Leitfaden zur Absicherung von Rechnersystemen in Netzen

DFN-CERT, Zentrum für sichere Netzdienste GmbH

11. Juli 2000

Version 1.0

Copyright

Die diesem Leitfaden zugrunde liegenden Forschungsvorhaben wurden mit Mitteln des Bundesministers für Bildung und Forschung gefördert.

Vorwort

Die Idee zum Leitfaden entstand vor circa drei Jahren. Bei den zu behandelnden Inhalten zum Thema IT-Sicherheit ist natürlich ein bestimmender Faktor der Kenntnisstand der Zielgruppe, an die sich ein solcher Leitfaden richtet. Der Kreis der potentiell Angesprochenen wird von unserer Seite als sehr groß angesehen, da sich das DFN-CERT als Ansprechspartner in Fragen der Rechner- und Netzwerksicherheit für alle nationalen Hochschulen und wissenschaftlichen Forschungseinrichtungen betrachtet. Bei einem so großen Klientel muß man sich klar sein, daß bei den Anzusprechenden von einem stark differenzierten Kenntnisstand über IT-Sicherheit auszugehen ist. Es erschien uns daher unmöglich, alle möglichen Zielgruppen mit derem unterschiedlichen Wissenstand beim Erstellen eines Leitfadens zu berücksichtigen.

Heute ist eine gute Absicherung von in Netzen betriebenen Rechnern ein zwingendes Muß, denn immer mehr Rechner erlangen Zugriff auf das Internet und immer häufiger sind Angriffe auf schlecht administrierte und abgesicherte Rechner erfolgreich, die dann entweder selbst für längere Zeit ausfallen oder dazu genutzt werden, um von diesen aus andere Rechensysteme anzugreifen.

Inzwischen gibt es eine Vielzahl sehr guter Fachliteratur zur Absicherung von IT-Systemen (siehe auch Literaturverzeichnis ab S. 161), die sich speziell an Experten richten, die heute schon große Netze administrieren. Mit der Herausgabe dieses Leitfadens wollen wir dieser Fachliteratur in keiner Weise Konkurrenz machen. Als Zielgruppe betrachten wir speziell solche Leser, die keine Experten auf dem Gebiet der IT-Sicherheit sind und die wissen möchten, wie und wodurch IT-Sicherheit „in kleinem Rahmen“ erreicht werden kann.

Da sich die Herausgabe des Leitfadens über längere Zeit hinzog, waren am Erstellen des Leitfadens viele Mitarbeiter aus den Projekten DFN-CERT, DFN-PCA und DFN-FWL in unterschiedlichem Maße stark beteiligt. Ich möchte diesen Mitarbeitern danken; mitgewirkt haben (in alphabetischer Reihenfolge): Ingmar Camphausen, Uwe Ellermann, Peter Janitz, Stefan Kelm, Wolfgang Ley, Britta Liedtke, Heino Peters, Dirk Reimers, Rüdiger Riediger und Les Schaefer.

Wir würden uns freuen, wenn Sie uns mit Ihrem Feedback helfen würden, diesen Leitfaden fortschreiben und weiterentwickeln zu können.

Hans-Joachim Mück

Inhaltsverzeichnis

Kapitel 1

Einführung

Gerade in der heutigen Zeit wird der Sicherheit von Rechnern, Netzwerken und verteilten Systemen eine immer größer werdende Beachtung geschenkt. Dies liegt vor allem an den offenkundigen Defiziten der eingesetzten Systeme und den immer wieder bekannt werdenden konkreten Vorfällen. Aber auch der Einsatz verwundbarer Systeme in neuen Anwendungsbereichen, in denen Sicherheitslücken nicht toleriert werden können, trägt dazu bei.

Die Klientel von DFN-CERT und DFN-PCA setzt sich vor allem aus Universitäten, Fachhochschulen und Forschungseinrichtungen zusammen. Dies heißt jedoch nicht, daß bei dem Betrieb der an das Deutsche Forschungsnetz angeschlossenen Systeme die Sicherheit keine Rolle spielen würde. Die Existenz der Einrichtungen ist nicht unbedingt so eng an die gespeicherten Informationen gebunden wie z.B. bei Wirtschaftsunternehmen oder Regierungseinrichtungen. Allerdings können grundsätzlich etwaige Ausfälle, eine unberechtigte Nutzung (z.B. für weitere Angriffe) oder eine Beeinträchtigung des öffentlichen Ansehens nicht hingenommen werden. Die Verantwortung für die betroffenen Benutzer, die Haftung für Schäden und Konsequenzen sowie geltende Gesetze erzwingen vorbeugende Maßnahmen. Die Nutzung von Systemen für kostenintensive Experimente sowie die Bearbeitung personenbezogener Daten (z.B. für Verwaltungsaufgaben oder im Rahmen von medizinischer Forschungen) bewirkt, daß spezielle Sicherheitsanforderungen an diese Systeme gestellt werden. Auch die Beeinträchtigungen durch solche Vorfälle selbst, bei denen die Beseitigung der Folgen notwendige Arbeitsleistung für eigentliche Aufgaben nicht mehr zur Verfügung stehen läßt, zeigt die Notwendigkeit von Sicherheitsmaßnahmen.

Gerade die weltweite Vernetzung von Rechnern und die dadurch möglich werdenden neuen Formen der Kooperation und Information machen den Nutzen für die Anwender aus. Deshalb müssen alle Anstrengungen unternommen werden, die Sicherheit der angeschlossenen Systeme zu verbessern und Maßnahmen vorzubereiten, die negative Auswirkungen reduzieren oder ganz verhindern. Die eigentliche Herausforderung dabei ist, mit der existierenden Technologie ein Höchstmaß an Sicherheit zu erreichen, und die angebotenen Dienste auch weiterhin nutzen zu können.

Die Schwachstellen von Computersystemen und -Netzen sind zwar seit vielen Jahren bekannt, aber immer noch ist es nötig, Hinweise und Warnungen, die bereits seit 1994 verbreitet wurden, immer wieder in solch einen Leitfaden aufzunehmen, weil sie bisher nicht durchgängig umgesetzt wurden und Systemanbieter wesentliche Schwachstellen ihrer Betriebssysteme nicht umfassend genug beseitigt haben (z.T. weil es systembedingte Schwachstellen sind).

Erschwerend kommt im Hochschulbereich hinzu, daß an diesen Einrichtungen das Personal (Wissenschaftler und Studenten), welches für die Administration der Rechensysteme häufig eingesetzt wird, sehr schnell wechselt, so daß immer wieder „Neulinge“ für die Aufgaben der Rechner- und Netzwerksicherheit ausgebildet werden müssen.

In den Kapiteln 2-4 wird daher versucht, in kompakter Form die Sicherheitsrisiken der Betriebssysteme Unix und Windows NT, sowie die inherenten Sicherheitsprobleme der derzeitigen Internet-Protokolle darzustellen, wobei jeweils bekannte Sicherheitsangriffe benannt und erprobte Absicherungsmaßnahmen dazu angeboten werden. Eine ausführliche Erläuterung der Fachbegriffe und Hintergrundinformationen finden sich dazu im Anhang (A-D).

Dazu gehören wichtige Maßnahmen, die inzwischen weitgehend etabliert sind, aber dennoch nicht zur Grundausstattung heutiger Betriebssysteme gehören oder noch nicht hinreichend verbreitet sind:

- z.T. mangelhafte Grundsicherung der Betriebssysteme -bei Auslieferung- bezüglich Schutz des Dateisystems vor Angriffen (sichere Rechtevergabe); hier sind die ausführlich erläuterten weitergehenden Schutzmaßnahmen dringend durchzuführen.
- Vermeidung der unverschlüsselten Übermittlung sensibler Daten über das Netzwerk (u.a. Paßworte). Hier gibt es inzwischen keinerlei Grund mehr, alte Programme wie telnet, rsh, ... einzusetzen, weil es dazu sichere Alternativen gibt (*SSH*), die zudem für non-profit Organisationen inzwischen kostenlos sind und sich für kommerzielle Anwendungen ohnehin rechnen.
- Das menschliche Sicherheitsrisiko lauert immer noch im leichtfertigen Umgang mit Paßworten, obwohl es hier viele Hilfsmittel zur Absicherung gibt. Hierzu gehört natürlich auch eine Sicherung der Paßwortdatei durch Verschlüsselung.

Ausführlicher werden dann in Kapitel 5 und 6 die besonders relevanten Themen: Firewalls und Verfahren zur sicheren Übertragung im Netz behandelt.

Kapitel 2

Analyse der Systemsicherheit von UNIX

2.1 Zugangskontrolle über Paßworte

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
schwache Paßworte	Paßwort-Cracker	ja (häufig)	Benutzer-schulung	hoch
			npasswd	mittel
Speicherung der Paßworte	Wörterbuch-Attacke	ja	shadow files	hoch
Klartext-Paßwort-Übertragung im Netzwerk	Sniffer-Programme	ja (häufig)	S/Key	hoch
			ssh	hoch
Weitergabe des Paßwortes durch Benutzer	„Social Engi-neering“	ja (selten)	Benutzer-schulung	hoch

Aufgaben

Das Paßwort-Prinzip basiert auf einem gemeinsamen Geheimnis zwischen einem Benutzer und dem System. Mit diesem Geheimnis identifiziert sich der Benutzer gegenüber dem System und erhält dadurch Zugang. Paßworte sind der erste Schritt, einen Benutzer-Account abzusichern und nicht autorisierte Personen vom Zugriff auf das System abzuhalten.

Funktionsweise

Die Paßworte werden mit einem unumkehrbaren Verfahren verschlüsselt und in einer Datei gespeichert. Bei der Anmeldung am System wird das vom Benutzer eingegebene Paßwort ebenfalls verschlüsselt und mit dem gespeicherten Eintrag verglichen. Nur bei einer Übereinstimmung wird ein Zugang zum System gewährt.

Schwachpunkte

Bei der Verwendung von Paßworten zur Authentisierung unter UNIX-Systemen können — bedingt durch die Art der Implementierung — gewisse Sicherheitsprobleme auftreten. Weitere Probleme ergeben sich durch die Unachtsamkeit bzw. Unwissenheit der Benutzer und durch den Einsatz der Paßwort-Authentisierung in einer Netzwerk-Umgebung.

Grundsätzliche Schwachpunkte

Der Zugang zu einem Computersystem ist an einen *Account* gebunden, der wiederum durch ein Paßwort geschützt sein sollte. Als problematisch erweist sich dabei allerdings, daß auch Accounts ohne Paßwort zugelassen werden und daß viele Hersteller in ihre Betriebssysteme sogenannte *Vendor-Accounts* und Accounts mit bekannten Standard-Paßworten integriert haben, die bei einer Installation des Systems angelegt werden.

Schwachpunkte des Mechanismus:

- Wahl der Paßworte (vgl. C.2.1)
- Geheimhaltung des Paßwortes
- Speicherung der Paßworte in Konfigurationsdateien (vgl. C.2.3)
- Klartext-Übertragung im Netzwerk bei Anmeldung auf entfernte Rechner
- Begrenzung der Paßwort-Länge auf 8 Zeichen (vgl. C.2.5)

Bedrohung:

- unautorisierter Zugang zum System

Angriffsmöglichkeiten:

- Paßwort raten
- Paßwortweitergabe
- Abhören von übertragenen Paßworten

Bekannte Angriffe

Die Paßwortinformationen sind für Angreifer schon immer von großem Interesse gewesen und sind vielfältigen Angriffen ausgesetzt. In den letzten Jahren wurden diverse Programme zur Unterstützung dieser Angriffe verbreitet.

Dazu gehört vor allem das Programm „Crack“[1], das in dem Informations-Bulletin DIB-93:01-Crack[2] beschrieben ist, und dessen Linux-Portierung „John (the Ripper)“. Beide Programme werden bei **Wörterbuch-Attacken** verwendet. Dabei werden die Einträge aus einem Wörterbuch sukzessive verschlüsselt und mit den verschlüsselten

[1] `ftp://ftp.cert.dfn.de/pub/tools/password/Crack/`
[2] `http://www.cert.dfn.de/infoserv/dib/dib-9301.html`

Paßworten verglichen. Weitere bekannte Angriffe sind **Brute-Force-Attacken**, bei denen alle möglichen Kombinationen über ein gegebenes Alphabet erzeugt und probiert werden (vgl. C.2.5).

Nicht zu unterschätzen sind auch die **Sniffer-Programme**, mit denen über das Rechner-Netz übertragene Benutzernamen und Paßworte aufgezeichnet werden (vgl. DSB-94:01[3] und DSB-94:03[4]). Diese werden vor allen nach erfolgreichen Angriffen auf den SuperUser-Account *root* eingesetzt, um Angriffe auf weitere Systeme vorzubereiten und unbefugten Zugang zu anderen Systemen zu erlangen.

Absicherungsmaßnahmen

1. Technische Lösung durch strukturierte Verkabelung (Twisted Pair–Verkabelung mit Switches)

2. Schulung der Benutzer, so daß schwer zu erratene Paßworte gewählt und ein sicherer Umgang (z. B. keine Weitergabe, Notiz, ...) erlernt wird.

3. Einsatz von „*shadow password*"-Dateien zur Speicherung der verschlüsselten Paßworte, auf die nur der SuperUser *root* Zugriff hat (vgl. C.2.3).

4. `npasswd`[5] ist ein Ersatz-Programm für `passwd`, das den Benutzer bei der Wahl eines Paßwortes unterstützt.

5. `cracklib`[6] stellt eine Library zum Überprüfen von Paßworten bereit, die in andere Paßwort-Programme integriert werden kann.

6. Nachträgliche Erkennung schwacher Paßworte mit Hilfe von „Crack"[7], das in dem Informations-Bulletin DIB-93:01-Crack[8] beschrieben ist.

7. Die Verwendung von sogenannten Einmalpaßworten[9] (vgl. C.2.6.3) bietet eine Möglichkeit, abgehörte Paßworte unbrauchbar zu machen.

8. Die Gültigkeitsdauer von Paßworten (vgl. C.2.2) sollte auf einen bestimmten Zeitraum begrenzt werden.

---

[3] `http://www.cert.dfn.de/infoserv/dsb/dsb-9401.html`
[4] `http://www.cert.dfn.de/infoserv/dsb/dsb-9403.html`
[5] `ftp://ftp.cert.dfn.de/pub/tools/password/npasswd/`
[6] `http://www.cert.dfn.de/infoserv/dib/dib-9303.htmlsub`
[7] `ftp://ftp.cert.dfn.de/pub/tools/password/Crack/`
[8] `http://www.cert.dfn.de/infoserv/dib/dib-9301.html`
[9] Eine komplette Beschreibung zu diesem Thema ist in dem Informationsbulletin DIB-94:04-S/Key[10]zu finden.
[10] `http://www.cert.dfn.de/infoserv/dib/dib-9404.html`

9. Periodische Überprüfung der Paßwort-Dateien auf unerlaubte Ergänzungen oder Inkonsistenzen.

10. Regelmäßige Überprüfung auf nicht mehr benutzte Accounts und deren Deaktivierung oder Löschung.

11. Überprüfung der vom Hersteller vorgegebenen Accounts (nicht mehr benötigte Accounts löschen, noch benötigte System-Accounts durch „Paßwort-Locking“[11] absichern)

12. System-Accounts (die nicht zum interaktiven Einloggen vorgesehen sind) durch sog. „Paßwort-Locking“ sperren.

13. Verschlüsselte Übertragung von Paßwort-Informationen über das Rechner-Netz (z. B. mittels `ssh`, siehe Abschnitt A.4).

Einschränkungen

Die Sicherheit der Zugangskontrolle über Paßworte hängt zu einem Großteil von dem Verantwortungsbewußtsein der einzelnen Benutzer ab. Neben der Wahl ihres Paßwortes sind sie auch für dessen Geheimhaltung verantwortlich. Weitere Einschränkungen ergeben sich durch die Speicherung der Paßworte in Konfigurationsdateien, die besonders in vernetzten Systemumgebungen besonders abgesichert werden müssen.

Fazit

Der Vorteil einer Authentisierung über Paßworte ist, daß sich diese leicht realisieren läßt, da jedes UNIX-System über entsprechende Mechanismen verfügt. Werden die oben erwähnten Maßnahmen zur Absicherung eingesetzt, stellt die Authentisierung über Paßworte ein durchaus akzeptables Zugangsverfahren dar. Ist eine Authentisierung über das Netzwerk notwendig, sollten allerdings zusätzlich kryptographische Verfahren zur Verhinderung der Klartext-Übertragung der Paßwort-Informationen eingesetzt werden (siehe *SSH*, *SSL*).

letzte Änderung	02-Mai-2000
überprüft am	25-Mai-2000

[11] d.h. daß ein interaktives Einloggen dadurch unterbunden wird, wenn das intern gespeicherte verschlüsselte Paßwort durch eine Zeichenkette definiert wird, welche nie das Ergebnis eines verschlüsselten Paßwortes sein kann

2.2 Zugriffskontrolle auf Dateien

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Standardrechte	Lesen & Veränderung von Daten	ja (selten)	`umask`	sehr hoch
Zugriffe durch SuperUser (*root*)	Lesen & Veränderung von Daten	ja (häufig)	Kryptographie	mittel
Verzeichnisse auch für *Andere* schreibbar	Veränderung von Dateien	ja	Sticky-Bit	hoch

Aufgaben

Daten und Programme werden häufig auf speziellen Netzwerk-Servern gespeichert und von allen Benutzern gemeinsam genutzt. Dabei sollten diese Daten und Programme nicht ohne Einschränkungen für alle Benutzer zugänglich sein. Einem Benutzer sollte nur der Zugang zu den von ihm benötigten Programmen und Dateien erlaubt werden.

Funktionsweise

Jeder Datei wird ein Besitzer und eine Gruppe mit den jeweiligen Zugriffsrechten zugewiesen, die bestimmen, wer die Dateien lesen und modifizieren darf. Damit lassen sich separate Zugriffsrechte für Benutzer, Gruppen und *Andere* (Welt) realisieren (vgl. C.3.1.1).

Schwachpunkte

Die Sicherheit im System hängt in großem Maß von dem Zusammenspiel zwischen Besitzrechten und Zugriffsrechten und seinen Benutzer-Accounts und Gruppen-Strukturen ab. Damit wird die Administration der Zugriffskontrolle zu einer komplexen Aufgabe, die dementsprechend fehleranfällig ist.

Grundsätzliche Schwachpunkte

Für das korrekte Setzen der Zugriffsrechte auf Dateien ist der jeweilige Datei-Besitzer zuständig. Damit liegt die Vergabe von Zugriffsrechten für Dateien in den jeweiligen Benutzer-Verzeichnissen im Verantwortungsbereich des einzelnen Benutzers.

Schwachpunkte des Mechanismus:

- Der SuperUser *root* kann alle Dateien im System lesen und verändern.
- Oft erhalten Dateien immer noch die Rechte (`rw-rw-rw-`), womit sie sowohl für den Besitzer und der zugeordneten Gruppe als auch für *Andere* les- und veränderbar sind.
- Oft erhalten Verzeichnisse immer noch die Rechte (`rwxrwxrwx`), womit sie ebenfalls sowohl für den Besitzer und der zugeordneten Gruppe als auch für *Andere* les- und veränderbar sind.
- Gewisse Verzeichnisse, z. B. `/tmp`, müssen für alle Benutzer, also auch für *Andere* schreibbar sein.

Bedrohung:

- Unerlaubtes Lesen von Daten.
- Veränderung (einschließlich Löschen) von Programmen und Dateien.

Angriffsmöglichkeiten:

- Unterschieben von gefälschten Programm-Versionen.
- Löschen von Logdaten zur Verschleierung unerlaubter Aktivitäten auf dem System.
- Ausspähen von Daten.

Bekannte Angriffe

Der Zugriff auf den SuperUser-Account *root* eines Unix-Systems wird immer das oberste Ziel eines Angreifers sein. Kann der Angreifer nicht direkt *root* werden, weil z. B. das Paßwort gut gewählt wurde, werden häufig Umwege über das Dateisystem genommen, um den SuperUser-Status zu erlangen.

Ein möglicher Weg ist das Ersetzen von gutartigen Programmen durch gefälschte, evtl. bösartige Versionen (sog. trojanische Pferde oder kurz „Trojaner“). Dies kann dazu führen, daß der SuperUser bei Ausführung eines untergeschobenen Programms dem Angreifer eine Hintertür ins System öffnet.

Nach erfolgreicher Erlangung von SuperUser-Privilegien werden Logdaten gelöscht und Systemprogramme durch weitere Trojaner ersetzt, um weiterhin Zugang zum System zu haben bzw. die Aktivitäten zu verschleiern. Auch werden oft Programme installiert, die weitere Angriffe gegen andere Systeme ermöglichen.

Absicherungsmaßnahmen

Zur Absicherung des Systems bieten sich alle Maßnahmen an, die verhindern, daß nicht-autorisierte Personen Zugriff auf Daten erhalten:

1. Verwendung des `umask`-Befehls, um bessere Default–Rechte einzustellen (vgl. C.3.1.1).
2. Setzen des Sticky-Bits auf Verzeichnisse, die für alle Benutzer schreibbar sein müssen, damit die Schreibrechte eingeschränkt werden.
3. Die Verwendung von *Access Control Lists* (ACLs) ermöglicht eine speziellere Vergabe von Rechten (vgl. hierzu auch C.3.1.1).
4. Setzen von angemessenen Rechten auf Dateien und Verzeichnisse.
5. Setzen von korrekten Suchpfaden.
6. Alle Systemprogramme, die vom SuperUser *root* gestartet werden, sollten auch *root* gehören und **keine** Schreibrechte für die Gruppe und *Andere* haben.
7. Kontrolle der Zugriffe auf das Dateisystem über das Rechner-Netz.
8. Regelmäßige Überprüfung des Systems auf Veränderungen (z. B. mittels *Tripwire*, siehe Abschnitt A.8).

Einschränkungen

Die Kontrolle der Zugriffe auf Benutzerdaten ist nur durch die einzelnen Benutzer möglich. Wählen die Benutzer unsichere Einstellungen, lassen sich ihre Daten kaum gegen unerlaubte Zugriffe schützen. Ein weiteres Problem ist, daß es keinen Schutz vor Veränderung durch den SuperUser *root* gibt (auch mit *Tripwire* lassen sich solche höchstens erkennen). Auch die ACLs lassen sich nur schwer einsetzen, da die einzelnen Hersteller keine einheitliche Unterstützung bieten.

Ebenso ist es dem SuperUser möglich, sämtliche Dateien des Systems zu lesen. Hier kann nur der Einsatz kryptographischer Methoden einen sicheren Schutz gewähren.

Fazit

Das Dateisystem-Konzept für den Schutz von Dateien — separate Zugriffsrechte für Benutzer, Gruppen und *Andere* — ist nur schwer zu administrieren, da das korrekte

Setzen der jeweiligen Zugriffsrechte seitens des Benuzters sehr fehleranfällig ist. Um Daten wirksam gegen unerlaubte Zugriffe abzusichern, sollten zusätzliche Techniken wie Verschlüsselung der Daten eingesetzt werden.

letzte Änderung	27-Mar-2000
überprüft am	25-Mai-2000

2.3 Zugriffskontrolle auf privilegierte Programme

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
SUID Shell-Skripte		ja (selten)	S-Bit entfernen, falls unnötig	hoch
Programme mit gesetztem S-Bit	Ausnutzung eines möglichen Buffer-Overflow	ja (häufig)	Überprüfung der Programme/Quelltexte	hoch
	Versteckte Funktionen, „Shell-Escapes“	ja		

Aufgaben

Im laufenden Rechnerbetrieb müssen die normalen Benutzer hin und wieder Programme benutzen, die Privilegien benötigen, die der betreffende Benutzer nicht hat. Beispielsweise benötigt das `/usr/bin/passwd` Programm SuperUser-Privilegien, damit Eintragungen in die Paßwortdatenbank vorgenommen werden können.

Funktionsweise

Ein privilegiertes Programm ist dadurch gekennzeichnet, daß das S-Bit (für SUID (Set-UserID) bzw. SGID (Set-GroupID)) gesetzt ist. Ein solches SUID- bzw. SGID-Programm wird mit den Rechten des Besitzers bzw. der zugehörigen Gruppe ausgeführt und ermöglicht damit den Zugriff auf ansonsten geschützte Dateien.

Schwachpunkte

Das Vorhandensein von privilegierten Programmen stellt an sich schon ein Sicherheitsrisiko dar, das aber nicht ausgeschlossen werden kann, da es auf einen Design-Fehler von UNIX zurückzuführen ist. Dieser ist leider so grundlegend, daß er nicht ohne weiteres verhindert werden kann.

Grundsätzliche Schwachpunkte

Grundsätzlich stellen alle Programme, die ein S-Bit gesetzt haben, eine potentielle Gefährdung der lokalen Sicherheit dar, da zumeist nicht bekannt ist, ob ein betreffendes

Programm über eine besondere Option oder einen Programmfehler verfügt, der die Ausführung eines Kommandos aus dem Programm heraus ermöglicht.

Schwachpunkte des Mechanismus:

- SUID-*root* Programme laufen mit SuperUser-Privilegien.
- Programme mit gesetztem S-Bit können das Starten weiterer Programme ermöglichen.
- Programme mit unnötig gesetztem S-Bit sind ein vermeidbares Risiko.

Bedrohung:

- Erlangung von erweiterten, schlimmstenfalls SuperUser Privilegien für normale Benutzer.
- Starten von anderen Programmen mit erweiterten Privilegien.

Angriffsmöglichkeiten:

- Ausnutzung eines möglicherweise vorhandenen *Buffer-Overflows*.
- Starten anderer Programme mit den erweiterten Privilegien des ursprünglichen Programmes durch Veränderung der Laufzeitumgebung (`PATH`, `IFS`, etc.).
- Sog. „Shell-Escapes“, also Ausführung von Shell-Kommandos aus dem Programm heraus (mit dessen erweiterten Privilegien).

Bekannte Angriffe

Der Zugriff auf den SuperUser-Account *root* eines Unix-Systems wird immer das oberste Ziel eines Angreifers sein. Kann der Angreifer nicht direkt *root* werden, weil z. B. das Paßwort gut gewählt wurde, werden häufig Umwege über das Dateisystem genommen, um den SuperUser-Status zu erlangen.

Dazu eignen sich insbesondere privilegierte Programme, die das Starten von weiteren Programmen ermöglichen. Ist z. B. für den Kommando-Interpreter `csh` SUID-*root* gesetzt, so haben alle Benutzer, die diesen Kommando-Interpreter starten, alle Rechte, die der SuperUser *root* besitzt. Das bedeutet, daß die in dieser `csh` aufgerufenen Kommandos mit SuperUser-Privilegien ausgeführt werden.

Nicht zu unterschätzen sind auch Angriffe mittels (durch Programm-Fehler bedingte) Buffer-Overflows auf SUID Programme, die zur Ausführung beliebiger Kommandos mit den Privilegien des Programm-Eigentümers führen.

