in einem tcp-Päckchen durchs Internet.
- ftp über einen anonymous ftp Server
  Die Datei muß von A auf den anonymous ftp Server mittels ftp übertragen werden, A muß B mit einer e-mail informieren, daß dieser die Datei abholen kann und B muß dann ebenfalls ftp aufrufen. Als Nebenbedingung muß ein ftp Server gefunden werden, der von beiden gut erreicht werden kann, und dieser Server muß ein allgemein schreibbares Verzeichnis aufweisen. Wärend die Datei auf dem anonymous ftp Server liegt, kann sie aber praktisch von jedermann gelesen, gelöscht oder verändert werden.
- verschicken via e-mail
  A schickt B die Datei als e-mail. Nach RFC 822 darf eine e-mail nur Zeichen aus dem NVT-ASCII Zeichensatz enthalten, welches eine Untermenge von 7 bit ASCII ist. Somit ist der Dateientransfer auf englische Textdokumente beschränkt, wenn man nicht die zu verschickende Datei entsprechend kodiert, so daß sie nur noch NVT-ASCII Zeichen enthält. Als Kodierung bieten sich an: uuencode oder MIME. Beide sind aber umständlich zu handhaben, unterstützen nicht alle Dateiattribute und vergrößern zwangsläufig die zu übertragende Datenmenge. Zudem sind die real existierenden MTAs nicht in der Lage, mit größeren Übertragungen fertig zu werden.

Um diesen Mangel zu beseitigen, wurde das SAFT-Protokoll (Simple Asynchronous File Transfer) entwickelt und mit sendfile eine Referenzimplementation für Unix geschaffen.

Wesentliche Merkmale des SAFT-Protokolls sind:

- Betriebsystemsunabhängigkeit
  SAFT soll auf möglichst allen Computersystemen im Internet verfügbar sein.
- Einfachheit
  „keep it simple“ ein überschaubares Protokoll auf ASCII-Basis.
- Erweiterbarkeit
  Beliebige Dateiattribute spezieller Betriebssysteme können jederzeit in einer erweiterten Definition mit aufgenommen werden.

Quasi als „Abfallprodukt“ wurden noch asynchrone Nachrichten („messages“) als weiterer Internetdienst integriert. Bei solch einer Nachricht handelt es sich um einen einzeiligen Text, der direkt auf das Terminal des Empfängers geschrieben wird.

SAFT ist ein Client/Server-Protokoll. Der SAFT-Client verschickt Dateien oder Nachrichten via Internet an den SAFT-Server, welcher sie entgegennimmt und sie an den lokalen Adressaten direkt ausliefert oder in einem Spoolbereich zwischenlagert, wo sie dann der Adressat mittels eines Receice-Clients abholen kann. Die Funktionsweise ist dieselbe wie bei normaler Internetmail. Der Spooling Mechanismus und der Receive-Client werden nicht von SAFT beschrieben, sondern bleiben der jeweiligen Implementation überlassen. SAFT definiert nur das reine Übertragungsprotokoll.

SAFT kann Dateien mittels gzip Algorithmus komprimiert übertragen, so daß Netzbandbreite geschont wird. Ebenso besteht die Möglichkeit,

Dateien nach dem PGP-Standard zu verschlüsseln und/oder zu signieren. Eine Umwandlung bzw. Anpassung der Zeichensätze (Umlaute etc) erfolgt automatisch.

SAFT wird in nächster Zukunft als neuer RFC eingereicht.

Sendfile, die SAFT-Implementation für Unix, umfasst die Teile:

- sendfiled
  der Server
- sendfile
  ein Client zum Verschicken von Dateien
- sendmsg
  ein Client zum Verschicken von einzeiligen Nachrichten
- receive
  ein Client zum Abholen von empfangenen Dateien

Der sendfile-Client ist ein Userprogramm, das Dateien an den sendfiled des Empfängersystems verschickt. Der sendfiled nimmt die Dateien entgegen, legt sie im lokalen Spoolbereich ab und informiert den Empfänger. Der Empfänger kann sich dann die Dateien mittels des receive-Clients in sein Directory kopieren, wobei das Original aus dem Spool gelöscht wird.

Beispiele für den sendfile-Client:

```
sendfile doku.ps uranus@vax.inka.de
```

```
sendfile -a Source *.f90 Makefile \
rusframs
```

Sendfile komprimiert automatisch vor dem versenden und spart so Netzbandbreite. Ausserdem ist es möglich, Directories oder mehrere Files als ein Archivfile zu verschicken (siehe zweites Beispiel oben).

Der sendmsg-Client ist ein Userprogramm, das einzeilige Textnachrichten an den sendfiled verschickt, welcher sie dann direkt auf das Terminal des Empfängers schreibt.

Beispiel für sendmsg:

```
sendmsg framstag@moep.bb.bawue.de
message: schieb schon mal die Pizza
in den Ofen, wir kommen!
```

Zum Abholen aus dem lokalen Spool benutzt der Empfänger den receive-Client.

Beispiel:

```
$ receive -l
From zrxh0370@baracke.rus.uni-
stuttgart.de (Ulli Horlacher)
  1) 1995-08-10 15:41:24     3 KB
LIESMICH
  2) 1995-08-10 15:41:37    30 KB
doku.txt
  3) 1995-08-10 15:42:09 91345 KB
Sourcen (archive)
```

```
$ receive L*
%receive-I, LIESMICH received
```

Existieren bereits Dateien gleichen Namens, fragt receive nach, ob diese überschrieben werden dürfen. Ebenso wird der Benutzer gefragt, ob Filenamen mit kritischen Zeichen wie Steuercodes oder Shell-Metazeichen erzeugt werden dürfen.

Als weitere Option kann sendfile Dateien mittels pgp (welches zuvor installiert sein muss) zu verschlüsseln:

```
sendfile -pe geheim.gif beate
```

oder zu signieren:

```
sendfile -ps wichtig.txt scheff@hq
```

Natürlich können auch die Optionen `-pe` und `-ps` kombiniert werden.

Sicherheitstechnisch ist sendfile unbedenklich, da die Clients als normale Userprogramme laufen und der Server nur vom inetd kontrolliert aufgerufen wird. Der sendfiled schreibt ausschliesslich in das Userspool-Directory und dies auch nur mit der UID des Empfängers. Der Administrator kann eine Datei anlegen mit Usern, die vom sendfile-Service ausgeschlossen sind. Zusätzlich kann jeder User für seine eine Liste anlegen von Adressen, von denen er nichts emfangen möchte

Sendfile läuft bisher auf AIX, BSDI, Convex-OS, Digital Unix, FreeBSD, HP-UX, IRIX, Linux, NeXTstep/Mach, OSF/1, SunOS 4, SunOS 5 (Solaris) und Ultrix. Eine Portierung für OS/2 befindet sich in der Endphase.

Weiterhin läuft gerade ein Projekt in der BelWü-Koordination, in dem eine MaTA-Praktikantin eine Java-Implementation von SAFT erstellt. Damit wären dann auch die Windows-Systeme abgedeckt.

Sendfile muß von root installiert werden, weil der sendfiled ein privilegierter Internetdämon ist.

Ein automatisches Installationsskript ist vorhanden, ebenso eine ausführliche 40-seitige Dokumentation, sowie man-pages für die Clients.

Sendfile benötigt einen ANSI-C-Compiler (zB gcc) zum Übersetzen und gzip und pgp zur Laufzeit. Wer sich für sendfile interessiert und es auf seinem System benutzen möchte, kann sich die Sourcen und eine ausführliche Dokumentation via ftp besorgen:

`ftp://ftp.uni-stuttgart.de/pub/unix/comm/misc/sendfile.tar.gz`

Teilnehmer am /sw-Konzept können sich sendfile noch einfacher installieren: Die Clients (sendfile, sendmsg und receive) stehen bereits unter /client/bin zur Verfügung. Die Serverinstallation und alle notwendigen Konfigurationsschritte übernimmt das Script /client/bin/sf_install (Dieses muß von root ausgeführt werden).

SAFT/sendfile enstand als Projektarbeit im Rahmen der Ausbildung zum Mathematisch-technischen Assistenten am Rechenzentrum der Universität Stuttgart und wird jetzt von der BelWü-Koordination gepflegt und weiterentwikkelt.

Glossar

- ASCII - American Standard Code of Information Interchange
  7 bit Zeichensatzkodierung der wichtigsten im amerikanischen Sprachraum vorkommenden Zeichen. Siehe RFC 1345.
- Bitnet - Because It`s Time Network
  Aussterbendes weltweites Forschungsnetzwerk auf IBM RSCS Technologie. In vielen Belangen der Vorgänger zum Internet.
- ftp - filetransfer-protocol/program
  Standardmethode, um Dateien zu transferieren (synchron). Siehe RFC 959.
- IANA - Internet Assigned Numbers Authority
  „Behörde“ im Internet, die Adressen reserviert. Siehe RFC 1700 ”
- MIME - Multipurpose Internet Mail Extensions
  Multimediaerweiterungen für Mail. Siehe RFC 1341.
- MTA - Mail Transport Agent
  Der E-mail-Server (z.B. für smtp). Siehe RFC 822.
- NVT - Network Virtual Terminal
  Remote login im Internet: telnet. Siehe RFC 854.
- RFC - Request for Comment
  Internet Standards und Protokoll Beschreibungen.
  ftp://ftp.uni-stuttgart.de/pub/doc/standards/rfc

Ulli Horlacher
BelWü-Koordination
framstag@belwue.de

TCP-Wrapper

Bernd Lehle / Oliver Reutter

In Analogie mit den Fantasy-Rollenspielen sehen die Hacker im Internet sich gerne als Nachfolger der mittelalterlichen Ritter und benutzen auch entsprechende Terminologie. In diesem Sinne wollen wir nach dem zweischneidigen Schwert SATAN nun einen recht wirksamen Schild vorstellen - den TCP-Wrapper.

Entwickelt hat dieses genial einfache Hilfsmittel der schon im letzten Artikel genannte Wietse Venema aus Holland. Es ist schon relativ alt und liegt momentan in der sehr stabilen Version 7.2 vor.

Das Prinzip ist denkbar einfach. Hauptproblem der Angriffe im Internet ist die Tatsache, daß man annähernd von jedem beliebigen Rechner aus fast jeden beliebigen anderen Rechner erreichen kann. Ein normaler Computer wird im Laufe seiner Online-Zeit vielleicht mit wenigen millionstel aller anderen Rechner Kontakt aufnehmen, obwohl er in der Lage ist, dies mit allen zu tun. Das Protokoll sieht hier keine Beschränkung vor. Wenn man seine Rechner vor Zugriff aus dem Internet wirksam schützen will, hilft meist nur ein aufwendiger Firewall, der sehr teuer wird, wenn er wirkungsvoll sein soll.

In diese Bresche springt der TCP-Wrapper: Eine Netzverbindung wird entweder direkt zu einem Unix-Daemon, der zu diesem Zweck an einem bestimmten Port lauscht, aufgebaut oder der Gast wendet sich an den Internet-Daemon (`inetd`) oder `portmapper` und wird von diesem an den entsprechenden Prozeß verwiesen. Der `inetd` hat ein simples Konfigurationsfile, das es erlaubt, hier eine weitere Kontrollinstanz vorzuschieben.

Das Konfigurationsfile (`/etc/inetd.conf`) sieht typischerweise so aus:

```
telnet stream tcp nowait root /etc/telnetd telnetd -h
```

Hier wird dem `inetd` gesagt, daß der service `telnet` (er gehört zum port 23, was in `/etc/services` steht) folgendermaßen zu behandeln ist:„Wenn eine Nachfrage nach port 23 kommt, dann baue einen tcp-stream auf, warte nicht, bis er beendet ist und verbinde ihn mit dem Programm `/etc/telnetd`, das mit der Option `-h` aufzurufen ist und unter der uid `root` läuft.“ Wenn man nun aber kontrollieren will, wer da `telnet` machen will, bringt man folgende kleine Änderung an:

```
telnet stream tcp nowait root /etc/tcpd telnetd -h
```

`/etc/tcpd` ist der angesprochene TCP-Wrapper (tcp-Daemon). Er wird nun zuerst aufgerufen und erst, wenn er die Verbindung zuläßt, wird sie an den telnet-Daemon weitergegeben, der ein login veranlaßt. In dieser Form läßt sich der Zugang zu allen Diensten regulieren, die in `/etc/inetd.conf` stehen.

Wie entscheidet nun der TCP-Wrapper, wer eine Verbindung aufbauen darf? Wie fast alle Unix-Programme benötigt er dazu Konfigurationsfiles. In diesem Fall heißen sie `etc/hosts.allow` und `/etc/hosts.deny`
Die Regel heißt dann: „Verbinden darf der, der entweder explizit in `hosts.allow` zugelassen oder in `hosts.deny` nicht explizit verboten ist.“ Das Format dieser beiden Files ist ebenfalls denkbar einfach. Es besteht aus Dienst und Quellrechner oder -netz. Ein typisches `/etc/hosts.allow` - File könnte so aussehen:

```
telnetd :.rus.uni-stuttgart.de

ftpd    :.uni-stuttgart.de

fingerd : ALL
```

Hier wird der telnet-Zugang von allen Rechnern des RUS, der ftp-Zugang von allen Rechner der Universität Stuttgart erlaubt und `finger` von jedem Rechner. Ein sinnvolles `/etc/hosts.deny` sähe dann so aus:

```
ALL:ALL
```

Das heißt, daß alle außer den soeben zugelassenen Rechnern abgewehrt werden.

Will man die Sache etwas weiter spezifizieren kann man das EXCEPT-Konstrukt verwenden. Will man z.B. keine telnet-Verbindungen von öffentlich zugänglichen Studentenrechnern haben, ändert man die erste Zeile auf:

```
telnetd .rus.uni-stuttgart.de
EXCEPT 129.69.21. 129.69.31.133
```

Hierdurch werden alle telnet-Versuche vom rpool, dem PC-Pool und dem studentischen Modemserver abgewehrt, während der Rest des

RUS darf. Wenn man sich auf ganze Subnetze bezieht, läßt man einfach die entsprechenden Zahlen hinter dem Punkt weg. (`129.69.`) Wenn man sich auf ganze Domains bezieht, läßt man einfach den Rechnernamen vor dem Punkt weg (`.uni-stuttgart.de`).

Bevor man den Wrapper „scharf“ macht, sollte man mit dem mitgelieferten Programm `tcpdmatch` testen, ob die Konfiguration auch den gewünschten Schutz bringt oder ob man sich nach dem Ausloggen selbst nicht mehr einloggen kann. Ebenso gibt es ein Programm `tcpdcheck`, das nach Syntaxfehlern in den Konfigurationsfiles sucht.

Es gibt hier noch weitere Möglichkeiten, den Zugang auf Netgroups oder ähnliche Sachen zu beschränken, deren genaue Ausführung hier allerdings zu weit führen würde. Es wird eine gute man-Page mitgeliefert: `host_access(5)`. Dort ist ebenfalls die Möglichkeit beschrieben, auf bestimmte Connects mit Aktionen zu reagieren. Beispielsweise kann man einen Rechner, von dem eine abgelehnte Connection ausgeht, automatisch anfingern lassen, um die eingeloggten Benutzer herauszufinden.

Im Makefile kann man auch noch eine Option PARANOID spezifizieren, mit der der TCP-Wrapper alle Connects abweist, bei denen IP-Adresse und Hostname laut Nameserver nicht zusammenpassen, was begrenzt vor der Fälschung von Adressen oder Namen schützen kann. Ebenso kann man dem TCP-Wrapper beibringen, über den ident-Daemon beim Ausgangsrechner nach dem Username zu fragen, der die Verbindung aufbaut.

Eine wichtige Sache, die ebenfalls große Lükken in den verbreiteten Betriebssystemen schließt, ist die Logging-Fähigkeit des TCP-Wrappers. Während sich manche Betriebsysteme erst zu einer Meldung im syslog hinreißen lassen, wenn mehrfach failed logins kommen, protokolliert der TCP-Wrapper alle Verbindungen mit, über die er entschieden hat. Etwas unglücklich ist, daß er defaultmäßig in den gleichen Kanal wie sendmail loggt. Man kann dies jedoch im Makefile umbiegen und ein eigenes Logfile mit Hilfe von `/etc/syslog.conf` erzeugen, das dann etwa so aussehen würde:

```
Jan 19 09:10:01 4Q:visbl ftpd[10116]:refused connect from evil.com
Jan 19 09:10:01 4Q:visbl telnetd[10115]:refused connect from some.where.edu
Jan 19 09:10:01 4Q:visbl fingerd[10117]:refused connect from rpool2.rus
Jan 21 10:59:21 6Q:visbl rlogind[11738]:connect from friend.uni.edu
Jan 22 17:55:34 6Q:visbl rlogind[2139]:connect from friend.uni.edu
```

Man wundert sich manchmal, wer einen alles anfingert ... :-)

An zusätzlichen Features bietet das Paket dann noch einige nützliche Utilities, mit denen man z.B. suspekte Rechner sofort beim Verbindungsaufbau anfingern kann, wer eingeloggt ist oder eine Library, mit der man die Fähigkeit die Konfigurationsfiles zu lesen auch anderen Programmen beibringen kann.

Zum Artikelende noch kurz eine Zusammenfassung, was der TCP-Wrapper bietet, was er nicht kann und wo man ihn bekommt.

Was der TCP-Wrapper kann

Er kann wirksam den Zugriff auf Netzwerkdienste regulieren und protokollieren, die über `inetd` gestartet werden.

Was der TCP-Wrapper nicht kann

Er kann nicht den Zugriff auf Netzwerkdienste regeln, die standalone oder über den `portmapper` angesprochen werden. Dafür gibt es eigene Wrapper oder gepatchte portmapper. Speziellere Details stehen in der Dokumentation.

Wo man den TCP-Wrapper herbekommt

Hier zwei wichtige Adressen:

- `ftp://ftp.uni-stuttgart.de/pub/unix/security`
- `ftp://ftp.cert.dfn.de/pub/tools`

Warnung!

Dies ist keine plug-and-play-Anleitung, die zum wirksamen Schutz ausreicht. Jeder, der den TCP-Wrapper verwendet, sollte gründlich die beigefügte Dokumentation lesen und verstehen, was er tut.