Absicherungsmaßnahmen

1. Überprüfung, ob Programme mit gesetztem S-Bit wirklich diese erweiterten Privilegien benötigen.
2. Überprüfung der Quelltexte aller SUID-*root* Programme.
3. Entfernen der Schreibrechte für Gruppe und *Andere* von allen Programme mit gesetztem S-Bit.
4. Periodische Überprüfung des Systems auf neue oder veränderte Programme mit gesetztem S-Bit.
5. Sicherstellen, daß kein Benutzer mit temporären SuperUser-Privilegien[12] einen SUID-*root* gesetzten Kommando-Interpreter im System hinterläßt.
6. Setzen der `umask` für *root* auf `027`.

Einschränkungen

Es wird in jeder System-Umgebung Programme geben, die mit erweiterten Privilegien arbeiten müssen. Dazu gehört z. B. das `/usr/bin/passwd` Programm, das die SuperUser-Privilegien benötigt, um neue Paßworte in die Datei `/etc/shadow` einzutragen.

Fazit

Nach der Neuinstallation eines Systems oder Programms sollten ausführbare Dateien mit gesetztem S-Bit gefunden und überprüft werden (vgl. Anhang C.3.2). Anhand der eigenen Anforderungen muß entschieden werden, welche privilegierten Programme wirklich benötigt werden. Grundsätzlich sollten möglichst wenig Programme unter erweiterten Privilegien, insbesondere SuperUser-Privilegien, laufen. Als Alternative können durchaus noch spezielle Benutzer-Gruppen eingeführt werden, die dann z. B. das Recht haben, auf ein CD-ROM Laufwerk zuzugreifen, was dem normalen Benutzer nach wie vor verwehrt bleiben sollte.

letzte Änderung	02-Mai-2000
überprüft am	26-Mai-2000

[12] Beliebte Technik bei einem „*root-compromise*“ sich für einen späteren Zeitpunkt eine Hintertür zur erneuten Erlangung von SuperUser-Privilegien einzubauen.

2.4 Zugriffskontrolle auf Ressourcen

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Beschädigung/Löschen von Ressourcen	destruktiver Denial-of-Service	ja (selten)	Schutz der System-Dateien	mittel
			root-Account schützen	mittel
Belegung aller Prozesse	Overload-Attack	ja (sehr selten)	Prozeß-Limits setzen	gering
Belegung des gesamten Plattenplatzes	Disk-full-Attack	ja (sehr selten)	Partitionierung der Platte	mittel
			Vergabe von Plattenquotas	
Überlastung von System-Diensten	Service-Overloading	ja (selten)	—	—

Aufgaben

Eine Kontrolle der Zugriffe auf Ressourcen soll einen ordnungsgemäßen Rechenbetrieb ermöglichen. Da z. B. nur eine begrenzte Anzahl von Prozessen auf einem System laufen kann, muß gewährleistet werden, daß diese maximale Anzahl von Prozessen nicht erreicht wird und grundsätzliche Wartungsarbeiten immer noch durchgeführt werden können.

Funktionsweise

Der Zugriff auf Ressourcen kann sowohl systembezogen durch die Vergabe von Parametern (sog. „*hard limits*“) eingeschränkt werden, als auch benutzerspezifisch über Quotas (sog. „*soft limits*“) geregelt werden.

Schwachpunkte

Als problematisch erweist sich die Tatsache, daß es nach einer Standardinstallation keine Restriktionen der Ressourcenverwendung gibt. Es gibt keine Ressourcen-Beschränkung oder Zuweisung von Ressourcen, sondern jeder Prozess/Benutzer belegt die Ressourcen, die er benötigt.

Grundsätzliche Schwachpunkte

Ressourcen sind technisch gesehen beschränkt und nicht erneuerbar, so daß eine fehlende dynamische Ressourcen-Verwaltung zu Problemen führt.

Schwachpunkte des Mechanismus:

- Die Festlegung der Parameter von Systemkern-Ressourcen erfolgt bei der Übersetzung des Kernels (sog. „*hard limits*“), so daß die nachträglich nicht geändert werden können.
- Benutzer können die an sie vom Administrator vergebenen Beschränkungen (sog. „*soft limits*“) ändern, wie CPU–time, filesize und memorysize.
- Die Beschränkungen werden prozeßbezogen vergeben; es existiert keine Methode, die Beschränkungen einem Prozeß mit seinen sämtlichen Unter-Prozessen zuzuweisen, z.B. gilt ein Zeitlimit für jeden einzelnd erzeugten Prozeß für Kindprozesse neu.

Bedrohung:

- Verhinderung eines ordnungsgemäßen Rechenbetriebes.
- Beschädigung oder Löschen von Ressourcen.
- Überlastung von System-Diensten oder Ressourcen.

Angriffsmöglichkeiten:

- Destruktive Denial-of-Service Angriffe
- Überlastungs-Angriffe (*Overload-Attack*)
- Belegung des gesamten Festplatten-Platzes (*Disk-full-Attack*)
- Überlastung von System-Dienste (*Service Overloading*)
- Überlauf der System-Meldungen (*Message Flooding*)

Bekannte Angriffe

Durch Belegung des gesamten Festplatten-Platzes kann verhindert werden, daß Systemmeldungen, z. B. über erfolgte Angriffe auf das System, protokolliert werden können. Ein weiterer möglicher Angriff besteht darin, die Systemmeldungen durch Falschmeldungen zu ersetzen, die die eigentlichen Aktivitäten verschleiern.

Absicherungsmaßnahmen

1. Partitionierung der Festplatte in unterschiedliche Bereiche, für entsprechende Daten und Zugriffe: Benutzer-, Log-, Mail- und System-Daten (näheres hierzu siehe C.1.1).
2. Schutz der System-Dateien vor unerlaubtem Zugriff.
3. Priorisierung der CPU Prozesse.
4. Beschränkungen für zu startenden Prozesse und Unter-Prozesse definieren.
5. Maximale Anzahl von Dateien/Prozessen für Benutzer definieren.
6. Vergabe von Quotas als Begrenzung des maximal nutzbaren Speicherplatzes.

Einschränkungen

Aktuell verfügbare UNIX Ressourcen-Limitierungen sind vom administrativen Standpunkt her nutzlos, weil sie nicht vom System-Administrator zu Systemlaufzeiten geändert werden können. Eine weitere Einschränkung ergibt sich dadurch, daß es gerade im Bereich der Vergabe von System-Ressourcen keine festen Regeln gibt, nach denen die Ressourcen vergeben werden können. Abhängig von den eingesetzten Systemen und zu erfüllenden Aufgaben ergeben sich deutlich differenzierte Anforderungen an die Ressourcen-Vergabe, die nur individuell gelöst werden können.

Fazit

Eine generelle Lösung zur Beschränkung der Ressourcen-Zugriffe wird es nicht geben können, da diese grundsätzlich systemabhängig sind. Gewisse Absicherungsmaßnahmen lassen sich jedoch pauschal einsetzen, z. B. das Partitionieren der Festplatte in unterschiedliche Bereiche, um „*Disk-full-Attacks*“ zu verhindern.

letzte Änderung	02-Mai-2000
überprüft am	26-Mai-2000

2.5 Syslog

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Speicherung der Log-Daten	Löschen der Dateien	ja (häufig)	mehrfache Speicherung	hoch
fehlende Authentizität	Erzeugung von Falschmeldungen	unbekannt	Paketfilter auf Port 514/UDP	mittel

Aufgaben

Der *syslogd* stellt einen Mechanismus zur Verfügung, mit dem von jedem Kommando erzeugte Fehlermeldungen und Informationen auf die System-Konsole oder in Log-Dateien geschrieben werden können. Es stellt ein zentrales Protokoll-Konzept zur Verfügung, mit dem Informationen zur Protokollierung verschickt, kategorisiert und weiterverarbeitet werden können.

Funktionsweise

Programme schicken Textnachrichten an den *syslogd*, der anhand seiner Konfigurationsdatei (`syslog.conf`) und den vorgebbaren Merkmalen der Nachricht (Klasse und Priorität) über Speicherung und/oder Weiterleitung der Nachricht entscheidet. Damit können die Protokoll-Informationen zentral gesammelt, verwaltet und vor der Speicherung gefiltert werden.

Schwachpunkte

Als problematisch erweist sich die Tatsache, daß bei einigen UNIX-Systemen die Log-Datei, in die der *syslogd* schreiben soll, existieren muß, damit die Konfigurationseinstellungen auch berücksichtigt werden. Dies bedeutet, daß zunächst eine leere Log-Datei angelegt werden muß, bevor der *syslogd* gestartet werden kann. Ein weiterer zu berücksichtigender Stolperstein ist die Konfigurationsdatei `syslog.conf`, in der zur Erzeugung der Leerräume auf einigen Systemen zwischen den einzelnen Spalten Tabulator-Abstände (*tabs*) und keine Leerzeichen (*spaces*) verwendet werden müssen.

Grundsätzliche Schwachpunkte

Ein weiteres Problem ist, daß die zu protokollierenden Log-Meldungen grundsätzlich an die Log-Datei angehängt werden und diese dadurch ständig größer wird. Werden

die Log-Dateien nicht regelmäßig gekürzt, ergeben sich notgedrungen auch Probleme mit Festplattenplatz.

Schwachpunkte des Mechanismus:

- lokale Speicherung der Log-Dateien
- ständiges Wachsen der Log-Dateien

Bedrohung:

- Verschleierung von erfolgtem Angriff

Angriffsmöglichkeiten:

- Veränderung/Löschung der Log-Daten
- Erzeugung von Falschmeldungen

Bekannte Angriffe

Wie im CERT Advisory[13] CA-95:13 näher erläutert wird, kann mit der *syslog(3)* Subroutine ein interner Buffer-Overflow verursacht werden, der anschließend das Ausführen beliebiger Programme ermöglicht und letztendlich zur Erlangung von SuperUser-Rechten führt.

Absicherungsmaßnahmen

1. Speicherung der Log-Daten auf weiteren Rechnern
2. regelmäßiges Sichern der Log-Daten
3. regelmäßige Überprüfung und Auswertung der Log-Daten
4. regelmäßige Verkleinerung der Log-Dateien
5. Zugriff auf Port 514/UDP beschränken, um Angriffe auf den *syslogd* (z. B. durch Erzeugung von Falschmeldungen) zu vermeiden
6. Anlegen einer eigenen Partition für die Log-Daten, damit Probleme anderer Programme (z.B. unter `/var/spool/lpd` liegende Druckaufträge oder das absichtliche Füllen der Festplatte mit Daten (vgl. 2.4)) nicht eine Protokollierung verhindern.

[13] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories`

Einschränkungen

Da es keinerlei Authentisierung gibt, kann im Prinzip jeder Benutzer (lokal oder über das Rechner-Netz) Log-Meldungen fälschen. Der Einsatz eines Paketfilters am UDP-Port verhindert zwar die Erzeugung von gefälschten Meldungen von außerhalb des eigenen Netzes, er läßt sich allerdings nicht gegen die Erzeugung von Log-Floods aus dem eigenen Netz einsetzen.

Fazit

Die Speicherung der Log-Daten auf weiteren Rechnern ist eine wichtige Maßnahme zur Absicherung, hat aber im Falle einer Netzüberlastung auch gewisse Probleme. Da die Zustellung der Log-Meldungen paketbasiert erfolgt, wird dementsprechend auch keine Zustellung der Log-Meldungen an den entfernten Rechner gewährleistet.

letzte Änderung	02-Mai-2000
überprüft am	19-Mai-2000

2.6 Auditing

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Speicherung der Daten	Veränderung der Daten	ja	externe Speicherung, regelmäßige Datensicherung, entsprechende Festplattenpartitionierung	hoch
Dienst- & Benutzersitzungen ohne Audit-Anmeldung	keine Protokollierung von Aktivitäten	ja	nur „auditfähige“ Programme einsetzen	hoch
Dienst- und Benutzersitzungen vor dem Audit-System gestartet	keine Protokollierung von Aktivitäten	ja	Reihenfolge der Systeminitialisierung beachten	hoch

Aufgaben

Auditing wird zur Protokollierung von System-Aktivitäten verwendet, kann aber auch zur Überwachung eines benutzerbezogenen Systemverhaltens eingesetzt werden.

Funktionsweise

Bei der Initialisierung von einer Benutzer- oder Systemdienst-Sitzung (eng. „*user session*“, „*service session*“) wird eine Audit-ID (AID) generiert und der Sitzung zugewiesen. Sämtliche Aktivitäten, die während der Benutzer-Sitzung getätigt werden, werden mit der AID behaftet, um Überwachungsdaten in einen sogenannten *Audit-Trail* zusammen fassen zu können. Hier ist zu beachten, daß der *Audit-Trail* nicht durch einen `chuid`-Systembefehl (Änderung der Benutzerkennung) unterbrochen wird, da die Aktivitäten der neu angenommenen Benutzerkennung auch mit der gleichen AID behaftet werden.

Das Audit-System funktioniert, indem die Ausführungen von *System-Calls* festgehalten werden. Dabei können unterschiedliche Klassen von Ereignissen berücksichtigt werden. Außerdem kann für jedes Ereignis spezifiziert werden, was protokolliert werden soll. Zusätzlich ist für die einzelnen Benutzer eine differenzierte Protokollierung möglich, um so die Menge der zu speichernden Daten minimieren zu können.

Das hier beschriebene Auditingsystem ist ein Bestandteil des Betriebssystems und darf nicht mit sog. Netzwerkauditingpaketen[14] (eng. „*network monitoring software*“) verwechselt werden.

Schwachpunkte

Da Auditing sehr stark von dem jeweiligen UNIX-System abhängig ist, lassen sich keine allgemeinen Regeln zur Administration aufstellen.

Grundsätzliche Schwachpunkte

Auditing-Systeme zeigen zwei grundsätzliche Schwachpunkte: Erstens, falls Systemdienste vor dem Audit-Subsystem gestartet werden, kann es passieren, daß sie nicht nachträglich gemeldet werden und somit keine AID zugewiesen bekommen. Ist dies der Fall, so werden keine Aktivitäten des Systemdienstes protokolliert. Zweitens werden u. U., falls Programme eingesetzt werden, die eine Systemanmeldung ermöglichen[15], aber die nötige Audit-Systemanmeldung nicht durchführen, überhaupt keine Session-Aktivitäten vom Auditsystem protokolliert.

Schwachpunkte des Mechanismus:

- lokale Speicherung der Audit-Daten

Bedrohung:

- Bildung von Benutzerprofilen - möglicher Mißbrauch von (personenbezogenen) Daten

Angriffsmöglichkeiten:

- Veränderung/Löschen der Audit-Daten
- Verwendung der Daten aus den Auditdateien für weitere Angriffe [16]
- DoS-Angriffe durch gezielte Generation von Auditdaten (Plattenüberlauf)

Bekannte Angriffe

Zur Zeit sind keine Angriffe auf das Auditing bekannt.

[14] z. B. `argus`
[15] z. B. das `login`-Programm, der `sshd`- oder `ftpd`-Dæmon
[16] z. B. um Informationen bzgl. der Arbeitszeiten zu erhalten

Absicherungsmaßnahmen

1. Speicherung der Audit-Daten auf weiteren Rechnern
2. regelmäßige Sicherung der Audit-Daten
3. regelmäßige Überprüfung und Auswertung der Audit-Daten
4. Anlegen einer eigenen Partition für die Audit-Daten, damit andere Programme die Abspeicherung der Daten nicht verhindern.

Fazit

Auditdaten bieten eine Möglichkeit, Informationen über das Verhalten eines Angreifers zu erhalten. Diese Daten können auch zur Bildung von Benutzerprofilen herangezogen werden und sind somit aus der Sicht des Datenschutzes bedenklich.

letzte Änderung	21-Mär-2000
überprüft am	29-Mai-2000

2.7 Prozeß-Accounting

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Speicherung der Daten	Veränderung der Daten	ja	Externe Speicherung, regelmäßige Datensicherung, entsprechende Festplattenpartitionierung	hoch

Aufgaben

Accounting wurde entwickelt und wird hauptsächlich benutzt, um Kostenabrechnungen entsprechend der System-Benutzung durchführen zu können. Es kann aber auch zur benutzerbezogenen Protokollierung von Programm-Nutzungen eingesetzt werden.

Funktionsweise

Bei der Zuteilung von Ressourcen durch das Betriebssystem wird aufgezeichnet, wieviel und welche Ressourcen von welchen Prozessen verbraucht werden. Es wird auch festgehalten, unter welcher Benutzerkennung der Prozeß läuft.

Das Betriebssystem stellt andere Mechanismen bereit, die verwandte Funktionalität aufweisen[17], aber nicht mit Prozeß-Accounting verwechselt werden dürfen.

Schwachpunkte

Da Accounting von der jeweiligen UNIX-Variante abhängig ist, lassen sich keine allgemeinen Regeln zur Administration aufstellen.

Grundsätzliche Schwachpunkte

Accounting ist nur bedingt geeignet, um den Rechnerbetrieb zu überwachen. Nur die Namen der ausgeführten Programme, bzw. Kommandos werden aufgezeichnet – weder Argumente noch die Laufzeit-Umgebung (das Verzeichnis aus dem das Programm

[17] Das `last`-Kommando liefert z. B. Benutzer-Accounting-Information

gestartet wurde, usw.) werden protokolliert. Die Auswertung der Daten ist problematisch, da die Annahme von anderen Benutzerkennungen[18] einen Bruch in der Überwachungsinformation verursacht, die eine vollständige, automatische Bearbeitung der Daten fast unmöglich macht.

Schwachpunkte des Mechanismus:

- lokale Speicherung der Accounting-Daten

Bedrohung:

- Bildung von Benutzerprofilen - möglicher Mißbrauch von (personenbezogenen) Daten

Angriffsmöglichkeiten:

- Veränderung/Löschen der Accounting-Daten
- DoS-Angriffe durch gezielte Generation von Accountingdaten (Plattenüberlauf)
- Gezielte Generation von überflüssigen Daten, um Aktivitäten zu verschleiern

Bekannte Angriffe

Zur Zeit sind mit Ausnahme des Löschens von Log-Dateien keine weiteren Angriffe auf das Prozeß-Accounting bekannt.

Absicherungsmaßnahmen

1. Speicherung der Accounting-Daten auf weiteren Rechnern
2. regelmäßige Sicherung der Accounting-Daten
3. regelmäßige Überprüfung und Auswertung der Accounting-Daten
4. Anlegen einer eigenen Partition für die Accounting-Daten, damit andere Programme die Abspeicherung der Daten nicht verhindern.

[18] z. B. durch den Aufruf von `su`

Fazit

Accountingdaten bieten eine Möglichkeit, Informationen über das Verhalten eines Angreifers zu erhalten. Da das System nicht für diesen Zweck entwickelt wurde, sind die Daten nur bedingt für Überwachungszwecke geeignet. Die Daten können auch zur Bildung von Benutzerprofilen herangezogen werden und sind somit aus der Sicht des Datenschutzes bedenklich.

letzte Änderung	21-Mär-2000
überprüft am	29-Mai-2000

Kapitel 3

Analyse der Systemsicherheit von Windows NT 4

3.1 Zugangskontrolle über Paßworte

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
schwache Paßworte	Paßwort-Cracker	ja (häufig)	Benutzer-schulung	hoch
			passfilt.dll	mittel
Speicherung der Paßworte	Wörterbuch-Attacke	ja	Absicherung der Registry	hoch
Übertragung im Netzwerk	Sniffer	ja	Syskey	hoch

Aufgaben

Paßworte zur Authentisierung sind ein weit verbreitetes Verfahren und ermöglichen den ersten Zugang zu Systemen. Ein nur dem Benutzer bekanntes Paßwort ermöglicht die Überprüfung einer angegebenen Identität. Dies stellt den ersten Schritt dar, einen Benutzer-Account abzusichern und nicht autorisierte Personen vom Zugriff auf das System abzuhalten.

Funktionsweise

Die Paßworte werden mit einem Verfahren verschlüsselt, das nicht rückgängig gemacht werden kann. Das so erhaltene verschlüsselte Paßwort wird in einer Datei gespeichert. Bei der Anmeldung am System, wird das vom Benutzer eingegebene Paßwort ebenfalls verschlüsselt und mit dem gespeicherten Eintrag verglichen. Nur bei einer Übereinstimmung wird ein Zugang zum System gewährt. Die Paßworte werden in der Standardinstallation in zwei Varianten gespeichert (vgl. D.2.3).

Schwachpunkte

Bei der Verwendung von Paßworten zur Authentisierung unter Windows NT-Systemen können — bedingt durch die Art der Implementierung — gewisse Sicherheitsprobleme auftreten. Weitere Probleme ergeben sich durch die Unachtsamkeit bzw. Unwissenheit der Benutzer und durch den Einsatz von Paßwort-Authentisierung in Netzwerk-Umgebungen.

Grundsätzliche Schwachpunkte

Der Zugang zu einem Computersystem ist an einen Account gebunden, der wiederum durch ein Paßwort geschützt sein sollte. Als problematisch erweist sich dabei allerdings, daß auch Accounts ohne Paßwort zugelassen werden.

Schwachpunkte des Mechanismus:

- Wahl der Paßworte (vgl. D.2.1)
- Geheimhaltung des Paßwortes
- Speicherung der Paßworte in Konfigurationsdateien (vgl. D.2.3)
- Klartext-Übertragung im Netzwerk bei Anmeldung auf Remote-Rechner
- Begrenzung der Paßwort-Länge auf 14 Zeichen (vgl. D.2.5)
- Verwendung von schwachen Verfahren zur Paßwortverschlüsselung (NT-LM) (vgl. D.2.4)

Bedrohung:

- unautorisierter Zugang zum System

Angriffsmöglichkeiten:

- Paßwort raten
- Paßwortweitergabe
- Abhören von übertragenen Paßworten

Bekannte Angriffe

Die Paßwortinformationen sind für unautorisierte Personen schon immer von großem Interesse gewesen und sind vielfältigen Angriffen ausgesetzt. In den letzten Jahren wurden diverse Programme zur Unterstützung der Angreifer verbreitet.

Dazu gehört vor allem das Programm „L0phtcrack“[1], das bei **Wörterbuch-Attacken** verwendet wird. Dabei werden die Einträge aus einem Wörterbuch sukzessive verschlüsselt und mit den verschlüsselten Paßworten verglichen. Weitere mit L0phtcrack durchführbare Angriffe sind **Brute-Force-Attacken**, bei denen alle möglichen Kombinationen über einem gegebenen Alphabet erzeugt und probiert werden (vgl. C.2.5).

Nicht zu unterschätzen sind auch die **Sniffer-Programme**, mit denen Benutzernamen und Paßworte aufgezeichnet werden (vgl. DSB-94:01[2] und DSB-94:03[3]) L0phtcrack bietet auch diese Funktionalität.

[1] `ftp://ftp.cert.dfn.de/pub/tools/password/l0pht/`
[2] `http://www.cert.dfn.de/infoserv/dsb/dsb-9401.html`
[3] `http://www.cert.dfn.de/infoserv/dsb/dsb-9403.html`

Absicherungsmaßnahmen

1. Schulung der Benutzer, so daß sichere, d.h schwer erratbare Paßworte gewählt werden und ein sicherer Umgang (z.B. keine Weitergabe, Niederschreiben, ...) erlernt wird.
2. „passfilt.dll“ (enthalten ab Service Pack 3[4]) ist eine Bibliothek, die Regeln definiert, nach denen Paßworte gebildet werden müssen.
3. Nachträgliche Erkennung schwacher Paßworte mit Hilfe von „L0phtcrack“[5].
4. Die Gültigkeitsdauer von Paßworten (vgl. D.2.2) sollte auf einen bestimmten Zeitraum begrenzt werden.
5. Nach Möglichkeit Verwendung von NT-LM-Paßworten zur Authentisierung deaktivieren (vgl.D.2.4).
6. Regelmäßige Überprüfung auf nicht mehr benutzte Accounts.
7. Überprüfung des Systems auf Default-Kennungen (Nicht mehr benötigte Kennungen löschen).
8. Sperrung eines Accounts nach einer bestimmten Anzahl fehlgeschlagener Anmeldungen.
9. Verschlüsselung der Passwortdatei mit „Syskey“ (enthalten ab Service Pack 3[4]).

Einschränkungen

Die Sicherheit der Zugangskontrolle über Paßworte hängt zu einem Großteil von dem Verantwortungsbewußtsein der einzelnen Benutzer ab. Neben der Wahl ihres Paßwortes sind sie auch für dessen Weitergabe verantwortlich. Weitere Einschränkungen ergeben sich durch die Speicherung der Paßworte in Konfigurationsdateien, die besonders in vernetzten Systemumgebungen besonders abgesichert werden müssen.

Fazit

Der Vorteil einer Authentisierung über Paßworte ist, daß sich diese leicht realisieren läßt, da jedes Windows NT-System über entsprechende Mechanismen verfügt. Werden die oben erwähnten Maßnahmen zur Absicherung eingesetzt, stellt die Authentisierung über Paßworte ein durchaus akzeptables Zugangsverfahren dar. Ist eine Au-

[4] `ftp://ftp.cert.dfn.de/pub/vendor/microsoft/winnt/ger/nt40/SP3/`
[5] `ftp://ftp.cert.dfn.de/pub/tools/password/l0pht/`

thentisierung über das Netzwerk notwendig, sollten allerdings noch zusätzlich kryptographische Verfahren zum Einsatz kommen, die den Schwächen bei der Klartext-Übertragung im Netzwerk begegnen.

letzte Änderung	26-Mai-2000
überprüft am	26-Mai-2000

3.2 Zugriffskontrolle auf Dateien

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
schwache Default-Zugriffsrechte	Veränderung von Daten	ja (häufig	Manuelle Anpassung	hoch
Zugriffe durch *Administrator*	Modifikation von Daten	ja (häufig)	Kryptographie	mittel

Aufgaben

Daten und Programme werden häufig auf speziellen Netzwerk-Servern gespeichert und von allen Benutzern gemeinsam genutzt. Diese Daten und Programme sollten nicht ohne Einschränkung für alle Benutzer zugänglich sein. Dem Benutzer sollte nur der Zugang zu den von ihm benötigten Programmen und Dateien erlaubt werden.

Funktionsweise

Während der Anmeldung am System wird jedem Benutzer ein Sicherheits-Token zugeordnet. Dieses Token enthält Informationen über den Benutzer und über die Gruppen, in denen der Benutzer Mitglied ist.
Wird Zugriff aus eine Datei gewünscht, wird durch die ACL der Datei überprüft, ob der gewünschte Zugriff erlaubt ist (vgl. D.3).

Schwachpunkte

Die Sicherheit im System hängt in großem Maß von dem Zusammenspiel zwischen Besitzrechten, Zugriffsrechten und Gruppenzugehörigkeit ab.

Die Vergabe von Zugriffsrechten auf Dateien in den Benutzerverzeichnissen liegt auch im Verantwortungsbereich des einzelnen Benutzers, so daß sich diese Zugriffsrechte nur schwer administrieren lassen.