Ansprechpartner bei sicherheitsrelevanten Fragen

An wen kann ich mich in so einem Fall wenden? Hier zwei wichtige Adressen und Ansprechpartner:

- `sneakers@rus.uni-stuttgart.de`
- `dfncert-request@cert.dfn.de`

Bernd Lehle
Lehle@rus.uni-stuttgart.de

Oliver Reutter
Oliver.Reutter@rus.uni-stuttgart.de

Universität Stuttgart

Viren, Würmer und Trojaner

Bernd Lehle / Oliver Reutter

Die Computerviren haben etwas an Publicity in der Presse verloren, was sie allerdings nicht ungefährlicher macht. Durch die Verbreitung virenanfälliger Betriebssysteme am RUS und den Instituten der Universität ist es daher auch für die RUS-Sicherheitsgruppe an der Zeit, sich zu diesem Thema zu äußern.

Was sind Viren?

Virus ist lateinisch und bedeutet Schleim oder Gift [6]. Diese Übersetzung trifft die Sache aber nicht ganz. Viren sind submikroskopisch (20-300 nm) kleine Gebilde, die aus einem Stück Erbinformation in Form eines DNA- oder RNA-Moleküls mit einem Proteinmantel und meistens einer Schutzhülle bestehen. Einzeln betrachtet sind sie unbelebte Kristalle aus organischem Material.

Trifft so ein Kristall auf eine Zelle in einem Organismus, legt er seine Unbelebtheit ab: Er dringt in die Zelle ein und implantiert seine Erbinformation in den Zellkern. Die Zelle interpretiert den neuen genetischen Code und produziert neue Viren, anstatt ihrer eigentlichen Aufgabe nachzugehen. Nach kurzer Zeit stirbt die Zelle, die Zellmembran platzt, und die neuen Viren suchen sich weitere Zellen. Es gibt auch Viren, die die Wirtszellen zu ungehemmter Zellteilung anregen. Wird der Stoffwechsel des betroffenen Lebewesens dabei gestört, spricht man von einer Vireninfektion. Die Auswirkungen auf den Organismus können von Unbemerktheit bis zu tödlichen Erkrankungen reichen. Einige bekannte Viruskrankheiten sind Schnupfen, Grippe, Herpes, Tollwut, Pocken und AIDS.Viren sind durch ihre einfache Struktur sehr schwer mit Medikamenten zu behandeln. Der befallene Organismus muß sie durch sein Immunsystem selbst abwehren. Dies kann Tage, Wochen, manchmal sogar Jahre dauern. Jeder Mensch trägt ständig Viren in sich, die je nach aktuellem Zustand des Immunsystems aktiv werden können.

Was sind Computerviren?

Der Begriff Computervirus oder einfach Virus hat sich in der Umgangssprache für eine ganze Gruppe von Programmen eingebürgert, die vom Fachmann als Malicious Software (böswillige Software) oder kurz Malware bezeichnet wird. Je nach Funktion wird sie als Virus, Wurm, Trojanisches Pferd, Logische Bombe oder Hoax bezeichnet. Die Charakteristika seien kurz erläutert:

1. Viren sind Code-Fragmente, die sich an andere Daten anhängen und sich bei deren Ausführung oder Verarbeitung vermehren. Die Daten können Programme, Bootsektoren oder Dokumente sein. Für sich alleine ist eine Computervirus meist nicht reproduktionsfähig. Die Analogie zu den biologischen Viren liegt auf der Hand.
2. Würmer [1] sind komplette Programme, die sich aktiv fortbewegen und vermehren können. Das bekannteste Beispiel ist der Internet-Wurm, der sich 1988 in kürzester Zeit über das damalige Internet verbreitete. Würmer sind relativ selten.
3. Trojanische Pferde [4] sind Programme, die sich äußerlich wie normale Anwendungssoftware verhalten, intern aber Anweisungen enthalten, die Schaden anrichten können. Trojanische Pferde können sich nicht selbst vermehren, sondern werden von Anwendern kopiert.
4. Logische Bomben sind Programmfragmente, die von Entwicklern in Betriebssystemen oder Anwendungsprogrammen versteckt werden und bei Eintreten von bestimmten Bedingungen (beispielsweise Datum, Systemaktivität, etc.) anlaufen und Schaden anrichten. Diese Gruppe überschneidet sich teilweise mit allen vorher genannten, hat aber auch reine Vertreter, die zu keiner anderen Kategorie passen. Logische Bomben können sich nicht vermehren.

5. Hoaxes sind falsche Meldungen über Malware. Sie warnen vor Bedrohungen, die nicht existiert. Dadurch wird zwar kein Schaden angerichtet, aber das Beseitigen kann genauso viel Arbeit kosten wie bei echter Malware. Beispiele sind die in letzter Zeit verschickten Warnungen vor den angeblichen Viren Good Times und Penpal, die einige Mailing-Listen verstopften.

Diese Charakteristika sind nicht als Definitionen gedacht, sondern als grobe Einteilung dessen, was einem passieren kann. Es wird immer Programme geben, die in mehrere Kategorien passen und immer welche, die in keine passen. Es gibt auch genug gut gemeinte Software, die unter gewissen Bedingungen in eine der Kategorien fällt.

Wo kommt Malware her?

Malware ist natürlich keine Mutation von normaler Software, sondern wird gezielt von Spezialisten programmiert. Zum Programmieren eines überlebensfähigen Computervirus oder eines Wurmes gehören sehr hohe Fachkenntnis und Wissen über das zugrundeliegende Betriebssystem. Daher gibt es wohl nur wenige Programmierer, die selbstständig solche Programme entwickeln können. Was häufiger auftritt, sind sogenannte Mutationen, bei denen ein weniger erfahrener Programmierer eine bestehende Spezies abändert, ihr z. B. eine neue Botschaft oder Aktion mitgibt. Trojanische Pferde und Logische Bomben sind sehr einfach zu programmieren, da man die böswilligen Aktionen nur in beliebigen anderen Source Code einfügen muß. Ebenfalls sehr leicht zu programmieren sind Makroviren, da diese in mächtigen, leicht verständlichen Hochsprachen geschrieben sind.

Die Motivationen, Malware zu schreiben und im Umlauf zu setzen, sind schwer herauszufinden, da man von den meisten Beispielen höchstens das Herkunftsland kennt. Der klassische elektronische Vandalismus gehört sicher dazu wie politische oder gesellschaftliche Motive. So wird vermutet, daß der Israel-Virus von Sympathisanten der PLO programmiert wurde, um israelische Computer lahmzulegen. Der Stoned-Virus verbreitet die Botschaft, Marihuana zu legalisieren. Die meisten Viren wurden aber wahrscheinlich aus Abenteuerlust oder Geltungsdrang programmiert. Es gibt auch Fälle, in denen sich Forschungsprojekte verselbständigt haben.

Welche Computer und Betriebssysteme sind bedroht?

Malware ist grundsätzlich auf jedem Betriebssystem denkbar. Je stärker das Betriebssystem jedoch seine Ressourcen kontrolliert, desto weniger Schaden kann angerichtet werden. Die klassischen Opfer sind daher die Single User Desktop-Systeme. Viren und Trojanische Pferde sowie von ihnen angerichtete Schäden gibt es vor allem auf den Systemen Amiga, Atari, Macintosh, MS-DOS und OS/2. Windows 3.X und 95 sind lediglich grafische Benutzeroberflächen von MS-DOS und haben auf die Funktion systemnaher Malware keinen Einfluß.

Auf Mehrbenutzersystemen wie UNIX, Windows NT oder VMS ist systemnahe Malware praktisch unbekannt. Hier tritt hauptsächlich anwendungsnahe Malware wie Word- oder Excel-Makroviren (NT), der Internet-Wurm (UNIX) oder der IBM Christmas Trojaner (VM/CMS) auf. Für Linux wurden in letzter Zeit verstärkt Trojanische Pferde mit wurmähnlichen Eigenschaften gefunden.

Funktionsweise von Malware

Viren

Wie bereits erwähnt, hängen sich Viren, ihren biologischen Vorbildern folgend, an bestehende Software an und vermehren sich durch Benutzung dieser Software. Es gibt drei Haupttypen von Viren, die man unterscheiden muß:

- **Boot-Sektor-Viren:** Wie Ihr Name schon andeutet, befällt diese Art von Viren den Boot-Sektor von Disketten oder Festplatten. Der Boot-Sektor besteht aus wenigen Bytes, die beim Boot-Vorgang als erstes vom BIOS in den Speicher geladen und ausgeführt werden. Der Virus ersetzt Teile des Boot-Sektors durch eigenen Code oder Zeiger auf eigenen Code und wird immer dann aktiv, wenn versucht wird, von der Diskette oder Festplatte zu booten. Der Virus kopiert sich dann in den Speicher und infiziert von dort aus weitere Disketten oder Festplatten. Diese Viren können sich unabhängig vom Betriebssystem auf alle Boot-Sektoren

setzen. Sogar eine Diskette, die gar nicht bootfähig ist und nur vor dem Booten im Laufwerk vergessen wurde, kann einen Boot-Sektor-Virus verbreiten. Auch eine Linux-Boot-Diskette oder der Boot-Sektor einer NT-Festplatte können infiziert sein. Der Virus kann sich dann allerdings nicht ausbreiten, da er das Umschalten des Kernels in den Protected Mode nicht überlebt. Er könnte aber vorher eine Diskette im zweiten Laufwerk infizieren.

- **Dateiviren:** Diese Viren befallen ausführbare Dateien. Dabei kopiert der Virus sich selbst oder eine Sprunganweisung auf sich selbst an den Anfang der Datei und wird immer dann ausgeführt, wenn die Datei aufgerufen wird. Die Dateien werden dadurch verlängert, was sich u.U. im Verzeichniseintrag bemerkbar macht. Manche Viren manipulieren allerdings diese Einträge.
- **Makroviren:** Diese Spezies ist relativ jung. Sie trat als erstes bei dem Textverarbeitungsprogramm Word auf, das eine an BASIC angelehnte Makrosprache hat. Mittlerweile gibt es sie auch bei anderen Dokumenten, die in der Lage sind, Informationen in Makros abzulegen. In der Vorgehensweise unterscheiden sich Makroviren kaum von Dateiviren. Der Virus-Code wird am Anfang beim Laden des Makros ausgeführt, kopiert sich dann auf andere Dokumente und erfüllt u. U. noch eine einprogrammierte Aufgabe. Diese Viren sind besonders gefährlich, da sie sich schnell verbreiten, wenn infizierte Dokumente mit Electronic Mail verschickt werden. So schaffte es der erste Word-Makrovirus Concept schon sechs Monate nach seiner Entdeckung, der häufigste Virus überhaupt zu sein. Makroviren sind anders als andere Viren leicht zu verstehen und zu programmieren. So besteht der auch hier am RUS schon aufgetretene Makrovirus NOP:DE aus fünf Zeilen Word-Makro-BASIC. Er kann von jedem halbwegs qualfizierten Programmierer beliebig verändert werden.

Zusätzlich zu diesen grundsätzlichen Vorgehensweisen haben Virenprogrammierer im Lauf der Zeit Techniken entwickelt, Viren vor Entdekkung und Bekämpfung zu schützen. So gibt es Viren, die Ihre Anwesenheit durch Manipulation von Verzeichniseinträgen tarnen (Stealth-Viren). Ebenso gibt es Viren, die ihren Code von Generation zu Generation ändern können (polymorphe Viren). Besonders tückisch sind Viren, die in verschlüsselter Form vorliegen und sich als erste Aktion selbst entschlüsseln, bevor sie aktiv werden (Crypto-Viren). Eine besondere Art Boot-Sektor-Viren benutzt das CMOS-RAM im BIOS als Speicher für Viren-Code.

Sehr unterschiedlich sind die sogenannten Nutzlasten (Payloads), die die Viren zusätzlich zu ihrem Reproduktionscode mit sich herumtragen. Die friedlichsten Viren tun gar nichts (z.B. der Word-Makro-Virus NOP, was für No OPeration steht). Dies kann allerdings schon gefährlich werden, wenn bei der Reproduktion irrtümlich Daten überschrieben werden, die dann verloren gehen. Diese Gefahr besteht vor allem im Boot-Sektor. Weniger freundliche Viren machen sich durch Textbotschaften bemerkbar (z.B. “Your Computer is now stoned - legalize Marihuana!“ vom Stoned-Virus). Andere Viren benutzen dazu den Lautsprecher (Yankee Doodle oder Oh Tannenbaum) oder lassen die Buchstaben wie Blätter im Herbst vom Bildschirm fallen (Herbst-Virus alias Cascade). Eine unangenehmere Spezies von Viren simuliert Hardware-Defekte (Parity Boot Virus) oder läßt den Rechner abstürzen. Die übelsten Viren löschen Daten oder komplette Festplatten, wenn sie aktiv werden.

Eine neue Qualität von Viren nutzt Online-Dienste aus, um Schaden anzurichten. Vom Chaos Computer Club wurde ein Virus INFEKT.EXE entwickelt, der gezielt Internet Banking Software stört, indem er Netscape befällt und auf Überweisungen Einfluß nimmt, bevor diese verschlüsselt werden können. Angeblich wird der Empfänger der jeweils letzten Überweisung durch Amnesty International ersetzt. Dieser Virus wurde dokumentiert als Anschauungsobjekt verbreitet, so daß noch keine Berichte über seine Wirksamkeit vorliegen.

Würmer

Diese Kategorie von Malware ist sehr selten und schwer zu programmieren. Eine einheitliche

Funktionsweise läßt sich schwer definieren. Wir wollen uns daher exemplarisch auf einen Wurm beschränken, der in freier Wildbahn beobachtet wurde.

Es ist der berühmt-berüchtigte Internet-Wurm, der auch schon in früheren Artikeln erwähnt wurde. Er war wohl als Testprojekt gedacht, eine Art künstliche Lebensform im Internet. Leider hat sein Entwickler die Sicherheit der Rechner im damaligen Internet völlig überschätzt. Anstatt langsam durch das Netz zu kriechen, explodierte die Wurm-Infektion innerhalb weniger Stunden und konnte erst nach etwa einer Woche gestoppt werden. Nach Aussagen der Spezialisten, die den Wurm bekämpften, war das Programm noch unvollständig und wahrscheinlich nur eine Testversion.

Die Funktion war denkbar einfach. Der Wurm, dessen Sourcen heute offen liegen, besteht aus einem aktiven Programm namens `sh,` das sich sofort aus der Prozeßtabelle streicht, wenn es anläuft. Durch Analyse von Routing Tables, `/etc/hosts` und NIS-Information sucht sich der Wurm potentielle Opfer. Diese werden dann durch drei verschiedene Sicherheitslücken angegriffen. Führt einer der Angriffe zum Ziel, wird der Wurm auf den Rechner kopiert und versucht von dort erneut, andere Rechner zu infizieren. Seine Hauptwaffe ist ein damals weit verbreiteter Buffer Overflow Bug im finger-Daemon, dem die meisten Rechner zum Opfer fielen. Kam er damit nicht durch, versuchte er mit dem DEBUG-Loch von sendmail (s. BI. 3/4 97) einzubrechen. Schlug auch dies fehl, wurde ein Brute Force-Angriff auf schlechte Paßwörter gemacht (s. BI. 1/2 97). Ab und zu schickte er ein paar bytes an `ernie.berkley.edu`, was schließlich auf den Programmierer führte. Robert Morris jr., ein 23jähriger Doktorand an der Cornell University und Sohn des Leiters des National Computer Security Institute, der Öffentlichkeitsabteilung der National Security Agency.

Als positive Konsequenz des Wurms wurde das erste CERT (Computer Emergency Response Team) gegründet. Dies war wohl hauptsächlich eine Konsequenz der Plan- und Hilflosigkeit der betroffenen Benutzer.

Wie weit dieser Wurm im heutigen Internet käme, möchte ich an dieser Stelle lieber nicht abschätzen. Im Stile der Sicherheitstestprogramme SATAN und ISS ist es aber technisch sicher möglich einen Wurm zu programmieren, der ähnliche Effekte wie der ursprüngliche Internet-Wurm hätte.

Eine Abart der Würmer sind die Kaninchen [1]. Sie sind die älteste dokumentierte Form der Malware. Es handelt sich dabei um kleine Jobs, die auf frühen Großrechnern in den Job Queues lagen. Kamen sie an die Reihe, schauten sie nach, ob der Programmierer einen Auftrag für sie hatte, den sie dann ausführten. War das das nicht der Fall, kopierten sie sich wieder an das Ende der Job Queue und stellten so einen permanenten Platzhalter dar, der einem schnellen Zugang zur Job Queue sicherte. Insbesondere, wenn die Queue nach dem Shortest-Job-Next-Prinzip funktionierte, konnte man sich mit einem kleinen Kaninchen am Job Scheduler vorbeimogeln, das dann einen wesentlich größeren Job startete.

Trojanische Pferde und Logische Bomben

Der Vergleich mit dem historischen Vorbild [4] paßt hier ganz gut. Dem Benutzer wird ein äußerlich harmloses, oft sogar attraktiv erscheinendes Programm vorgespiegelt, daß beim Start Unheil anrichtet. Oft sind die bösartigen Teile unmerklich in normalem, funktionierenden Code versteckt, so daß sie erst später bemerkt werden. Von den Viren und Würmern unterscheiden sie sich durch die Tatsache, daß der Benutzer aktiv eingreifen muß, um das Trojanische Pferd zu starten. Manche Trojanische Pferde können sich fortpflanzen, wie das folgende Beispiel verdeutlicht.

Ein bekanntes Trojanisches Pferd auf dem als sehr sicher geltenden Mainframe-Betriebssystem VM/CMS war Christmas. Es wurde als Mail verschickt und zeichnete einen Christbaum auf den Bildschirm. Dann wurde der Benutzer aufgefordert, die angehängte Datei XMAS EXEC auszuführen. Diese war eine in der Programmiersprache REXX geschriebene Prozedur, die den NAMES-File (Datei mit Mail-Aliases unter VM/CMS) auslas und sich an alle dort aufgeführten Benutzer weiterverschickte. Das Pferd kam vermutlich aus Deutschland und schaffte es immerhin, die komplette VM/CMS-Infrastruktur von IBM für 72 Stunden lahmzulegen.

Ein anderes Beispiel ist AOLGOLD.ZIP, das als Upgrade für die Zugangs-Software von America Online (AOL) getarnt per E-Mail verschickt wurde. Ausgepackt und ausgeführt löschte das Programm alle AOL-Software von der Platte und gab einige unschöne Kommentare zu AOL auf dem Bildschirm aus. Es gibt bereits eine Nachfolgeversion, die sowohl als Hoax wie auch als echter Trojaner existiert und AOL4FREE heißt.

Trojanische Pferde sind sehr einfach zu programmieren. Jeder, der Zugriff zu Source oder Object Code eines Programmes hat, kann dort ein Trojanisches Pferd einbauen. Besonders gefährdet ist daher Freeware im Internet.