Grundsätzliche Schwachpunkte

Ein normaler Benutzer kann seine Daten, d.h. Daten, deren Besitzrecht ihm gehört, vor unbefugtem Zugriff durch das Setzen von Zugriffsrechten schützen, die nur ihm Zugriff gewähren. Dieser Schutzmechanismus kann jedoch vom *Administrator* übergangen werden, da dieser in der Lage ist, die Besitzrechte von allen Dateien zu übernehmen und anschließend die Rechte so zu verändern, daß er Zugriff auf diese Dateien

erhält. Da der *Administrator* die Besitzrechte nicht wieder an den Benutzer zurückgeben kann, kann die Einsichtnahme nicht verdeckt erfolgen.
Außerdem werden bei der Installation eines Windows NT Systems die Rechte auf viele Verzeichnisse mit Systemdaten derart gesetzt, daß normale Benutzer Lese- und Schreibrechte auf diese Verzeichnisse haben.

Schwachpunkte des Mechanismus:

- Festplatten erhalten standardmäßig das Recht *Everyone Full Control*. Damit haben alle Benutzer Zugriff auf die Daten im Hauptverzeichnis.
- Die Superuser (*Mitglieder der Administrator-Gruppe*) können alle Dateien im System lesen und modifizieren
- Gewisse Verzeichnisse müssen für alle Benutzer schreibbar sein (dazu gehört je nach installierter Applikationssoftware auch das Windows NT Verzeichnis)

Bedrohung:

- Unerlaubte Modifikation von Daten
- Zugriff auf Sicherheitsinformationen wie Paßwortdaten
- Ersetzen und Löschen von Programmen und Dateien

Angriffsmöglichkeiten:

- Lesen der verschlüsselten Paßwortinformationen zur Benutzung mit einem Paßwort-Cracker (siehe D.2.4)
- Löschen von Log-Dateien zur Vertuschung eines System-Einbruchs

Bekannte Angriffe

Angriffe auf den Administrator sind beliebt, da dieser Account nach einer Standard-Installation nicht nach *n* fehlerhaften Logins gesperrt wird. D.h. ein Angreifer kann beliebig viele Login-Versuche unternehmen. Durch Modifikation der Paßwortdatenbank können neue Benutzer dem System zugefügt oder Paßworte geändert werden.

Nach einem erfolgreichen Einbruch und Erlangung von *Administrator*-Privilegien können Logdaten gelöscht und Systemprogramme durch trojanische Pferde ersetzt werden, um weiterhin Zugang zum System zu haben bzw. Aktivitäten zu vertuschen.

Absicherungsmaßnahmen

1. Die Rechte in den verschiedenen Verzeichnissen sollten entsprechend D.8 gesetzt werden.
   Achtung: Nach der Installation eines Service Pack müssen die Änderungen ggf. erneut durchgeführt werden.

2. Verwendung des `passpro` Programms aus dem MS-Ressource-Kit[6]. Durch dieses Programm wird ein Administrator-Account bzgl. Zugriffen über das Netzwerk gesperrt. D.h. nach *n* fehlerhaften Login-Versuchen muß sich der Administrator einmal am Domänen-Kontroller direkt einloggen, damit sein Account wieder freigeschaltet wird.

Einschränkungen

keine

Fazit

Das Dateisystem-Konzept der Sicherheit von Dateien — separate Zugriffsrechte für Benutzer und Gruppen — ist zwar relativ einfach zu verstehen, aber schwer zu administrieren, da das korrekte Setzen aller einzelnen Zugriffsrechte (File Permissions) sehr komplex ist. Um Daten wirksam gegen unerlaubte Zugriffe abzusichern, sollten zusätzliche Techniken wie Verschlüsselung der Daten eingesetzt werden. Um einen Überblick über die gesetzten Rechte in einem Dateisystem zu erhalten, bietet Windows NT keine geeigneten Tools, so daß auf Drittanbieter, wie z.B. DUMP-ACL (vgl. B.4) zurückgegriffen werden muß.
Mit dem Ende November 1999 veröffentlichen Tool *Microsoft Management Console* (vgl. Kapitel B.1) lassen sich sog. Templates mit Sicherheitsdefinitionen für Domänen definieren und anwenden. Damit können die Rechner einer Domäne mit wenig Aufwand auf ein einheitliches Sicherheitsniveau gebracht werden.

letzte Änderung	26-Mai-2000
überprüft am	26-Mai-2000

[6]Das MS-Ressource-Kit ist als Beilage in der **Technische Referenz Windows NT 4.0 Server** zu finden.

3.3 Zugriffskontrolle auf Privilegierte Programme

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Schedule-Dienst	Programme mit besonderen Rechten starten	nein		niedrig
IE 5 Task-Schedule-Dienst	Programme mit besonderen Rechten starten	ja	Update auf IE 5.01	hoch

Aufgaben

Bestimmte Dienste müssen in Bereiche schreiben dürfen, die für die Benutzer gesperrt sind (z.B. Mail, Schedule). Um dieses zu gewährleisten, werden diese Programme mit besonderen Rechten ausgestattet und unter besonderen Benutzerkennungen gestartet.

Funktionsweise

Dienste werden auf einem Server installiert und dort automatisch beim Hochfahren gestartet. Es läßt sich zu jedem Dienst definieren, unter welcher Benutzerkennung dieser Dienst gestartet werden soll.

Schwachpunkte

Die Aktionen, die Dienste ausführen, finden unter den Berechtigungen des Benutzers statt, unter dem ein Dienst gestartet wurde. D.h., daß Programme, die z.B. vom Schedule-Service gestartet werden, mit System-Privilegien ausgeführt werden.

Bedrohung:

- Erlangung erweiterter Benutzerrechte

Angriffsmöglichkeiten:

- Einloggen auf der Konsole und Dienst starten

Bekannte Angriffe

Bei der Installation des Internet Explorers 5 kann optional der NT Scheduler-Dienst durch den „Task Scheduler“ ersetzt werden. Durch eine Schwachstelle im „Task Scheduler“ ist es einem normalen Benutzer möglich administrative Rechte zu erlangen. Zu dieser Schwachstelle hat Microsoft das Sicherheits-Advisory MS99-051[7] herausgegeben.

Absicherungsmaßnahmen

1. Rechte, Dienste zu starten, durch entsprechende ACL-Einstellungen so einschränken, daß nur Systemverwaltungspersonal dazu befugt ist.
2. Nur Systemverwaltungspersonal das Recht geben, sich auf Servern einzuloggen.
3. Nur benötigte Dienste starten.
4. Wenn der Schedule-Service benötigt wird, unter einer Benutzerkennung starten, die nur mit den erforderlichen Rechten ausgestattet ist.

Fazit

Privilegierte Dienste werden eher selten angegriffen. Die Absicherung über ACL ist einfach.

letzte Änderung	26-Mai-2000
überprüft am	26-Mai-2000

[7] `http://www.microsoft.com/Security/Bulletins/ms99-051.asp`

3.4 Zugriffskontrolle auf Ressourcen

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Belegung des Prozessorzeit	Denial-Of-Service	ja	Hot-Fixe	hoch
Belegung des Speicherplatzes	Disk-Full-Attack	ja	Partitionierung	mittel
Überlastung von System-Services	Service-Overloading	ja	-	mittel

Aufgaben

Ressourcen werden unter Windows NT etwas anders definiert als unter UNIX. Windows NT Systeme sind keine Mehrbenutzersysteme in dem Sinne, daß mehrere Benutzer gleichzeitig auf einem Rechner Programme ausführen können. Windows NT–Server Systeme liefern Dienste an Dritte. Durch Belegung aller Prozessorzeit durch einen Angriff wird ein Server an der Ausführung seiner Dienste gehindert (File-Server, Proxy-Server, DNS-Server).

Funktionsweise

Durch Ausnutzung von fehlerhaften Implementationen wird die CPU komplett ausgelastet. Weitere Dienste können dann nicht mehr geleistet werden.

Schwachpunkte

Als problematisch erweist sich die Tatsache, daß es per Default keine Restriktionen auf Ressourcen-Allozierung (Aufteilung) gibt. Es gibt keine Ressourcen-Beschränkung oder Zuweisung von Ressourcen, sondern jeder Prozeß (Benutzer) nimmt sich, was er benötigt. Da die Ressourcen technisch gesehen beschränkt und nicht erneuerbar sind, führt dies unweigerlich zu Problemen.

Bedrohung:

- Verhinderung eines ordnungsgemäßen Rechenbetriebes
- Beschädigung oder Löschen von Ressourcen
- Überlastung von System-Services oder Ressourcen

Angriffsmöglichkeiten:

- Denial-Of-Service
- Überlastungs-Angriffe (Overload-Attack)
- Belegung des gesamten Festplatten-Platzes (Disk-Full-Attack)
- Überlastung von System-Services (Service Overloading)

Bekannte Angriffe

Durch Belegung des gesamten Festplatten-Platzes kann verhindert werden, daß Systemmeldungen, z.B. über erfolgte Angriffe auf das System, geschrieben werden können. Eine weiterer möglicher Angriff besteht darin, die Systemmeldungen durch Falschmeldungen zu ersetzen, die die eigentlichen Aktivitäten verschleiern.

Absicherungsmaßnahmen

1. Schutz der System-Dateien vor unerlaubtem Zugriff
2. Entsprechende Partitionierung der Festplatte

Einschränkungen

Plattenquotas werden standardmäßig von Windows NT 4.0 nicht unterstützt.

Fazit

Denial-Of-Service Angriffe sind die häufigste Angriffsart für Windows NT Rechner. Nach Möglichkeit sollten entsprechende Hot-Fixes eingespielt werden, wenn der Verdacht besteht, daß ein System einem Denial-Of-Service Angriff ausgesetzt ist.

letzte Änderung	26-Mai-2000
überprüft am	26-Mai-2000

3.5 Monitoring

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Standardmäßig deaktiviert			aktivieren	mittel
Logdaten öffentlich zugänglich	Lesen der Logdaten	unbekannt	Logdaten nur lokal lesbar machen	mittel

Aufgaben

Die alleinige Installation von Schutzmechanismen ist nicht ausreichend, da diese in der Regel Ergebnisse – in Form von Meldungen – produzieren, die aufgezeichnet und vor allem auch ausgewertet werden müssen. Zum einen, um sicherzugehen, daß die Mechanismen korrekt funktionieren, zum anderen, um auf die Aktionen, die nicht den eigenen Sicherheitspolitiken entsprechen, richtig reagieren zu können.

Monitoring ist ein wichtiger Vorgang, um die Sicherheit, Verfügbarkeit und Integrität von Systemen zu gewährleisten. Monitoring setzt sich im allgemeinen aus den Teilbereichen Logging und Auditing zusammen.
Unter Audit-Daten werden die Daten verstanden, die direkt vom Kernel bereitgestellt werden. Logging-Daten werden dagegen von Anwendungsprogrammen erzeugt.
Eine klare Trennung ist unter Windows NT nicht möglich. **Achtung:** die Windows NT Terminologie für *Monitoring* ist *Auditing*.

Funktionsweise

Beim Auditing werden detaillierte Auskünfte über die gestarteten Prozesse und deren Aktivitäten protokolliert.
Die von den Schutzmechanismen produzierten Meldungen werden zumeist in Dateien gespeichert, den sog. Log-Dateien. Die in diesen Dateien gesicherten Informationen sind wichtig, um ein System zu sichern, da sie eine Art Vorgangsgeschichte beschreiben (z.B. **welche** Programme **wann** von **wem** gestartet wurden). Durch diese Log-Dateien ist es möglich, bereits geschehene Ereignisse zu verfolgen bzw. nachzuvollziehen und die Ursache für das Fehlverhalten zu finden.

Schwachpunkte

Grundsätzliche Schwachpunkte

Audit ist unter Windows NT generell deaktiviert. Um Audit zu aktivieren, müssen an zwei Stellen die entsprechenden Einstellungen vorgenommen werden (vgl. D.5).
Es werden drei Kategorien von Audit-Daten bereitgestellt:

- System-Audit
- Security-Audit
- Application-Audit.

Benutzerprogramme und Programme von Drittanbietern können nur in die Application-Audit-Kategorie schreiben. Weitere Kategorien können nicht erstellt werden.
Da die Log-Dateien außerdem meist lokal auf der Maschine gespeichert werden, kann es für einen erfolgreichen Angreifer möglich sein, die erzeugten Daten zu löschen, bevor sie ausgewertet werden konnten.

Schwachpunkte des Mechanismus:

- Auditdaten werden nur lokal gespeichert.
- Auditdaten sind über das Netzwerk zugreifbar.
- Es gibt nur drei verschiedene Auditkategorien.
- Ist der für die Audit-Daten reservierte Platz verbraucht, werden alte Audit-Daten gelöscht oder Benutzer können sich nicht mehr an dem Rechner anmelden.

Angriffsmöglichkeiten:

- Durch Erzeugung einer Vielzahl von Auditereignissen können alte Auditdaten überschrieben werden.

Bekannte Angriffe

unbekannt

Absicherungsmaßnahmen

1. Setzen der entsprechenden Registry-Werte (vgl. D.6)
2. Verwendung von Drittanbieter-Tools zur zentralen Speicherung der Audit-Daten (vgl. B.2)

Einschränkungen

keine

Fazit

Windows NT Systeme bieten von der Art der Log-Daten nur wenig Differenzierungsmöglichkeiten, so daß bei konsequenter Verwendung von Audit schnell eine unüberschaubare Menge von Audit-Daten entsteht.
Dennoch ist es wichtig, die Auditfunktionalität zu benutzen, damit bestimmte Verhaltensmuster eines Fehlers oder Angriffe bestimmt werden können.

letzte Änderung	10-April-2000
überprüft am	10-April-2000

Kapitel 4

Sicherheitsanalyse der Internet-Protokolle

4.1 IPv4: Internet Protocol

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Klartextübertragung der Nutzdaten	Packet-Sniffing	ja	Verschlüsselung	hoch
Absender frei wählbar	IP-Spoofing	ja	IP-Spoofing Unterdrückung	hoch
keine gesicherte Paket-Zustellung	Umleitung des Verkehrs	unbekannt	—	gering

Aufgaben

Die Aufgabe des „Internet Protocol“ (IP) ist die Übertragung von Dateneinheiten (*Datagrammen*) zwischen der Netzwerk Schicht (*Network Access Layer*) und der Transport Schicht (*Transport Layer*) sowie der Transport von Datagrammen über das Rechner-Netz. Durch eine Quell- und eine Zieladresse werden in jedem Datagramm die Kommunikationspartner festgelegt.

Das „Internet Protocol“ ist der grundlegende Baustein des Internets; alle weiteren TCP- und UDP-Protokolle benutzen IP zur Übertragung von Datenpaketen.

Funktionsweise

IP unterstützt die Übertragung von Datagrammen durch mehrere Netze und deren Gateways, auf denen die zulässigen Nachrichtenlängen variieren können. Das Internet Protokoll übernimmt die Fragmentierung der Daten. Die entstehenden Datagramme werden unabhängig voneinander transportiert, erhalten aber spezielle Kennungen, die anschließend ein Wiederzusammenfügen der Nutzdaten ermöglichen.

Schwachpunkte

Grundsätzliche Schwachpunkte

Das „Internet Protocol“ ist beschränkt auf die Übermittlung einzelner unabhängiger Pakete und weist keinerlei Mechanismen zur Gewährleistung der Übertragungszuverlässigkeit und Übertragungsreihenfolge auf. Im einzelnen werden die Datagramme übertragen, ohne daß beim Aufbau der End-zu-End Verbindung Kontroll-Informationen ausgetauscht werden. Des weiteren sind auch keine Fehlererkennung und keine Überprüfung der Authentizität bei der Datenübertragung enthalten. Dadurch sind insbesondere die verwendeten IP-Adressen vom Absender frei wählbar und bei gefälschten IP-Adressen ist eine Ermittlung des Absenders anhand der Datagramme nicht möglich.

Schwachpunkte des Protokolls:

- ungesicherter Verbindungsaufbau
- unzuverlässige Datenübertragung
- unverschlüsselte Datenübertragung
- IP-Adressen können gefälscht werden

Bedrohung:

- Datenverlust bei Übertragung
- Abhören
- nicht authentisierter Zugang
- Ausnutzung fehlerhafter Implementationen des IP-Stacks, um einen Rechner zum Absturz zu bringen (**Denial-of-Service**)

Angriffsmöglichkeiten:

- **Sniffer** erlauben das Aufzeichnen von Daten-Paketen, die z. B. Paßwortinformationen oder andere sensitive Daten enthalten können.
- Das Fälschen von IP-Adressen (**IP-Spoofing**) kann in Zusammenhang mit anderen Protokollen ausgenutzt werden, um sich hierüber einen nicht autorisierten Zugang zu einem Rechner zu verschaffen.
- IP-Pakete mit bestimmten Charakteristiken können zum Absturz des angesprochenen Rechners führen (Implementationsfehler im IP-Stack).
- Durch Manipulation von Routing-Tabellen kann der IP-Verkehr umgeleitet werden und dadurch Informationen erlangt bzw. unterdrückt werden.

Bekannte Angriffe

Sniffer-Programme, mit denen über das Rechner-Netz übertragene Benutzernamen und Paßworte aufgezeichnet werden (vgl. 2.1, DSB-94:01[1] und DSB-94:03[2]), werden vor allen in „shared–LANs“ nach erfolgreichen Angriffen auf den SuperUser-Account *root* eingesetzt, um Angriffe auf weitere Systeme vorzubereiten und unbefugten Zugang zu anderen Systemen zu erlangen.

[1] `http://www.cert.dfn.de/infoserv/dsb/dsb-9401.html`
[2] `http://www.cert.dfn.de/infoserv/dsb/dsb-9403.html`

Konkrete Informationen über Angriffe sind z. B. in den CERT Advisories CA-94:01[3] und CA-95:18[4] enthalten.

Das Fälschen von IP-Adressen (IP-Spoofing) wird ausgenutzt, um Zugangskontrollen, die IP-Adressen zur Authentisierung benutzen, zu umgehen oder den Ursprung eines Angriffs zu verschleiern. Hinweise zu dem IP-Spoofing finden Sie in dem CERT Advisory CA-95:01[5].

Absicherungsmaßnahmen

1. Verschlüsselung des IP-Verkehrs (IPsec)

2. Authentisierung des IP-Verkehrs (IPsec)

3. Bei dem Übergang zwischen zwei Netzen (z. B. Anbindung des LANs an das Internet) sollten nur die Pakete weitergeleitet werden, die eine gültige IP-Absendeadresse haben. Dieses läßt sich durch Firewalls realisieren, die in dem Kapitel 5 beschrieben werden (Stichworte „*ingress Filtering*“ und „*egress Filtering*“).

4. Nutzung einer Netzwerkstruktur, bei der Verkehr zwischen zwei Rechnern nicht von einem unbeteiligten Dritten abgehört werden kann (Einsatz von Switches und getrennter Verkabelung zu den einzelnen Rechnern)

5. Einspielen der jeweils aktuellen Patches des Betriebssystem-Herstellers zur Schließung bekannter Sicherheitslücken.

Einschränkungen

Das Problem der fehlenden Zuverlässigkeit der Datenübertragung ist auf IP-Ebene ungelöst und muß auf höheren Protokoll-Ebenen gelöst werden.
Durch Manipulation von Routingtabellen kann der IP-Verkehr unbemerkt umgeleitet oder unterdrückt werden. Ebenso können bei einer Überlastung des Netzes Datenpakete verloren gehen.

[3] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-94:01.network.monitoring.attacks`
[4] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-95:18.widespread.attacks`
[5] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-95:01.IP.spoofing`

Fazit

Sniffer zum Abhören der Kommunikation sind weit verbreitet und werden von Angreifern häufig eingesetzt. Durch die fehlende Verschlüsselung der übertragenen Daten sind Paßwortinformationen und sensitive Daten nicht gesichert.

Das Hauptproblem der unverschlüsselten Datenübertragung kann nur durch den Einsatz von kryptographischen Verfahren (wie z. B. IPsec, das Bestandteil von IPv6 ist) gelöst werden. Leider sind die entsprechenden Implementationen noch nicht weit verbreitet verfügbar, so daß die Umstellung auf eine gesicherte Verbindung erst langsam erfolgen kann. Auf jeden Fall müssen hier die aktuellen Entwicklungen in diesem Bereich weiter verfolgt werden. Als Übergangslösung können die einzelnen Protokolle, die IP benutzen, abgesichert werden. Auf diese Protokolle wird im folgenden eingegangen.

letzte Änderung	04-Mai-2000
überprüft am	17-Mai-2000

4.2 UDP: User Datagram Protocol

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Schwachpunkte von IP	siehe IP	siehe IP	siehe IP	siehe IP
Fehlende Loop-Erkennung	UDP Denial-of-Service	selten	Kontrolle am Firewall	mittel
Unkontrollierte Vervielfältigung mittels Broadcasts	UDP Denial-of-Service	ja (oft)	Kontrolle am Firewall/auf Routern	hoch

Aufgaben

Das „User Datagram Protocol“ (UDP) erweitert IP im wesentlichen um sogenannte „Portnummern“. Die Portnummern dienen der Festlegung bestimmter Dienste in den kommunizierenden Systemen. Über die Portnummern kann entschieden werden, welcher Anwendung die übertragenen Daten verarbeiten soll. Eine Vielzahl dieser Portnummern ist daher durch die Internet-Gemeinde in einem RFC [RFC 1700] und später durch die IANA (Internet Assigned Numbers Authority, [IANA Numbers]) standardisiert worden, um eine einheitliche Benutzung der Portnummern zu ermöglichen.

Funktionsweise

UDP fungiert als eine Art Multiplexer, der den eintreffenden Datenstrom an die zuständigen Anwendungen bzw. an weitere Protokolle verteilt. Es dient zur Datenübertragung zwischen der Anwendungsschicht (*Application Layer*) und der Internet Schicht (*Internet Layer*) und erlaubt den Austausch von Nachrichten über das Netzwerk mit einem minimalen Protokoll-Overhead.

Schwachpunkte

Grundsätzliche Schwachpunkte

UDP ist ein unzuverlässiges Protokoll zum Versenden von Datenpaketen. Es ist nicht gewährleistet, daß abgesandte Pakete auch zugestellt werden oder daß die Pakete in der korrekten Reihenfolge zugestellt werden. Ebenso ist die Integrität der Nutzdaten nicht gewährleistet.

UDP setzt auf dem IP auf und hat daher zusätzliche die Schwachpunkte von IP (siehe Kapitel 4.1).

Schwachpunkte des Protokolls:

- Authentisierung über IP-Adressen
- unverschlüsselte Datenübertragung
- Fehlende Loop-Erkennung

Bedrohung:

- nicht autorisierter Zugang
- Abhören
- Denial-of-Service Angriff mittels *UDP-Loops*
- Denial-of-Service Angriff mittels *UDP-Floods*

Angriffsmöglichkeiten:

- Alle Angriffsmöglichkeiten von IP (siehe Kapitel 4.1)
- Das Fälschen von UDP-Portnummern und IP-Adressen ermöglicht einem Angreifer ein Paket zu konstruieren, auf das eine Anwendung reagiert und ein Antwortpaket versendet. Hierduch kann es zu Endlos-Schleifen mit entsprechendem Ressourcen-Verbrauch (Netzwerk, CPU-Nutzung) kommen (sog. *UDP-Loops*).
- UPD Datagramme können durch eine Versendung an die *Broadcast* IP-Adresse eines Rechner-Netzes vervielfältigt werden. Unter Verwendung einer gefälschten Absender-Adresse kann somit ein **Denial-of-Service** erreicht werden (sog. *UDP-Floods*), da die von den verschiedenen Rechnern des Netzwerkes versendeten Antwort-Datagramme an diesen (gefälschten) Absender zugestellt werden.

Bekannte Angriffe

Neben den bereits in Kapitel 4.1 erwähnten Angriffen werden insbesondere die Denial-of-Service Angriffe mit UDP-Paketen häufiger gesehen. Informationen hierzu finden Sie in dem CERT Advisory CA-96:01[6].

[6] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-96.01.UDP_service_denial`

Absicherungsmaßnahmen

1. Verwendung von TCP statt UDP (wenn möglich und sinnvoll)
2. Authentisierung der kommunizierenden Rechner (IPsec)
3. Nutzung UDP-basierter Protkolle mit eigenen Sicherheitsmechanismen (z.B. secure RPC)
4. Einschränkung des UDP Verkehrs am Firewall
5. Kontrolle des UDP Verkehrs (insbesondere an IP-Broadcasting-Adressen) auf Routern

Einschränkungen

Die Einschränkungen von IP gelten auch für UDP, da UDP auf dem IP aufsetzt.

Fazit

UDP bietet eine einfache und sehr effiziente Möglichkeit, kleine Datenmengen zu übertragen. Die fehlende Absicherung des Protokolls kann derzeit nur durch den Einsatz kryptographischer Verfahren (IPsec) oder auf Anwendungsebene kompensiert werden.

letzte Änderung	04-Mai-2000
überprüft am	17-Mai-2000

4.3 TCP: Transmission Control Protocol

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Schwachpunkte von IP	siehe IP	siehe IP	siehe IP	siehe IP
ungesicherter Verbindungs-kontext	TCP-Hijacking	selten	Verschlüsselung	mittel

Aufgaben

Das „Transmission Control Protocol“ (TCP) kann als zuverlässiger „Basic Data Transfer“ einen kontinuierlichen Datenstrom austauschen, in dem Daten in Segmente verpackt werden. TCP erweitert IP ebenfalls um die Portnummern (wie bereits bei UDP in Kapitel 4.2 diskutiert) und unterstützt mehrere Ports, um die gleichzeitige Kommunikation mehrerer Applikationen zu ermöglichen. Zusätzlich bietet TCP folgende Dienste an:

- Gesicherte End- zu End-Verbindung
  Es wird sichergestellt, daß die übertragenen Daten sowohl korrekt als auch in der richtigen Reihenfolge übertragen werden. Verlorene oder verfälschte Datagramme werden von TCP automatisch neu übertragen.
- Übertragung von Vorrangdaten
  Durch diesen Dienst können wichtige Datagramme bevorzugt behandelt werden, d.h. sie werden unmittelbar nach dem Eintreffen an die Applikation weitergegeben und „überholen“ u.U. Daten, die zeitlich vor ihnen versendet wurden. Diese Vorrangdaten werden daher als „out of band“ Daten bezeichnet, also als „Daten außerhalb der normalen Reihenfolge“.

Funktionsweise

Das TCP Protokoll benutzt zwei spezielle Bits im Paket-Header, `SYN` (Synchronize Sequence Numbers) und `ACK` (Acknowledgment), um den Aufbau einer neuen Verbindung über einen „*three-way handshake*“ zu initiieren. Dies dient zur Synchronisation der Sequenz-Nummern, damit die Daten als zusammenhängender/fortlaufender Datenstrom und nicht als unabhängige Pakete gesendet werden können.

Schwachpunkte

Grundsätzliche Schwachpunkte

TCP setzt auf IP auf und hat daher auch die Schwächen von IP (siehe Kaptiel 4.1) (mit Ausnahme der unzuverlässigen Paketzustellung, die durch die Verwendung der Sequenz-Nummern sichergestellt ist).