Eine weitere beliebte Art, Trojanische Pferde unter die Leute zu bringen, sind böswillige Links, Java Applets oder ActiveX Controls auf Web-Seiten. Java hat zwar ein sicheres Design, allerdings fehlt es oft an der Sicherheit der Implementierung. Der Schaden, den böswillige Java Appplets anrichten können, ist relativ begrenzt, da der Zugriff auf kritische Bereiche in der Sprache gar nicht vorgesehen ist.

ActiveX hat ein völlig unsicheres Design, da es auf lokalen Betriebssystem-Calls beruht und vollen Zugriff auf den Rechner hat. Das heißt, ein ActiveX Control ist nichts anderes als ein ausführbares Programm, das aus dem Web geladen wird. Die Implementierung von irgendwelchen Schutzmechanismen ist daher prinzipiell nicht möglich. ActiveX-fähige Browser wie der Microsoft Internet Explorer führen bereitwillig auf Mausklick jedes beliebige Trojanische Pferd aus, das man auf eine Web-Seite packt. Davor ist man auch durch Abschalten von ActiveX und Erhöhen der Sicherheitsstufe nicht sicher, denn der Explorer hat noch eine ganze Menge anderer Schwächen, die auch ohne ActiveX vollen Systemzugriff erlauben. Die permanente Installation von Malware auf dem Rechner ist damit sehr einfach. Die Vergabe von Zertifikaten an ActiveX Controls durch Microsoft und Partnerfirmen ist dabei reine Augenwischerei. Für wenige Dollars sind diese Zertifikate zu erhalten und schützen absolut nicht vor negativen Folgen.

Eindrucksvoll demonstriert wurden die möglichen Auswirkungen dieser Schwächen Anfang des Jahres vom Chaos Computer Club. Dort wurde ein ActiveX Control entwickelt, das im Hintergrund das Telebanking-Programm Quicken startet und eine Überweisung von DM 20,-- einfügt, die beim nächsten Login zusammen mit den anderen Überweisungen an die Bank geht. Details sind in der Zeitschrift iX in der Ausgabe 3/1997 nachzulesen. Eine weitere gute Beschreibung der horrenden Sicherheitslöcher von ActiveX findet sich auch in der Mai-Ausgabe von Internet professionell (früher pl@net). Einige Firmen gehen schon soweit, daß ActiveX durch das Sicherheitskonzept grundsätzlich verboten ist und dessen Verwendung damit zum Kündigungsgrund wird.

Logische Bomben lassen sich meist sehr schwer von den bisher beschriebenen Verwandten unterscheiden. Klassische Vertreter werden in umfangreichen Software-Paketen versteckt und treten erst nach langer Zeit durch Eintreten bestimmter Begleitumstände in Aktion. Ein friedliches Beispiel ist die Angewohnheit mancher Entwickler, durch bestimmte Tastenkombinationen einen graphischen Gruß vom Entwickler-Team ablaufen zu lassen. Eine unfreiwillige Logische Bombe ist die Nachlässigkeit der Software-Entwickler bei der Berücksichtigung des Jahres 2000. Die meiste Software kann Jahreszahlen nur zweistellig verwalten, was bei Altersberechnungen zu Fehlern von 100 Jahren führt. Diese Bombe wird am 1. Januar 2000 explodieren und nach Schätzungen von Experten weltweit einige hundert Milliarden US$ kosten.

Logische Bomben werden weniger im Massenbereich eingesetzt. Ihre Hauptanwendung liegt in großen Softwaresystemen von Banken oder anderen Großunternehmen, da dort wesentlich mehr Geld zu holen ist.

Abwehr von Malware

Malware verursacht weltweit etwa doppelt so viel finanziellen Schaden wie Einbrüche in fremde Systeme [5]. Daher wollen wir auch auf die Möglichkeiten hinweisen, sich vor Schaden zu schützen.

Viren

Der beste Schutz gegen Computerviren ist, wie auch gegen die biologischen Pendants, die Enthaltsamkeit, d. h. der bewußte Verzicht auf Software oder Datentransfer, die durch Viren bedroht sind. Dies hat natürlich nur Sinn, wenn sie im Rahmen eines Sicherheitskonzeptes auch konsequent durchgeführt wird und Alternativen zu bedrohter Software bereitstehen. Am Ende dieses Artikels sind einige konkrete Beispiele aufgeführt.

Wenn man gezwungen ist, Software zu verwenden, bei der Datenverlust durch Virenbefall nicht auszuschließen ist, muß man natürlich anders vorgehen. Seit einiger Zeit hat sich eine große Menge Antiviren-Software etabliert, die auf verschiedenste Weise versucht, Daten vor Viren zu schützen. Es gibt im wesentlichen zwei Arten von Vorgehensweisen, die diese Programm verfolgen:

1. **Virenscanner**: Diese Programme durchsuchen Datenmengen oder Datenströme auf Muster, die sie in einer Datenbank gespeichert haben. Wird eine Übereinstimmung erkannt, zeigt

sie das Programm an und versucht danach, den Virus so zu entfernen, daß die ursprüngliche Funktionalität wieder hergestellt wird. Die Hauptanwendungen sind Scanner für Dateisysteme oder Scanner für Netzverbindungen wie E-Mail, FTP oder WWW. Einige Systeme bedienen sich sogenannter heuristischer Methoden, die auch nicht gespeicherte Viren erkennen sollen.

2. **Virenschilde**: Diese Programme laufen im Hintergrund und überwachen Systemkomponenten oder Betriebssystemschnittstellen , auf Anzeichen viraler Aktivität oder bekannter Virenmuster. Werden diese gefunden, schlägt das Programm Alarm und versucht den auslösenden Prozeß zu stoppen, bzw. den Verursacher zu finden und den Virus zu entfernen.

Die Virenscanner betreiben reines Pattern Matching und sind immer nur so gut, wie ihr aktuelles Virenverzeichnis. Für einige Viren sind speziellere Verfahren nötig, da ihr Code emuliert werden muß, um polymorphe, getarnte oder verschlüsselte Viren zu finden. In von unabhängigen Spezialisten durchgeführten Tests [7] erkennen gute Virenscanner im wesentlichen alle (95-100 %) in der freien Wildbahn auftretenden Datei- und Boot-Sektor-Viren. Die Erkennungsrate von Makroviren liegt meist weit darunter (60-80 %).

Die Schilde können unabhängig von ihrer Datenbank Funktionen überwachen, die für Viren typisch sind und so Viren, die nicht bekannt sind, zumindest stoppen und melden. Leider fallen diesen Schilden auch immer wieder schlecht programmierte normale Programme zum Opfer.

Wer Antivirus-Software zuverlässig einsetzen will, der sollte folgende Regeln beachten:

- Wenn irgend möglich mehrere unabhängige Scanner verwenden, die regelmäßig eine aktuelle Virendatenbank erhalten
- Jede neue Diskette und jedes per Netzwerk geladene Programm muß zuerst gescannt werden. Manche Scanner machen dies automatisch, bevor sie einen Datenträger freigeben
- Ein Virenschild sollte installiert sein
- In regelmäßigen Abständen (z.B. jede Woche oder jeden Monat) sollten alle Datenträger gescannt werden
- Ein regelmäßiges Backup schützt vor Datenverlust

Würmer

Da Würmer zu ihrer Fortpflanzung nicht auf die Mithilfe von Benutzern angewiesen sind, kann er wenig tun, um sie abzuwehren. Am Beispiel des Internet-Wurms wird deutlich, daß hier hauptsächlich die allgemeine Systemsicherheit verbessert werden muß. Die Wurmabwehr ist am ehesten mit der Abwehr von Systemeinbrüchen zu vergleichen.

Trojanische Pferde

Trojanische Pferde sind immer auf die Mitwirkung der Benutzer angewiesen. Daher sind sie am besten durch vorsichtigen Umgang mit unbekannter Software abzuwehren. Dies fängt damit an, daß bei der Verwendung von Freeware aus dem Internet vorher ein Viren-Scanner benutzt wird, da diese oft auch Trojanische Pferde finden. Dasselbe gilt für Virenschilde. Desweiteren sollte der Benutzer bei jeder ungewöhnlichen Aufforderung, ein Programm abzuspeichern oder auszuführen, extrem vorsichtig sein.

Manche Entwickler von Freeware liefern zu ihrem Source Code eine digitale Signatur, mit der der Code auf Veränderungen überprüft werden kann. Das sollte, wenn möglich, genutzt werden.

Logische Bomben

Diese Spezies fällt entweder unter eine der anderen Kategorien oder ist derart exotisch und versteckt, daß es keine generellen Abwehrstrategien gibt. Es gilt hier wie allgemein, daß man sensibel gegenüber seltsamem Verhalten sein muß.

Was tun, wenn Malware aktiv wird?

Wenn der Benutzer merkt, daß Malware aktiv wurde, ist es meistens schon zu spät. Die auffälligen Zeichen von Malware wie Meldungen auf dem Bildschirm oder das Abspielen von Liedern über den Lautsprecher bilden fast immer den Abschluß der zerstörerischen Arbeit, so daß man lieber auf subtilere Zeichen achten sollte.

Einige dieser Zeichen sind:

- Ungewöhnliche Verlängerung von Dateien
- Ungewöhnliche Aktivitäten von Festplatten oder Diskettenlaufwerken
- Auffällige Verzögerungen beim Ausführen von Programmen
- Probleme beim Booten von mehreren Betriebssystemen

Leider gibt es genug Programme und Betriebssysteme, bei denen solche Anzeichen zum normalen Betrieb gehören.

Wird das Vorhandensein von Malware vermutet, empfehlen wir folgende Vorgehensweise:

1. Ruhe bewahren, es kann immer noch falscher Alarm sein
2. Rechner vom Netzwerk trennen, falls der Verdacht besteht, die Malware käme von dort oder könne sich so weiterverbreiten
3. Versuchen, wichtige Daten vom Speicher auf Platte zu sichern
4. Rechner möglichst bald ausschalten (DOS/Windows sofort, NT/UNIX nach Shut Down). Bei Makroviren das Anwendungsprogramm beenden
5. Booten des Rechners durch saubere Systemdiskette (vorher anlegen !). UNIX sollte im Single User Mode gebootet werden
6. Check der Filesysteme und des Boot-Sektors auf Befall durch Malware
7. Suche nach der Quelle (Netz, Mail, Disketten) und Bekämpfung der Malware dort
8. Information von weiteren möglichen Betroffenen, Daten, Fakten, Namen

Nach den manchmal recht theoretischen Ausführung hier noch einige konkrete Tips:

Grundsätzlich: Niemand sollte sich vor Malware sicher fühlen. Durch die Einführung von weltweiter Kommunikation und die Steigerung der Komplexität von Betriebssystemen und Anwendungen steigt die Anfälligkeit. Die Personen, die vor fünf Jahren Viren programmiert haben, über die man heute lacht, entwickeln heute mit ziemlicher Sicherheit Dinge, die uns so treffen können wie damals die ersten Viren. Der Phantasie sind wenig Grenzen gesetzt!

Einige Zahlen

(Stand April 1997, Quellen [2], [5], [7]:

- Anzahl der Dateiviren (inkl. Mutationen und Abarten): ca. 10 000
- Anzahl Boot-Sektor-Viren: ca. 800
- Anzahl Makroviren: MS Word ca. 350, andere Programme ca. 10
- Anteil der Viren am Gesamtschaden im Computerbereich: 4 %

Für Malware anfällige Software

- **Extrem anfällig, nicht verwenden!:** Microsoft Internet Explorer und alle anderen Browser, die ActiveX unterstützen
- **Stark anfällig, vorsichtig verwenden oder Alternativen suchen**: Microsoft Word ab Version 5 insbesondere in Verbindung mit E-Mail, MS-DOS in den Versionen 2.0 bis 7.0 (Windows 95), PC-DOS, DR-DOS
- **Software für die Makroviren existieren**: Word, Excel, PowerPoint, Access, Lotus Notes, Lotus 123, AmiPro, Ghostscript. Windows-Viewer für das Portable Document Format (PDF) sind ebenfalls anfällig für Malware, da man mit Ihnen sogar Programme starten kann [9]. Reine Word-Viewer (z.B. Netscape Plugins) sind vor Makroviren bisher sicher

Qualitativ hochwertige Viren-Scanner

Testergebnissen des Virus Test Centers der Universität Hamburg [7] zufolge sind nachfolgende als qualitativ hochwertige Viren-Scanner beurteilt worden:

Dr. Solomon Antivirus 768, AVP (Kamis) 2.2, AVAST! 77/1 (Alwil), Alert41/15 (Look) Sweep 294 (Sophos), F-PROT 2.25 (Data Fellows), Scan 2.53 (McAffee)

Scan 2.5.3 von McAffee ist als Campuslizenz erhältlich. Aus eigener Erfahrung können wir noch das Antivirenpaket von Pandasoft empfehlen.

Bekannte Hoaxes

Irina, Good Times, Ghost, Deeyenda, Penpal Greetings, Make Money Fast, Naughty Robot, AOL4FREE (Vorsicht, auch als echtes Trojanisches Pferd vorhanden!).

Literatur

[1] Brehm, A.: Das neue Tierreich nach Brehm, Meyer ,1973

[2] Cameron, D.: Security Issues for the Internet and the World Wide Web, CTR, 1996

[3] Hofmann, M.: Viren erkennen und beseitigen, Falken Verlag, 1990

[4] Homer: Ilias, 9. Jhd. vor Christus

[5] Icove, D. et.al.: Computer Crime, O‘ Reilly & Associates, 1995

[6] Pschyrembel, W.: Klinisches Wörterbuch, 257. Auflage, Walter de Gruyter, 1994[7] Virus Test Center Universität Hamburg

http://agn-www.informatik.uni-hamurg.de/vtc/navdt.htm

[8] Ferbrache, D.: A Pathology of Computer Viruses, Springer, 1992

[9] Kuri, J.: Und Äktschn! Sicherheitsrisiko Acrobat PDF, c‘ t 6/97, S. 48, Heise

http://www.ix.de/ct/art_ab97/9706048/

[10] Virus Help Munich:

http://www.vhm.haitec.de/

Ansprechpartner in sicherheitsrelevanten Fragen

- sneakers@rus.uni-stuttgart.de
- dfncert-request@cert.dfn.de

Bernd Lehle
Lehle@rus.uni-stuttgart.de

Oliver Reutter
Oliver.Reutter@rus.uni-stuttgart.de

Universität Stuttgart

Tripwire

Bernd Lehle / Oliver Reutter

„Er ... ging in die Küche und näherte sich vorsichtig dem Kühlschrank. Dirk kauerte sich vor den Kühlschrank und inspizierte eingehend den Rand der Tür. Er fand, wonach er suchte. Besser gesagt, er fand mehr als wonach er suchte. Nahe der Unterseite der Tür, auf dem schmalen Spalt, der die Tür von dem eigentlichen Kühlschrank trennte und der den grauen Dichtungsgummistreifen enthielt, lag ein einzelnes menschliches Haar. Es war dort mit getrockneter Spucke befestigt. Das hatte er erwartet. Er hatte es selbst vor drei Tagen dort hingeklebt und seitdem mehrere Male nachgesehen. ...“Auszug aus *Der lange dunkle Fünfuhrtee der Seele* von Douglas Adams)

Bisher haben wir hauptsächlich Tools vorgestellt, die helfen, das eigene System gegen Angriffe sicherer zu machen. Etwa einmal im Monat stehen wir allerdings als Arbeitsgruppe Systemsicherheit vor dem Problem, daß es irgendwo an der Universität Stuttgart jemandem gelungen ist, vorhandene/nicht vorhandene Sperren eines Rechners zu überwinden und dort Systemprivilegien zu erhalten. Die betroffenen Systembetreuer stehen dann meist vor dem Problem, ihre Maschinen so schnell wie möglich wieder sicher und benutzbar zu machen.

Das kann sehr schnell in eine Sisyphos-Arbeit ausarten, da gewitzte Angreifer sofort nach dem Eindringen Hintertüren installieren, die ihnen in Zukunft den Zugang zu dem betroffenen Rechner leichter machen, selbst wenn ihre erste Einstiegsluke entdeckt und gestopft wurde. Es gibt bereits fertige Programmpakete (root kits) die veränderte Versionen einiger wichtiger Systemdateien (`ifconfig`, `telnetd`, etc.) enthalten, mit denen man in Minutenschnelle ein einmal geknacktes System mit zahlreichen weiteren Löchern versehen kann. Die veränderten Versionen sind oft auf den ersten Blick nicht von den Originalen zu unterscheiden.

Was kann nun der geplagte Systembetreuer tun, wenn er unter diesem Damokles-Schwert sein System so schnell wie möglich wieder flott bekommen will ?

Die sicherste Methode ist natürlich die Neuinstallation von einem sauberen Boot-Medium. Das einzige Problem liegt dann darin, sicherzustellen, daß sich unter den Benutzerdaten kein trojanisches Pferd in Gestalt eines `.rhosts`-Files oder eines SetUID-Programmes befindet.

Eine elegantere Methode ist es, die Dateigrößen und Zugriffsdaten mit dem letzten Backup zu vergleichen, bzw. diesen einfach wieder einzuspielen. Problematisch ist hierbei, daß man nicht sicher sein kann, was der letzte saubere Backup ist, wenn der Angriff erst Wochen oder Monate später entdeckt wurde. Außerdem ist es für qualifizierte Angreifer kein Problem, Dateigröße, Zugriffszeiten und sogar einfache Prüfsummen der veränderten Programme gezielt zu fälschen.

Wenn man sich allerdings schon vor einem solchen Angriff mit dem Tag X auseinandersetzt, gibt es ein Programm, daß einem im Ernstfall viel Arbeit sparen kann und die Angriffswege und Hintertüren aufdecken hilft. In Analogie zu unserer einleitenden Story wurde es von seinen Autoren `tripwire` (Stolperdraht) genannt.

`tripwire` durchsucht einen bestimmten Teil der Dateisysteme nach allen Dateien und speichert wichtige Informationen wie Zugriffszeiten, Inodes und Längen ab. Zusätzlich werden von jeder Datei zwei unabhängige kryptographische Prüfsummen (snefru und MD-5) berechnet. Im Gegensatz zu der Standard-Unix-Prüfsumme `sum` sind diese kryptographischen Summen nicht gezielt fälschbar. Sollte es doch mit einer der beiden gelingen, was einen mehrmonatigen Rechenaufwand bedeuten würde, steht die andere als Backup zur Stelle.

So kann nach einmaligem Aufbau einer Datenbank immer wieder genau nachvollzogen werden, welche Dateien verändert wurden.