TCP verwendet den o. g. Drei-Wege-Handshake. Bedingt hierduch muß jedes System eine Tabelle der aktuellen Zustände einer TCP-Verbindung vorhalten. Um Überlastungen (und damit zu hohen Ressourcenverbrauch) zu vermeiden, sind die Größen der verwendeten Tabellen beschränkt, was bei Überlast dazu führt, daß keine weiteren Verbindungen mehr entgegegen genommen werden.

Schwachpunkte des Protokolls:

- alle Schwachpunkte von IP
- fälschbare Portnummern
- Limitierungen in den einzelnen Implementationen (z. B. Anzahl der offenen oder der zu öffnenden Verbindungen)

Bedrohung:

- alle Bedrohungen von IP (siehe Kapitel 4.1).
- keine Kontrolle über Datenstrom (fehlende Authentisierung der kommunizierenden Rechner)
- Überlastung eines Rechners führt zur Blockade (oder zum Abbruch) von TCP-Verbindungen

Angriffsmöglichkeiten:

- alle Angriffsmöglichkeiten von IP (siehe Kapitel 4.1).
- **TCP-Hijacking** wird benutzt, um die Kontrolle über eine bereits existierende interaktive Session zu erlangen.
- **TCP SYN-Flooding** ermöglichen (Netzwerk-)Blockade eines Rechners.

Bekannte Angriffe

TCP Hijacking ist zwar für einen Angreifer sehr effektiv, wird derzeit aber noch relativ selten ausgenutzt. Die Angriffe sind in dem CERT Advisory CA-96:21[7] beschrieben.

Denial-of-Service Angriffe durch **TCP SYN-Flooding** werden häufiger eingesetzt. Ziel der Angriffe sind in der Regel Server (WWW-Server, FTP-Server, ...). Informationen zu diesen Angriffen finden Sie ebenfalls im o. g. CERT Advisory CA-96:21.

[7] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-96.21.tcp_syn_flooding`

Absicherungsmaßnahmen

1. Absicherungen zu IP ergreifen (siehe Kapitel 4.1)
2. Anwendungen, die TCP nutzen, kryptographisch absichern
3. Implementation von sog. SYN-Cookies zur Abwehr von TCP SYN-Flooding Angriffen

Einschränkungen

TCP beugt einem IP-Spoofing durch den Drei-Wege-Handshake vor, da der die Verbindung initiierende Rechner (Rechner 1) auf das Antwortpaket des kontaktierten Rechners (Rechner 2) ebenfalls ein gültiges Antwortpaket senden muß. Diese Antwortpakete enthalten die Sequenz-Nummern der TCP-Verbindung, die von den entsprechenden Rechnern zufällig gewählt werden. Rechner 1 kann also ohne die Kenntnis der im ersten Antwortpaket von Rechner 2 versandten Sequenz-Nummer keine gültige Antwort generieren.
Bei einigen Betriebssystemen ist jedoch die für neue Verbindungen verwendete Sequenz-Nummer erratbar, so daß ein Angreifer (unter Benutzung des die Verbindung initiierenden Rechners 1) die Antwortpakete des mit dieser Schwachstelle behafteten Rechners 2 nicht benötigt, um ein gültiges Antwortpaket zu generieren. Damit ist die Absender-Adresse in dem ursprünglich zum Verbindungsaufbau versandten Datagramm beliebig wählbar, so daß Authentisierungs-Mechanismen, die auf IP-Adressen basieren, umgangen werden können.

Fazit

Das Hauptproblem des TCP Protokolls ist die fehlende Verschlüsselung und Authentisierung der Daten. Dadurch können Informationen abgehört, unterdrückt, eingefügt oder modifiziert werden.

letzte Änderung	04-Mai-2000
überprüft am	26-Mai-2000

4.4 TFTP: Trivial File Transfer Protocol

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
keine Authentisierung	Paßwortdatei stehlen	ja	Dateizugriffe einschränken	sehr hoch
keine Authentisierung	falsche TFTP-Daten einschleusen	nein	Verzicht auf TFTP	mittel
unverschlüsselte Datenübertragung	Sniffer	ja (häufig)		sehr hoch

Aufgaben

Das „Trivial File Transfer Protocol“ (TFTP) ist ein primitives UDP-basiertes Protokoll zur Datenübertragung, das absichtlich möglichst einfach gehalten wurde. Es versorgt Klienten auf einfachem Wege mit Informationen, welche diese für den Betrieb benötigen, ohne daß sie dafür eigene Kapazitäten für die dauerhafte Speicherung der Daten besitzen müssen.

Funktionsweise

TFTP führt einen Datentransfer über Port 69/UDP ohne vorherige Authentisierung durch. Der Zugriff kann bei besseren Implementationen über `chroot` auf einen in sich abgeschlossenen Bereich des Dateisystems beschränkt werden.

Schwachpunkte

Grundsätzliche Schwachpunkte

TFTP ermöglicht den Zugang zu einem Remote-System, ohne eine Form der Authentisierung zu verlangen. Damit kann jeder Benutzer aus dem Internet auf das System zugreifen; man benötigt keinen Account und kein Paßwort auf dem Server, der den TFTP-Dienst zur Verfügung stellt.

Schwachpunkte des Protokolls:

- **keine Authentisierung**
- **unverschlüsselte Datenübertragung**

Bedrohung:

- öffentlicher Zugang zum System
- Zugriff auf Dateien
- Manipulation von Daten
- Abhören der übertragenen Daten

Angriffsmöglichkeiten:

- Stehlen der Paßwortdatei
- Sniffer
- Verbreiten gefälschter Informationen (Täuschung von TFTP-Klienten)

Bekannte Angriffe

Der Zugriff auf Paßwortdateien via TFTP, mit anschließender Weiterverarbeitung durch Angriffs-Tools (z. B. „Crack“) auf dem eigenen Rechner ist schon lange ein bekanntes Problem. Die so erlangten Paßworte können für weiteren Mißbrauch eingesetzt werden, weshalb in zahlreichen *CERT Advisories* auf die Schwächen im TFTP-Protokoll hingewiesen wird: CA-90:02[8], CA-90:11[9], CA-91:18[10], CA-94:05[11], CA-95:06[12] und CA-95:18[13].

Absicherungsmaßnahmen

Da dieses Protokoll über keinerlei Authentisierungsmöglichkeiten verfügt, sollte überprüft werden, ob dieser Dienst überhaupt angeboten werden muß. Falls TFTP eingesetzt werden soll, muß der Dateizugriff auf einen stark begrenzten Teil des gesamten Dateisystems eingeschränkt werden. Zu empfehlen ist auch, für den Zugriff auf TFTP nur bestimmte Rechner im lokalen Netz zuzulassen.

[8] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-90:02.intruder.warning`

[9] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-90:11.Security.Probes`

[10] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-91:18.Active.Internet.tftp.Attacks`

[11] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-94:05.MD5.checksums`

[12] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-95:06.satan`

[13] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-95:18.widespread.attacks`

Einschränkungen

Die Schwachstellen des TFTP Dienstes sind so grundlegend, daß eine Absicherung ohne eine Protokolländerung kaum durchführbar ist.

Fazit

TFTP bietet eine sehr einfache Möglichkeit der Dateiübertragung. Die Anfragen, Antworten und die übertragenen Daten sind in keiner Weise authentisiert oder verschlüsselt.

Ein TFTP-Server sollte nur öffentlich zugängliche Informationen ohne Sicherheitsrelevanz über TFTP anbieten (z. B. Fonts ohne lizenzrechtlichen Schutz). Sensitive Informationen (z. B. Paßwort- oder Benutzerdaten) sollten nicht über TFTP zugreifbar sein.

letzte Änderung	02-Mai-2000
überprüft am	23-Mai-2000

4.5 Remote Services

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Authentisierung über IP-Adressen von *„trusted hosts“*	IP-Spoofing	ja	Verwendung sicherere Protokolle oder Blockade am Firewall	sehr hoch

Aufgaben

Die sog. r-Dienste (`rsh` und `rlogin`) ermöglichen den Aufbau von Verbindungen zu entfernten Rechnern im Netzwerk, ohne erneut den Benutzernamen und das Paßwort abzufragen. Der Benutzername wird automatisch beim Verbindungsaufbau übermittelt. Kommt eine Verbindung von einem vertrauenswürdigen Rechner (*trusted host*), startet der Remote-Rechner automatisch den entsprechenden Dienst.

Funktionsweise

Das *rsh*-Protokoll erlaubt es einem Benutzer, ein Kommando auf einem anderen Rechner ohne die Angabe eines Paßwortes auszuführen. Das *rlogin*-Verfahren stellt ein interaktives Login ohne Paßwortangabe auf einem anderen Rechner zur Verfügung. Dabei nimmt der r-Dienst (z. B. *rlogind*) auf dem Zielrechner die Verbindung entgegen, überprüft die Authentizität des anfragenden Rechners und Benutzers und startet ein lokales Programm (z. B. `/bin/login`). Voraussetzung für die Akzeptanz eines fehlenden Paßwortes ist die Akzeptanz des Quellrechners und -benutzers durch den Zielrechner und -benutzer.

Schwachpunkte

Grundsätzliche Schwachpunkte

Der große Nachteil liegt in der Möglichkeit, daß jeder Benutzer eigenverantwortlich bestimmen kann, welchen Rechnern und Benutzern er vertraut. Dadurch ist es möglich, daß Benutzer ihre Benutzerkennung auch anderen Benutzern zur Verfügung stellen und damit die Sicherheit des gesamten Systems gefährden. Darüber hinaus besteht das Problem, daß zur Authentisierung IP-Adressen verwendet werden, die beliebig fälschbar sind.

Schwachpunkte des Protokolls:

- **schwache Authentisierung durch IP-Adressen**
- **unverschlüsselte Datenübertragung**
- **Benutzer kann Zugangskontrolle zum System verwalten**

Bedrohung:

- unautorisierter Zugang
- Abhören

Angriffsmöglichkeiten:

- IP-Spoofing
- Eingeschleuste Einträge in `.rhosts` Dateien

Bekannte Angriffe

Verschiedene Angriffstools und Trojaner schreiben die Folge „`+ +`“ in die `.rhosts`-Datei. Dieser Eintrag ermöglicht jedem Benutzer von jedem Rechner einen Login zu der Benutzerkennung, in deren Home-Verzeichnis sich die `.rhosts`-Datei befindet.

Speziell für die Nutzung von IP-Spoofing zwecks Zugang über die Dienste `rsh` oder `rlogin` stehen verschiedene fertige Angriffsprogramme zur Verfügung, die auch häufig von Angreifern benutzt werden (vgl. 4.3).

Absicherungsmaßnahmen

Statt `rsh` und `rlogin` sollte ein sichereres Protokoll verwendet werden, das eine sichere Authentisierung und Verschlüsselung unterstützt. Eine solche Alternative bietet die „Secure Shell“ (`ssh`) (siehe Abschnitt A.4).

Sollte der Einsatz von `rsh` und/oder `rlogin` innerhalb des eigenen Netzes zwingend notwendig sein, so sollten am Firewall die Portnummern 513 und 514 (jeweils TCP) blockiert werden, um Angriffe von außerhalb des eigenen Netzes abzuwehren.

Einschränkungen

Einige Backup-Lösungen benutzen die C-Funktion `rcmd()`, um auf andere Rechner zuzugreifen. Diese Funktion nutzt intern `rsh`, um einen Befehl auszuführen. Weitere Programme, wie z. B. `rdist`, verwenden den gleichen Mechanismus. Bei einem

Ersatz von `rsh`/`rsh` durch `ssh` sind eventuell Änderungen an den Programmen, die `rcmd()` nutzen, notwendig. Hinweise hierzu sind in der Dokumentation zu `ssh` zu finden.

Fazit

Die r-Dienste nutzen fälschbare IP-Adressen zur Authentisierung. Angriffsprogramme zum Ausnutzen dieser Schwachstelle sind frei verfügbar und werden häufig eingesetzt. Die Dienste bieten einem Angreifer einen direkten Zugang zu dem System, das diese Dienste zur Verfügung stellt.

letzte Änderung	04-Mai-2000
überprüft am	26-Mai-2000

4.6 Telnet

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Übertragung unverschlüsselter Paßworte	Sniffer	ja (häufig)	`ssh`	sehr hoch
			Einmal-Paßworte	sehr hoch
ungesicherter Verbindungskontext	TCP-Hijacking	selten	`ssh`	sehr hoch

Aufgaben

Das Telnet Protokoll ([RFC 854]) ist auf der Applikations-Ebene angesiedelt. Es erlaubt einem Benutzer, sich über das Rechner-Netz auf jedem erreichbaren System anzumelden, sofern er eine Zugangsberechtigung (Benutzerkennung mit Paßwort) zu diesem System besitzt.

Funktionsweise

Der `telnet`-Dienst läuft unter UNIX am privilegierten TCP-Port 23. Die Anmeldung erfolgt durch einen *telnet*-Klienten am *telnet*-Server.

Ein Benutzer startet einen Klienten-Prozeß auf seinem lokalen System, der die Verbindung zu einem entfernten System über das Rechner-Netz aufzunehmen versucht. Der *telnet*-Server des entfernten Systems erwartet vor dem Aufbau der `telnet`-Verbindung die Authentisierung des Benutzers durch Angabe eines Benutzernamens und Paßwortes. Ist die Verbindung aufgebaut, stellt `telnet` auf dem entfernten System ein „virtuelles Terminal“ bereit.

Schwachpunkte

Sowohl die Kontrollinformationen beim Verbindungsaufbau als auch die eigentlichen Daten werden im Klartext übermittelt, so daß die übermittelten Informationen ungeschützt sind.

Grundsätzliche Schwachpunkte

Sämtliche Kommunikation zwischen Klient und Server kann *abgehört* werden, weil alle Daten im Klartext übermittelt werden. Außerdem können Daten (insbesondere die

Kontrollinformationen) gefälscht werden und bereits aufgebaute Telnet Verbindungen umgeleitet werden (Hijacking).

Schwachpunkte des Protokolls:

- **unverschlüsselte Datenübertragung**
- **schwache Authentisierung durch Paßworte**
- **ungesicherter Verbindungskontext**

Bedrohung:

- Abhören der übertragenen Daten-Pakete
- Aufzeichnung von Paßwortinformationen
- unautorisierter Zugang

Angriffsmöglichkeiten:

- Sniffer
- Telnet-Hijacking

Bekannte Angriffe

Da die Telnet-Pakete auf dem gesamten Übertragungsweg — nicht nur im lokalen Netz — ungesichert sind, werden sie immer wieder von speziellen Programmen, sogenannten Sniffern (vgl. 4.1), aufgezeichnet. Die abgehörten Daten werden von dem Angreifer weiter analysiert, so daß dieser daraus an Benutzernamen und Paßwortinformationen gelangen kann, welche ihm einen unerlaubten Zugang zum System ermöglichen.

Eine weitere Gefahr besteht darin, daß eine bereits aufgebaute Telnet-Verbindung von einem Angreifer durch sog. „session hijacking“ übernommen wird. Nachdem der Benutzer sich mit seinem Paßwort authentisiert hat, kann der Angreifer die Telnet-Verbindung übernehmen und beliebige Kommandos ausführen.

Absicherungsmaßnahmen

Da Telnet ein weit verbreiteter Service ist, auf den viele nicht verzichten wollen bzw. können, sollte er zumindest dahingehend abgesichert werden, daß keine wiederverwendbaren Paßworte zugelassen werden. Durch den Einsatz von *SSH* oder Einmal-Paßworten können zumindest Sniffer Angriffe auf die Paßwortinformationen verhindert werden.

Soll auch Telnet-Hijacking verhindert werden, müssen Verschlüsselungsverfahren, z. B. `ssh` (siehe Abschnitt A.4) eingesetzt werden.

Einschränkungen

Ein Schutz gegen Telnet-Hijacking ist nur möglich, wenn auf das Telnet-Protokoll verzichtet wird und statt dessen ein Protokoll verwendet wird, das die komplette Verbindung verschlüsselt (siehe *SSH*).

Fazit

Das Abhören von Paßwortinformationen bei Telnet-Verbindungen stellt eine der wichtigsten Schwachstellen dar, über die Angreifer in Systeme einbrechen. Weitere Schwachstellen in den Endsystemen führen schnell von einem nicht autorisieren Zugang zu einer Benutzerkennung zu einem *root-compromise*.

letzte Änderung	04-Mai-2000
überprüft am	23-Mai-2000

4.7 FTP: File Transfer Protocol

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Übertragung unverschlüsselter Paßworte	Sniffer	ja (häufig)	`ssh`	sehr hoch
Unsichere Konfiguration	Umschlagplatz für nicht autorisierte Daten	ja	sichere Konfiguration	hoch
FTP `PORT` Befehl	Port-Scanning über FTP-Server	ja (selten)	Einschränkung des `PORT` Befehls	mittel

Aufgaben

Das Ziel von FTP besteht darin, Dateien zwischen verschiedenen Systemen zuverlässig zu transferieren. Das innerhalb der Applikationsebene angesiedelte FTP setzt auf den darunter liegenden Transport-Protokollen auf. Die Spezifikation von FTP ([RFC 959]) setzt einen geordneten und zuverlässigen Paket-Transport – wie ihn z. B. TCP bietet – voraus.

Funktionsweise

Das FTP-Protokoll unterstützt zwei Operationsarten „active mode“ und „passive mode“, die bestimmen, ob der FTP-Server oder der FTP-Klient die TCP-Verbindung zur Datenübertragung initiieren.

Zunächst wird vom FTP-Klienten eine Verbindung zu Port 21 am FTP-Server geöffnet (Kontrollkanal). Im „active mode“ teilt der Klient dem Server die Portnummer mit, auf der er die Daten erwartet (`PORT`-Kommando). Der Server akzeptiert und öffnet von Port 20 einen Datenkanal zurück zu dem betreffenden Port des Klienten. Anschließend bestätigt dies der Klient und kann mit dem Transfer der Daten beginnen.

Im „passive mode“ initiiert der FTP-Klient die Verbindung, die der Server zur Datenübertragung zum Klienten benutzt. Der Klient schickt eine Passiv-Anfrage an den Server. Dieser akzeptiert und schickt die für den Daten-Kanal zu verwendende Portnummer zurück. Der Klient öffnet die Datenverbindung zu dem betreffenden Port und kann mit der Datenübertragung beginnen.

Grundsätzlich startet ein Benutzer auf seinem lokalen System einen Klienten-Prozeß, der die Verbindung zu einem FTP-Server aufzunehmen versucht. Der entfernte Server erwartet normalerweise beim Kontaktieren die Angabe eines Benutzernamens und Paßwortes. Ist der FTP-Server für einen anonymen Zugriff aufgesetzt, so kann jeder

Benutzer — auch die, die keinen Account auf dem System haben — Dateien aus bestimmten Verzeichnissen herunterladen oder in bestimmte Verzeichnisse ablegen, ohne sich vorher durch Angabe eines Benutzernamens und Paßwortes zu authentisieren.

Schwachpunkte

Grundsätzliche Schwachpunkte

Sämtliche Kommunikation zwischen Klient und Server kann *abgehört* werden, weil alle Daten im Klartext übermittelt werden.

Betreiber von „anonymous“-FTP Servern müssen darauf achten, daß der angebotene Dienst nicht zur Weitergabe von nicht autorisierten Daten (z. B. Raubkopien kommerzieller Software oder Musikstücken) genutzt werden kann.

Schwachpunkte des Protokolls:

- **unverschlüsselte Datenübertragung**
- **schwache Authentisierung**
- `PORT` **Befehl** erlaubt das Aufbauen von Verbindungen zu dritten Rechnern

Bedrohung:

- nicht autorisierter Zugang
- Abhören von Paßwortinformationen und sensitiven Daten
- Unerlaubtes Ablegen fremder Daten
- Mißbrauch des FTP-Servers als Proxy für Port-Scanning Angriffe

Angriffsmöglichkeiten:

- IP-Spoofing
- Sniffer
- Ressourcen Nutzung (Nutzung des Festplatten-Speicherplatzes von FTP-Servern)
- Portscanner mit FTP `PORT` Befehlen

Bekannte Angriffe

Durch **Sniffer-Programme** werden über das Rechner-Netz übertragene Benutzernamen und Paßworte aufgezeichnet (siehe Abschnitt 4.1). Zusätzlich zu dem Problem der Klartextübertragung der verwendeten Paßworte, werden FTP-Server häufig schlecht konfiguriert, so daß ein Mißbrauch des Servers möglich ist. Diese Angriffe sind in dem CERT Technical Tip *Anonymous FTP Abuses*[14] dokumentiert.

Absicherungsmaßnahmen

Hinweise zur sicheren Konfiguration des FTP Servers finden sich in den *Anonymous FTP Configuration Guideline*[15] des CERT/CC.

Da bei FTP die verwendeten Paßworte im Klartext übertragen werden, sollten Alternativen in Betracht gezogen werden, die eine sicherere Authentisierung ermöglichen. Mögliche Lösungen sind die Verwendung von Einweg-Paßworten oder die Verwendung von `scp` (aus dem `ssh`-Paket, siehe Abschnitt A.4). Auch eine „Tunnelung“ des Kontrollkanales durch eine bestehende `ssh`-Verbindung ist möglich.

Einschränkungen

Analog zu der Gefährdung bei Telnet ist auch bei FTP ein Session-Hijacking möglich. Die einzige Lösung für dieses Problem ist die Verwendung eines alternativen Protokolls, das die gesamte Verbindung verschlüsselt, oder die oben beschriebene „Tunnelung“ des Kontrollkanales.

Fazit

Das Abhören von Paßworten bei FTP-Verbindungen stellt eine der wichtigsten Schwachstellen dar, über die Angreifer in Systeme einbrechen. Weitere Schwachstellen im FTP-Server, zu dem durch das abgehörte Paßwort nicht autorisierter Zugang erlangt wurde, führen schnell zu einem *root-compromise*.

letzte Änderung	04-Mai-2000
überprüft am	30-Mai-2000

[14] `ftp://ftp.cert.dfn.de/pub/csir/cert/tech_tips/anonymous_ftp_abuses`
[15] `ftp://ftp.cert.dfn.de/pub/csir/cert/tech_tips/anonymous_ftp_config`

4.8 SMTP: Simple Mail Transfer Protocol

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Ungeschützte Absenderadresse	beliebig einfach	ja (sehr häufig)	Signatur	sehr hoch
Ungesicherter Nachrichteninhalt	Sniffer	ja (noch selten)	Kryptographie	hoch
Ableugnen der Kommunikation		nein		

Aufgaben

Das Ziel von SMTP besteht darin, Email zwischen verschiedenen Rechnern zuverlässig und effizient zu transportieren. Das innerhalb der Applikationsebene angesiedelte SMTP arbeitet dabei gänzlich unabhängig von den darunterliegenden Transport-Protokollen. Die Spezifikation von SMTP ([RFC 821]) setzt lediglich einen zuverlässigen und geordneten Paket-Transport – wie ihn z. B. TCP bietet – voraus und beschreibt die Mechanismen des Nachrichten-Transports. Das Format solcher Nachrichten wurde in [RFC 822] spezifiziert; im Internet werden die beiden Protokolle in der Regel gemeinsam eingesetzt.

Funktionsweise

Ein Benutzer startet einen Klienten-Prozeß auf seinem lokalen System und übergibt diesem die vorher erzeugte Nachricht. Der Klient versucht dann, die Verbindung zu einem entfernten SMTP-Server aufzubauen.[16] Ist die Verbindung zustande gekommen, tauschen Klient und Server bestimmte Informationen aus, bevor die eigentliche Nachrichtenübermittlung beginnen kann. Kommt keine Verbindung zustande, weil z. B. der Server momentan nicht verfügbar ist, speichert der Klient die zu übermittelnde Nachricht im Dateisystem (*Mail Spool Area*) und versucht die Verbindung zu einem späteren Zeitpunkt erneut herzustellen. Der Klient arbeitet also im Hintergrund, so daß der Absender einer Nachricht nicht auf die Auslieferung der Nachricht warten muß.

Bei den übertragenen Kommandos handelt es sich um lesbare Kommandos im ASCII-Format; den Antworten des Servers ist dabei eine drei-stellige Zahl vorangestellt. Diese Zahl dient der vereinfachten Kommunikation zwischen Klient und Server und macht die Interpretation der eigentlichen Antwort des Servers durch den Klient überflüssig.

[16] Bei dem Server, welcher unter UNIX am privilegierten Port 25 läuft, muß es sich nicht unbedingt um den Ziel-Server handeln; es ist auch möglich, daß zunächst eine Verbindung zu einem Server auf einem sog. *Mail Gateway* aufgebaut wird, welcher anschließend die Nachricht weiterleitet.

Sämtliche Kommunikation zwischen Klient und Server wird dabei im Klartext übertragen.

Vor der eigentlichen Nachricht werden Adreßinformationen (der sog. *Envelope*) zu dieser Nachricht übertragen. Diese enthalten den Adressanten der Nachricht (SMTP-Kommando `MAIL FROM`) und den Adressaten (SMTP-Kommando `RCPT TO`). Im Anschluß an diese Adreßinformationen wird die Nachricht selbst übertragen.

Eine Nachricht besteht aus Kontrollinformationen (dem sog. *Header*) sowie der eigentlichen Nachricht (dem sog. *Message Body*). Der Header enthält Informationen, die die fehlerfreie Übermittlung der Nachrichten garantieren sollen. Die minimal notwendigen Header-Einträge sind: „**From:**“ (Absender der Nachricht), „**To:**“ (vorgesehener Empfänger der Nachricht), sowie „**Date:**“ (Datum des Versands); weitere optionale Einträge sind z. B.: „**Cc:**“ (weitere Empfänger), „**Subject:**“ (Betreff), oder „**Message-ID:**“ (eindeutige Nachrichtenidentifizierung). Header und eigentliche Nachricht sind dabei durch eine Leerzeile getrennt und beinhalten in der Regel ausschließlich Zeichen des 7-Bit ASCII-Alphabets. Sollen Binärdateien per Email versendet werden, müssen diese vorher in ein geeignetes ASCII-Format umgewandelt werden.

Schwachpunkte

Weder die Vertraulichkeit, noch die Integrität der im Internet transportierten Nachrichten läßt sich garantieren. Auch die Authentizität (also die Echtheit) eines Nachrichten-Senders läßt sich mit SMTP nicht verifizieren. Da neben den Adreß- und Kontrollinformationen beim Verbindungsaufbau auch die eigentlichen Nachrichten im Klartext übermittelt werden, sind diese Informationen ungeschützt.

Grundsätzliche Schwachpunkte

Sämtliche Kommunikation zwischen Klient und Server kann *abgehört* werden, weil alle Nachrichten im Klartext übermittelt werden; außerdem können alle Daten (insbesondere die des Headers) aber auch die Adreßinformation im Envelope *gefälscht* werden, da weder der Ursprung, noch die Unversehrtheit der Daten kontrolliert werden kann.