Installation von tripwire

Am besten besorgt man sich die aktuelle Version:

```
ftp://ftp.uni-stuttgart.de/pub/unix/security/Tripwire-1.2.tar.gz
ftp://ftp.cert.dfn.de/pub/tools/admin/Tripwire/Tripwire-1.2.tar.gz
```

Nach dem Auspacken mit `gunzip` und `tar` sollte im `Makefile` sichergestellt werden, daß die Definitionen für C-Compiler und Optionen richtig gesetzt sind. Tips für die richtigen Einstellungen befinden sich in der `Ported`-Datei.

Dann sollte im Verzeichnis `./configs` nach einem passenden `conf-<os>.h` ge-sucht werden, das am besten zum lokal verwendeten Betriebssystem paßt. Dort wird haupsächlich festgelegt, welche Teile des Dayeisystems mit einbezogen werden. Teile, die sich im Normalbetrieb oft ändern wie Spoolbereich, Logfiles oder Benutzerdaten müssen natürlich ausgeklammert werden.

Nun wird `./include/config.h` den lokalen Gegebenheiten entsprechend angepaßt. Hier einige Auszüge aus einer Beispielinstallation:

```
#include "../configs/conf-svr4.h"
#define CONFIG_PATH     "/opt/sonst/tripwire-1.2/bin/databases"
#define DATABASE_PATH   "/opt/sonst/tripwire-1.2/bin/databases"
```

In dieser Datei wird auch festgelegt, wo die Datenbank für `tripwire` später angelegt wird. Von den Autoren des Programms wird empfohle, die Pfade, die in `CONFIG_PATH` und `DATABASE_PATH` spezifiziert sind, auf eine Partion zu legen, die nur lesbar ist, um Modifikationen entgegenzuwirken. Am sichersten ist natürlich die Lagerung der Daten auf Disketten, zumal die Datenbanken meist klein genug sind, um auf eine Diskette zu passen.

Die eben erwähnte Konfigurationsdatei muß dann unter dem Namen `tw.config` im Verzeichnis `CONFIG_PATH` stehen.

Ein Auszug aus einem `tw.config` file:

```
# Unix itself
/kernel/unix                    R
# Now, some critical directories and files
# Some exceptions are noted further down
/dev                          L
/devices                      L
=/devices/pseudo              L
/etc                          +pnugsm12-iac
```

Hierbei besteht ein Eintrag pro Zeile aus zwei Feldern, die durch ein TAB getrennt sind. Das erste Feld bezeichnet die Datei oder das Verzeichnis, das von `tripwire` untersucht wird, wobei man auch mit Hilfe der Operatoren `!` und `=` Dateien und Verzeichnisse ausschließen kann. Im zweiten Feld gibt man an, welche Dateiattribute in die Datenbank aufgenommen werden. Dies kann detailliert erfolgen, wie es bei `/etc` demonstriert ist oder man kann die vordefinierten Makros R, L, N und E benutzen.Die genaue Bedeutung der Operatoren, Modifikatoren und Makros ist im Vorspann der Beispielconfig-Dateien erläutert.

Zum Schluß ist eigentlich nur noch `make` auf der obersten Ebene des `tripwire`-Dateibaums aufzurufen.

Benutzung von `tripwire`

Bevor überhaupt ein Vergleich mit den gespeicherten Daten möglich ist, muß mit dem Kommando eine Referenzdatenbank erzeugt werden:

```
# ./tripwire -initialize
### Phase 1: Reading configuration file
### Phase 2: Generating file list
### Phase 3: Creating file information database
```

Jetzt wird eine Datei `tw.db_[hostname]` generiert. Sie wird an der in der Variable `DATABASE_PATH` angegebenen Stelle abgespeichert.

Bei den regelmäßigen Kontrollen (am besten wöchentlich) startet man `tripwire` einfach mit

```
# ./tripwire
```

oder

```
# ./tripwire -interactive
```

Im ersten Fall wird nur über gefundene Unterschiede berichtet, was dann auch in regelmäßigen cron-Jobs gemacht werden kann. Im zweiten Fall besteht auch die Möglichkeit die Unterschiede interaktiv als legitim zu kennzeichnen und damit die Datenbank gleich auf den neusten Stand zu bringen. Es gibt noch einige weitere Optionen, die man mit

```
./tripwire -help
```

erhält, die aber im Routinebetrieb kaum von Bedeutung sind.

Tip: Einige der Dateien, die `tripwire` lesen muß, um Prüfsummen zu berechnen, sind nur für root lesbar. Daher sollte es immer nur als root benutzt werden, um unnötige Fehlermeldungen und falsche Ergebnisse zu verhindern.

Interpretation der Ergebnisse

Man wird sich wundern, wieviele unerwartete Unterschiede beim ersten Durchlauf auftauchen. Nach und nach wird man dann allerdings feststellen, daß beispielsweise das Betriebssystem jedes Wochenende seine Platten neu strukturiert oder seine logfiles archiviert. Es erinnert an Programme, die abends noch mal schnell installiert und dann vergessen wurden. Ein zuverlässiges Bild des Dateisystems, das im Falle eines zu reproduzierenden Angriffs wertvolle Informationen liefert, entsteht erst nach einigen Durchläufen und Veränderungen der `tw.config`-Datei.

Literaturverzeichnis

- Kim, G. H., Spafford, E. H. : Experience with Tripwire: Using Integrity Checkers for Intrusion Detection, Purdue Technical Report CSD-TR-93-071, 21 February 1994
- Kim, G. H., Spafford, E. H. : Writing, Supporting, and Evaluating Tripwire: A Publically Available Security Tool, Purdue Technical Report CSD-TR-94-019, 12 March 1994
- Kim, G. H., Spafford, E. H. : The Design and Implementation of Tripwire: A File System Integrity Checker,Purdue Technical Report CSD-TR-93-071, 19 November 1993

Warnung!

Die hier gegebenen Hinweise reichen nicht zum vollständigen Gebrauch des Programms aus. Jeder, der die Verwendung des Programms erwägt, sollte gründlichst die beigefügte Dokumentation durchlesen und verstehen, was er tut. Für Risiken und Nebenwirkungen lesen Sie die Programmdokumentation oder fragen Sie Ihren zuständigen Security-Administrator.

Ansprechpartner in sicherheitsrelevanten Fragen

An wen kann ich mich in so einem Fall wenden? Hier zwei wichtige Adressen und Ansprechpartner:

`sneakers@rus.uni-stuttgart.de`
`dfncert-request@cert.dfn.de`

Bernd Lehle
Lehle@rus.uni-stuttgart.de

Oliver Reutter
Oliver.Reutter@rus.uni-stuttgart.de

Universität Stuttgart

Paßwörter - Ein ewiges Problem?

Bernd Lehle/Oliver Reutter

Unsere bisherigen Artikel beschäftigten sich hauptsächlich mit Dingen, über die sich ein Systembetreuer Gedanken machen muß. Nun kommen wir zu einem Thema, das wirklich alle Benutzer angeht und bei dem jeder Benutzer durch leichtfertiges Verhalten Sicherheitslücken schaffen kann. Es ist daher nötig, daß diese Informationen an alle Benutzer weitergegeben werden.

Ein alltäglicher Vorgang: Jemand betritt ein Büro, in dem ein Computer steht, auf dem ein Mehrbenutzer-Betriebssystem installiert ist. Der potentielle Benutzer setzt sich an ein Terminal, findet einen Login Prompt vor und tippt seinen Benutzernamen ein. Der Rechner gibt sich damit nicht zufrieden und fragt nach einem Paßwort. Auch das tippt der potentielle Benutzer ein, und wenn es stimmt, hat er Zugang zum System und kann damit arbeiten. Jeder, der dies liest, kennt diesen Vorgang und hat ihn sicher schon unzählige Male durchexerziert. Die wenigsten jedoch machen sich Gedanken darüber, was dabei abläuft und was dieser Ablauf im einzelnen bedeutet.

Grundlagen

Was bedeutet nun eigentlich das Paßwort? Wenn wir einen Account mit einem privaten Raum vergleichen, so ist das Paßwort eine Art Schlüssel zu diesem Raum. Genauso wird es von den meisten Benutzern auch verwendet. Leider versagen in der elektronischen Welt der Netzwerke solche einfachen Vergleiche sehr schnell. Übertragen auf die normale Welt ist ein Paßwort nämlich Schlüssel, Kreditkarte und Personalausweis zugleich. Es öffnet den Account wie ein Schlüssel, verschafft in dem Moment aber auch Zugang zu Ressourcen wie Rechenzeit und Plattenplatz und kann somit Kosten verursachen. Vor allem verschafft es aber demjenigen, der das Paßwort kennt, eine Identität innerhalb des Systems bzw. innerhalb des angeschlossenen Netzwerks.

Wer also sein Paßwort an andere Personen weitergibt, sollte sich bewußt sein, was er damit von sich preisgibt: Nicht nur, daß der Mitwisser nun auf alle Daten des Benutzers zugreifen, sondern auch in seinem Namen Ressourcen benutzen (z.B. Drucker oder kostenpflichtige Rechenanlagen), elektronische Post verschikken oder über andere Wege elektronisch kommunizieren kann. Alle positiven/negativen Folgen fallen auf den Benutzer zurück, der sein Paßwort und damit seine Identität weitergegeben hat. Dies kann besonders unangenehm werden, wenn der Mitwisser andere Netzteilnehmer belästigt oder versucht in andere Rechner einzudringen.

Die Entschuldigung „Ich war‘s nicht, ich hab nur mein Paßwort weitergegeben!“ wird im Fall von Regreßansprüchen nicht akzeptiert. Eine derartige Regelung, die deshalb explizit die Weitergabe von Paßwörtern verbietet, ist in den meisten Benutzungsordnungen vorgesehen.

Ab und zu passiert es allerdings auch, daß ein Paßwort bekannt wird, ohne daß der Benutzer es freiwillig weitergegeben hat. Um zu verstehen, wie das passiert und wie man sich dagegen schützen kann, begeben wir uns auf einen kleinen Ausflug in die Technik.

Technisches zur Verwaltung von Paßwörtern

Es war nicht immer üblich, daß man sich auf Mehrbenutzersystemen unbedingt mit einem Paßwort ausweisen mußte. UNIX verbrachte einige seiner ersten Jahre ohne Paßwortschutz und selbst nach Einführung der Paßwörter war es noch ein langer Weg bis zu der Paßwortverwaltung wie wir sie heute kennen. Mittlerweile hat jedes ernstzunehmende Workstation-Betriebssystem eine Paßwortverwaltung, die in ähnlicher Form wie bei UNIX funktioniert. Wir wollen uns daher bei den Details exemplarisch auf UNIX beschränken.

Nun soll betrachtet werden, was im einzelnen abläuft, wenn sich ein Benutzer über das Netz auf einer entfernten Maschine einloggt, wie z.B. auf dem SERVus-Cluster:

Als erstes wird ein Terminalemulator - z.B. `telnet` - gestartet:

```
telnet servint1.rus.uni-stuttgart.de
```

Der angesprochene Rechner bekommt über den `inet-daemon` das Signal, daß seine Dienste im Netz gewünscht sind, und startet darauf-

hin erst einen `telnet-daemon`, dann einen `login`-Prozeß, der sich mit einem Login Prompt meldet:

```
AIX Version 4
(C) Copyrights by IBM and by others
1982, 1994.
login:
```

Der Rechner verlangt hier den Benutzernamen und sofort darauf das Paßwort. Alles, was getippt wird, geht natürlich in dieser Form als Klartext über das Netz, bzw. befindet sich an beiden Enden kurzfristig in den Puffern des Kernels oder der graphischen Benutzeroberfläche:

```
login: ruslehle
ruslehle's Password:
```

Nach kurzer Zeit kommen dann wie gewohnt die Nachrichten und der Shell Prompt zu Tage, so daß gearbeitet werden kann. Was passiert in dieser Zeit?

Der Rechner muß in irgendeiner Form feststellen, ob das eingegebene Paßwort mit dem übereinstimmt, das von dem Benutzer früher festgelegt wurde. Das könnte einfach dadurch realisiert werden, daß eine Datei existiert, in der alle Benutzernamen mit den zugehörigen Paßwörtern und sonstigen Daten gespeichert sind. Da diese Datei allerdings für alle Benutzer lesbar sein sollte, um Informationen über andere Benutzer zugänglich zu machen, können die Paßwörter dort nicht im Klartext stehen. In der Tat zeigte ein Blick auf die Datei `/etc/passwd` auf einem UNIX-System bis vor kurzem noch folgenden Anblick:

```
zohn:V41B6Zgffh91c:237:302:Hermann Zohn:/home/zohn:/bin/csh
doedel:*:415:303:Doedel-Account:/home/doedel:/bin/sh
kigel:Dgh34KaR4hMq.:419:302:Hans Kigel:/home/kigel:/bin/tcsh
maier:33qal5Bd0IpF2:368:302:Johann Maier:/home/maier:/bin/bash
```

Bis vor kurzem heißt, daß moderne UNIXe die Paßwortinformationen heute meist an sichereren Orten speichern, aber dazu später mehr.

Hier steht im einzelnen: Der Benutzername, ein verschlüsselter Text, der Paßwortinformation beinhaltet, die Benutzer- und die Gruppennummer, der echte Name des Benutzers und/oder ein Kommentar (Gecos Field), das Home-Verzeichnis und die Login Shell.

Wie wird nun verglichen, ob das Paßwort stimmt? Das Paßwort wird bei dem Prozeß strenggenommen nicht verschlüsselt, sondern als Schlüssel benutzt. Im einzelnen läuft es wie folgt: Die ersten acht Zeichen des Paßwortes werden in einen Schlüssel für das DES (Data Encryption Standard)-Verfahren umgewandelt. Dieses Verfahren benutzt Schlüssel von 56 bit Länge, daher kann ein UNIX-Paßwort nur acht Zeichen lang sein, von denen je 7 bit verwendet werden.

Mit diesem Schlüssel wird nun per DES eine Kette von Nullbits verschlüsselt. Dabei wird nicht exakt das DES-Verfahren aus der Literatur [1] verwendet, sondern ein abgewandeltes, in das während des Verschlüsselungsverfahrens noch 12 bit weitere Information eingestreut werden. Diese eingestreute Information wird als Salt (Salz) bezeichnet und besteht aus zwei alphanumerischen Zeichen.

Was steht nun in der Paßwortdatei? Man erkennt eine Kette aus 13 Zeichen. Die ersten beiden bilden den Salt, die restlichen elf sind das Ergebnis der Verschlüsselung der Nullen, so gestaltet, daß keine Doppelpunkte oder nicht druckbare Zeichen darin vorkommen.

Beim Login-Vorgang wird also mit dem eingegebenen Paßwort unter Verwendung der ersten beiden Zeichen der 13er-Kette eine Kette von Nullbits verschlüsselt. Stimmt das Ergebnis der Verschlüsselung mit den restlichen elf Zeichen überein, ist das Paßwort richtig und der Benutzer wird zugelassen. Um aus den 13 Zeichen das Paßwort zu rekonstruieren, müßte man das modifizierte DES-Verfahren brechen, was selbst beim Einsatz von schneller Hardware heutzutage sehr aufwendig ist. Normale DES-Verschlüsselungs-Chips können hier nicht eingesetzt werden, da sie den Salt-Trick nicht beherrschen.

Bei so viel Verschlüsselungstechnik mag sich der Benutzer nun fragen, warum die Sache nicht absolut sicher ist.

Angriffstechniken auf Paßwörter

Die Sicherheit der Paßwörter läßt sich auf verschiedene Weisen umgehen, dazu wird am besten nochmals der bereits beschriebene Login-Prozeß betrachtet.

Das erste Problem stellt der Benutzer dar. Um vernünftig arbeiten zu können, muß er sein Paßwort irgendwo außerhalb des Rechners speichern, so daß er es immer wieder findet, wenn er es braucht. Idealerweise weiß er seine Paßwörter auswendig. Viele Benutzer vertrauen aber ihrem Gedächtnis nicht und benutzen andere Hilfsmittel. Eines der schlechtesten Hilfsmittel ist das Aufschreiben des Paßwortes. Selbst wenn dieser Zettel im Portemonnaie oder verschlossen im Schrank liegt, ist diese Lösung nicht akzeptabel. Ein neugieriger Blick im falschen Moment reicht aus, um das Paßwort herauszufinden. Wesentlich besser ist es, sich für seine Paßwörter ein System zu überlegen, mit dem man anhand einfacher Hilfsmittel (z.B. Telefonbuch oder andere öffentliche Quellen großer Informationsmengen) ein vergessenes Paßwort schnell wieder rekonstruieren kann, obwohl das Paßwort selbst kompliziert und schlecht zu merken ist.

Der nächste Angriffspunkt ist das Eintippen des Paßwortes. Die meisten Menschen, die häufig am Computer arbeiten, können relativ schnell tippen, so daß es für einen normalen Beobachter nicht möglich ist, durch Verfolgen der Finger den getippten Text zu lesen. Probleme tauchen dann auf, wenn jemand sehr langsam tippt oder das Paßwort so kompliziert ist, daß man sich auf der Tastatur stark verrenken muß und so wiederum nur sehr langsam tippen kann. Es zählt zum guten Ton, wenn man gemeinsam an einem Computer arbeitet, wegzuschauen, wenn jemand sein Paßwort eintippt. Was nämlich trotz schnellen Tippens immer wieder passiert ist das Folgende:

```
login: ruslehle
ruslehle‘s Password:
login incorrect
login: sf%hsgt
sf%hsgt‘s Password:
```

Was ist hier passiert? Der Benutzer hat sich beim ersten Eingabeversuch des Paßwortes vertippt. Er weiß dies. und da er in Eile ist, kann er es nicht erwarten, das Paßwort richtig einzugeben, wobei er dann vergißt, daß auch der Login-Name erneut gefragt wird. Statt des Login-Namens gibt er nun sein Paßwort ein, was nun jeder in Sichtweite des Bildschirms mühelos lesen kann. Nun heißt es, sich schnell richtig einzuloggen, den Screen zu löschen und das Paßwort zu ändern, denn normalerweise steht der zweite, nun auch ungültige, Login-Versuch wie folgt in den System-Logdateien:

```
Jan  7 09:54:45 login failed from servint1 as sf%hsgt
```

Da der Rechner das Paßwort als Login-Name interpretiert, den es natürlich nicht gibt, protokolliert er den Vorgang als ungültiges Login durch den Benutzer `sf%hgst`. Auf manchen Systemen sind die Logdateien für alle Benutzer lesbar und somit kann jeder die Paßwörter bekommen.

Hat der Benutzer nun auch diese Klippen umschifft, droht ihm neue Gefahr durch die interne Verarbeitung seines Paßwortes. Gibt der Benutzer einen Text auf der Tastatur ein, wird er vom Tastaturtreiber des Kernels entgegengenommen und verbleibt dann kurz in einem Puffer, bis er an die Anwendung weitergeleitet wird. Diese Puffer können von einem Benutzer mit entsprechenden Privilegien gelesen und verändert werden. So ist es leicht möglich, Tastatureingaben abzuhören oder zu verändern.

Noch leichter, nämlich ohne Privilegien, funktioniert das ganze, wenn die Anwendung, an die die Eingabe weitergeht, eine mangelhaft gesicherte graphische Benutzeroberfläche ist. Wird z.B. X-Windows ohne die Sicherheitsmechanismen `xhost` oder `Magic Cookies` betrieben, kann jeder beliebige Internet-Teilnehmer die Tastatureingaben abhören oder verändern. Der Schutz durch die `Secure Keyboard`-Funktion, den einige Versionen des `xterm`-Programmes bieten, ist dabei völlig wirkungslos.

Wie man X-Windows sicher betreibt, wurde in dem Artikel von Markus Müller (BI. 3 94) ausführlich dargelegt.

Auch wenn das Paßwort sicher durch alle Oberflächen gedrungen ist, lauern neue Gefahren, sobald es sich auf ein Netzwerk begibt. Ist das Paßwort nicht für den Rechner bestimmt, an dem der Benutzer sitzt, sondern für einen, der entfernt im Netzwerk sitzt, muß es erst dorthin gelangen.

Leider ist eine Verschlüsselung des Paßwortes auf diesem Weg nicht vorgesehen und jemand, der Zugang zu einem Netzwerk-Segment hat, an dem das Paßwort vorbeikommt, kann es leicht mitlesen. Abhilfe schaffen hier nur kryptographische Login-Verfahren wie ssh oder Kerberos, die im rpool, im SERVus-Cluster oder im HWW verwendet werden.

Neben diesen Abhör- und Ablese-Angriffen gibt es aber noch eine ganz andere Klasse von Methoden, an Paßwörter heranzukommen. Man bezeichnet sie als Wörterbuch-Angriffe (Dictionary Attacks). Zum Ausführen dieser Angriffe benötigt man folgende Dinge:

1. Die Paßwortdatei mit den verschlüsselten Paßwortinformationen
2. Ein Wörterbuch mit möglichst vielen Worten, die als Paßwörter in Frage kommen könnten
3. Ein Programm, das die beschriebene Paßwort- Verschlüsselung möglichst schnell und einfach durchführt
4. Viel Rechenzeit oder einen sehr schnellen Rechner.

2. und 3. sind im Internet sehr leicht zu beschaffen. Ein Repertoire von über einer Million Wörtern aus vielen europäischen Sprachen sowie wichtige Fachwörter, Sagengestalten, Vornamen, Nachnamen, Popgruppen, Geburtsdaten, etc. steht auf ftp-Servern zur Verfügung. Ein komfortables Programm, um die Wörter durchzuprobieren findet man mit Alec Muffet's `Crack`, das mittlerweile als Version 5.0 frei verfügbar ist. `Crack` testet nicht nur alle ihm zur Verfügung gestellten Wörter, sondern berechnet von jedem Wort 280 Abarten (vorwärts, rückwärts, groß, klein, mit angehängten Zahlen und Sonderzeichen, etc.). Weitere Abarten von Wörtern sind frei konfigurierbar. Auch alle verfügbare Information über den Benutzer (Login-Name, echter Name, Rechnername, etc.) wird als Paßwort in siebzig Abarten durchgetestet. Ebenso ist das Programm darauf angelegt, auf Workstation Clustern oder Parallelrechnern die volle Performance herauszuholen; Wörter durchzutesten ist trivial parallelisierbar.

Geringfügige Probleme gibt es heute mit Punkt 1, da die meisten modernen Betriebssysteme die Paßwortinformation mittlerweile getrennt von der Benutzerinformation aufbewahren. In der eigentlichen Paßwortdatei stehen nur Name, Verzeichnis und Login Shell, während eine weitere Datei existiert, die dann die Paßwortinformation trägt, aber nur vom Systembetreuer lesbar ist.

Es gibt allerdings noch eine Vielzahl von Systemen, die dieses sogenannte Shadowing zwar beherrschen, aber nur auf Anforderung auch anwenden. Eine verbreitete Anwendung, die das Shadowing nicht beherrscht, ist NIS (Yellow Pages). Dort kann man die komplette Paßwortinformation jederzeit von allen beteiligten Rechnern mit `ypcat passwd` anfordern. Wenn der Portmapper oder eine andere Sicherungsinstanz nicht ausreichend gegen Zugriffe von außen geschützt ist, kann jeder Internet-Teilnehmer die Paßwortdatei anfordern. Das Shadowing kann aber auch noch anders umgangen werden. Ein Eindringling, der es geschafft hat, auf einem Rechner Systemverwalterrechte zu bekommen, kopiert als erstes die geschützte Datei. Diese kann er an anderer Stelle dann in aller Ruhe knacken, um Zugriff auf das System zu haben, auch wenn sein erster Zugang entdeckt und gesperrt wurde. Shadowing ist also auch hier nicht die ultimative Lösung.

Auf Punkt 4 werde ich gleich in einem Beispiel eingehen.

Beispiel für einen Angriff gegen eine Paßwortdatei

Alle Daten und Fakten des folgenden Beispiels wurden von Mitarbeitern unserer Arbeitsgruppe mit Unterstützung einiger Systemverwalter am RUS bestätigt.

Als erstes besorgten wir uns Paßwortdateien. Das kann quasi legal über NIS oder Kopieren realisiert werden, oder man sucht gezielt nach schlecht gesicherten NIS-Systemen und holt sich dort die Paßwörter. Unsere Dateien enthielten jeweils etwa 200 Paßwörter. Dann wurde auf zwei Rechnerarchitekturen das Programm `Crack` installiert. Als Wörterbuch nahmen wir 1,2 Millionen Wörter, die wir uns im Internet zusammenkopierten. Die verwendeten Rechner waren ein PC 486DX4/100 mit 16 MB RAM unter Linux und die intel Paragon des Rechenzentrums mit 105 intel 860 XP-Prozessoren mit je 32 MB RAM unter OSF/1. Das Programm wurde leicht verändert, so daß jeder Prozessor nur noch etwa 10 000 Wörter zu probieren hatte.

Auf der Paragon wurde nur die idle-Zeit genutzt, d.h. Rechenzeit, die kein anderer Benutzer anforderte. Der Linux-PC widmete sich ausschließlich dem Paßwortknacken.

Die Ergebnisse waren typisch und könnten sich jederzeit überall wiederholen. Die gefundenen Paßwörter verteilen sich wie folgt:

a. Nach wenigen Sekunden bis wenigen Minuten kamen 2-5 Prozent der Paßwörter heraus, die einfach dem Login-Namen entsprachen

b. Innerhalb einer Stunde (Paragon), bzw. eines halben Tages (PC) kamen weitere 15 Prozent der Paßwörter heraus, die die Struktur <Wort><Ziffer> hatten (z.B. herbert4) oder unverändert im Wörterbuch standen
c. In den folgenden 55 Stunden (Paragon), bzw. 14 Tagen (PC) kamen weitere 3-5 Prozent der Wörter heraus, die etwas komplizierter waren (groß/klein, mit Jahreszahlen oder Sonderzeichen).

Insgesamt wurden 20-25 Prozent der Paßwörter gefunden, was gutem Durchschnitt entspricht.

Man kann also davon ausgehen, daß ein beliebiger Angreifer, wenn er schlecht ausgestattet ist, in zwei Wochen, wenn er gut ausgestattet ist an einem Wochenende oder schneller zum selben Ergebnis kommen kann.

Schutzmaßnahmen

Sie werden sich zurecht fragen, was man nun dagegen tun kann. Im folgenden werden wir einige Tips zum besseren Umgang mit Paßwörtern geben, die man als Systembetreuer kennen muß und seinen Benutzern weitergeben sollte. Diese Regeln gelten für alle paßwortgeschützten Vorgänge wie Accounts auf UNIX, VMS, Windows NT, o.ä. aber auch für Telebanking oder weniger offensichtliche Paßwörter:

1. Ein Paßwort wird niemals und an niemanden weitergegeben. Wenn mehrere Personen an einem Projekt arbeiten, bekommt jeder ein eigenes und der Rest wird über Dateiberechtigungen oder `setuid`-Programme gemacht
2. Ein Paßwort darf kein Wort sein, das in einem Wörterbuch irgendeiner Sprache zu finden ist oder in irgendeiner Form Bezug zum Benutzer hat
3. Ein Paßwort sollte Groß- und Kleinbuchstaben sowie Zahlen oder Sonderzeichen enthalten
4. Ein Paßwort wird nirgends aufgeschrieben
5. Ein Paßwort muß leicht und schnell zu tippen sein
6. Ein Paßwort sollte die maximale Länge nutzen (8 Zeichen unter UNIX, 14 unter Windows NT, Pass Phrases bei PGP oder anderen Anwendungen).

Nach so vielen Regeln hier auch ein paar konstruktive Vorschläge, das Paßwortproblem in den Griff zu bekommen:

1. Verwenden Sie Paßwortsysteme. Paßwörter kann man beispielsweise aus einzelnen Elementen zusammensetzen, die zusammen keinen Sinn machen und durch Sonderzeichen getrennt sind
2. Sichern Sie Ihr System weitestmöglich durch die Verwendung von Shadowing und abgesichertem NIS bzw. X-Windows gegen Paßwortdiebstahl ab
3. Nutzen Sie Programme wie `npasswd` oder `passwd+`, die es dem Benutzer nicht erlauben, schlechte Paßwörter zu wählen, da sie diese sofort beim Ändern mit einem Wörterbuch vergleichen
4. Bevorzugen Sie, wenn möglich, moderne, sichere Authentisierungsmechanismen wie kerberos und ssh
5. Greifen Sie in kritischen Umgebungen zu Einweg-Paßwörtern.

Der Punkt 5 ist leider etwas heikel, da er nicht leicht zu implementieren ist und meistens das Mit-sich-herumtragen einer längeren Liste von Einweg-Paßwörtern bedingt. Interessenten, die damit experimentieren wollen, sei das Programmpaket S/Key empfohlen. In kritischen Umgebungen im Industriebereich kommen Smart Cards zum Einsatz, die aus einer Vorgabe beim Login und einer PIN das Einwegpaßwort errechnen und so die Listen überflüssig machen.

Ansprechpartner in sicherheitsrelevanten Fragen

- `sneakers@rus.uni-stuttgart.de`
- `dfncert-request@cert.dfn.de`

Literatur

[20] Bruce Schneider, Applied Cryptography, Wiley & Sons, 1996

[21] manpage zu crypt (3)

Bernd Lehle
Lehle@rus.uni-stuttgart.de

Oliver Reutter
Oliver.Reutter@rus.uni-stuttgart.de

Universität Stuttgart

Schulen im BelWü

Friedrich Achtstätter

Internet im Unterricht - Zukunftsmusik oder Realität?

- Schüler: „Herr Lempel, in unserem Erdkundebuch steht nur wenig über die Aborigines in Australien, wir wollen aber etwas über diese Menschen erfahren.Wenn Sie mit uns in der nächsten Erdkundestunde in den Internetraum gehen, forschen wir gern auf eigene Faust nach und stellen sogar eine Schauwand über die Aborigines für unsere Aula zusammen!“

Höhlenmalerei der Aborigines in Australien

- Deutschunterricht Klasse 10 im Computerraum mit Internetanbindung, Thema „Zeitungen“: Schüler gestalten mit Hilfe der aktuellsten Meldungen von Nachrichtenagenturen und mit einem DTP-Programm sehr motiviert eine eigene Zeitung. Diese vergleichen sie am nächsten Tag mit den realen Tageszeitungen und diskutieren lebhaft und engagiert die Gemeinsamkeiten und Unterschiede.

- „Sabine, schau mal nach, was unsere Partnerklasse in Israel auf die E-Mail unserer letzten Unterrichtsstunde geantwortet hat!“. Sabine geht in die Ecke des Klassenzimmers zum Computer mit Internetanschluss ...

Die Beispiele zeigen die möglichen Stärken des Interneteinsatzes im Unterricht auf: große Informationsfülle, Aktualität, schnelle Kommunikation und motivierende Angebote. Sind diese Beispiele nur visionäre Zukunftsmusik oder bereits ein Stück Realität an unseren Schulen?

Sowohl weltweit als auch innerhalb Baden-Württembergs gibt es hier große Unterschiede. Besonders wenn an einer Schule engagierte LehrerInnen als Internet-Pioniere tätig sind, gehören Anwendungen des Internet im Unterricht, wie sie oben anklingen, auch heute schon da und dort zum Schulalltag. Inzwischen beschäftigen sich in Baden-Württemberg LehrerInnen an etwa 800 Schulen (Schätzung) mit den Möglichkeiten, die das Internet für den Unterricht bietet. Gemessen an der Gesamtzahl der LehrerInnen sind sie Vorreiter und haben manchmal keine Anschlussmöglichkeit in der Schule selber.

In Nordbaden wurde schon frühzeitig erkannt, daß der Einsatz des Internet an der Schule mit der Akzeptanz bei den LehrerInnen steht und fällt. Im Frühjahr 1995 initiierten das Oberschulamt Karlsruhe (RSD Buhmann) und die Universität Karlsruhe (Prof. Gerhard Schneider, Rechenzentrum) das Internet-Projekt, bei dem Lehrern und Lehrerinnen aller Schularten der Zugang ins Internet mit E-Mail-Adresse und Plattenplatz für WWW-Seiten ermöglicht wurde. Im Juli 1997 sind über 1200 Lehrer angemeldet und die Teilnehmer am Projekt erwerben zunehmend Erfahrung und Kompetenz bei der Internetnutzung. Obwohl das Angebot sich an die LehrerInnen als Einzelperson richtet, profitieren auch die Schulen davon. Die Früchte zeigen sich u.a. in etwa hundert Schul-Homepages, die es ohne dieses Projekt so nicht gegeben hätte.

Messestand des Internetprojekts auf der Multimediamesse in Stuttgart Mai 1997

v.l.n.r.: StD Berberich (OSA Karlsruhe), Gisa Schultze-Wolters (IBM, Sponsor des Internetprojekts), Prof. Gehard Schneider (Universität Karlsruhe), OStR Achtstätter (OSA Karlsruhe)

Die Erfahrungen aus dem Internetprojekt in Nordbaden zeigen, daß die technischen Schwierigkeiten bei der Internetanbindung ausgeräumt werden mußten, bevor eine gewinnbringende Arbeit auf breiter Basis an den Schulen möglich war. Die LehrerInnen erkannten beim Interneteinsatz für den Unterreicht schnell, daß wünschenswerte, fächerübergreifende Aspekte allein schon durch das Medium gegeben sind. Die Rolle der Unterrichtenden besteht weniger in der Leitung des Unterrichts, sondern mehr in der Moderation der Lernerfahrungen der Schülerinnen.

Von SAN bis BelWü - Hilfen für die Schulen

Von sehr kleinen Schulen abgesehen ist die Anbindung einer Schule nur mit einem lokalen Netz und der Internetanbindung über einen Router zu einem Provider sinnvoll.

Zunächst stehen die meisten Schulen dabei vor einem Berg von Problemen: oft fehlt es an der notwendigen Hardwareausstattung, die Schulträger zahlen ungern die schnell in die Höhe steigenden Verbindungsgebühren, es fehlen Fachleute vor Ort, die mit dem neuen Medium und der dazu erforderlichen Technik umgehen können. Eine ganze Reihe von Initiativen versucht derzeit diese Stolpersteine aus dem Weg zu räumen, um in den nächsten Jahren schließlich alle Schulen an das Internet anzubinden.

Durch die bundesweite „Schulen ans Netz“-Initiative des Bundesministeriums für Bildung, Wissenschaft, Forschung und Technologie und der Deutschen Telekom AG werden für Einstiegsprojekte insbesondere Gebührengutschriften und kleinere Hardwareanschaffungen getragen, Modellprojekte erhalten Unterstützung bis 20.000,- DM.

Für das Land Baden-Württemberg wird die Landesmedieninitiative einen großen Schub bei der Einführung des Interneteinsatzes an Schulen bringen. Sie enthält mehrere Komponenten, durch die einerseits Hardwaremindestanforderungen an jeder Schule erfüllt werden können und andererseits das Knowhow der Lehrer sowohl beim Multimediaeinsatz als auch bei der Schaffung und Betreuung von Schulnetzen (mit Internetanschluß) gefördert wird.

Welcher Provider ist der Richtige? - BelWü als erste Wahl

Die Bedürfnisse einer Schule bei der Internetanbindung werden nicht von allen Providern voll abgedeckt. Hier ist das Angebot von BelWü kaum zu überbieten.

Universitäten, Fachhochschulen und Berufsakademien kommen als Einwählpunkte oder Anbindungspartner für Standleitungen in Frage. Durch den Einsatz vorhandener Ressourcen kann dies meist sehr kostengünstig verwirklicht werden. Dieses Netz soll weiter verdichtet und ausgebaut werden, bis irgendwann jede Schule in Baden-Württemberg zum Citytarif ans BelWü angebunden werden kann.

Eine anschlusswillige Schule erhält einen Domainnamen der Form skz.ak.bw.schule.de (skz steht für das Schulkürzel, ak für das Autokennzeichen in diesem Ort). WWW-Seiten der Schule werden auf dem BelWü-Rechner abgelegt und sind über einen virtuellen Webserver (Adresse: www.skz.ak.bw.schule.de) abrufbar. Die unbeschränkte, eigenverantwortliche E-Mail-Adressenvergabe vor Ort in der Form „irgendwas@skz.ak.bw.schule.de“ zusammen mit der Zwischenpufferung der Mail im BelWü-Rechner bis zur nächsten Verbindungsherstellung durch die Schule (sofern keine Standleitung besteht) ist ein ganz wichtiges Angebot.

Durch unterschiedliche Netzbetriebssoftware und verschiedene Routerrealisierungen (als Software auf einem PC mit ISDN-Karte oder als dezidierter Router) ist eine Vielzahl von Anbindungslösungen samt gewünschter Funktionalitäten (z.B. Intranet-Webserver, Proxy, Mailserver, Firewall) denkbar. Die meisten Schulen besitzen nicht das notwendige Knowhow dafür. Auch hier hilft BelWü, ein Arbeitskreis für Schulen (AK3-BelWü) trifft sich mehrmals jährlich, um Probleme und Beipielslösungen zu diskutieren. Hier sind die Partner aus dem Beruflichen Bereich mit ihrer LEU-ZPG-Gruppe durch die Herausgabe der „Online-News“ besonders hilfreich. Im WWW werden Musterlösungen von „Pionieren“ vorgestellt. In nächster Zeit wird es regionale Ansprechpartner an den Universitäten geben, die bei Fragen der Anbindung weiterhelfen können.