Schwachpunkte des Protokolls:

- **unverschlüsselte Datenübertragung**
- **schwache Authentisierung via Hostnamen**

- **mangelnde Protokollierungsmechanismen**

Bedrohung:

- Verlust der Integrität / Unbefugte Manipulation
- Verlust der Vertraulichkeit
- Verlust der Anonymität
- Maskerade
- Leugnen der Teilnahme an der Kommunikation
- Änderung von Nachrichten-Reihenfolgen
- Wiedereinspielen von Nachrichten (*Replay*)
- Verzögerung von Nachrichten (*Delay*)
- Vorzeitige Übermittlung von Nachrichten (*Preplay*)
- Verkehrsanalysen (*Traffic analysis*)
- Fehlleiten einer Nachricht
- Verhinderung des Dienstes (*Denial of Service*)
- Mißbrauch des Dienstes (z. B. durch unerwünschte Emails, sog. *Spam*)

Angriffsmöglichkeiten:

- IP-Spoofing
- Sniffer
- Entsprechend veränderte/konfigurierte Klient-Programme

Bekannte Angriffe

Der Mißbrauch von SMTP ist seit der Anfangszeit des Internet ein relevantes Problem. Dies hat sich bis heute nicht verändert: Durch die ständig steigende Zahl von Diensteanbietern und der damit verbundenen großen Anzahl neuer Internet-Teilnehmer werden Mail-Dienste – inbesondere wegen der Einfachheit der Angriffe – immer wieder mißbraucht. Allein die Anzahl der *CERT Advisories* zum Thema Email ist deut-

lich: CA-90:08[17], CA-91:14[18], CA-95:02[19], CA-95:08[20], CA-96.20[21], CA-96.24[22], CA-96.25[23], CA-97.05[24] und CA-97.14[25].

Immer wieder werden unerwünschte Emails mit gefälschten Absenderadressen verschickt (sog. „Spam“), um beispielsweise aggressiv Werbung zu betreiben. Das Abhören von sensitiven Daten (z. B. Kreditkartennummern) einer Email stellt ebenfalls im Zeitalter des *e-commerce* ein zunehmendes Problem dar.

Absicherungsmaßnahmen

Fast alle in diesem Zusammenhang stehenden Schwachstellen lassen sich durch den Einsatz von Verschlüsselungssoftware erheblich eingrenzen, teilweise sogar komplett beseitigen. Beispielhaft sei hier das Programm „PGP“ (Pretty Good Privacy [RFC 1991, RFC 2440]) genannt, das momentan im Internet einen großen Stellenwert hat. Viele Hersteller von Email-Programmen sind ebenfalls dazu übergegangen, Verschlüsselungs-Funktionalität durch eine Implementation des S/MIME-Standards [RFC 1848, RFC 2311, RFC 2633] anzubieten.

Aufgrund des Umfangs und der Bedeutung kryptographischer Verfahren werden diese in Kapitel 6 eingehend beschrieben.

Einen grundsätzlichen Schutz gegen Mißbrauch durch unerwünschte Emails gibt es leider nicht. Es ist jedoch eine Filterung der ankommenden Email anhand der Adreß- und Kontrollinformationen oder eine vollständige Blockierung von Verbindungen von bestimmten Rechnern möglich. Weitere Informationen hierzu sind in der DIB-99:01 *Technische Maßnahmen gegen Spam*[26] zu finden.

[17] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-90:08.irix.mail`

[18] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-91:14.IRIX.mail.vulnerability`

[19] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-95:02.binmail.vulnerabilities`

[20] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-95:08.sendmail.v.5.vulnerability`

[21] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-96.20.sendmail_vul`

[22] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-96.24.sendmail.daemon.mode`

[23] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-96.25.sendmail_groups`

[24] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-97.05.sendmail`

[25] `ftp://ftp.cert.dfn.de/pub/csir/cert/cert_advisories/CA-97.14.metamail`

[26] `http://www.cert.dfn.de/infoserv/dib/dib-9901.html`

Einschränkungen

Generell können kryptographische Verfahren *destruktive Angriffe* zwar mit hoher Wahrscheinlichkeit entdecken, jedoch nicht verhindern. Da die per Email versendeten Nachrichten über das ansonsten ungeschützte Internet transportiert werden, kann ein Angreifer mit physikalischem Zugriff auf das Netzwerk diese Nachrichten willkürlich modifizieren. Kritisch ist dies prinzipiell jedoch nur dann, wenn es einem Angreifer gelingt, die Nachricht komplett zu löschen. Diese Form der Attacke wird üblicherweise weder vom Absender noch vom Empfänger bemerkt, wenn dieser nicht im voraus über das Eintreffen einer Email informiert wurde.

Da kryptographische Verfahren üblicherweise lediglich den Inhalt der Nachricht, nicht jedoch die Header-Einträge einer Email nach RFC 822 schützen, ist eine anonyme Kommunikation nicht möglich. In jeder versendeten Email sind sowohl Absender als auch Empfänger im Klartext enthalten. Damit könnten *Kommunikationsprofile* erstellt werden; z. B. die Angaben, ob und wieviele Nachrichten zwischen bestimmten Benutzern ausgetauscht wurden, die Häufigkeit der Kommunikation, die Richtung des Nachrichtenflusses, etc.

Zu den im weitesten Sinne durch physikalischen Zugriff auf das Netzwerk möglichen destruktiven Attacken gehört auch das Aufzeichnen einer kompletten Email mit dem Zweck, diese zu einem späteren Zeitpunkt *erneut einzuspielen*. Der Empfänger hat keine Möglichkeit festzustellen, ob es sich bei der zweiten Nachricht um eine kopierte Email handelt, oder ob der Absender diese Email tatsächlich noch einmal gesendet hat. In Fällen, in denen das Wiedereinspielen kritisch ist, sollte daher der Absender Datum und Zeitpunkt des Absendens innerhalb der Nachricht vermerken und diese kryptographisch schützen.

Fazit

Verschiedene Tools zur Absicherung des Email-Verkehrs sind auf dem Markt verfügbar und zum Teil weit verbreitet. Sie bieten einen deutlich Schutz gegenüber der ungesicherten Kommunikation und werden derzeit in immer mehr Standardprodukte integriert, was die Anwendbarkeit solcher Verfahren weiter erhöhen sollte. Aufgrund der stetig wachsenden Bedeutung der elektronischen Kommunikation nimmt auch die Anzahl der sensitiven Daten zu, die über derartige Medien übermittelt werden. Kryptographische Verfahren können wirksam vor einer Vielzahl von Angriffen schützen, ohne jedoch sämtliche Probleme zu beseitigen.

letzte Änderung	10-Mai-2000
überprüft am	30-Mai-2000

4.9 NFS: Network File System

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
schwache Authentisierung	Umgehung von Zugriffsrechten	ja (selten)	secure NFS	mittel
falsche Konfiguration	Zugriff auf NFS-Daten	ja	sichere Konfiguration	hoch

Aufgaben

Eine der Grundanwendungen in Netzwerken ist die Nutzung der Plattenkapazitäten eines Servers durch seine Klienten. In dem Bereich der reinen IP Netzwerke kommt diesbezüglich das „Network File System“ (NFS) zum Einsatz. Für andere Netze gibt es ähnliche Lösungen.

Funktionsweise

Bei der Verwendung von NFS werden Daten zentral auf einem Server verwaltet. Dieser stellt z. B. Festplatten-Partitionen[27] bereit, auf deren Daten andere Rechner (Klienten) über das NFS-Protokoll zugreifen können. Diese Art des gemeinsamen Datenzugriffs basiert konkret auf zwei Protokollen:

Mount: Das Mount-Protokoll dient zur Feststellung, welche Partitionen für welche Rechner verfügbar sind. Aus der Sicht des Servers übernimmt dieser Dienst die Vergabe von Zugriffsrechten auf Daten an andere Rechner. Klienten holen sich über dieses Protokoll eine Zugriffsnummer, die sie bei allen nachfolgenden NFS-Aktionen benutzen.

NFS: Das NFS-Protokoll übernimmt alle aktiven Zugriffe auf ein Dateisystem des Servers. Hier sind alle üblichen Zugriffsmethoden implementiert (also z. B. Datei lesen, Verzeichnis anlegen, Links erzeugen, . . .)

Für die Zugriffe (sowohl das sog. „mounten“ als auch die Dateizugriffe via NFS) werden „Remote Procedure Calls“ (siehe 4.11) verwendet. Die RPC Anfragen werden mit Hilfe der UDP oder TCP Protokolle übertragen. NFS Server unterstützen in der Regel beide Übertragungsarten (UDP und TCP), während die Klienten oft nur auf eine der beiden Arten zugreifen.

[27] Der Begriff „Festplatten“ stellt dabei nur eine Vereinfachung dar. Natürlich können auch CD-ROMs oder andere Datenträger gemeinsam von verschiedenen Rechnern genutzt werden.

Schwachpunkte

Grundsätzliche Schwachpunkte

Ein großes Problem besteht darin, daß zur Authentisierung IP-Adressen verwendet werden, die beliebig fälschbar sind.

Schwachpunkte des Protokolls:

- **schwache Authentisierung durch IP-Adressen**
- **unverschlüsselte Datenübertragung**

Bedrohung:

- unautorisierter Zugang zu Daten
- Abhören

Angriffsmöglichkeiten:

- Mounten von NFS-Verzeichnissen
- indirekter NFS-Zugriff via Portmapper/Rpcbind
- Raten von NFS-Filehandles
- IP-Spoofing
- Sniffer

Bekannte Angriffe

Schwachstellen in der Konfiguration führen manchmal dazu, daß Verzeichnisse für alle Rechner im Internet freigegeben werden. Angreifer probieren daher die exportierten Dateisysteme abzufragen (`showmount -e`) und dann gezielt anzusprechen.

NFS benutzt zur Prüfung der Zugriffsrechte lediglich ein Verfahren, das unter dem Namen `AUTH_UNIX` bekannt ist. Bei diesem Verfahren schickt der Klient seine *User-ID* (UID, numerische Benutzerkennung) und seine *Group-IDs* (GIDs, numerische Gruppennamen) zum Server, der diese für die Zugriffskontrolle nutzt. Da die Kontrolle über die verwendeten UIDs und GIDs beim Klienten liegt, kann dieser sehr leicht falsche Informationen zum Server schicken und dadurch Zugang zu den Daten anderer Benutzer erhalten.

Absicherungsmaßnahmen

Statt der üblichen (schwachen) Authentisierung sollte eine sicherere Authentisierung mit Hilfe eines Public-Key Verfahrens verwendet werden. Dieses ist unter dem Stichwort „secure NFS“ verfügbar und nutzt zur Absicherung „secure RPC“. Alternativ kann Kerberos oder AFS (kommerzielles Produkt) zur Absicherung verwendet werden.

Der Zugriff zu dem rpcbind/portmapper Dienst sollte eingeschränkt werden. Dieses kann entweder durch einen Filter am Firewall auf Port 111 (UDP und TCP) oder durch den Einsatz eines Ersatzprogrammes, das eine Zugriffskontrolle ermöglicht (siehe Abschnitt A.3), erfolgen.

Die Zugangkontrolle zu den NFS-Diensten am Firewall ist nur möglich, wenn alle Ports geblockt werden und nur einzelne Dienste freigeschaltet werden (die Dienste `mountd` und `lockd` verwenden keine festen, sondern dynamisch vergebene Portnummern). Eine Blockade von Port 2049 (UDP und TCP) verhindert zwar den Zugriff zum `nfsd`, nicht aber zu den anderen NFS-Komponenten.

Als Minimalabsicherung sollte zumindest die Verwendung von privilegierten Portnummern für NFS aktiviert werden.

Hinweise zu den einzelnen Absicherungsmaßnahmen finden Sie in dem DFN-CERT Sicherheitsbulletin DSB-94:05 *Sicherheitslücken bei der Benutzung von NFS*[28].

Einschränkungen

Sichere Erweiterung für NFS (secure NFS, Kerberos oder AFS) müssen nicht nur auf dem NFS-Server sondern auch auf allen Klienten zur Verfügung stehen.

Fazit

NFS bietet nur eine sehr schwache Zugangskontrolle. Es sollten auf jeden Fall sicherere Authentisierungsprotokolle, wie z. B. secure NFS, Kerberos oder AFS, verwendet werden. Ob und welche dieser Optionen unterstützt werden, hängt vom verwendeten Betriebssystem ab.

letzte Änderung	11-Mai-2000
überprüft am	30-Mai-2000

[28] `http://www.cert.dfn.de/infoserv/dsb/dsb-9405.html`

4.10 NIS: Network Information System

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
keine Authentisierung	eigenen NIS-Server installieren	ja (selten)		sehr hoch
	Modifikation der Antworten	ja		hoch

Aufgaben

Das „Network Information System“ NIS wurde eingeführt, um wichtige Informationen einer immer größeren Anzahl vernetzter Rechner zentral verwalten zu können. Dazu gehören Informationen wie die Zuordnung von Rechnernamen zu Internetadressen, die symbolischen Namen der Portnummern, Paßworte der Benutzer oder Mail-Adressen.

Funktionsweise

Ein sogenannter „Master-Server“ übernimmt dabei die Verwaltung der o. g. Informationen. Damit die Rechner in einem Netzwerk bei dem Ausfall des Master-Servers auch weiterhin auf die zentral verwalteten Daten zugreifen können, kann man weitere Rechner damit beauftragen, die Informationen des Master-Servers zu spiegeln und ihrerseits Anfragen von anderen Rechnern bzgl. dieser Informationen zu beantworten. Diese Rechner werden dann „Slave-Server“ genannt.

Schwachpunkte

Grundsätzliche Schwachpunkte

Ein grundsätzliches Problem stellt die fehlende Authentisierung des Servers gegenüber den Klienten und umgekehrt dar. Damit ist ein Angreifer in der Lage, von jedem Rechner aus Anfragen an den Server zu stellen oder Klienten-Anfragen zu beantworten.

Schwachpunkte des Protokolls:

- **keine Authentisierung der Anfragen und Antworten**
- **unverschlüsselte Datenübertragung**

Bedrohung:

- Fälschen von NIS-Informationen
- Zugang zu NIS-Daten nur schwach geschützt

Angriffsmöglichkeiten:

- Installation eines eigenen NIS-Servers
- Fälschen einzelner NIS-Anfragen
- Abfrage sämtlicher NIS-Daten

Bekannte Angriffe

Der häufigste Angriff auf NIS-Systeme ist derzeit noch der „passive“ Angriff. Dabei werden von einem Angreifer die NIS-Daten (insbesondere die Paßwortdaten) abgefragt, um diese später zum Erraten von Paßworten zu nutzen. Seltener (aber auch schwieriger zu entdecken) werden aktiv NIS-Antworten gefälscht, um sich direkt einen Zugang zu einem anderen Rechner zu verschaffen.

Weitere Angriffe und Absicherungsmöglichkeiten sind in dem DFN-Bericht Nr. 81[29] im Artikel *Angriffe auf NIS-Systeme und Möglichkeiten einer kryptographischen Absicherung*[30] zu finden.

Absicherungsmaßnahmen

Statt NIS sollte ein sichereres System verwendet werden, das eine starke Authentisierung ermöglicht. Dieses kann z. B. durch NIS+ (verwendet secure RPC) oder Kerberos geschehen.

Sollte NIS trotzdem verwendet werden müssen, so sollte der Zugriff auf die NIS-Daten abgesichert werden. Hierzu kann auf dem NIS-Server in der Datei `/var/yp/securenets` eingestellt werden, welche Subnetze Zugang zu den NIS-Informationen haben sollen.

Einschränkungen

Die Alternativen wie NIS+ sind leider noch nicht für alle Betriebssysteme verfügbar.

[29] `http://www.cert.dfn.de/dfn/berichte/db081/`
[30] `http://www.cert.dfn.de/dfn/berichte/db081/nis/`

Fazit

NIS bietet keine Authentisierung und stellt ein sehr unsicheres Protokoll dar. Die NIS-Schwachstellen können sowohl zum unberechtigten Zugriff über das Netzwerk als auch zum Erlangen von SuperUser-Privilegien ausgenutzt werden.

letzte Änderung	23-Mai-2000
überprüft am	30-Mai-2000

4.11 RPC: Remote Procedure Calls

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
schwache Authentisierung	Umgehung von Zugriffsrechten	ja (selten)	secure RPC	mittel
fehlerhaftes Verwaltungsprogramm	Umgehung von Zugriffsrechten	ja	sichereres Verwaltungsprogramm und Zugangskontrolle	hoch

Aufgaben

Das RPC Protokoll ermöglicht es Programmen, bestimmte Unterprogramme auf anderen Rechnern ausführen zu lassen. Dieses ist ein beliebtes Paradigma für Client-Server Anwendungen. Beispiele für solche Anwendungen sind NFS (siehe Kapitel 4.9) und NIS (siehe Kapitel 4.10).

Funktionsweise

Der Server registriert sich und das Port, auf dem er den Service anbietet (UDP und/oder TCP) bei einem Verwaltungsprogramm („portmap“ bzw. „rpcbind“). Klienten können die entsprechende Portnummer von dem Verwaltungsprogramm auf Port 111 sowohl über UDP als auch über TCP erfragen und dann Kontakt zu diesem Port aufnehmen. Alternativ können sie ihre Anfragen von dem Verwaltungsprogramm an den Server weiterleiten lassen und bekommen dann die Antwort auch vom Verwaltungsprogramm zurück.

Schwachpunkte

Grundsätzliche Schwachpunkte

Das Protokoll erlaubt zwar starke Authentisierungen, diese sind jedoch normalerweise nicht implementiert. Die meisten weit verbreiteten Implementationen kennen Zugriff ohne Authentisierung und die schwache Authentisierung über IP-Adressen.

Schwachpunkte des Protokolls:

- **schwache Authentisierung**
- **fehlerhaftes Verwaltungsprogramm**

Bedrohung:

- Umgehung von Zugriffsrechten

Angriffsmöglichkeiten:

- IP-Spoofing
- Mißbrauch des Verwaltungsprogramms

Bekannte Angriffe

Durch IP-Spoofing können Server dazu gebracht werden, Aktionen auszuführen, die nur bestimmte Maschinen auslösen dürfen.

Das Verwaltungsprogramm leitet Aufträge an Server auf der lokalen Maschine weiter. Für den Server sieht das dann so aus, als ob die Anfrage von der lokalen Maschine käme, die oft mehr darf, als der Angreifer. Auf diese Art kann ein Angreifer alles tun, was ein Benutzer der lokalen Maschine via RPC tun darf.

Das Verwaltungsprogramm liefert auch eine Liste aller registrierten Services und deren Portnummern an jeden Anfrager. Da viele Services sicherheitsrelevant sind, sollte auch diese Information als sicherheitsrelevant und vertraulich betrachtet werden.

Absicherungsmaßnahmen

Statt der üblichen (schwachen) Authentisierung sollte eine sicherere Authentisierung mit Hilfe eines kryptographischen Verfahrens verwendet werden. Dieses ist unter dem Stichwort „secure RPC“ verfügbar und beruht auf einem Mix von Public- und Private-Key Verfahren. Alternativ kann Kerberos zur Absicherung verwendet werden.

Der Zugriff zu dem rpcbind/portmapper Dienst sollte eingeschränkt werden. Dieses kann entweder durch einen Filter am Firewall auf Port 111 (UDP und TCP) oder durch den Einsatz eines Ersatzprogrammes, das eine Zugriffskontrolle ermöglicht (siehe Abschnitt A.3), erfolgen.

Einschränkungen

„Secure RPC“ ist nicht überall verfügbar. Es erfordert auch einen relativ hohen administrativen Aufwand. Die Kerberos-Variante ist noch weniger verfügbar.

Fazit

RPC bietet nur eine sehr schwache Zugangskontrolle. Es sollten auf jeden Fall sicherere Authentisierungsprotokolle, wie z. B. „secure RPC“ oder Kerberos, verwendet werden. Ob und welche dieser Optionen unterstützt werden, hängt vom verwendeten Betriebssystem ab.

letzte Änderung	04-Mai-2000
überprüft am	30-Mai-2000

4.12 DNS: Domain Name Service

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Keine Authentisierung	Einfügen falscher Informationen	ja	Secure DNS	mittel
Unverschlüsselte Datenübertragung	Abhören	selten		
ungesicherter Verbindungskontext	TCP-Hijacking	selten		

Aufgaben

Das DNS Protokoll dient der Zuordnung von IP-Adressen zu Domainnamen und umgekehrt.

Funktionsweise

Der Domain-Namensraum ist ausgehend von den sogenannten „Root Nameservern“ hierarchisch gegliedert. Die IP-Adressen der „Root Nameserver“ sind allgemein bekannt. Um einen Namen einer IP-Adresse zuzuordnen, wird ein „Root Nameserver“ nach den IP-Adressen der Nameserver der entsprechenden Top-Level Domain gefragt, diese dann rekursiv wiederum nach den Nameservern der nächsten Hierarchiestufe, bis die für den Namen zuständigen Nameserver gefunden wurden. Diese liefern dann die gewünschte IP-Adresse.

Um einer IP-Adresse einen Namen zuzuordnen, wird aus der Adresse nach einem einfachen Algorithmus ein spezieller Domainname erzeugt. Für diesen Namen wird der zuständige Nameserver gesucht, der dann den Namen zu der IP-Adresse liefert.

Schwachpunkte

Grundsätzliche Schwachpunkte

Dieses sehr wichtige Protokoll benutzt keinerlei Authentisierung. Antworten von Nameservern wird blind vertraut.

Schwachpunkte des Protokolls:

- **keine Authentisierung**
- **Unverschlüsselte Datenübertragung**
- **ungesicherter Verbindungskontext**

Bedrohung:

- Verlust der Integrität / Unbefugte Manipulation
- Verlust der Vertraulichkeit

Angriffsmöglichkeiten:

- DNS-Spoofing
- Sniffer

Bekannte Angriffe

Es gibt zwei bekannte Möglichkeiten, den DNS-Prozess zu manipulieren. Bei der direkteren Methode sendet der Angreifer von einem regulären Nameserver aus falsche Informationen. Diese Informationen können sich auf die Anfrage beziehen, können sich aber auch auf ganz andere Namen beziehen. Oft werden diese zusätzlichen Informationen einfach geglaubt. Der Angreifer kann diesen Angriff also sowohl von einem Nameserver aus starten, den er legitimerweise betreibt, als auch von einem Nameserver, auf den er sich unerlaubterweise Zugriff verschafft hat.

Bei der indirekteren Methode wird die Registrierungsstelle dazu gebracht, die Einträge für die Nameserver einer Domain zu ändern, so daß sie jetzt auf die Nameserver des Angreifers zeigen.

In beiden Fällen gelingt es dem Angreifer, Verbindungen, die zu der Domain gehen sollten, auf einen seiner Rechner umzuleiten. Dabei will er entweder den Inhaber der Domain schädigen, indem er z. B. seine Webseite entstellt, oder er will denjenigen schädigen, der die Verbindung aufbaut. Dieser glaubt ja, mit einem vertrauenswürdigen Server zu kommunizieren.

Durch Abhören von DNS-Verkehr können Kommunikationsprofile erstellt werden.

Absicherungsmaßnahmen

Es gibt Erweiterungen für DNS, die eine kryptographisch sichere Authentifizierung erlauben. Diese sind in den RFCs 2535 bis 2539 beschrieben [RFC 2535, RFC 2536, RFC 2537, RFC 2538, RFC 2539]. Neuere Implementationen der „bind“ Nameserversoftware unterstützen diese Erweiterungen.

Einschränkungen

Die Erweiterungen befinden sich derzeit nicht im breiten Einsatz. Ohne eine breite Grundlage, können sie leider nur minimale Verbesserungen bewirken.

Fazit

Es wird dringend Zeit, dieses wichtige Protokoll abzusichern. Die RFCs haben die nötige Grundlage geliefert, jetzt sollten sie zügig umgesetzt werden.

letzte Änderung	03-Mai-2000
überprüft am	30-Mai-2000

4.13 X11

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Schwache Authentisierung	X-Anwendungen	ja	sichere Konfiguration	sehr hoch
Übertragung unverschlüsselter Paßworte	Sniffer	ja (selten)	SSH	hoch
ungesicherter Verbindungskontext	TCP-Hijacking	nein		

Aufgaben

X11 ist ein Windowsystem, das unter Unix und anderen Betriebssystemen funktioniert. Es gibt auch X11-Software für MS-Windows und spezielle X11-Terminals. X11 Anwendungen können auf beliebigen Rechnern im Netzwerk laufen und ihre (auch graphischen) Ein- und Ausgaben auf dem lokalen Arbeitsplatz tätigen.

Funktionsweise

Auf dem lokalen Arbeitsplatz läuft ein X11-Server. Dieser wartet an einer oder mehreren Stellen auf Verbindungen. Bei Unix-Systemen wartet der Server auf Verbindungen über einen Unix-Socket und über ein TCP-Port. Weiterhin wird gegebenenfalls auch DECnet unterstützt.

Klienten bzw. Anwendungen nehmen über einen dieser Wege Kontakt mit dem Server auf. Für die Authentisierung der Klienten gibt es mehrere Möglichkeiten. Möglich sind prinzipiell Verbindungen ohne Authentisierung, mit schwacher Authentisierung über IP-Adressen, mit einem wiederverwendbaren Klartextpasswort („MIT-MAGIC-COOKIE“), sowie mit Authentisierung basierend auf „secure RPC“ oder auf Kerberos.

Schwachpunkte

Grundsätzliche Schwachpunkte

Die Entwickler von X11 sind davon ausgegangen, daß die Sicherheit in die Transportschicht gehört. Deshalb haben sie die Sicherheit nicht in die Anwendungsschicht integriert.

Schwachpunkte des Protokolls:

- **schwache Authentisierung**
- **Übertragung unverschlüsselter Paßworte**

Bedrohung:

- unautorisierter Zugang
- Abhören

Angriffsmöglichkeiten:

- Einfaches Verbinden mit beliebigen Anwendungen
- IP-Spoofing
- Sniffer

Bekannte Angriffe

Es gibt Programme, die alle Tastatureingaben protokollieren, die an einem X11-Server gemacht werden, Programme, die den Bildschirminhalt kopieren, sowie Programme, die den Server unbrauchbar machen. Alle diese Programme setzen lediglich voraus, daß sie eine Verbindung zum Server herstellen können.

Absicherungsmaßnahmen

Zunächst muß sichergestellt werden, daß der X-Server lediglich Verbindungen akzeptiert, die authentisiert sind. Da die Authentisierung mittels „secure RPC“ und Kerberos nicht sehr weit verbreitet ist, wird hier von einer Authentisierung mittels „MIT-MAGIC-COOKIE“ ausgegangen. Unter Unix startet man dazu den Server mit dem Parameter „-auth <authfilename>“. In dem File „authfilename“ muß sich bereits ein Cookie für diesen Server befinden. Es empfiehlt sich dringend, vor jedem Start des Servers ein neues Cookie zu generieren (z. B. mit dem Programm „mkcookie“). Keinesfalls darf mit „xhost +<Rechnername>“ die Authentisierung mittels Cookies umgangen werden.

Da die Cookies vom Protokoll im Klartext über das Netz übertragen werden, müssen Verbindungen über das Netz verschlüsselt werden. Hierfür eignet sich SSH sehr gut, da es über einen eingebauten Mechanismus für das Tunneln von X11-Verbindungen durch SSH-Verbindungen verfügt.

Einschränkungen

Selbst beim Einsatz von SSH kann der Cookie von jedem mißbraucht werden, der in der Lage ist, die Datei zu lesen, in der er gespeichert ist. Das betrifft Angreifer mit (zu Unrecht erworbenen) administrativen Rechten, aber möglicherweise auch Angreifer, die Schwächen bei der Speicherung des Files ausnutzen können (z. B. durch Abhören des NFS-Verkehrs).

Fazit

X11 ist mit etwas Aufwand relativ sicher zu konfigurieren. Leider kann die Sicherheit durch kleine Fehler schnell wieder zusammenbrechen, ohne daß der Benutzer es merkt. Eine neue Protokollversion mit inherenter Sicherheit wäre wünschenswert.

letzte Änderung	11-Mai-2000
überprüft am	30-Mai-2000

4.14 Sendmail

Schwachpunkt	Angriff	ausgebeutet	Absicherung	Priorität
Verbindungsaufbau ohne Authentisierung	Spam	ja (sehr häufig)	sichere Konfiguration	sehr hoch
Keine Authentisierung auf Anwendungsebene	gefälschte Mail	ja (selten)	Kryptographie	hoch
Übertragung unverschlüsselter Mails	Sniffer	häufig	Kryptographie	hoch
ungesicherter Verbindungskontext	TCP-Hijacking	nein		gering

Aufgaben

Unter Unix ist „sendmail“ der am weitesten verbreitete „Mail Transfer Agent“. Er akzeptiert Emails von Benutzern und von anderen „Mail Transfer Agenten“ und leitet diese entweder an lokale Mailboxen oder an andere „Mail Transfer Agenten“ weiter.

Funktionsweise

Es gibt zwei Wege, auf denen „sendmail“ Emails erhalten kann. Die Mail kann auf dem lokalen System via „stdin“ von einem Benutzer kommen. In diesem Fall kann über die Kommandozeile zusätzlich der Envelope geändert werden.

Alternativ kann die Mail von einem anderen Rechner über das Netzwerk kommen. Zu diesem Zweck läuft „sendmail“ als Daemon und wartet auf dem SMTP-Port auf Verbindungen. Bei diesen Verbindungen wird zunächst der Envelope und dann die eigentliche Mail übertragen.

Unabhängig davon, wie die Mail empfangen wurde, gibt es zwei Möglichkeiten, die Mail weiterzuleiten. Bei lokalen Empfängern wird die Mail in der Regel an ein externes Programm übergeben, das die eigentliche Auslieferung übernimmt.

Bei entfernten Empfängern wird über den DNS Mechanismus der Zielrechner einer Mail identifiziert oder es wird ein System identifiziert, das bereit ist, Mail für den Zielrechner entgegenzunehmen. Zu dem SMTP-Port des identifizierten Systems wird dann eine TCP-Verbindung aufgebaut, über die die Mail mit dem selben Protokoll übertragen wird, mit dem „sendmail“ auch Mail von außen empfängt.

Schwachpunkte

Grundsätzliche Schwachpunkte

„Sendmail“ ist ein sehr komplexes Programm, das leider administrative Privilegien benötigt, um seine Aufgaben zu erfüllen. Es hat in seiner Geschichte diverse Probleme gehabt, die zu Kompromittierungen sowohl duch lokale Benutzer, als auch über das Netz führten. Es ist deshalb dringend zu empfehlen, die jeweils neueste Version des Programms zu benutzen.

Schwachpunkte:

- **Verbindungsaufbau ohne Authentisierung**
- **Keine Authentisierung auf Anwendungsebene**
- **Übertragung unverschlüsselter Mails**
- **ungesicherter Verbindungskontext**

Bedrohung:

- Spam
- Verlust der Integrität / Unbefugte Manipulation
- Verlust der Vertraulichkeit

Angriffsmöglichkeiten:

- `telnet` auf Port 25
- Jeder Mailclient
- Sniffer

Bekannte Angriffe

Immer wieder werden Emails mit gefälschten Absenderadressen verschickt (sog. „Spam“), um beispielsweise aggressiv Werbung zu betreiben. Das Ablauschen von sensitiven Daten (z.B. Kreditkartennummern) einer Email stellt aus heutiger Sicht ein viel kleineres Problem dar, welches aber in Zukunft stark an Bedeutung gewinnen wird.

Absicherungsmaßnahmen

Gegen den Empfang von Spam kann man sich fast nicht wehren, aber auf jeden Fall sollte man verhindern, daß man Spam weiterleitet (sog. „relaying“). Dazu benutzt man in der M4-Konfiguration von „sendmail“ die Features „access_db“ und „rbl“. Mit „access_db“ kann man angeben, für wen Mail weitergeleitet wird und mit „rbl“ kann man Mail von bekannten Übeltätern ablehnen.

Fast alle anderen in diesem Zusammenhang stehenden Schwachstellen lassen sich durch den Einsatz von Verschlüsselungssoftware erheblich eingrenzen, teilweise sogar komplett beseitigen. Beispielhaft sei hier das Programm “PGP” (Pretty Good Privacy) genannt, das momentan im Internet einen großen Stellenwert hat. In Zukunft wird wahrscheinlich „Secure Mime“ an Bedeutung gewinnen.

Aufgrund des Umfangs und der Bedeutung kryptographischer Verfahren werden diese in Kapitel 6 eingehend beschrieben.

Einschränkungen

Es ist mit technischen Mitteln nicht zu verhindern, daß die eigenen Benutzer „Spam“ verschicken.

Zu den Einschränkungen der kryptographischen Methoden siehe Kapitel 4.8.

Fazit

„Sendmail“ ist das unter Unix am weitesten verbreitete Programm zur Mailweiterleitung, weil es sehr flexibel ist. In der gegenwärtigen Version sind keine Schwachstellen mehr bekannt, die über die generellen Schwachstellen des zugrunde liegenden Protokolls hinausgehen. In besonders kritischen Bereichen sollte man jedoch über den Einsatz eines einfacheren und damit überschaubareren Programmes nachdenken oder auf den Quellcode zugreifen können.

letzte Änderung	26-Apr-2000
überprüft am	30-Mai-2000

Kapitel 5

Firewalls

Firewalls sind Zugriffskontrollmechanismen in Netzen. Sie werden eingesetzt, um eine Trennung der Netze nach Sicherheitsgesichtspunkten zu erreichen. Üblicherweise wird durch einen Firewall ein internes, sicherheitskritisches Netz vor Angriffen aus einem externen Netz geschützt.[1] Der Firewall setzt vorher spezifizierte Sicherheitsrichtlinien durch. Dem externen Netz gegenüber stellt der Firewall nur eine kleine Anzahl gut gesicherter und streng überwachter Dienste zur Verfügung. Damit der Firewall die gesamte Kommunikation zwischen internem und externem Netz überwachen kann, muß durch organisatorische Maßnahmen dafür gesorgt werden, daß der Firewall den alleinigen Übergang zwischen beiden Netzen bildet. Die Kommunikation zwischen Systemen innerhalb des internen Netzes sowie zwischen Systemen im externen Netz wird durch den Firewall nicht kontrolliert.

Firewall: *"Ein Firewall ist eine Schwelle zwischen zwei Netzen, die überwunden werden muß, um Systeme im jeweils anderen Netz zu erreichen. Es wird dafür gesorgt, daß jede Kommunikation zwischen den beiden Netzen über den Firewall geführt werden muß. Auf dem Firewall sorgen Zugriffskontrolle und Audit dafür, daß das Prinzip der geringsten Berechtigung durchgesetzt wird und potentielle Angriffe schnellstmöglich erkannt werden."* [Ellermann 1993]

Zunächst wurden Firewalls primär zur sicheren Anbindung von Unternehmensnetzen an das (öffentliche) Internet eingesetzt.[2] Neben diesen "Internet-Firewalls" ist heute die Absicherung besonders sensitiver Abteilungen innerhalb eines Unternehmensnetzes ein wichtiger Einsatzbereich von Firewalls ("Intranet-Firewalls"). Diese beiden Einsatzbereiche unterscheiden sich durch:

Kommunikationsanforderungen: Zwischen verschiedenen Teilen eines Unternehmensnetzes bestehen in der Regel engere Kommunikationsbeziehungen als mit dem Internet. So ist meist der Zugriff vom abgesicherten Netz auf Datenbanken im Unternehmensnetz notwendig.

Da das durch einen Intranet-Firewall abgetrennte Netz oft nur wenige Systeme umfaßt, ist der Aufwand zur Absicherung selten genutzter Dienste kaum zu rechtfertigen. Die Liste der zu unterstützenden Protokolle wird sich daher für Internet- und Intranet-Firewalls unterscheiden.

Sicherheitsanforderungen: Intranet-Firewalls sind einer höheren Bedrohung ausgesetzt, da die zu schützenden Systeme besonders kritische Informationen enthalten. Auch der Kreis der potentiellen Angreifer unterscheidet sich, da primär mit

[1] Für das durch den Firewall geschützte Netz wird im folgenden die Bezeichnung "*internes Netz*" verwendet. Das Netz auf der anderen Seite des Firewalls wird als "*externes Netz*" bezeichnet.

[2] Praktisch alle Firewalls werden derzeit für TCP/IP-Protokolle eingesetzt. Eine Anwendung des Firewall-Konzeptes ist prinzipiell aber auch bei anderen Protokollen (ATM, DECNet, ISO-OSI, ...) möglich.

Angriffen durch firmeninterne Mitarbeiter gerechnet wird. Externe Angreifer werden bereits durch den Internet-Firewall erkannt und aufgehalten.

Performanzanforderungen: Hochgeschwindigkeitsnetze werden innerhalb von Unternehmen früher eingesetzt als bei der Internet-Anbindung. Intranet-Firewalls bilden daher früher einen Engpaß.

Für den Aufbau eines Firewalls wird eine Reihe von Komponenten benötigt, die exakt aufeinander abgestimmt sein müssen. Die wichtigsten Firewall-Komponenten — Packet Screen, Proxy-Server und Anwendungs-Gateway — sollen im folgenden Abschnitt 5.1 vorgestellt werden. Die wichtigsten Firewall-Architekturen, die sich aus den vorgestellten Firewall-Komponenten aufbauen lassen, werden im Abschnitt 5.2 diskutiert. Auf die Vor- und Nachteile jeder Firewall-Architektur wird dabei explizit hingewiesen.

5.1 Firewall-Komponenten

Firewall-Komponenten sind Mechanismen, aus denen Firewalls aufgebaut werden. Die Mechanismen werden auf unterschiedlichen Ebenen realisiert. Die erreichte Granularität der Zugriffskontrolle hängt von den auf der jeweiligen Ebene zur Verfügung stehenden Informationen ab. Die folgende Tabelle zeigt im Überblick die Eigenschaften der in den nächsten Abschnitten vorgestellten Firewall-Komponenten.

Firewall-Komponente	Granularität	Vorteile	Nachteile
Packet Screen	IP-Adressen, Portnummern	Transparenz, hohe Performanz, geringer Installationsaufwand, niedrige Kosten	geringe Granularität der Zugriffskontrolle, kein Audit, mangelhafte Kontrolle freigegebener Ports, Überprüfung der Korrektheit der Filterregeln kaum möglich
Proxy	IP-Adressen, Dienst, Benutzer, Kontext	feine Granularität der Zugriffskontrolle, Authentisierung möglich	keine Transparenz, Performanzprobleme möglich, hoher Installationsaufwand
Anwendungs-Gateway	IP-Adressen, Dienst, Benutzer, Kontext	feine Granularität der Zugriffskontrolle, Verdecken interner Struktur möglich	keine Transparenz, Performanzprobleme möglich, hoher Installationsaufwand, keine Benutzerinteraktion für Authentisierung möglich

5.1.1 Packet Screen

Packet Screens führen eine Zugriffskontrolle auf IP-Datagrammen durch. Sie gewähren bzw. unterbinden die Kommunikation zwischen mindestens zwei Netzen, indem sie nur die Datagramme weiterleiten, deren Protokoll-Header auf Vermittlungs- und Transportschicht den vom Verwalter konfigurierten Filterregeln genügen.[3] Einen Überblick über Header-Informationen, die von einer Packet Screen ausgewertet werden können, bietet Abbildung 5.1.

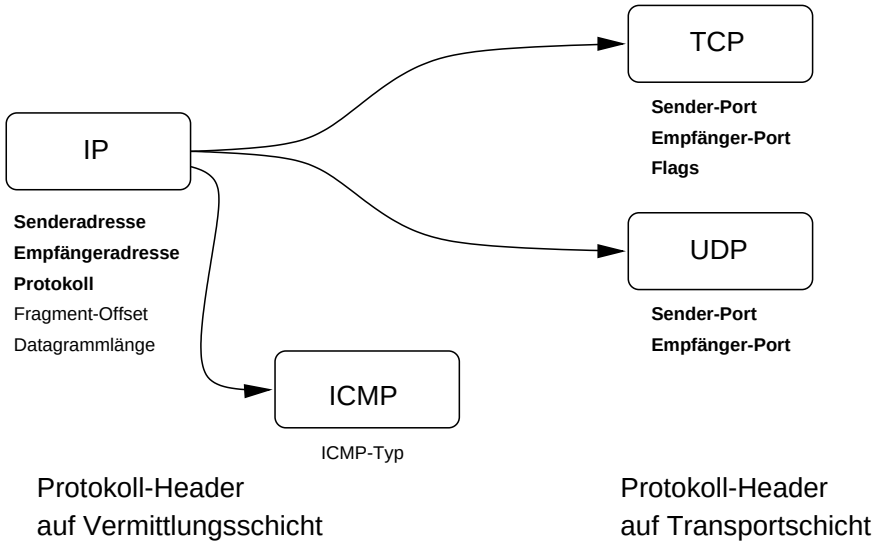


Abbildung 5.1: Von Packet Screens ausgewertete Informationen

Über die Sender- und Empfänger-Adressen des IP-Headers ist eine Beschränkung der Endsysteme möglich, die miteinander kommunizieren dürfen. Dadurch kann zum Beispiel der Zugriff auf besonders kritische Systeme verwehrt werden. Das Protokoll-Feld des IP-Headers gibt an, zu welchem Protokoll die enthaltene Dateneinheit gehört (TCP, UDP, ICMP oder andere). Fragment-Offset und Datagrammlänge — in Abbildung 5.1 schwach gedruckt — werden nur von einigen Packet Screens überprüft, sind aber zur Erkennung von Fragmentierungsangriffen notwendig.[4] Über Sender- und Empfänger-Ports der Protokoll-Header auf der Transportschicht können die zugegriffenen Dienste

[3] Häufig werden die Begriffe Paket-Filter (“Packet Filter”) und Packet Screen fälschlicherweise synonym verwendet. Ein Paket-Filter ist ein Filter in Endsystemen, mit dem Applikationen wie `tcpdump` empfangene Pakete nach bestimmten Merkmalen selektieren können.

[4] Bei fragmentierten Dateneinheiten sind die Transport-Header nur im ersten Fragment enthalten, so daß Packet Screens folgende Fragmente (Offset > 0) ohne vollständige Überprüfung weiterleiten können. Durch nachfolgende Fragmente mit einem kleinen Offset können beim Empfänger Teile des im ersten Fragment enthaltenen Transport-Headers überschrieben werden. So können beispielsweise andere Portnummern angewählt werden, als im ersten Fragment angegeben.

identifiziert werden. Die Identifikation der Dienste ist jedoch nicht besonders zuverlässig, da durch Manipulation der Endsysteme auch nicht standardisierte Ports verwendet werden können und zudem für RPC-basierte Protokolle keine feste Zuordnung der Dienste zu bestimmten Portnummern existiert. Über die Auswertung des SYN-Flags im TCP-Header kann zwischen der ersten Dateneinheit einer neuen Verbindung und den nachfolgenden Dateneinheiten unterschieden werden. Die Auswertung des SYN-Flags ermöglicht es der Packet Screen, die Richtung eines Verbindungsaufbaus zu erkennen. Da die Übertragung einer Dateneinheit mit SYN-Flag für den Aufbau einer TCP-Verbindung erforderlich ist, kann unter der Annahme, daß die Dateneinheit mit SYN-Flag von der Packet Screen bereits überprüft wurde, die Prüfung der nachfolgenden Dateneinheiten (ohne SYN-Flag) entsprechend einfacher ausfallen. Die meisten Packet Screens unterstützen keine Filterung von ICMP-Dateneinheiten anhand des ICMP-Typs (Echo, “Destination Unreachable”, “Source-Quench”, “Timestamp”, “Redirect”, ...). Meist kann nur über die Auswertung des Protokoll-Feldes im IP-Header die Übertragung von ICMP-Nachrichten insgesamt unterbunden werden. Dies ist jedoch wenig empfehlenswert, da ICMP wichtige Kontroll-Informationen für die ordnungsgemäße Verarbeitung von IP-Dateneinheiten transportiert.[5]

Dynamische Packet Screens: Im Gegensatz zu den bisher beschriebenen “*Statischen Packet Screens*”, bei denen jede eingehende Dateneinheit nach einem festgelegten Satz von Filterregeln unabhängig von bereits verarbeiteten Dateneinheiten ausgewertet wird, kann bei “*Dynamischen Packet Screens*”[6] die Filterentscheidung vom Kontext bereits empfangener Dateneinheiten abhängen.

Die Möglichkeit, Filterentscheidungen anhand des Kontextes bereits empfangener Dateneinheiten durchführen zu können, hat im Wesentlichen zwei Vorteile:

- Da bei UDP kein Verbindungsaufbau durchgeführt wird, kann die Richtung des Zugriffs nicht wie bei TCP über das SYN-Flag festgestellt werden. Ohne Kenntnis vorangegangener Dateneinheiten kann nicht entschieden werden, ob es sich bei einem empfangenen UDP-Datagramm um eine Reaktion (Antwort) oder um einen neuen Zugriff handelt. Bei einer “Dynamischen Packet Screen” wird eine Filterregel ergänzt, die Antworten auf übertragene UDP-Datagramme ebenfalls passieren läßt. Nach längerer Inaktivität dieser Kommunikationsbeziehung wird die Filterregel wieder entfernt.

- FTP stellt für Packet Screens ein Problem dar, weil Kontroll- und Datenverbindung von unterschiedlicher Richtung aus geöffnet werden. “Dynamische Packet

[5] Über ICMP kann einem Sender mitgeteilt werden, daß der Empfänger nicht erreichbar ist. Ohne ICMP kann der Sender bei verbindungsorientierten Protokollen erst nach einem mehrere Minuten betragenden “Timeout” erkennen, daß seine Dateneinheiten nicht bestätigt werden. Bei verbindungslosen Protokollen ist keine Erkennung möglich.

[6] Gelegentlich wird auch die Bezeichnung “Stateful Packetfilter” verwendet.

Screens" können das Zulassen einer FTP-Datenverbindung davon abhängig machen, ob gleichzeitig auch eine Kontrollverbindung zwischen den beiden Systemen existiert.

Granularität: Die Granularität der Zugriffskontrolle ist bei Packet Screens auf die im Protokoll-Header enthaltenen Informationen beschränkt. Somit kann der Zugriff nur anhand der Host-Adressen (**IP-Adresse**) und des zugegriffenen Dienstes (**Port-Nummer**) kontrolliert werden. Eine Authentisierung von Benutzern[7] oder eine Kontrolle der übertragenen Daten ist nicht möglich.

Vorteile einer Packet Screen:

Transparenz: Die Überprüfung von Dateneinheiten kann in der Packet Screen ohne Interaktion mit dem Benutzer erfolgen. Es sind auch keine Konfigurationsänderungen auf den Endsystemen notwendig.

Performanz: Der Filter-Aufwand der Packet Screen ist relativ gering, so daß eine gute Performanz erreicht wird. Die Performanzauswirkungen der Packet Screen werden vom Benutzer nicht bemerkt.[8]

Geringer Installationsaufwand: Die Packet Screen Funktionalität ist in heutigen Routern meist schon vorhanden, so daß nur die Definition geeigneter Filterregeln notwendig ist. Die Konfiguration erfordert nur einen geringen Zeitaufwand.

Kostengünstig: Da Packet Screens meist im Router schon verfügbar sind, ist dank des geringen Zeitaufwands zur Konfiguration der Filterregeln, der Einsatz einer Packet Screen relativ kostengünstig. Wenn kein vorhandener Router als Packet Screen verwendet werden kann, bietet frei verfügbare Packet Screen Software (z. B. "Drawbridge"[9] oder "IP-Filter"[10]), die auf einfacher PC-Hardware eingesetzt werden kann, eine ebenfalls kostengünstige Alternative. Kommerzielle Lösungen wie die dynamische Packet Screen "Firewall-1" sind deutlich teurer.

Nachteile einer Packet Screen:

[7] Eine Authentisierung von Benutzern wäre zukünftig mit Hilfe von IPSEC denkbar [Ellermann 1996], die Verfügbarkeit von entsprechenden Packet Screen Implementationen läßt sich heute noch nicht absehen.

[8] In Hochgeschwindigkeitsnetzen kann eine Packet Screen jedoch zum Engpaß werden, wie die Analysen in [Ellermann & Benecke 1998] zeigen.

[9] `ftp://ftp.cert.dfn.de/pub/firewalls/software/drawbridge/`

[10] `ftp://coombs.anu.edu.au/pub/net/ip-filter/`

Granularität der Zugriffskontrolle: Die Zugriffskontrolle kann bei Packet Screens nur anhand von Informationen im Protokoll-Header erfolgen.

Schlechtes Audit: Die in gebräuchlichen Routern enthaltenen Packet Screens können nur sehr rudimentäre Audit-Daten aufzeichnen. Über das Internet verbreitete Packet Screen Software ("Drawbridge" und "IP-Filter") sowie kommerzielle Lösungen (z. B. "Firewall-1") erlauben eine detailliertere Aufzeichnung von Auditdaten, jedoch stehen konzeptbedingt bei Packet Screens nur geringe Informationen für das Audit zur Verfügung. (vgl. "Granularität")

Kontrolle der freigegebenen Ports: Eine Packet Screen kann nicht kontrollieren, ob an einer freigegebenen Portnummer tatsächlich der vorgesehene Dienst angeboten wird. Durch einfache Manipulationen an den Endsystemen kann eine Packet Screen leicht umgangen werden.

Korrektheit erstellter Filterregeln: Die korrekte Konfiguration der Filterregeln erfordert größte Sorgfalt, da bereits kleine Fehler zu einem nicht erwünschten Verhalten führen können. Bisher bieten nur wenige Packet Screens dem Verwalter eine Unterstützung bei der Konfigurierung an. Wünschenswert wäre auch eine automatische Überprüfung neu erstellter Filterregeln.

5.1.2 Proxy-Server

Ein Proxy-Server — kurz Proxy (Stellvertreter) — ist ein Prozeß auf Anwendungsebene, der Anfragen für einen Clienten an einen Server stellt und die Antworten des Servers an den Clienten weiterleitet. Bei dem Einsatz von Proxies in Firewall-Konzepten wird durch zusätzliche Maßnahmen die direkte Kommunikation zwischen Client und Server unterbunden, so daß die Kommunikation nur über einen Proxy möglich ist. Da der Proxy Zugriff auf die Anfragen des Clienten und auf die Antworten des Servers erhält, wird eine sehr gute Zugriffskontrolle möglich. Der konzeptionelle Aufbau eines Proxies ist in Abbildung 5.2 wiedergegeben.

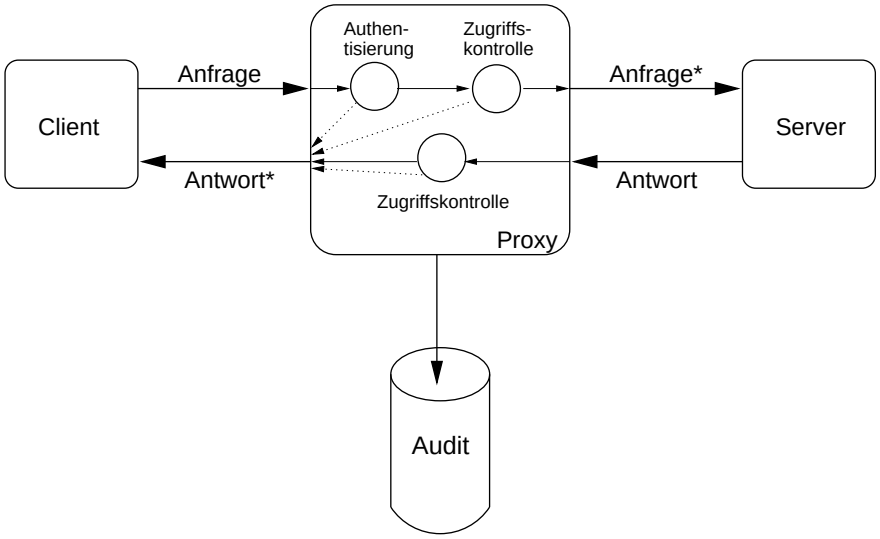


Abbildung 5.2: Konzeptionelle Sicht auf einen Proxy

Eine Anfrage eines Clienten muß vom Proxy zunächst authentisiert werden. Hierfür werden kryptographische Verfahren, Einmal-Paßworte, aber auch unsichere Verfahren wie die Überprüfung der IP-Absender-Adresse eingesetzt. Je nach Sicherheitsanforderungen kann die Authentisierung einmalig beim Verbindungsaufbau oder wiederholt für jede Anfrage durchgeführt werden. Bei der anschließenden Zugriffskontrolle wird überprüft, ob der Zugriff des Clienten den spezifizierten Sicherheitsrichtlinien entspricht. Der Zugriff kann für bestimmte Server, Dienste oder auch bestimmte Uhrzeiten eingeschränkt sein. Es ist auch möglich, den Zugriff auf ausgewählte Bereiche eines Servers zu beschränken. Die vom Proxy an den Server gesendete $Anfrage^*$ kann sich daher durchaus von der ursprünglichen Anfrage des Clienten unterscheiden. Die vom Server zurückgesendeten Daten werden vom Proxy ebenfalls zunächst überprüft und gegebenenfalls modifiziert ($Antwort^*$), bevor sie an den Clienten weitergeleitet werden. Bei der Überprüfung der Antwort kommen hauptsächlich Scanner für ausführbare Inhalte (Java-Applets, Java-Script, Makros usw.) zum Einsatz. Alle

drei Mechanismen können Zugriffe verweigern und sollten in einem derartigen Falle dem Clienten dieses über eine Fehlermeldung mitteilen (gestrichelte Pfeile). Das Audit protokolliert normale und verweigerte Zugriffe sowie gegebenenfalls weitere Vorkommnisse. Abhängig von der Sicherheitspolitik kann auf die Implementierung einzelner Mechanismen auch verzichtet werden.

Granularität: Mit Proxies ist eine sehr detaillierte Zugriffskontrolle möglich. Die Kontrolle kann abhängig von **Host-Adressen**, vom **Dienst**, vom **Benutzer**, von **zeitlichen Beschränkungen**, oder vom **Kontext** des Zugriffs erfolgen. Insbesondere stehen der Zugriffskontrolle alle zwischen Client und Server ausgetauschten Daten zur Verfügung.