Konkret kann eine Schule die erforderlichen Kenntnisse vor Ort fast auf Null reduzieren, wenn sie einen von BelWü fertig konfigurierten Router finanziert, der anschließend von BelWü auch gewartet und ständig mit neuester Software auf dem laufenden gehalten wird.

Was fangen die Schulen mit einem Internetanschluß an?

Diese Frage hat sicher mehr Antworten, als hier dargestellt werden kann. Zuerst werden meist eigene WWW-Seiten gebastelt und eigene Einstiegsseiten ins WWW zusammengestellt. Gerne werden überregionale Projekte im Internet gesucht (und auch gefunden), durch die Teilnahme an diesen Projekten ist fächerübergreifender Unterricht ganz natürlich (Fremdsprache, Thema des Projektes, das oft mehreren Fächern zugeordnet werden kann). Dabei werden Kontakte zu anderen Schulen irgendwo in der Welt geknüpft, die wiederum zu Erfahrungen über die Lebenswelt von Schülern in anderen Ländern führen.

Schulpartnerschaften gewinnen durch E-Mail-Kontakte neues Leben, durch die Homepages kommen Schulen oder Gruppen in den Schulen mit ähnlichen Interessen zusammen (Schülerzeitungen, Orchester, Theatergruppen).aaaaa

Die Schüler nutzen mit großer Selbstverständlichkeit das Internet. Mancher Lehrer wurde in seinem Unterricht schon durch Beiträge von Schülern überrascht, die diese zum Stoff passend im Internet gefunden hatten. Auch das Schreiben von Referaten kann bei geeignetem-Thema für Schüler seinen Schrecken verlieren ;-)

Erweiterbare Listen zu Einzelfächern

)urch Klicken auf die ausgeschriebenen Fächernamen finden Sie fachspezifische Listen von Informationsangeboten im *Vorld Wide Web* , die Sie vielleicht für Ihren Unterricht nutzen können. Diese Listen sind durch Sie selbst erweiterbar, .h. Sie können hier Ihre Kollegen oder Schüler auf interessante Informationen aufmerksam, eventuell auch Reklame für ıre eigene Homepage machen. Das Klicken auf die dazugehörige Abkürzung bringt Sie bei einigen Fächern bereits zu hrplangerecht sortierten Unterrichtseinheiten. Dies ist ein im Aufbau befindlicher Service. Hier ist nachzulesen, wie Sie ıre eigenen Materialien einbringen können.. Ein Autorenmodul erlaubt Ihnen die Online-Eingabe Ihrer Texte und deren infache Einsortierung in die richtige Rubrik.

Fach	Lehrpläne	Fach	Lehrpläne	Fach	Lehrpläne
ev Religion	evR	Englisch	E	Mathematik	M
kath. Religion	kR	Französisch	F	Physik	Ph
Ethik	Eth	Russisch	Ru	Astronomie	A
Gemeinschaftskunde	Gk	Spanisch	S	Chemie	Ch

Auszug aus einer WWW-Seite von ZUM (Zentrale für Unterrichtsmedien)

Lehrer hoffen, im Internet Material zu finden, mit dem sie ihren Unterricht aktueller und motivierender gestalten können. Hier gibt es Initiativen, um im Internet Angebote speziell für den Unterricht zu machen. Dazu gehören in Baden-Württemberg „ZUM“ (Zentrale für Unterrichtsmedien), das u.a. einen nach Fächern und Lehrplan gegliederten Einstieg ins Internet anbietet, und „SIN96“ (Schule im Netz 96) mit ähnlicher Struktur. Beide erlauben die Mitwirkung aller Benutzer zur Erweiterung des Angebots. Das Land selber entwickelt zur Zeit den Landesbildungsserver, der dann das Forum und die Zentrale im Internet für alle unterrichtlichen Belange in Baden-Württemberg sein soll.

Rosige Zeiten fürs Internet an den Schulen?

Das alles klingt ein bisschen euphorisch, was aber den Möglichkeiten des Interneteinsatzes für den Unterricht durchaus angemessen ist. Zeigt man Lehrern das WWW oder die einfache E-Mail-Bedienung, sind sie zumeist begeistert und wollen auch „ins Internet“. Auf privater Basis gibt es hierzu einfache Verwirklichungen.

Anders sieht es aus, wenn die Anbindung einer Schule verwirklicht werden soll, damit auf breiter Basis und gewinnbringend im Internet gearbeitet werden kann. Dann steht und fällt die Umsetzung an der Schule mit der Anwesenheit eines/einer Experten/Expertin. Solche sind aber insbesondere an der großen Zahl der Grund-, Haupt- und Realschulen noch dünn gesät. Es bleibt zu hoffen, dass das ehrgeizige Fortbildungs-Projekt des Landes Baden-Württemberg hier die erforderliche Wirkung zeigt und tatsächlich innerhalb drei Jahren an jeder Schule Fachleute für Multimedia vorhanden sind und Netzwerkberater in der näheren Umgebung der Schule gefunden werden können.

Bei allen Fortbildungsaktivitäten bleibt offen, ob die Fachleute für Multimedia und Netzwerke durch geeignete Deputatsanrechnungen zeitlich in die Lage versetzt werden, ihre Kenntnisse auch ihren eigenen oder benachbarten Schulen zugänglich zu machen. Ebenso ungeklärt ist für die Kommunen, wo sie das Geld für die gestiegenen Bedürfnisse in diesem Bereich (Hardware, Anbindungskosten) dann aufbringen sollen.

Probleme sind dazu da, um gelöst zu werden. Es ist keine Frage mehr, daß in den Schulen zum einen der Umgang mit dem Internet als Zukunftstechnik gelehrt werden muss und zum anderen die Chance ergriffen werden muss, durch das Internet neue Motivationen in die Schule zu tragen. Irgendwie wird es gelingen.

Friedrich Achtstätter
Oberstudienrat
Leibniz-Gymnasium Östringen
Internet-Projekt Oberschulamt Karlsruhe
Achtstaetter@lehrer1.rz.uni-karlsruhe.de

Die IBM RS/6000 SP am Rechenzentrum der Universität Karlsruhe

Prof. Dr. Willi Schönauer

Leistungsdaten und Einsatzmöglichkeiten des neuen 256-Knoten-Höchstleistungsparallelrechners

Seit Dezember 1995 war am Rechenzentrum der Universität Karlsruhe ein Parallelrechner mit 100 Prozessoren installiert: Die Universität brachte 16 Thin66 Node2 sowie 56 Wide77 Nodes ein, und das Forschungszentrum Karlsruhe (FZK) 28 Wide77 Nodes, die im Rahmen des Virtuellen Rechenzentrums gemeinsam betrieben wurden. Dieser Höchstleistungsrechner wurde Ende März mit weiteren 168 Thin120 Power2 Super Nodes zu einer hochmodernen 256-Knoten-Großanlage ausgebaut und am 17. Juni 1997 feierlich in Betrieb genommen. Der neue High-Tech-Rechner wurde sowohl in den Computerverbund der hww als auch in das Höchstleistungsrechenzentrum Karlsruhe „HLRKA“ (http://www.hlrka.de/) eingebracht. Der Supercomputer steht damit bundesweit Universitäten und Forschungsreinrichtungen sowie der Wirtschaft zur Verfügung.

Um den Benutzerbetrieb so wenig wie möglich zu stören, wurde mit den neuen Knoten eine separate Maschine eingefahren und hard- und softwaremässig stabilisiert. Am Rechenzentrum der Universität ist weltweit noch immer die einzige SP installiert, die unter DCE/DFS läuft. Daher sind über die Standardsoftware hinaus viele Anpassungen an lokale Besonderheiten zu machen. IBM selbst wird bald standardmässig nur noch DCE/DFS für die SP anbieten. Die Benutzer des Rechenzentrums wurden dann im laufenden Betrieb von der „alten“ SP auf die neue SP migriert. Als das geglückt war, wurde die „alte“ SP abgebaut und neben der neuen Maschine aufgebaut. Danach wurden beide Maschinen abgeschaltet und zu einer einzigen 256-Prozessoranlage zusammengeschlossen.

Am RZ wird jetzt die grösste IBM RS/6000 SP, die es in Europa gibt, betrieben. Um neue Betriebssystemteile auszutesten, wurde eine separate Testmaschine eingerichtet mit 12 Wide77, 256 MB. Hier soll nach Möglichkeit auch Produktion unter LoadLeveler gefahren werden.

Die gesamte Produktionsmaschine hat eine theoretische Spitzenleistung von 107 GFLOPS, die Testmaschine von ca. 4 GFLOPS. Das Wichtigste an diesem Rechner aber ist sein Hauptspeicher von 122 GB, der in einem vernünftigen Verhältnis zu den 107 GFLOPS steht und einer der Hauptgründe für die Auswahl dieses Rechners war. Zum Vergleich: Die NEC SX-4 der Universität Stuttgart hat bei 64 GFLOPS theoretischer Spitzenleistung 8 GB Hauptspeicher, die CRAY T3E mit 512 Prozessoren und 307 GFLOPS theoretischer Spitzenleistung 66 GB Hauptspeicher. Bei grossen Problemen begrenzt der Hauptspeicher i.a. die lösbare Problemgrösse, die Rechenleistung kann man über die Rechenzeit ausgleichen. Nach unserem Wissen ist der Hauptspeicher der Karlsruher SP von 122 GB der grösste Hauptspeicher eines Supercomputers in Europa. Damit können in Karlsruhe grössere Probleme als auf jedem anderen Supercomputer in Europa gelöst werden. An jedem Knoten sind jetzt je eine Platte mit 2 und mit 4,5 GB direkt angeschlossen, das sind zusammen 1,7 TB, wovon allerdings ein Teil für das Betriebssytem benötigt wird. Der Rest ist knotenlokaler Plattenplatz. Ferner ist ein separates Plattensystem mit 350 GB direkt an die SP angeschlossen.

Da wir verschiedene Prozessor-Typen in der SP haben (das zeigt deren Flexibilität) wird man jedoch für einen einzigen Paralleljob maximal die 168 Thin120 Nodes nehmen, was 80 GFLOPS und 86 GB Speicher entspricht. Bis Herbst 1997 werden aber aus technischen Gründen 8 der 168 Thin120 Knoten in der Testmaschine laufen müssen und dafür 8 Wide77 in der Produktionsmaschine.

Die Entscheidung für den Kauf der SP fiel schon 1995. Uns hatte die Leistung der dort angebotenen Wide77-Knoten und der damals projektierten Wide135-Knoten überzeugt.

Tabelle 1: Die endgültige Konfiguration wird wie folgt aussehen:

32 Server Nodes	16	Thin66	128 MB	PIOFS
	4	Wide77	256 MB	DFS-Fileserver
	4	Wide77	256 MB	DCE, Loadleveler,...
	4	Wide77	512 MB	LOGIN
	2	Wide77	256 MB	Gateway nach aussen
	2	Wide77	256 MB	ADSM (Archive, Backup)
224 Compute Nodes	8	Wide77	2 MB	Serial Batch, FEM, ...
	4	Wide77	512 MB	Serial, interaktiv
	16	Wide77	256 MB	parallel Entw., interaktiv
	168	Thin120	512 MB	parallel Batch
	28	Wide77	256 MB	FZK-Regeln

Als es dann zur Entscheidung über den Ausbau kam, hatten wir erneut eine Wahl zwischen den inzwischen verfügbaren preisgünstigeren Thin120- und den teureren Wide135-Knoten. Tabelle 2 zeigt Messungen für die 3 Knoten: Wide77, Thin120, Wide135 für die Vektor-Triade

$$a_i = b_i + c_i \cdot d_i,$$

die für Ingenieuranwendungen die wichtigste Operation darstellt. Die Arithmetik-Leistung für diese Operation, die 4 Speicherzugriffe braucht (3 load, 1 store), wird durch die Cache-Bandbreite für Daten aus dem Cache und die Memory-Bandbreite für Daten aus dem Memory bestimmt. Ingenieuranwendungen sind i.a. memory-bound. Wir haben aus zwei Alternativangeboten das bezüglich Rechenleistung und Hauptspeicher günstigste Angebot ausgewählt, und zwar die Variante mit den Thin120-Knoten. Ein Thin120-Knoten hat eine bessere Speicherzugriffseinheit und bringt daher trotz niederer Frequenz eine höhere reale Leistung für Daten aus dem Memory als der Wide135-Knoten. Für Daten aus dem Cache skalieren die Prozessoren etwa mit der Frequenz, da sie bezüglich der Cache-Bandbreite die gleiche Architektur haben.

Verwunderlich ist, dass der Thin120 für Daten aus dem Memory kaum höhere, der Wide135 sogar geringere Leistung als der Wide77 bringt. Der Grund dafür liegt darin, dass man zwar die Prozessorfrequenz erhöht, die Memory-Zykluszeit aber gleich gelassen hat. Alle drei Prozessoren haben 256 Bit Bandbreite zwischen Cache und Memory, aber für die neuen Prozessoren arbeitet das Memory mit der halben Frequenz des Prozessors, so dass beim Thin120 und Wide135 effektiv nur eine Bandbreite von 128 Bit gegeben ist. Beim Thin120 klappt das Zusammenspiel zwischen dem langsameren Memory und dem schnellen Prozessor besser, so dass dieser eine höhere Leistung für Daten aus dem Memory hat. Bild 1 zeigt ein Block-Diagramm des Thin120 Node, aus dem die Breite der Datenpfade zu sehen ist. Man beachte das „Half Speed“ zwischen DCU (Data Cache Unit) und Memory.

An dieser Stelle ist es vielleicht interessant, einen Leistungsvergleich zwischen verschiedenen Workstation-Prozessoren anzuführen, die als Knoten in heute relevanten Supercomputern

verwendet werden. In Tabelle 3 sind die Leistungsdaten für die Vektor-Triade für 5 Prozesoren angegeben. Dabei ist

$$\eta_a = \frac{r_{real}}{r_{theoret}}$$

ein „Architekturwirkungsgrad“, der angibt, welchen Anteil der theoretischen Spitzenleistung man bei der entsprechenden realen Operation erreicht. Die Werte der Tabelle sprechen für sich. Nicht die hohe theoretische Spitzenleistung, sondern die Cache- und die Memory-Bandbreite bestimmen die reale Leistung, und da sind die IBM-Prozessoren bisher am besten.

Tabelle 2: Messungen von IBM in MFLOPS für Wide77-, Thin120- und Wide135-Knoten für die Vektor-Triade bei verschiedenen Vektorlängen n (Nov. 96)

n	Wide77	Thin120	Wide135	Bemerkung
1	12.4	20.9	23.5	data from cache
10	60.1	106.4	119.6	
10^2	125.0	191.6	221.2	
10^3	135.9	211.0	237.0	
10^4	52.3	52.2	42.3	date from memory
10^5	46.5	53.1	426	

Tabelle 3: Leistung in MFLOPS für verschiedene Workstation-Prozessoren für die Vektor-Triade für Daten aus dem Cache und dem Memory

Prozessor	f MHZ	Theoret. Spitzenleistung	Vektor-Triade aus Cache	Vektor-Triade aus Memory	ηa Daten aus Cache	ηa Daten aus Memory
Wide77	77	308	135.9	46.5	0.441	0.151
Thin120	120	480	211.0	53.1	0.440	0.111
DEC Alpha im CRAY T3E	300	600	134.0	37.4	0.223	0.062
Mips R10000 in SGI Origin	200	400	93.5	17.3	0.234	0.043
HP PA8000 in Exemplar	180	720	143.4	28.1	0.199	0.039

Dieser interessante Wettlauf wird sich bei der nächsten Generation der Prozessoren fortsetzen. Es kommt übrigens immer darauf an, in welcher „Umgebung“ ein solcher Prozessor verwendet wird, z.B. auch, unter welchem Compiler und dort, unter welcher Option.

Tabelle 4: Leistung in MFLOPS für verschiedene Kernoperationen. Wo mehrere Werte angegeben sind, handelt es sich um die Streubreite aus mehreren Messungen

$$A: a_i = (b_i + c_i) \cdot d_i + \frac{(h_i - f_i)}{g_i}$$

	add. $a_i = b_i + c_i$	mult. $a_i = b_i \cdot c_i$	div. $a_i = \frac{b_i}{c_i}$	linked triad $a_i = b_i + s \cdot c_i$	vecotr triad $a_i = b_i + c_i \cdot d_i$	scal. prod. $s = s + a_i \cdot b_i$	expr. A
Spalte	1	2	3	4	5	6	7
n = 1	7.6-8.9	8.1	8.2	15.7-16.0	15.6-16.4	11.9-12.3	48.1
10	50.5-58.1	53.2	15.6	106.7-111.1	90.9-93.5	88.2-88.8	181.2
10^2	113.6-135.1	125.0	14.3	235.3-242.4	185.2-192.3	200-202.7	271.7
10^3	119.0-151.5	138.9	14.3	258.1-285.7	204.1-212.8	234.4	297.6
10^4	31.2-34.5	32.9	13.9	65.6-66.7	52.1-52.6	111.9-112.8	102.5
10^5	31.6-32.9	32.5	13.9	64.5-65.0	52.1-52.6	102.7	100.8

Zum Schluss sollen noch die Kernmessungen, die wir üblicherweise machen, um einen neuen Rechnertyp zu untersuchen, für unsere „neue“ SP, also für die Thin120-Knoten in der SP, angegeben werden. Wir haben mit dem neuen Fortran 90-Compiler XLF 4.1.0.3 mit den Optionen -o3, -qarch=pwr2, -qtune=pwr2, -qhot gemessen. Um die Operationen wurden Wiederholungsschleifen gelegt, so daß die Messzeiten zwischen 0,5 und 1,5 Sekunden lagen. Dabei haben wir signifikante Streuungen bei wiederholten Messungen festgestellt. Wurde die Messzeit um den Faktor 10 erhöht, verringerten sich die Streuungen beträchtlich, und die Messwerte lagen etwa im Mittelbereich der Streuung. Die Ursache der Streuung ist einerseits die Ungenauigkeit der CPU-Uhr (10 msec), besonders aber sind es die Aktionen des Betriebssystems, die sich zwischen die Rechnung einblenden und zusammen mit der Ungenauigkeit der CPU-Uhr nichtreproduzierbare Zeiten, selbst auf einer dedizierten Anlage, bewirken. Dort, wo mehrfach gemessen wurde, wird die beobachtete Streubreite angegeben. Übrigens: die in Tabelle 2 aufgeführten Leistungsdaten sind unter anderen Compilerversionen von IBM gemessen, was einen merklichen Einfluss auf die Leistung hat.