Vorteile eines Proxies:

Feine Granularität: Größter Vorteil von Proxies ist die **detaillierte Zugriffskontrolle**, durch die auch ein sehr gutes **Audit** möglich wird (s. o.).

Authentisierung: Es ist eine einmalige Benutzerauthentisierung beim Verbindungsaufbau oder eine wiederholte Authentisierung für jede Anfrage möglich.

Nachteile eines Proxies:

Transparenz: Zugriffe über einen Proxy können für den Clienten nicht transparent erfolgen. Abhängig von der Implementation des Clienten kann gegebenenfalls die fehlende Transparenz gegenüber dem Benutzer verborgen werden. Dafür ist jedoch eine auf den Proxy abgestimmte Implementation und Konfiguration des Clienten erforderlich, was einen hohen Installations- und Verwaltungsaufwand nach sich zieht. Authentisierungsverfahren machen oft eine Interaktion mit dem Benutzer erforderlich, so daß eine transparente Lösung hier nicht möglich ist.

Performanz: Die Verarbeitung von Anfragen auf Anwendungsebene erfordert eine höhere Rechenleistung als die Überprüfung der Protokoll-Header in einer Packet Screen. Durch aufwendige Authentisierungs- und Zugriffskontrollmechanismen kann der Proxy schnell zum Engpaß werden.

Installationsaufwand: Proxies können nicht allein eingesetzt werden. Aufbau und Konfiguration ebenfalls benötigter Komponenten wie Packet Screens und Bastionen (vgl. Abschnitt 5.2) stellen einen hohen Aufwand dar.

5.1.3 Anwendungs-Gateway

Bei Protokollen, die im Stapelbetrieb (“Batch-Mode”) verarbeitet werden, wie zum Beispiel SMTP (E-Mail) und NNTP (Netnews), ist es vorteilhaft, Zugriffe nicht über einen Proxy an einen Server weiterzuleiten, sondern auf dem Firewall bereits wesentliche Teile der Serverfunktionalität zu realisieren.

Die Bezeichnungen Proxy und Anwendungs-Gateway werden häufig synonym verwendet. Anwendungs-Gateways zeichnen sich gegenüber Proxies durch eine höhere Komplexität aus. Während sich Proxies meist auf eine einfache Authentisierung beschränken und anschließend lediglich Daten zwischen den beiden Verbindungen kopieren, enthalten Anwendungs-Gateways für die jeweilige Applikation spezialisierten Code, der bereits eine Vorverarbeitung durchführt oder einen Teildienst der Anwendung erbringt. Da auch bei Proxies Zugriffskontrollmechanismen eingesetzt werden können, die speziell auf die unterstützte Anwendung ausgerichtet sind, ist die Grenze zwischen den Anwendungs-Gateways und Proxies fließend.

Granularität: Die Granularität der Zugriffskontrolle und des Audits unterscheiden sich nicht von dem eines Proxies.

Vorteile von Anwendungs-Gateways: Zusätzlich zu den Vorteilen von Proxies haben Anwendungs-Gateways folgende Vorteile:

Verdecken der internen Struktur: Durch Erbringen wesentlicher Serverfunktionen kann die interne Struktur des durch den Firewall geschützten Netzes verdeckt werden.

Verzögerungszeiten weniger relevant: Die stapelorientierte Verarbeitung erlaubt aufwendigere Zugriffskontroll-Verfahren (z. B. das Scannen nach Viren in E-Mails), da höhere Verzögerungen als bei interaktiven Protokollen akzeptabel sind.

Nachteile von Anwendungs-Gateways: Zusätzlich zu den Nachteilen von Proxies haben Anwendungs-Gateways folgende Nachteile:

Keine Benutzerinteraktion zur Authentisierung: Stapelorientierte Anwendungen schränken die Liste der einsetzbaren Authentisierungsverfahren ein, da eine Benutzerinteraktion zur Authentisierung nicht möglich ist.

5.1.4 Weitere Firewall-Komponenten

Audit: Die von den Zugriffskontrollkomponenten gemeldeten Ereignisse müssen aufgezeichnet und ausgewertet werden. Um Angriffe schnell erkennen zu können, ist eine maschinelle Auswertung der aufgezeichneten Daten notwendig.

Verschlüsselung: In zunehmendem Maße werden Verschlüsselungsverfahren in Firewall-Konzepte integriert. Neben den bereits eingesetzten "Virtual Private Networks" (VPNs) werden zukünftig auch in Proxies und Anwendungs-Gateways kryptographische Verfahren zur Verschlüsselung der übertragenen Daten oder zur Authentisierung angewendet werden.

NAT: Über die "Network Address Translation" (NAT) können Packet Screens nicht-öffentliche auf öffentliche IP-Adressen abbilden. Systeme mit nicht-öffentlichen IP-Adressen können so auf Dienste im Internet zugreifen. Zudem bleibt durch NAT die Struktur des internen Netzes verdeckt.

Systemmanagement: Die Konfiguration und Wartung eines Firewalls stellt eine sehr komplexe Aufgabe dar, die einer Unterstützung durch geeignete Werkzeuge bedarf. Dies können zum Beispiel grafische Benutzeroberflächen für die Erstellung von Filterregeln oder Werkzeuge zur Auswertung von Auditdaten sein.

5.2 Firewall-Architekturen

Firewall-Architekturen werden aus Firewall-Komponenten aufgebaut. Aus der Vielzahl möglicher Kombinationen werden im Folgenden die wichtigsten Architekturen mit ihren Vor- und Nachteilen kurz vorgestellt.

Firewall-Architektur	Vorteile	Nachteile
Packet Screen	geringer Installationsaufwand, geringe Kosten	Große Anzahl bedrohter Rechner, Kontrolle interner Endsysteme erforderlich, geringe Granularität der Zugriffskontrolle
Kombination von Packet Screen und Bastion	feine Granularität der Zugriffskontrolle, Erweiterbarkeit	hohe Kosten, hoher Installationsaufwand
Gateway-Firewall	feine Granularität der Zugriffskontrolle, viele kommerzielle Angebote	eventuell hohe Kosten, mangelhafte Erweiterbarkeit

5.2.1 Packet Screens

Die Konfiguration einer Packet Screen stellt die einfachste Firewall-Architektur dar. Trotz der bereits diskutierten Einschränkungen (vgl. Abschnitt 5.1.1) bei der Granularität der Zugriffskontrolle und beim Audit kann die Konfiguration einer Packet Screen ausreichen, wenn die erkannten Defizite durch zusätzliche Maßnahmen kompensiert werden (vgl. [Ellermann 1995]).

Da die Möglichkeit, Filterregeln zu definieren, bereits in allen gängigen Routern enthalten ist und ein Router zudem zur Anbindung eines Netzes an das Internet oder an ein größeres Firmennetz ohnehin benötigt wird, ist die Packet Screen eine kostengünstige Lösung. Die konzeptionelle Sicht einer Firewall-Architektur auf Basis einer Packet Screen ist in Abbildung 5.3 wiedergegeben.

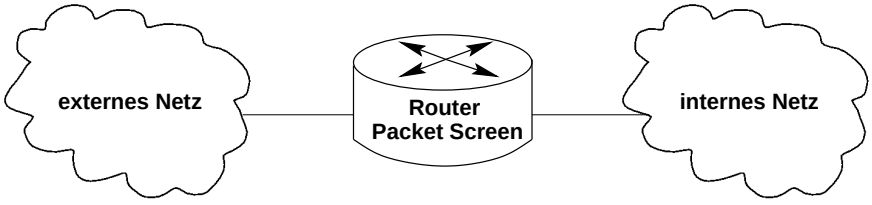


Abbildung 5.3: Packet Screen: konzeptionelle Sicht

Zur Erstellung der Filterregeln können zwei unterschiedliche Ansätze gewählt werden:

Verbotsregeln: Über eine Konfigurierung von Verbotsregeln für als gefährlich eingestufte Protokolle können Angriffe an der Packet Screen gestoppt werden. Da

alle anderen Dienste von diesen Regeln nicht betroffen sind, bleiben bei diesem Ansatz viele Dienste erreichbar. Für Benutzer hat dies den Vorteil, daß ihre Kommunikation nur selten durch die Filterregeln unterbunden wird. Allerdings bleiben so auch viele Dienste für Angriffsversuche geöffnet, so daß dieser Ansatz nur als "Ad-hoc Lösung" bei konkreten Angriffen auf einzelne Dienste gewählt werden sollte.

Erlaubnisregeln: Vor der Konfiguration einer Packet Screen mit Erlaubnisregeln muß genau analysiert werden, welche Kommunikationsanforderungen (benötigte Dienste) und Sicherheitsanforderungen (akzeptables Restrisiko) bestehen. Resultat dieser Analyse muß neben einer Aufstellung weiterer Absicherungsmaßnahmen eine kurze Liste sein, welche Dienste von welchen Hosts oder Gruppen von Hosts zugreifbar sein müssen. Aus dieser Liste können anschließend die Erlaubnisregeln für die Konfiguration der Packet Screen erstellt werden. Die Liste der Erlaubnisregeln muß immer abgeschlossen werden durch eine Verbotsregel für alle nicht explizit aufgeführten Dienste.

Eine wichtige Aufgabe von Packet Screens ist der Schutz gegen IP-Spoofing Angriffe (s. Kapitel 4.1). Durch einfache Plausibilitätsüberprüfungen kann die Packet Screen feststellen, ob die Absender-Adresse einer eingegangenen Dateneinheit zu einem Endsystem gehört, das über die jeweilige Schnittstelle erreichbar ist. [RFC 2267]

Wird eine Packet Screen in einem Router mit mehr als zwei Schnittstellen konfiguriert, muß die Konfiguration von Filterregeln sowohl vor als auch nach der Wegfindung (Routing) möglich sein. Auf einer Schnittstelle hereinkommende Dateneinheiten müssen vor der Wegfindung überprüft werden, um IP-Spoofing Angriffe erkennen zu können. Erst nach der Wegfindung sollten die für das Zielnetz geltenden Filterregeln angewendet werden, da zu diesem Zeitpunkt aufgrund der Routing-Entscheidung nur noch bestimmte Zieladressen vorkommen können und somit nur die auf diese Zieladressen zutreffenden Filterregeln ausgewertet werden müssen.

Neben den grundsätzlichen Vor- und Nachteilen der Firewall-Komponente "Packet Screen" (s. Abschnitt 5.1.1) werden abschließend Aspekte vorgestellt, die aus dem Einsatz einer Packet Screen als Firewall-Architektur resultieren.

Vorteile bei ausschließlichem Einsatz einer Packet Screen:

Kosten: Wesentlicher Vorteil einer Firewall-Architektur, die nur aus einer Packet Screen besteht, sind die niedrigen Kosten für Installation und Betrieb. Die niedrigen Kosten können jedoch durch zusätzlich notwendige Absicherungsmaßnahmen bei den Endsystemen in der Gesamtkalkulation zu hohen Kosten führen.

Nachteile bei ausschließlichem Einsatz einer Packet Screen:

Große Anzahl bedrohter Rechner: Der Einsatz einer Packet Screen führt üblicherweise zu einer Konfiguration mit einer großen Anzahl erreichbarer und somit bedrohter Rechner. Die Sicherheit dieser Lösung richtet sich dann nach dem schwächsten internen System, da Angreifer, die hier eindringen können, von diesem System aus auch alle anderen internen Systeme direkt erreichen können ("Island Hopping"). Um trotzdem eine sichere Installation zu erreichen, ist daher ein hoher Absicherungsaufwand bei den internen Endsystemen notwendig.

Kontrolle interner Endsysteme: Die Sicherheit dieser Lösung hängt in hohem Maße von den internen Endsystemen ab, da von der Packet Screen hereingelassene Dateneinheiten von den Endsystemen sicher verarbeitet werden müssen. Bei vielen Netzinstallationen unterliegt die Verwaltung interner Endsysteme nicht der Kontrolle des Firewall-Verwalters bzw. bei einigen Endsystemen ist eine Trennung zwischen Verwaltungsaufgaben und normaler Benutzung gar nicht möglich. Neben der versäumten Installation sicherheitsrelevanter Patches und der unsicheren Konfiguration stellt auch die beabsichtigte Umgehung der Packet Screen ein Problem dar, weil gesperrte Dienste an alternativen, nicht gesperrten Ports bereitgestellt werden können.

Granularität: Die Überprüfung von Protokoll-Headern reicht für die Abwehr vieler Angriffe nicht aus.

5.2.2 Kombinationen von Packet Screen und Bastion

Die Probleme, die durch die große Anzahl bedrohter Rechner und durch die mangelhafte Kontrolle der Endsysteme bei der Packet Screen Architektur auftreten können, lassen sich durch eine weitere Einschränkung des Zugriffs lösen. Bei Kombinationen von Packet Screen und Bastion werden eine oder mehrere Packet Screens so konfiguriert, daß nur noch ein einzelnes Endsystem mit dem internen und dem externen Netz kommunizieren kann; die direkte Kommunikation zwischen internen und externen Endsystemen wird von der Packet Screen unterbunden. Dieses Endsystem wird als “Bastion” bezeichnet. Eine **Bastion** wird allgemein definiert als ein besonders abgesichertes System, das überwunden werden muß, um Endsysteme im jeweils anderen Netz zu erreichen. Auf der Bastion werden Proxies (s. Abschnitt 5.1.2) und Anwendungs-Gateways (s. Abschnitt 5.1.3) eingesetzt, um die Kommunikation zwischen internen und externen Systemen mit einer sehr feinen Granularität bei Zugriffskontrolle und Audit zu ermöglichen. Da die Bastion neben der Packet Screen als einziges System von beiden Netzen aus erreichbar ist, sind hier besondere Sicherungsmaßnahmen erforderlich.

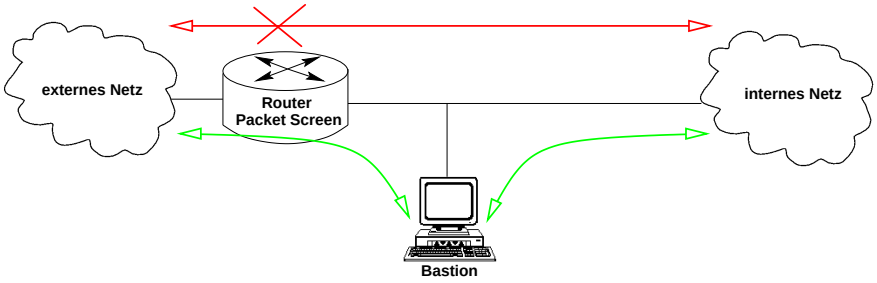


Abbildung 5.4: Kombination von Packet Screen und Bastion: konzeptionelle Sicht – Bastion innen

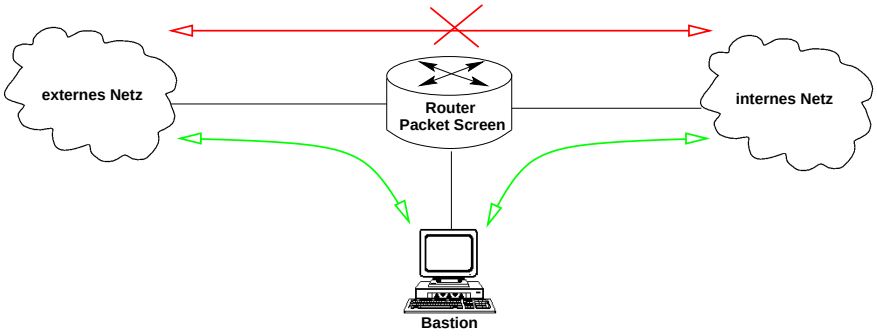


Abbildung 5.5: Kombination von Packet Screen und Bastion: konzeptionelle Sicht – Bastion mittig

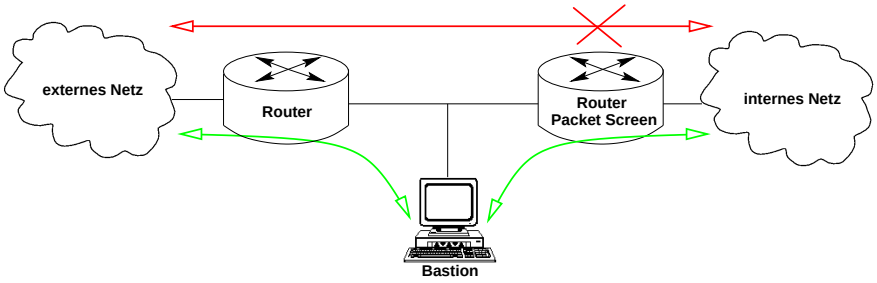


Abbildung 5.6: Kombination von Packet Screen und Bastion: konzeptionelle Sicht – Bastion außen

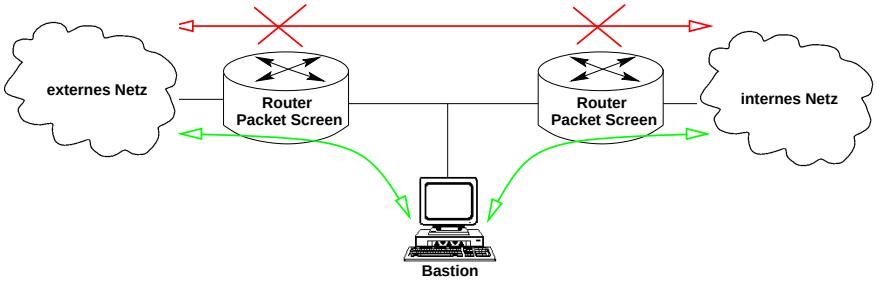


Abbildung 5.7: Kombination von Packet Screen und Bastion: konzeptionelle Sicht – beidseitig abgeschirmte Bastion

Bei der Kombination von Packet Screen und Bastion wird abhängig von der Position der Bastion zwischen vier Konzepten unterschieden: (siehe auch [Ellermann 1993])

Bastion innen (Abbildung 5.4)

Die Installation der Bastion im internen Netz ist die am einfachsten zu realisierende Lösung. Jedoch besteht hierbei zwischen internem Netz und der Bastion keine weitere Abschirmung. Angriffe auf die Bastion, die von internen Systemen ausgehen, müssen in diesem Fall von der Bastion allein abgewehrt werden.

Bastion mittig (Abbildung 5.5)

Durch die Installation der Bastion in einem eigenen Netz an einer weiteren Schnittstelle der Packet Screen kann der Zugriff auf die Bastion vom externen und vom internen Netz besser eingeschränkt werden.

Bastion außen (Abbildung 5.6)

Die Lösung, die Bastion vor der Packet Screen zu installieren, erscheint zunächst als schlechte Wahl, da Angriffe vom Internet aus auf die Bastion leicht möglich sind. Bei entsprechender Absicherung der Bastion kann mit dieser Lösung über

sogenannte „Paket Sauger“ (siehe [Ellermann 1993, Abschnitt 4.4.1]) eine sehr detaillierte Aufzeichnung von Angriffen auf den Firewall erfolgen. Zusätzlich können von der Bastion aus alle Dienste im Internet uneingeschränkt zugegriffen werden.

Bastion beidseitig abgeschirmt (Abbildung 5.7)

Bei der vorhergehend beschriebenen Installation der Bastion außen werden zwei Router benötigt, von denen aber nur einer als Packet Screen verwendet wird. Der zum externen Netz hin installierte Router ist in den meisten Fällen zusätzlich notwendig, um die Anbindung an das Internet technisch zu realisieren. Aus diesem Grund liegt es nahe, auch den zum Internet hin installierten Router als Packet Screen zu nutzen. Es ergibt sich dann die in Abbildung 5.7 vorgestellte Konfiguration mit einer beidseitig abgeschirmten Bastion. Das in beide Richtungen hin abgeschirmte Subnetz, an dem die Bastion installiert ist, wird als Grenznetz (“Screened Subnet”, “Demilitarized Zone” oder “DMZ”) bezeichnet.

Die Lösung der beidseitig abgeschirmten Bastion mit zwei Packet Screens kann zu der Lösung der mittig angeordneten Bastion optimiert werden, bei der eine Packet Screen logisch die Funktionen der beiden Packet Screens realisiert. Die Lösung mit zwei Packet Screens hat dieser vereinfachten Lösung gegenüber den Vorteil, daß ein Angreifer jeweils immer nur eine Packet Screen angreifen kann, während die zweite Packet Screen einen großen Anteil der Sicherheit des Firewalls weiterhin aufrechterhalten kann.

Vorteile der Kombination von Packet Screen und Bastion:

Granularität: Durch die Kombination von Packet Screen und Bastion kann eine sehr feine Granularität bei Zugriffskontrolle und Audit erreicht werden. Insgesamt kann ein sehr hohes Sicherheitsniveau erreicht werden.

Erweiterbarkeit: Durch Einsatz zusätzlicher Bastionen kann diese Lösung bei Bedarf an höhere Leistungsanforderungen angepaßt werden.

Nachteile der Kombination von Packet Screen und Bastion:

Kosten: Zur Realisierung diese Lösung ist zunächst ein hoher Installationsaufwand notwendig. Während des Betriebs können durch niedrigeren Aufwand bei Pflege und Überwachung der internen Systeme Kosten eingespart werden.

5.2.3 Gateway-Firewalls

Ein Gateway ist ein Rechner, der an mindestens zwei Netze angeschlossen ist und der auf Anwendungsebene eine Kommunikation zwischen beiden Netzen ermöglicht. Durch Erweiterungen um Zugriffskontrolle und Audit zusammen mit einer besonderen Absicherung des Gateways ist der Einsatz als Firewall möglich (vgl. Abb. 5.8).[11] Dabei muß sichergestellt werden, daß alle Kommunikationsbeziehungen zwischen den angeschlossenen Netzen nur über den Firewall möglich sind. Auf dem Firewall kann mit Hilfe von Proxies und Anwendungs-Gateways eine sehr präzise Zugriffskontrolle definiert werden.

Abbildung 5.8: Gateway Firewall: konzeptionelle Sicht

Der Gateway-Firewall erreicht ähnliche Eigenschaften wie eine Kombination von Packet Screen und Bastion. Während bei letzterer Lösung die Packet Screen dafür sorgt, daß nur die Bastion mit beiden Netzen kommunizieren kann, muß bei einem Gateway-Firewall der gesamte Verkehr über dieses System geführt werden, da das Gateway als einziger Rechner an beide Netze angeschlossen ist. Eine zusätzliche Abschirmung des Gateway-Firewalls durch vorgeschaltete Packet Screens kann die Sicherheit eines Gateway-Firewalls kaum erhöhen. Das Grenznetz zwischen Packet Screen und Gateway-Firewall eignet sich jedoch hervorragend für die Installation von Informationsdiensten (Anonymous-FTP, WWW usw.), deren Installation auf einem Gateway-Firewall aus Sicherheitsgründen nicht empfohlen werden kann.

Vorteile von Gateway-Firewalls:

Granularität: Durch einen Gateway-Firewall kann eine sehr feine Granularität bei Zugriffskontrolle und Audit erreicht werden. Insgesamt wird ein sehr hohes Sicherheitsniveau erreicht.

Viele kommerzielle Lösungen: Für Gateway-Firewalls gibt es viele Anbieter.

[11] Ein Gateway-Firewall ist per Definition eine Bastion.

Nachteile von Gateway-Firewalls:

Kosten: Zur Realisierung diese Lösung ist zunächst ein hoher Installationsaufwand notwendig. Während des Betriebs können durch niedrigeren Pflege- und Überwachungsaufwand bei internen Systemen Kosten eingespart werden.

Erweiterbarkeit: Im Gegensatz zu Kombinationen von Packet Screen und Bastion besteht keine Erweiterungsmöglichkeit durch den Einsatz zusätzlicher Bastionen.

5.2.4 Firewall-Ergänzungen

Nach der Vorstellung der Firewall-Architekturen sollen in diesem Abschnitt die wichtigsten Ergänzungen und Varianten kurz angesprochen werden.

Informationsdienste: Zusätzlich zur Absicherung des internen Netzes wird von Firewall-Konzepten oft auch die sichere Integration von Informationsdiensten (Anonymous-FTP, WWW usw.) gefordert. Diese Dienste, die meist eingesetzt werden, um öffentliche Informationen für das externe Netz bereitzustellen, haben andere Sicherheits- und Kommunikationsanforderungen als die Systeme im internen Netz. Um eine Gefährdung für das interne Netz durch die Informationsdienste ausschließen zu können, ist das Bereitstellen der Informationsdienste durch Systeme im internen Netz oder auf dem Firewall selbst nicht akzeptabel. Besser ist es, die Informationsdienste außerhalb des Firewalls anzubieten. Da der Informationsserver so nicht mehr durch den Firewall geschützt werden kann, erfordert diese Lösung jedoch eine zusätzliche Absicherung des Informationsservers.

Virtual Private Networks (VPNs): Die Verbindung entfernt voneinander liegender Teilnetze kann entweder über angemietete, teure Standleitungen oder beim Einsatz von Verschlüsselung auch über öffentliche Netze wie dem Internet erfolgen. Die Koppelung über öffentliche Netze mit Hilfe von Verschlüsselung wird als “Virtual Private Network” (VPN) bezeichnet. Eine Dateneinheit wird vom VPN-Endpunkt verschlüsselt und nach der Übertragung im öffentlichen Netz von einem weiteren VPN-Endpunkt wieder entschlüsselt. Die Ver- und Entschlüsselung geschieht für Sender und Empfänger transparent.

Die Planung eines VPNs muß im Zusammenhang mit dem Firewall-Konzept betrachtet werden, da je nach Positionierung der beiden Konzepte zueinander unterschiedliche Sicherheitseigenschaften erreicht werden [Kossakowski 1998, S. 6f]. In jedem Fall ist zu beachten, daß die miteinander gekoppelten Netzwerke dasselbe Sicherheitsniveau erreichen müssen, da andernfalls Zugriffe aus einem weniger stark gesicherten Netz durch das VPN eine Bedrohung für besser gesicherte Netze darstellen können.

5.3 Nicht durch Firewalls abgedeckte Sicherheitsprobleme

Firewalls werden oft als “Allheilmittel” für die Netzwerksicherheit angesehen. Ein Firewall ist ein Zugriffskontrollmechanismus, der Netze in verschiedene Sicherheitsbereiche aufteilt. Um falsche Erwartungen zu verhindern, soll auch aufgeführt werden, was Firewalls nicht leisten können.

Keine Kontrolle des internen Verkehrs: Über den Firewall findet nur die Kommunikation zwischen internem und externem Netz statt. Der Datenverkehr zwischen internen Systemen untereinander kann durch den Firewall nicht kontrolliert werden. Für den Schutz gegen Insider-Angriffe ist daher weiterhin eine Absicherung der internen Systeme erforderlich. In manchen Fällen kann auch eine weitere Aufteilung des internen Netzes in mehrere über (Intranet-)Firewalls getrennte Sicherheitszonen sinnvoll sein.

Kein Schutz gegen Abhören oder Manipulation übertragener Daten:

Ein Firewall kann Angriffe nur durch Überprüfung der übertragenen Daten erkennen. Da einer Packet Screen für diese Überprüfung nur die Protokoll-Header zur Verfügung stehen, sind hier nur sehr einfache Plausibilitätstests möglich.[12] Proxies können zwar auch die übertragenen Dateninhalte überprüfen, jedoch ermöglicht auch dies keine Erkennung von Manipulationen an übertragenen Daten (z. B. TCP-Hijacking). Statt eines Zugriffskontrollmechanismus sind kryptographische Verfahren zur Absicherung der übertragenen Daten erforderlich.