In Tabelle 4 sind die gemessenen MFLOPS für einige Kernoperationen angegeben. Man sieht die teilweise beträchtliche Streubreite. Oberhalb der feinen Trennlinie hat man „data from cache“, unterhalb „data from memory“. Der Abfall der Leistung macht drastisch klar, wie wichtig für einen solchen Mikroprozessor die Wiederverwendung der Daten im Cache ist. Man sollte, wo das möglich ist, die Daten in Cache-Blöcken abarbeiten, so dass die Daten möglichst selten aus dem Memory geholt werden müssen. Die Messungen haben ferner gezeigt, dass die Leistung ganz beträchtlich von den gewählten Compiler-Optionen (und der Compilerversion) abhängt.

Die Werte in Tabelle 4 sind mit dicht liegenden Elementen (contiguous elements) gerechnet. In Tabelle 5 sind die MFLOPS für Addition mit n=100 und Abstand (stride) s angegeben. Stride mit hoher Zweierpotenz ist fatal für die Leistung. In Tabelle 6 ist die Leistung in MFLOPS für die Addition mit indirekter Adresse rechts (gather) und links (scatter) angegeben, jeweils für invertierten und konstanten Index. Unterhalb der gestrichelten Linie sind „data from memory“ aufgeführt.

Nicht erklärt werden können die Werte für

$n = 10^4$

und

$n = 10^5$

für gather bei invertiertem Index, die höher als bei dichten Elementen liegen.

Tabelle 5: Leistung in MFLOPS für die Addition mit n=100 Elementen, die mit Abstand (stride) s im Speicher liegen

s	1	2	3	4	8	16
MFLOPS	65.2	66.7	69.8	69.8	68.2	66.7
s	32	48	64	128	256	
MFLOPS	69.8	69.8	1.7	1.1	1.2	

Tabelle 6: Leistung in MFLOPS für die Addition bei indirekter Adressierung.

n	gather a(i) = b(ind(i)) + c(i)		scatter a(ind(i)) = b(i) + c(i)	
	ind(i) = n – i + 1	ind(i) = 1	ind(i) = n – i + 1	ind(i) = 1
1	17.9	17.2	8.1	8.1
10	50.5	24.6	29.8	29.4
10^2	65.8	26.2	42.4	43.1
10^3	7.6	26.6	43.1	43.1
10^4	38.8	26.2	21.0	26.9
10^5	38.8	25.3	20.7	26.3

In Tabelle 7 sind die MFLOPS-Werte für die Matrix-Multiplikation für verschiedene Dimensionen n der Matrix angegeben für fünf verschiedene Algorithmen: Skalarproduktform, simultan spalten- und zeilenweise, zeilenweise mit 4-fach Unrolling der zweitinnersten Schleife sowie Bibliotheksroutine. Die Matrizen sind jeweils in einem mit 512*512 dimensionierten Feld (leading dimension = 512) gespeichert. Wenn man die jeweilige Matrix in einem n*n-Feld speichert, erhält man teilweise andere MFLOPS-Werte. Wo mehrere Messungen gemacht wurden, ist jeweils der beste Wert angegeben. Die zeilenweise Form wird offensichtlich durch Schleifenvertauschung in die spaltenweise Form überführt. Das 4-fach Unrolling bringt nur bei kleinem und grossem n einen Vorteil. Die Bibliotheksroutine ist teilweise schlechter als der spal-

tenweise Code. Diese Tabelle zeigt den signifikanten Einfluss des Algorithmus auf die Leistung: Es wird jeweils dasselbe Problem gelöst, nur geht man dabei auf verschiedene Weise durch die Daten. Die Bibliotheksroutine ist sicher nicht optimal. Sie müsste ab n=100 in allen Fällen eine Leistung nahe an der theoretischen Spitzenleistung von 480 MFLOPS bringen. Wir wissen, wie der „richtige“ Algorithmus aussehen müsste. Warum realisiert IBM ihn nicht?

Tabelle 7: Leistung in MFLOPS für die Matrix-Multiplikation für fünf verschiedeneAlgorithmen bei verschiedener Dimension n (die Matrizen sind n x n).

n	scal. prod. form	rowwise form	col.-wise form	with } 4-fold unroll.	library routine
10	58.8	84.5	89.3	100	125.0
100	135.1	375.0	379.7	357.1	342.5
350	93.2	364.9	368.0	189.7	296.7
500	7.7	362.3	362.3	201.6	316.5
512	7.7	106.9	107.4	179.0	308.5

Zusammenfassend kann man sagen, dass wir in der „neuen“ SP ein mächtiges Werkzeug für die numerische Simulation haben, von dem unsere Ingenieure vor wenigen Jahren kaum zu träumen wagten. Jetzt kommt es darauf an, dieses Werkzeug sinnvoll zu nutzen. Es kann unseren Ingenieuren einen entscheidenden Vorsprung verschaffen, wenn sie durch numerische Simulation, meist durch numerische Lösung von partiellen Differentialgleichungen, technische Fragestellungen berechenbar machen, die früher mühselig in langwierigen Experimenten beantwortet werden mussten. Jetzt kann man im Computer viele Fälle „durchspielen“ und schnell eine optimale Lösung finden. Die Mitarbeiter des Rechenzentrums freuen sichdarauf, ihren Benutzern hierbei zu helfen.

Noch ein technischer Hinweis:

Weitere Informationen zur Nutzung der „neuen“ SP stehen im WWW bereit unter der URL

http://www.uni-karlsruhe.de/~SP/

und im Rahmen des Höchstleistungsrechenzentrums Karlsruhe (HLRKA) unter

http://www.hlrka.de/

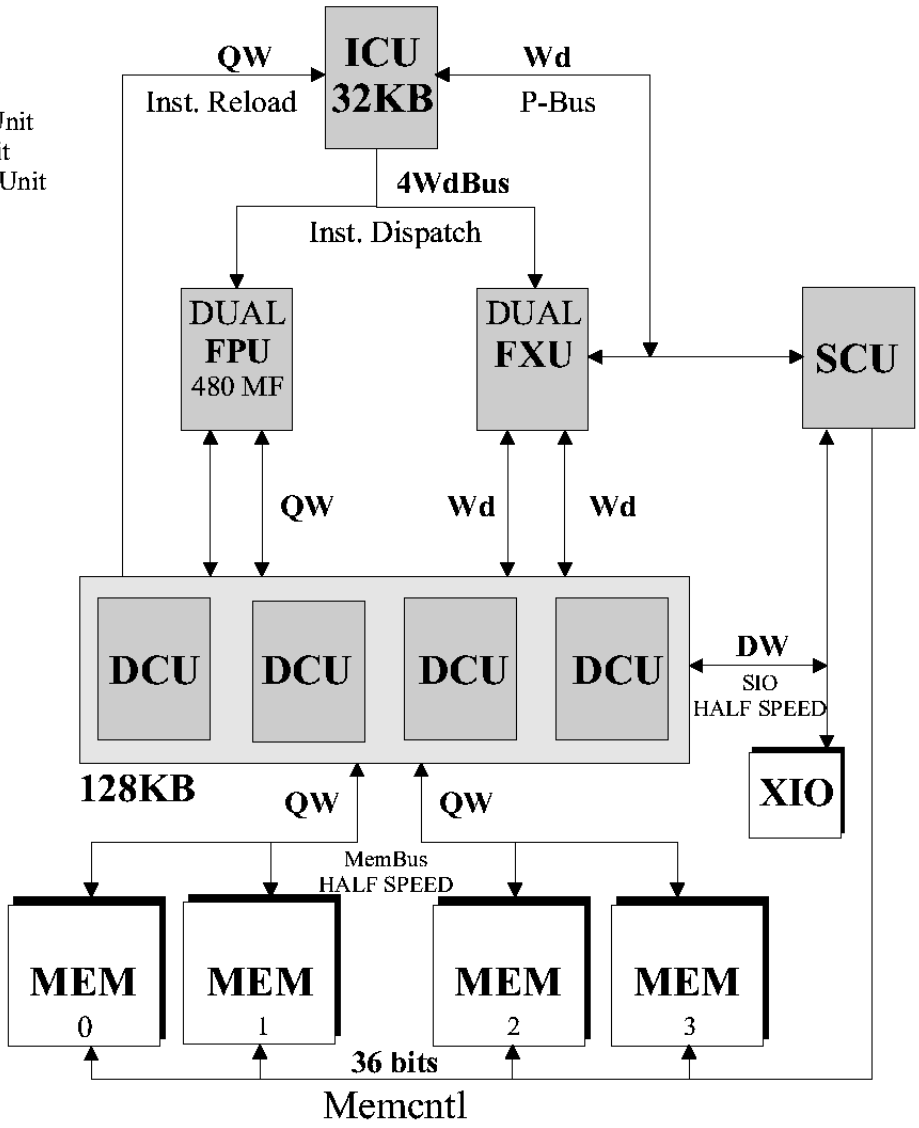
Prof. Dr. Willi Schönauer
Universität Karlsruhe

RS-2/120

Wd = 4 bytes
DW = 6 bytes
QW = 16 bytes
I/D Line Size = 128B/256B
I/D Line Xfer = 8/16 CPU cy
Freq = 120 MHz

ICU = Instrc Cache
DCU = Data Cache
FPU = Floating Pt. Unit
FXU = Fixed Pt. Unit
SCU = Storage Cntl Unit
XIO = MC bus cntl

SP Node:
THIN P2SC



Das Universitätsrechenzentrum (URZ) Ulm

Michaela Hering

Überblick:

Das Universitätsrechenzentrum (URZ) Ulm stellt sich vor

Anschriften

Unsere Adresse:

Universität Ulm
Universitätsrechenzentrum
Albert-Einstein-Allee 11
D-89081 Ulm

Unsere Postanschrift:

Universität Ulm
Universitätsrechenzentrum
D-89069 Ulm

Leiter des URZ

Prof. Dr. H.P. Grossmann
Abt. Organisation und Management von Informationssystemen

E-Mail: hans-peter.grossmann@rz.uni-ulm.de
Telefon ++49-(0)731-502-2502
Telefax ++49-(0)731-502-2471

Komm. stellv. Leiter:
Thomas Nau
E-Mail: thomas.nau@rz.uni-ulm.de
Telefon ++49-(0)731-502-2464

Aktivitäten des URZ

Basisdienstleistungen

Bereitstellung einer Netzwerksinfrastruktur für Forschung und Lehre . Nach einjähriger unproblematischer Laufzeit der FDDI-Glasfaserringe in der Baustufe A der Universität Ost sowie in der Universität West wurde zum Jahreswechsel 1994/1995 das bewährte Konzept zur Sanierung der Netzwerksinfrastruktur in den Baustufen B und C der Universität Ost mit der Installation eines separaten FDDI-Ringes fortgeführt. Der im Festpunkt O27 angesiedelte Fachbereich Informatik wurde aufgrund seines hohen Netzverkehrsaufkommens ebenfalls mit einem eigenen FDDI-Ring ausgestattet. Alle diese Ringe sind mittels eines Switches miteinander verbunden. An jedem Glasfaserring sind über diverse Festpunktrouter die Rechner der Abteilungen, Sektionen und Institute angeschlossen. Weitere Router erfüllen Sonderfunktionen wie z.B. Außenanbindungen via ISDN und per Modem, Zugang zum Internet mit 34 Mbit/s. Daneben existieren Firewalls zum Verwaltungs- und zum Kliniknetz der Universität Ulm. Der Netzausbau an der Universität Ulm ist durch zahlreiche Workgroup-(Glasfaser)-Ringe und den Einsatz von FDDI-/Ethernet-Switches in Abteilungen mit hohem Rechenaufkommen gekennzeichnet. Innerhalb der Workgroups ist damit eine hohe Performance garantiert, ohne daß der Backbone belastet wird. Im Juni 1997 waren ca 2700 Rechner an das Netz angeschlossen und ca 1600 Dial-In-Nutzer eingetragen: sie haben von zuhause aus via ISDN oder per Modem Anschluss an das lokale Netz der Universität Ulm und damit ins Internet.

Bereitstellung von Rechenanlagen und Peripheriegeräten. Das URZ stellt an das Netzwerk angeschlossene Rechnersysteme und Ausgabegeräte unterschiedlicher Kapazität und Leistungsstärke bereit (s. Technische Ausstattung). Ein Workstation-Pool sowie vier PC-Pools stehen zur Nutzung durch unterschiedliche Klientel zur Verfügung. Es organisiert einen Wartungspool für Workstations und PC‘s; kleinere Reparaturen werden am URZ ausgeführt.

Internet-Dienste. Das URZ ermöglicht die Nutzung der wichtigsten Internet-Dienste wie Electronic Mail (e-mail), World Wide Web (www), File Transfer (ftp) und News. Ausserdem betreibt es den zentralen WWW-Server der Universität Ulm (URL: http://www.uni-ulm.de/). Auf diesem Server können sich Abteilungen, Sektionen und sonstige Einrichtungen der Universität Ulm - auch das URZ - sowie alle Nutzer im WWW präsentieren.

Anwenderberatung . Das URZ unterstützt seine Nutzer hinsichtlich einer optimalen Nutzung der verfügbaren Hard- und Softwareressourcen für die Bereiche Scientific Computing, High Performance Computing, nicht-medizinische Statistik, Biocomputing, Datenbank, Visualisierung u.a.m..... Die Anwender finden kompetente Beratung bei der Auswahl geeigneter EDV-Verfahren für ihre Projekte. Selbstverständlich unterstützt das URZ auch Anwender von PC-Standardsoftware. Auch bei Problemen im Umgang mit den Ressourcen des URZ und abteilungseigenen EDV-Systemen gibt das URZ intensive Hilfe zur Selbsthilfe. Es berät Angehörige wissenschaftlicher Abteilungen bei Hard- und Software-Beschaffungsmassnahmen und verfaßt Stellungnahmen.

Kurse . Im URZ finden regelmäßig folgende Kurse statt: UNIX, Fortran, C/C++, SAS, MAPLE, verschiedene Visualisierungs-Tools, Internet-Dienste, PC-Standardanwendersoftware (Textverarbeitung, Tabellenkalkulation, Datenbank).

Neue Services

Zum Selbstverständnis des URZ .Die Dynamik der technischen Entwicklung im Bereich der Informationstechnik hat die Aktivitätsfelder von Hochschulrechenzentren stark ausgeweitet und wird sie auch weiterhin modifizieren. Die Rechenzentren, so auch wir, verstehen sich zunehmend als Service- und Kompetenzzentren, die sich nach neuen Belangen der Nutzer - vor allem im Bereich moderner Informationstechnologien - ausrichten.

Services rund um neue Informationstechnologien. Das URZ sammelt Erfahrungen in der Erstellung und in der Ausgabe von Publikationen, Präsentationen und Nutzerinformationen in Druck- und Online-Versionen. Ausserdem bie-

ten wir Nutzern Unterstützung beim Erstellen von Publikationen und beim Internet Publishing (Verfahren, Editoren etc., Layout) an und suchen nach Lösungen für Probleme bei der Ausgabe auf grafischen und Farb-Ausgabegeräten des URZ. In Zusammenarbeit mit der Zentrale für Foto, Grafik und Reproduktion der Universität Ulm ist ein Dia-Service eingerichtet, der Dia-Files direkt vom Rechner des Nutzers über einen Server des URZ via FTP zur ZPhGR transportiert. Auch die Erprobung neuer Lehr- und Lernmethoden wie Computer Based Training und video-unterstütztes Lernen nehmen einen breiten Raum ein.

Visualisierung wissenschaftlicher Simulations- und Ergebnisdaten. Am URZ wird seit kurzem das Competence Center für Deutschland von AVS/UNIRAS aufgebaut. Es soll in naher Zukunft detaillierte Nutzerinformationen liefern und intensive Unterstützung bei der Anwendung dieser Softwareprodukte bieten. Weitere Einzelheiten sind in Kürze auf dem WWW-Server des URZ (http://www.uni-ulm.de/urz) zu erwarten. Darüber hinaus finden Nutzer weitere Visualisierungstools für ihre Forschungsprojekte vor : PV Wave u.a. Einen besonderen Weg der Visualisierung verfolgt die Computeranimation mit Hilfe der „Analog Draw“. Dieses analoge Bildplattenspeichersystem (s. 4. Technische Ausstattung, 4.3 Besondere Geräte) stellt wissenschaftlich-technische Daten in Bewegtbildern dar.

Auslagern und Sichern individueller Nutzerdaten auf Magnetbändern . Dieser neue Service des URZ bedeutet eine Erweiterung des universitätsweiten Backup-Dienstes, der alle wichtigen Datenbereiche (Homedirectories, Mail, Software etc.) erfasst. Er bietet Nutzern zusätzlich die Möglichkeit, selbst zu entscheiden, welche ihrer Daten sie auslagern und sichern wollen. Er soll in den künftigen Backup- und Archiv-Service des URZ integriert werden, sobald der darauf zielende HBFG-Antrag genehmigt ist.

Bevor Sie die Dienste nutzen ...

Auf Antrag hin. Bevor Sie die Ressourcen des URZ nutzen bzw. seine Dienstleistungen in Anspruch nehmen können, muss das URZ auf Ihren Antrag hin eine Nutzungsberechtigung erteilen. Das „Allgemeine Antragsformular“ ist bei den Operateuren, Herrn Micheler und Herrn Rapp, im Gebäudefestpunkt O26/5101 erhältlich oder online auf unserem WWW-Server verfügbar:

http://www.uni-ulm.de/urz/Nutzeranträge/Anträge.html

Gültigkeitsdauer. Die Berechtigungen gelten maximal ein Jahr. Zu Beginn des Wintersemesters (Studenten) und zu Jahresbeginn können Verlängerungen beantragt werden.

Anschluss ans Lokale Netzwerk gewünscht. Wenn Sie Ihren Rechner ins lokale Netzwerk (LAN) der Universität einbinden wollen, benötigt er eine eindeutige Netzadresse. Diese kann ausschliesslich vom URZ erteilt werden. Der entsprechende Formular „Antrag auf Zuteilung einer Netzadresse“ ist ebenfalls bei den Operateuren, Herrn Micheler und Herrn Rapp, im Gebäudefestpunkt O26/5101 zu erhalten oder online auf unserem WWW-Server verfügbar (s.o.).

Projekte und Forschungsschwerpunkte des URZ

BELCANTO. Das Landesforschungsnetz von Baden-Württemberg, das bislang ausschliesslich zur Datenkommunikation zwischen den Anschlusspartnern genutzt wurde, soll um einen wichtigen Dienst erweitert werden: um die Telefonie. Diese innerhalb des Landesforschungsnetzes im Rahmen eines Corporate Network einzuführen, zu erproben und zu optimieren, ist das Ziel des Projekts BELCANTO. (Baden-Württemberg Extended LAN Corporate ATM Network Telephony Optimization). Bislang erfolgen Sprachübertragung und Datenübertragung in zwei voneinander unabhängigen Infrastrukturen. Im Rahmen des Projekts sollen 34 Mbit/s- und später 155 Mbit/s-Verbindungen auf ATM-Basis zum Einsatz kommen, die beide Kommunikationsarten über ein einziges Netz abwickeln können. An drei Standorten soll je eine Pilotinstallation zur Sprach-Daten-Integration erfolgen: in den Rechenzentren der Universitäten Ulm und Mannheim und im Ministerium für Wissenschaft, Forschung und Kunst (MWK) Baden-Württemberg, Stuttgart. Anhand der dort gewonnenen Erfahrungen soll am Ende des Projektes eine Empfehlung für die optimale technische Gestaltung des landesweiten Corporate Network vorgelegt werden. Parallel dazu wird ein praktikables Betriebsmodell für ein solches Corporate Network erarbeitet.

Firewall . Das URZ betreibt seit Anfang 1995 ein Firewallsystem, das das Netzwerk der Universitätsverwaltung mit dem offenen For-

schung&Lehre-Netz und dem Internet verbindet. Dieser Übergang soll auf der einen Seite verwaltungsinterne Daten vor Zugriffen von aussen schützen, auf der anderen Seite den Mitarbeitern in der Verwaltung den Zugriff auf Dienste im F&L-Netz und im Internet (E-Mail, WWW) erlauben. Im Auftrag der DFG wurde eine Studie erarbeitet, die den Einsatz von Firewallsystemen in Universitätsverwaltungen untersucht. Eine Volltextversion im Postscript-Format steht Interessenten auf dem WWW-Server des URZ zur Verfügung:

http://www.uni-ulm.de/urz/Projekte/dfgfir.ps

Windows NT. Seit Frühjahr 1996 wird am URZ das Betriebssystem Windows NT in einem realitätsnahen Testumfeld auf Servern und Clients betrieben. Nutzer sind im Augenblick Mitarbeiter des Rechenzentrums. Ziel dieser auf einen längeren Zeitraum ausgerichteten Untersuchungen ist es, die Verwendbarkeit von NT in Produktionsumgebungen zu evaluieren und unter technischen, administrativen sowie nutzerspezifischen Aspekten mit den im Hochschulbereich vielfach eingesetzten UNIX-Systemen zu vergleichen. Im Auftrag der DFG wurde ebenfalls eine Studie erarbeitet, die die Eignung von NT für den Einsatz in Universitätsverwaltungen untersucht. Auch hier steht eine Volltextversion im PDF-Format auf dem WWW-Server des URZ zur Verfügung:

http://www.uni-ulm.de/urz/Projekte/nt.pdf

Abgeschlossen: ULKOM. Zwischen 1989 und 1996 wurden am URZ drei Entwicklungsprojekte durchgeführt und abgeschlossen: ULKOM und UDINE und UDINE-II. ULKOM ist ein 1989 begonnenes und Mitte 1993 abgeschlossenes Pilotprojekt, bei dem das URZ mit der Deutschen Telekom (Stuttgart), der Fa. SNI (München) und der Fa. Zeiss (Oberkochen) zusammenarbeitete. Das Projekt sich hatte zum Ziel gesetzt, im Zuge des geplanten Ausbaus der Universität Ulm sowie der Wissenschaftsstadt am Oberen Eselsberg eine auf die Bedürfnisse der Anwender ausgerichtete Informations- und Kommunikationslösung zu schaffen. Deren Notwendigkeit ergab sich aus der Kooperation verschiedener, räumlich getrennt liegender Einrichtungen und im Hinblick auf die Nutzung gemeinsamer Ressourcen. Im Verlauf des Projekts wurden auf der Basis des Vermittelnden Breitbandnetzes VBN der Deutschen Telekom (max. 140 Mbit/s) Videokonferenzen, Bildtelefonie, schnelle Datenübertragung, LAN-LAN-Kopplung u.v.a. durchgeführt und getestet. Vor allem im medizinischen Bereich ergaben sich zahlreiche Testkonstellationen, z.B. in der Fernschnellschnittdiagnostik und im Rahmen konsiliarischer Begleitung von Operationen. Aus dem Teilbereich Telemedizin von ULKOM entwickelte sich das deutsch-russische Folgeprojekt RATEMA (Radiation Accident Telecommunication Medical Assistance System). Nach Abschaltung des VBN-Anschlusses durch die Deutsche Telekom im August 1994 wurden Tests auf der Basis des ISDN weitergeführt. Zwischen der Abteilung Pathologie der Universität Ulm und dem Bezirkskrankenhaus Günzburg bestand eine ISDN-Strecke, die bis vor kurzem im Routinebetrieb für Fernschnellschnittdiagnostik genutzt wurde.

Ebenfalls abgeschlossen: UDINE und UDINE-II. Das Ziel des 1992 gestarteten Projekts UDINE (Universal Document Information and Navigation Environment) war die Entwicklung eines flexiblen, multimedialen Informationssystems, das Dokumente (Text, Bild, Audio und Video) unter einer einheitlichen Benutzeroberfläche auf unterschiedlichen Rechnerplattformen darstellen kann. Eine weitere wesentliche Eigenschaft von UDINE bestand darin, daß Informationen dort in das System eingebracht werden sollten, wo sie entstehen, ohne spezielle Kenntnisse der Nutzer am Arbeitsplatz. Ausserdem wurde UDINE 1995 um ein Modul erweitert, welches den Zugang zu Online-Datenbanken wie OCLC und FIZ Karlsruhe vermittelt. 1994 begann mit dem Mosaic Browser der Siegeszug des World-Wide-Web. Um auf „Web“-Informationen zugreifen zu können, wurden die UDINE-Clients um einen WWW-Browser erweitert. Speziell der Browser für MS-Windows war sehr erfolgreich. In Zusammenarbeit mit der Abteilung Organisation und Management von Informationssystemen der Universität Ulm (Leiter: Prof. Dr. H.P. Grossmann) wurde er zu einem eigenständigen, sehr leistungsfähigen WWW-Browser (UdiWWW) weiterentwickelt. Das UDINE-System wurde 1994 und 1995 vielfach vorgestellt, u.a. im Ministerium für Wissenschaft und Forschung Baden Württemberg (MWF). Ende 1994 wurde das UDINE-Projekt in dieser Form abgeschlossen. Das Folgeprojekt UDINE-II begann 1995. Mit dessen Hilfe ist Interactive Publishing von einem gewöhnlichen WWW-Browser am Arbeitsplatz aus möglich.

Organisationsstruktur

Leitung

Prof. Dr. H.P. Grossmann
Abt. Organisation und Management von Informationssystemen
E-Mail: hans-peter.grossmann@rz.uni-ulm.de
Telefon ++49-(0)731-502-2502
Telefax ++49-(0)731-502-2471

Kommis. stellv. Leiter:

Thomas Nau
E-Mail: thomas.nau@rz.uni-ulm.de
Telefon ++49-(0)731-502-2464

Arbeitsgruppen (AGs)

Arbeitsgruppen sind permanente Einrichtungen zur Wahrnehmung der Routine-Arbeiten des URZ; die Mitglieder sind fest zugeordnet.

- AG Nutzerservices (Nutzerberatung, Schulung, Materialverkauf u.a.)
- AG Basissysteme (Betriebssysteme, übergreifende Campusservices, z.B. Backup= , Archivierung ...)
- AG Netzwerk (Netzplanung und -betrieb, Netzbasisdienste, Sicherheit, Telekommunikation)
- AG Verwaltung (Betreuung der Verwaltungsanwendungen)
- AG Informationssysteme und Visualisierung (Betreuung der Visualisierung und Grafiksysteme, Internetservices, Medienerstellung)
- AG Gesamtkoordination (Aussenvertretung, Stellen- und Mittelverwaltung, controlling, allgemeine Verwaltung des URZ)

Projektgruppen (PGs)

Projektgruppen sind temporäre Einrichtungen zur Wahrnehmung grösserer und / oder übergreifender Sonderaufgaben mit zeitlicher Befristung; die Mitglieder werden temporär zusammengestellt.

Technische Ausstattung

Rechner und Ausgabegeräte

SUN Enterprise Server. Lyra und Wega heissen die beiden Spitzenrechner des URZ. Diese zwei Enterprise Server der Fa. SUN sind seit Januar 1997 in Betrieb und sollen den Bedarf an allgemeiner Rechenleistung, der aus den Forschungsabteilungen der Universität Ulm kommt, decken. Plattenspeicher in RAID5-Konfiguration ist in der Grössenordnung von 27 x 4 GB vorhanden. Nähere Angaben zu diesen Rechnern sowie zu allen anderen vom URZ bereitgestellten Unix-Workstations verschiedener Plattformen sind in stets aktueller Form auf unserem WWW-Server zu erhalten:

http://www.uni-ulm.de/urz/Hard_Software/workstation.html

MPP-System nCUBE3. Für massiv-parallele Anwendungen betreibt das URZ das System nCUBE3 unter dem Betriebssystem Transit, einem PLAN9-Derivat.

Pools. Ein Unix-Workstation-Pool und vier PC-Pools sind für die jeweils berechtigte Klientel frei zugänglich.

Ausgabegeräte. An höherwertigen Ausgabegeräten verfügt das URZ über einen Farblaserdrucker, einen Farbtintenstrahlplotter und einen schnellen Postscript Laserdrucker für beidseitigen Druck. Mehr Informationen zu diesen und weiteren Geräten sind wiederum auf unserem WWW-Server zu finden:

http://www.uni-ulm.de/urz/Hard_Software/Druck/index.html

Besondere Geräte

Computeranimationssystem „Analog Draw“. Die Analog Draw ist ein analoges Bildplattenspeichersystem, mit dem am Computer erzeugte (digitale) Einzelbilder auf einer Bildplatte analog abgelegt werden können. Diese Einzelbilder können zu Sequenzen und die Sequenzen zu einem Film in der Art eines „elektronischen Daumenkinos“ zusammengefasst werden. Die Bedienung der Analog Draw erfolgt über eine X-Windows-basierte Software, die auf der SGI-Workstation Purple (s.o.) läuft, an die die Analog Draw angeschlossen ist. Um Bilder auf der Bildplatte ablegen zu können, müssen sie zunächst in ein Format gewandelt werden, das die Bildplatte schreiben kann. Diese Aufgabe übernimmt der Real-Time-Converter RTC. Weitere Angaben:

http://www.uni-ulm.de/Dienste/b_anima.html

Professionelle Video-Echtzeitkodierung im MPEG1-Standard. Wer seine analogen Videoclips auch in der Computerwelt präsentieren will, z.B. im WWW, kann sie mit der Maschine

URZMPEG1 des URZ im MPEG1-Format in Echtzeit digitalisieren lassen. Die URZMPEG1 ist ein Pentium-90-PC mit zusätzlicher Hardware. Details zu Hard- und installierter Software sind unter

http://www.uni-ulm.de/DiensMite/b_mpeg1.html

zu finden.

Aktuelles Softwareangebot

Die auf den Rechnern des URZ installierte Software sowie die von uns verwalteten Mehrfach-, Campus- und Landeslizenzen sind auf unserem WWW-Server zusammengestellt. Ab der Seite mit der URL

http://www.uni-ulm.de/Hard_Software/index.html

kann in die jeweilige Hard- und Software-Rubrik verzweigt werden.

Michaela Hering
Universität Ulm
michaela.hering@rz.uni-ulm.de

Zentrale BelWü-Dienste

Das BelWü bindet wissenschaftlichen Einrichtungen, Bibliotheken und Schulen per Wähl- und Festverbindung ans Internet an. Für die angebundenen Einrichtungen bietet die BelWü-Koordination eine Reihe von Netzdiensten, die nachfolgend aufgeführt sind, zentral an. Dies ist insbesondere für BelWü-Teilnehmer interessant, die mit nur geringer Bandbreite oder über eine Wählverbindung angeschlossen sind und solche Services nicht selbst erbringen können. Unter anderem betrifft dies die meisten an das BelWü angeschlossenen Schulen.

Zu den zentral angebotenen Diensten gehören:

- Primary und Secondary DNS-Nameservice auf noc.belwue.de
- SMTP-Mailrelay noc.belwue.de mit Zwischenspeicherung der Mail und Auslieferung bei Verfügbarkeit des SMTP-Servers des Teilnehmers
- UUCP-Mailserver news.belwue.de
- POP-Mailbox Service auf noc.belwue.de
- NTP-Network Time Service
- FTP-Server ftp.belwue.de mit Netzwerksoftware, Benutzerhandbüchern und Beispielkonfigurationen (z.B. sendmail.cf).
  - Virtuelle ftp-Server auf ftp.belwue.de unter dem Domainnamen des Teilneh mers
- WWW-Server www.belwue.de mit Informationen zum Anschluss an das BelWü undHinweisen auf Konfigurationsanleitungen anderer BelWü-Teilnehmer,insbesondere Schulen
  - Virtuelle WWW-Server auf www.belwue.de unter dem Domainnamen des Teilnehmers
- Newsserver news.belwue.de, Newsfeed und Client-Zugriffe (NNTP, UUCP) für BelWü-Teilnehmer
- Telnet/X.29-Gateway x29-gw.belwue.de
- OSS-Zugang zu SAP in Walldorf
- Verbindungsstatistik bei ISDN-Wählverbindungen

Die Universität Heidelberg bietet zentral für BelWü-Telnehmer an:

- Archie-Server archie.belwue.de zur Suche nach FTP Archiven

(Siehe auch http://www.belwue.de/BelWue/sv-home.html)

Dienst	Server
DNS	noc.belwue.de
SMTP-Mailrelay	noc.belwue.de
UUCP-Mailserver	news.belwue.de
POP-Mailboxserver	noc.belwue.de
NTP-Server	auf Anfrage
FTP-Server	ftp.belwue.de
WWW-Server	www.belwue.de
News-Server	news.belwue.de
Telnet/X.29 - Gateway	x29-gw.belwue.de
Archie-Server	archie.belwue.de

BelWü-Beauftragte

Willibald Meyer Universität **Freiburg** Rechenzentrum Hermann-Herder-Str. 10 79104 Freiburg 0761/203-4622 mywi@sun1.ruf.uni-freiburg.de	Hartmuth Heldt Universität **Heidelberg** Rechenzentrum Im Neuenheimer Feld 293 69120 Heidelberg 06221/54-5451, Fax: -5581 Hartmuth.Heldt@urz.uni-heidelberg.de
Andreas Tabbert Universität **Hohenheim** Rechenzentrum Schloß Westhof-Süd 70593 Stuttgart 0711/459-2838 tabbert@uni-hohenheim.de	Brian Worden Regionales Hochschulrechenzentrum Postfach 3049 Paul-Ehrlich-Straße 67653 **Kaiserslautern** 0631/205-2448 worden@uni-kl.de
Dr. Bruno Lortz Universität **Karlsruhe** Rechenzentrum Zirkel 2 Postfach 6980 76128 Karlsruhe 0721/608-4030 lortz@rz.uni-karlsruhe.de	Jörg Vreemann Universität **Konstanz** Rechenzentrum Postfach 5560 78434 Konstanz 07531/88-3893 joerg.vreemann@uni-konstanz.de
Ralf-Peter Winkens Universität **Mannheim** Rechenzentrum L15,16 68131 Mannheim 0621/292-5781, Fax: -5012 winkens@rz.uni-mannheim.de	Dr. Lisa Golka Rechenzentrum der Universität **Stuttgart** Allmandring 30 70550 Stuttgart 0711/685-5983 golka@rus.uni-stuttgart.de
Dr. Heinz Hipp Universität **Tübingen** Zentrum für Datenverarbeitung Brunnenstr. 27 72074 Tübingen 07071/29-6967, Fax: -5912 hipp@zdv.uni-tuebingen.d400.de	Pius Hieber Universität **Ulm** Rechenzentrum Albert-Einstein-Allee 11 89069 Ulm2463 0731/502- pius.hieber@rz.uni-ulm.de

BelWü-Koordination

Rechenzentrum der Universität Stuttgart
Allmandring 30
70550 Stuttgart

belwue-koordination@belwue.de
anschluss@belwue.de
schul-anschluss@belwue.de
schul-lan@belwue.de
schul-netzsw@belwue.de

Hotline:

0711/685-5797

Betrieb und Dienste

Peter Merdian
0711/685-5804
merdian@rus.uni-stuttgart.de

Jürgen Georgi
0711/685-5739
georgi@belwue.de

Joseph Michl
0711/685-5807
michl@belwue.de

Ulli Horlacher
0711/685-5868
framstag@belwue.de

Andreas Koppenhöfer
0711/685-7225
koppenh@belwue.de

Beate Herrmann
0711/685-5372
beate@belwue.de

BelWü-Maillisten

belwue@belwue.de
dialup@belwue.de
ag-netzdienste@belwue.de
netzbetrieb@belwue.de
netzqualitaet@belwue.de
netzplanung@belwue.de
netz-probleme@belwue.de

Entwicklung und Projekte

Paul Christ
0711/685-2515
christ@rus.uni-stuttgart.de

Chris Copplestone
0711/685-5987
copplestone@rus.uni-stuttgart.de

Tassilo Erlewein
0711/685-5871
erlewein@rus.uni-stuttgart.de

Holger Fahner
0711/685-5736
fahner@rus.uni-stuttgart.de

Jürgen Jähnert
0711/685-4273
jaehnert@rus.uni-stuttgart.de

Angela Giannitrapani
0711/685-4145
giannitrapani@rus.uni-stuttgart.de

Weiping Li
0711/685-5735
li@rus.uni-stuttgart.de

Andreas Rozek
0711/685-4514
rozek@rus.uni-stuttgart.de

Ingo Seipp
0711/685-5988
seipp@rus.uni-stuttgart.de

Robert Stoy
0711/685-5859
stoy@rus.uni-stuttgart.de

Stefan Wesner
0711/685-4145
wesner@rus.uni-stuttgart.de

Inhaltsverzeichnisse aller BelWü-Spots

Die Ausgaben der BelWü-Spots können Sie über die BelWü-Koordination (Rechenzentrum der Universität Stuttgart, Allmandring 30, 70550 Stuttgart, Tel.: 0711/685-5804, belwue-koordination@belwue.de) beziehen. Im übrigen sind die Ausgaben auch auf dem WWW-Server der BelWü-Koordination unter der URL http://www.belwue.de/BelWue/spots/belwuespots.html zu finden.