Keine Zugriffskontrolle auf verschlüsselten Daten: Der Einsatz kryptographischer Verfahren soll das Abhören oder die Manipulation von übertragenen Daten verhindern. Dadurch wird gleichzeitig auch eine Kontrolle dieser Daten durch einen Firewall verhindert.

Begrenzte Kontrollmöglichkeiten des Firewalls: Proxies und Anwendungs-Gateways erhalten zwar Zugriff auf alle übertragenen Daten, was eine Vielzahl von Angriffen theoretisch erkennbar macht. In der Praxis übersteigen aufwendige Überprüfungen jedoch die Leistungsfähigkeit des Firewalls.

Zur Absicherung der aufgeführten Probleme müssen andere Sicherheitsmechanismen eingesetzt werden. Über die aufgeführten Probleme hinaus kann der Firewall selbst zum Ziel von Angriffen werden, was schwerwiegende Konsequenzen haben kann, wenn die Sicherheit allein vom Firewall abhängt und die Sicherheitsmaßnahmen auf den internen Systemen vernachlässigt wurden.

[12] IP-Spoofing Angriffe können über einfache Regeln [RFC 2267] teilweise erkannt und abgewehrt werden.

Ein weiteres Problem sind “Denial of Service” Angriffe, da der Firewall gewollt als Flaschenhals in der Kommunikation zwischen internem und externem Netz eingeführt wird, um hier eine Kontrolle zu ermöglichen. “Denial of Service” Angriffe auf diesen Engpaß haben daher weitreichende Folgen, wenn der Firewall gegen diese Angriffe nicht resistent ist.

5.4 Planung eines Firewalls

Im Folgenden soll das Vorgehen beim Aufbau eines Firewalls skizziert werden.

Bevor ein Firewall aufgebaut werden kann, muß zuerst festgelegt werden, welche Kommunikation zwischen internem und externem Netz erlaubt werden darf und welche verboten werden muß. Bei der Kommunikation, die zwischen den beiden Netzen durchgeführt werden darf, muß weiter aufgeschlüsselt werden, welche zusätzlichen Sicherungmaßnahmen notwendig sind, um die Bedrohungen durch das jeweilige Protokoll auf ein akzeptables Niveau zu reduzieren. Erst wenn alle Sicherheitsanforderungen festgelegt sind, kann mit der Auswahl eines geeigneten Konzeptes und anschließend mit der technischen Realisierung des Firewalls begonnen werden.

5.4.1 Kommunikationsanforderungen

Bei den Kommunikationsanforderungen ist es unbedingt notwendig, zwischen hereinkommenden Verkehr und herausgehenden Verkehr zu unterscheiden.

- Bei **hereinkommendem Verkehr** wird vom externen Netz auf Server im internen Netz zugegriffen. Sicherheitsmaßnahmen müssen die Bedrohung der internen Server ausschließen bzw. reduzieren.
- Bei **herausgehendem Verkehr** wird von internen Systemen auf Server im externen Netz zugegriffen. Sicherheitsmaßnahmen müssen mögliche Gefährdungen der internen Clienten (z. B. durch ausführbare Inhalte) sowie unberechtigtes Abfließen von Informationen abwehren.

Es wird hier von dem Regelfall ausgegangen, daß das interne Netz vor dem externen Netz geschützt werden soll. Die höheren Risiken bestehen bei hereinkommendem Verkehr.

Zur Minimierung der Risiken ist es wichtig, nur für unbedingt notwendige Dienste hereinkommenden Verkehr zuzulassen (“Prinzip der geringsten Berechtigung”). Zur ersten Ermittlung der zur Zeit genutzten Dienste können Netzwerkmonitoring-Werkzeuge wie `argus` gute Dienste leisten.

Dienste, die in der Regel zugelassen werden sollten, sind:

- **SMTP** ist notwendig zum Empfangen von E-Mail.
- **DNS** wird für die Umwandlung von Rechnernamen in IP-Adressen benötigt.

- **Informationsdienste** (Anonymous FTP und WWW) werden benötigt, um Informationen anderen Benutzern im Internet zugänglich zu machen. (Produktinformation, Selbstdarstellung, ...)
- **RLogin/Telnet** ermöglicht den Zugang zu internen Systemen. Dies kann zum Beispiel für Heimarbeitsplätze oder Arbeitsplätze mit wechselnden Orten notwendig sein. Eine Verbindung sollte grundsätzlich über *SSH* aufgebaut werden.
- **IDENT** wird von vielen Servern im Internet abgefragt, um wiederum Benutzer zu identifizieren, die dort die Dienste abfragen.

Zugriffe auf andere Dienste müssen vom Firewall unterbunden werden. Die Einschränkung auf die genannten Dienste bedeutet gleichzeitig, daß auch nur diese wenigen Dienste von Angreifern für einen Einbruch ausgenutzt werden können.

Aufgabenteilung Da nicht alle Rechner die gleichen Aufgaben haben, ist es notwendig, die oben festgelegten Kommunikationsanforderungen noch einmal für unterschiedliche Rechner zu überprüfen. Entsprechend der jeweiligen Aufgabe sollen nur die Protokolle an einen Rechner weitergeleitet werden, die für die Erfüllung der Aufgabe unbedingt notwendig sind (Prinzip der geringsten Berechtigung).

So ist es zum Beispiel immer sinnvoll, den **Informationsserver** von den übrigen Systemen zu trennen; d. h außerhalb des internen Netzes aufzustellen. Auf dem Informationsserver sollten dann ausschließlich öffentliche Informationen gespeichert werden, die jedem Benutzer im Internet zugänglich gemacht werden. Hereinkommender WWW- und Anonymous FTP Verkehr kann dann außerhalb des internen Netzes verarbeitet werden und braucht nicht durch einen Firewall gelassen zu werden.

5.4.2 Auswahl der Firewall-Komponenten

5.4.2.1 Packet Screen

5.4.2.1.1 Aufgaben

Die Packet Screen soll Zugriffe nur auf bestimmte festgelegte Dienste zulassen.[13]

5.4.2.1.2 Funktionsweise

Durch Zugriffskontrolle auf IP-Dateneinheiten wird die Kommunikation zwischen zwei Netzen eingeschränkt. Von der Packet Screen werden Dateneinheiten nur dann weitergeleitet, wenn IP, TCP und UDP Protokoll-Header festgelegte Informationen enthalten.

5.4.2.1.3 Schutz gegen folgende Bedrohungen

Zugriffe von nicht lokalen Systemen auf ausgewählte Systeme (IP-Adressen) und Dienste (Portnummern) können verhindert werden.

5.4.2.1.4 Einschränkungen

Zugriffe von lokalen Systemen, die die Packet Screen nicht passieren müssen, können nicht kontrolliert werden. Zugriffe auf IP-Adressen und Portnummern können entweder erlaubt oder verboten werden; eine feinere Unterscheidung nach Benutzern, Art des Zugriffs oder übertragener Inhalte erfordern den zusätzlichen Einsatz von Proxies (s. Abschnitt 5.4.2.2) oder Anwendungs-Gateways (s. Abschnitt 5.4.2.3).

5.4.2.1.5 Vorgehen beim Einsatz

- Feststellen der zu unterstützenden Protokolle (Kommunikationsanforderungen, s. Abschnitt 5.4.1).
- Feststellen der verbleibenden Restrisiken bei den unterstützten Protokollen und den erreichbaren Servern (s. Abschnitt 4).
- Formulierung der Kommunikationsanforderungen in Form von Filterregeln.
- Konfiguration der Filterregeln.
- Absicherung der erreichbaren Server.

[13] Weitere Informationen zu Packet Screens finden sich im Abschnitt 5.1.1

5.4.2.1.6 Fazit

Packet Screens können den Zugriff auf bestimmte Hosts und Dienste einschränken. Für erreichbare Dienste muß eine zusätzliche Absicherung erfolgen. Zugriffe zwischen Systemen des internen Netzes können durch Packet Screens nicht kontrolliert werden.

5.4.2.2 Proxy

5.4.2.2.1 Aufgaben

Ein Proxy soll für einen Clienten Anfragen an einen Server weiterleiten und die dabei übertragenen Daten überwachen.[14]

5.4.2.2.2 Funktionsweise

Nach einer Authentisierung können autorisierte Benutzer über den Proxy auf Server zugreifen, deren direkte Erreichbarkeit durch weitere Maßnahmen (z. B. Einsatz einer Packet Screen) unterbunden wird. Die zwischen Client und Server ausgetauschten Daten werden vom Proxy auf der Anwendungsebene verarbeitet, wodurch eine vollständige Kontrolle über die Kommunikation erreicht wird.

5.4.2.2.3 Schutz gegen folgende Bedrohungen

- Der Zugriff unberechtigter Benutzer auf interne Server kann durch Authentisierungsmechanismen des Proxies verhindert werden.
- Angriffe auf bekannte Schwächen eines Servers können durch geeignete Filterung im Proxy verhindert werden.
- Bedrohungen für Clienten durch ausführbare Inhalte können durch Integration geeigneter Scanner in einem Proxy reduziert werden.

5.4.2.2.4 Einschränkungen

- Zugriffe von lokalen Systemen, die den Proxy nicht passieren müssen, können nicht kontrolliert werden.
- Verschlüsselte Inhalte können nicht überprüft werden.
- Durchführen zeitintensiver Scan-Mechanismen (Viren-Scanner) übersteigt zeitliche Restriktionen. (Alternativen Einsatz eines Anwendungs-Gateways s. Abschnitt 5.4.2.3) prüfen.

[14] Weitere Informationen zu Proxies finden sich im Abschnitt 5.1.2

5.4.2.2.5 Vorgehen beim Einsatz

- Feststellen der Risiken des unterstützten Protokolls (s. Abschnitt 4).
- Auswahl eines Proxies, der gegen die festgestellten Risiken geeignete Sicherheitsmechanismen anbietet.
- Installation und Konfiguration des Proxies.
- Eventuell weitere Maßnahmen, um Nutzung des Proxies zu erzwingen (Packet Screen, s. Abschnitt 5.4.2.1).

5.4.2.2.6 Fazit

Zugriffe auf einen Dienst können durch Proxies einer sehr detaillierten Zugriffskontrolle unterworfen werden. Zugriffe zwischen Systemen des internen Netzes können durch Proxies nicht kontrolliert werden.

5.4.2.3 Anwendungs-Gateway

5.4.2.3.1 Aufgaben

Ein Anwendungs-Gateway soll einen nach Sicherheitsgesichtspunkten reduzierten Dienst zur Verfügung stellen. Anfragen werden in erster Linie vom Anwendungs-Gateway selbst beantwortet. Stehen keine ausreichenden Informationen zur Beantwortung zur Verfügung, wird die Anfrage weitergeleitet. Die Sicherheitsrichtlinien regeln, welche Anfragen weitergeleitet werden dürfen.[15]

5.4.2.3.2 Funktionsweise

Das Anwendungs-Gateway implementiert ein Anwendungsprotokoll zumindest in Teilen. Anfragen, die nicht selbst beantwortet werden können, werden an einen regulären Server weitergeleitet. Die Antworten des Servers werden vor der Weitergabe an den Clienten gefiltert. Die direkte Erreichbarkeit der Server muß durch ergänzende Maßnahmen (z. B. Einsatz einer Packet Screen) unterbunden werden. Die zwischen Client und Server ausgetauschten Daten werden auf der Anwendungsebene verarbeitet und können vom Anwendungs-Gateway zwischengespeichert werden, wodurch eine aufwendigere Kontrolle als bei einem Proxy möglich wird.

5.4.2.3.3 Schutz gegen folgende Bedrohungen

- Der Zugriff unberechtigter Benutzer auf interne Server kann durch Authentisierungsmechanismen des Proxies verhindert werden.
- Angriffe auf bekannte Schwächen eines Servers können durch geeignete Filterung im Anwendungs-Gateway verhindert werden.
- Bedrohungen für Clienten durch ausführbare Inhalte können über die Integration auch zeitintensiver Scan-Mechanismen (Viren-Scanner) in ein Anwendungs-Gateway reduziert werden.

5.4.2.3.4 Einschränkungen

- Zugriffe von lokalen Systemen, die das Anwendungs-Gateway nicht passieren müssen, können nicht kontrolliert werden.
- Verschlüsselte Inhalte können nicht überprüft werden.

[15] Weitere Informationen zu Anwendungs-Gateways finden sich im Abschnitt 5.1.3

5.4.2.3.5 Vorgehen beim Einsatz

- Feststellen der Risiken des unterstützten Protokolls (s. Abschnitt 4).
- Auswahl eines Anwendungs-Gateways, das gegen die festgestellten Risiken geeignete Sicherheitsmechanismen anbietet.
- Installation und Konfiguration des Anwendungs-Gateways.
- Eventuell weitere Maßnahmen, um Nutzung des Anwendungs-Gateways zu erzwingen (Packet Screen, s. Abschnitt 5.4.2.1).

5.4.2.3.6 Fazit

Zugriffe auf einen Dienst können durch Anwendungs-Gateways einer sehr detaillierten Zugriffskontrolle unterworfen werden. Zugriffe zwischen Systemen des internen Netzes können durch Anwendungs-Gateways nicht kontrolliert werden.

5.4.3 Auswahl eines Firewalls

Die Auswahl eines Firewalls hängt von den Kommunikations- und Sicherheitsanforderungen sowie den ausgewählten Firewall-Komponenten ab. Wenn dabei festgestellt wurde, daß eine Packet Screen zur Absicherung ausreichend ist, dann stehen eine Reihe von Public-Domain Packet Screen Implementationen zur Verfügung. Gegebenenfalls können auch in bereits vorhandenen Routern Filterregeln konfiguriert werden.

Hat sich darüber hinaus jedoch auch die Notwendigkeit für Proxies oder Anwendungs-Gateways ergeben, kommt praktisch nur noch der Einsatz kommerzieller Firewall-Angebote in Frage.

5.4.4 Realisierung

Das Vorgehen bei der Realisierung eines Firewalls hängt in erster Linie von dem gewählten Firewall-Konzept und dem dafür eingesetzten Firewall-Produkt ab, so daß an dieser Stelle nur Aussagen anhand eines konkreten Szenarios möglich sind. In diesem Abschnitt sollen daher Konfigurationsbeispiele für ein konkretes Szenario vorgestellt werden. In dem Szenario wird davon ausgegangen, daß eine Kontrolle des herausgehenden Verkehrs nicht notwendig ist und daß für hereinkommenden Verkehr nur wenige ausgewählte Dienste (SMTP, DNS, SSH, IDENT) angeboten werden (vgl. Abschnitt 5.4.1). Für die angebotenen Dienste soll eine Kontrolle durch eine Packet Screen ausreichend sein.

Zunächst werden auf dem Mail-Server (“`mailserver`”, `192.168.14.1`) und dem Name-Server (“`nameserver`”, `192.168.14.2`) aktuelle Versionen der entsprechenden Server-Software (s. Kapitel 4.1) installiert. Für IDENT muß auf allen internen Systemen eine sichere Installation vorgenommen werden (s. Abschnitt A.6); an Systeme, für die keine sicheren Implementationen des IDENT-Dienstes verfügbar sind, dürfen von der Packet Screen keine Dateneinheiten für den IDENT-Dienst weitergeleitet werden. SSH stellt ein großes Sicherheitsproblem dar, da hierüber ein Einloggen vom externen Netz auf Systeme des internen Netzes möglich wird. Der Zugriff über SSH wird daher nur auf ein einzelnes besonders zu überwachendes internes System (`ssh-server`”, `192.168.14.3`) zugelassen. Auch hier muß eine aktuelle SSH Version installiert werden (s. Abschnitt A.4).

Um einen Zugriff aus dem externen Netz nur auf die bereits abgesicherten Dienste zuzulassen, soll die Packet Screen Drawbridge-2.0[16] eingesetzt werden.

Drawbridge verwendet eine relativ leicht verständliche Filtersprache, die mit einem “Filter Compiler” in eine für Drawbridge lesbare Form übertragen wird. Die “Filtersprache” soll im folgenden kurz vorgestellt werden.

[16] `ftp://ftp.cert.dfn.de/pub/firewalls/software/drawbridge/`

Service-Spezifikation: Zentrales Element der Sprache ist die "Service-Spezifikation", mit der Dienste (genauer Ports) angegeben werden. Als Beispiel wird folgende "Service-Spezifikation" betrachtet:

```
<smtp/tcp in-out>
```

Die Service-Spezifikation steht immer in `< ... >`. Als Protokoll kann eine Bezeichnung aus der `/etc/services` oder direkt eine Portnummer angegeben werden. Nach dem Schrägstrich wird das Transport-Protokoll angegeben, wobei zur Zeit nur `tcp` und `udp` zugelassen sind. Statt einzelner Ports können auch Portbereiche (z.B. `1-65535/tcp`) angegeben werden.

Nach der Angabe des Protokolls wird die Richtung angegeben. Es gibt 3 Möglichkeiten:

- `in`
  Verbindungen nach innen, also z.B. zu einem internen Mailserver.
- `out`
  Verbindungen von innen nach außen, also z.B. Verbindungen zu externen Mailservern.
- `in-out`
  Verbindungsaufbau in beiden Richtungen.

Durch das Voranstellen eines "`!`" können Angaben negiert werden:
`<!smtp/tcp in>` verbietet also explizit den Verbindungsaufbau an den internen Mailserver.

Gruppen-Spezifikation: Zur Vereinfachung des Aufbaus von Filtern besteht die Möglichkeit, Gruppen zu definieren:

```
define gruppe1 <smtp in>, <telnet in>;
```

Eine Gruppen-Definition beginnt immer mit dem Befehl `define`, danach folgt der Name der Gruppe und eine Aufzählung von Service-Spezifikationen oder anderen Gruppen. Eine Sonderstellung hat die Gruppe `default`. Alle für diese Gruppe definierten Dienste gelten für alle Hosts, die nicht explizit in der Konfigurations-Datei mit einem `host`-Eintrag (s.u.) enthalten sind.

Host-Spezifikation: Die Host-Spezifikation besteht aus dem Steuerwort `host`, dem Namen eines Endsystems und einer Liste von Service-Spezifikationen oder Gruppen.

```
host  mailserver  intern,<smtp/tcp in>;
```

Statt des vollen Rechnernamens kann auch die IP-Adresse des Rechners angegeben werden. Beim Compiler-Durchlauf werden Rechnernamen in IP-Adressen gewandelt.

Spezialitäten: Mit der `reject`-Regel können Pakete, die von bestimmten IP-Adressen kommen, verboten werden. Gibt man mit der `reject`-Regel die IP-Adressen der internen Hosts an, können keine Pakete mehr vom Internet in das interne Netz geschickt werden, die vorgeben, von internen Hosts zu kommen. Ein "IP-Spoofing" der internen Hosts ist somit unmöglich.

Filter Konfiguration: Diese wenigen Konstrukte reichen aus, um die durch die Sicherheitspolitik geforderten Filter zu erstellen. Die Filterkonfiguration für das beschriebene Szenario muß wie folgt aussehen:

```
# Drawbridge 2.0 Konfigurations-Datei
#-----------------------------------
#

# Verkehr von innen nach aussen wird nicht eingeschraenkt.
define  tcpoutok        <1-65535/tcp out>;

# Sicheres Default setzen:
# "Alles was nicht explizit erlaubt ist, ist verboten!"
#
# Die Gruppe "default" trifft fuer alle Rechner zu, die nicht explizit
# mit einem "host"-Eintrag erw"ahnt werden.
define  default         <!1-65535/tcp in-out>,<!1-65535/udp in-out>;


# Interne Systeme duerfen Verbindungen nach aussen aufbauen
# Von aussen duerfen IDENT-Anfragen gestellt werden
define  intern        tcpoutok,<ident/tcp in>;

#
# Gegen IP-Spoofing:
# Alle von aussen eingehenden Pakete, die eine IP-SRC-Addr eines
# internen Hosts (192.168.14.x) haben, muessen vernichtet werden.
#
reject 192.168.14.0    255.255.255.0;

# Die internen Server
host  mailserver  intern,<smtp/tcp in>;
host  nameserver  intern,<domain/tcp in>,<domain/udp in>;
host  sshserver   intern,<22/tcp in>;

# Die internen Hosts
host  hostinter1  intern;
host  hostinter2  intern;
```

```
host  hostinter3  intern;
host  hostinter4  intern;
```

5.4.5 Betrieb eines Firewalls

Bei Firewalls ist die Notwendigkeit zur Überwachung besonders hoch. Für das obige Szenario müssen neben den durch die Packet Screen erzeugten Auditdaten auch die Auditdaten der internen Server überwacht werden. Ein Firewall ist zudem ein guter Ansatzpunkt für Netzwerkmonitoring (s. Kapitel 4.1).

5.5 Weitere Informationen zum Thema Firewalls

Die in diesem Abschnitt vorgestellten Maßnahmen können nur einen groben Überblick über die Planung eines Firewalls geben. Interessierte Leser finden weiteführende Informationen in [**Chapman & Zwicky 1995**, **Cheswick & Bellovin 1994**, Ellermann 1994].

Kapitel 6

Sicherung der Übertragung im Netz

Das IP-Protokoll stellt die Basis für die Kommunikation im Internet dar. Gerade dieses Protokoll erlaubt aber eine Reihe von Angriffen, die seit Anfang 1994 in die Schlagzeilen geraten sind. Da dieses Protokoll auch für andere wesentliche Dienste der Netzinfrastruktur sowie für die Anwendungen genutzt wird, betreffen Sicherheitslücken potentiell alle Anwendungen und Anwender. Diese Situation existiert bereits seit der Entwicklung dieser Protokolle und Anwendungen, doch erst jetzt haben die Angriffe ein solches Maß erreicht, daß sie nicht mehr ignoriert werden können. Möglich ist es in der Regel für einen Angreifer (mit unterschiedlichem Aufwand), die übertragenen IP-Pakete der meisten Protokolle:

mitzulesen: Angreifer erhalten Zugriff auf die in den Paketen enthaltenen Daten und Informationen sowie auf Meta-Informationen über Kommunikationsverbindungen, etc.

fehlzuleiten: Verbindungen kommen nicht zustande und Anwendern werden Dienstleistungen entzogen bzw. Verbindungen werden zu anderen Systemen aufgebaut, die Benutzern die erwartete Oberfläche vortäuschen.

zu manipulieren: Die übermittelten Befehle und Daten werden verändert.

zu fälschen: Angreifer fügen neue Pakete in den Paket-Strom ein, erzeugen ganz neue Verbindungen und fälschen bestimmte Teile (z. B. IP-Adressen) oder ganze Pakete (z. B. Einfügung eines ganzen Pakets mit einer „geratenen“ TCP-Sequenznummer).

6.1 Vorbemerkungen

Konkreter Anlaß für die Anfang 1994 veröffentlichten Schlagzeilen über erfolgreiche Angriffe und die Unsicherheit des Internets war ein drastischer Anstieg gezielter Angriffen mit sogenannten Sniffern. Diese zeichnen die ersten Pakete einer neuen Verbindung auf. Diese Pakete enthalten z. B. bei den Protokollen FTP, Telnet und rlogin üblicherweise Benutzerkennungen und Paßworte. Die spätere Auswertung und das „Knacken“ der aufgezeichneten Paßworte erfolgte dann, um Zugangsberechtigungen zu Tausenden von Systemen zu gewinnen. Da die Sniffer ständig aktiv sein müssen, wurde durch trojanisierte Systemkommandos eine Entdeckung verdächtiger Prozesse erschwert.

Die Angriffe, über die die Öffentlichkeit im Februar 1994 informiert wurde, waren besonders kritisch, da einige der gefundenen Sniffer auf Workstations mit direktem Anschluß an einen Backbone installiert waren. Mit Hilfe der gewonnenen Kennungen wurden auf vielen weiteren lokalen Netzen ebenfalls Sniffer installiert, so daß sich

diese Lauschangriffe immer weiter ausbreiteten. Mitte 1994 wurden auch in Europa und Deutschland installierte Sniffer gefunden und heute gehören sie quasi zu den Standardwerkzeugen der Cracker.

Ein Jahr später machte ein weiteres Problem ähnlich große Schlagzeilen: IP-Spoofing. Darunter werden Angriffe verstanden, bei denen Paketen mit gefälschten Header-Angaben – insbesondere IP-Adressen – genutzt werden, um eine Verbindung aufzubauen und Kommandos an einen Server zu übermitteln, die dieser aufgrund der Absenderadresse akzeptiert.

Neben diesen grundlegenden, inhärenten Problemen der gesamten Infrastruktur des Internets gibt es immer neue Angriffe, durch die Daten abgehört, verfälscht oder vorgetäuscht werden. Daher müssen Sicherheitsmechanismen, die Vertraulichkeit und Authentizität gewährleisten können, sinnvoller Teil der Netzwerke werden. Da aber ein Re-Design des weltweiten Netzes nicht möglich ist, wird mittelfristig nur die Möglichkeit bestehen, einzelne Anwendungen sicherer zu gestalten und spezielle Angriffe abzuwehren.

Unterschiedliche Sicherheitsanforderungen

Aufgrund der vielfältigen Schwachstellen und Sicherheitslücken heutiger Protokolle stellt sich grundlegend die Frage, welche Anforderungen an die Nutzung des Netzes, und damit verbunden an die Sicherheitsmechanismen der verwendeten Anwendungen zu stellen sind. Wie jede Frage, die an ein so subjektiv begriffenes Gefühl wie „Sicherheit“ geknüpft wird, kann es keine allgemeine und für jeden gleichermaßen gültige Antwort geben.

Grundsätzlich bleibt jedoch festzuhalten, das jeder Benutzer Mindest-Sicherheitsanforderungen haben *sollte* – unabhängig von seinem subjektiven Empfinden einer Bedrohung oder einer tatsächlichen „Brisanz“ seiner Daten. Wir alle müssen lernen, daß viele kleine Informationen über uns in öffentlichen Datennetzen „entstehen“ bzw. verfügbar sind. Diese können zu einem Bild zusammengesetzt werden, und bevor wir alle zu „gläsernen Menschen“ werden, sollte bereits den Anfängen begegnet werden. Folgende Sicherheitsanforderungen sind für die meisten Anwendungssituationen angemessen:

- Der Zugriff erfolgt nur auf authentisierte Server und Informationsangebote, d. h. der Benutzer ist sich sicher, daß er den *richtigen* Server nutzt.
- Der Zugriff erfolgt vertraulich, d. h. während der Nutzung übertragene und empfangene Daten sowie dabei entstehende Meta-Informationen sind Angreifern nicht zugänglich.

- Übertragene und empfangene Daten können zu einem späteren Zeitpunkt nicht mißbraucht werden, insbesondere ist ein Wiedereinspielen nicht möglich.
- Übertragene und empfangene Daten können nicht (unbemerkt) durch Angreifer verändert werden.
- Empfangene Daten sind verbindlich, d. h. der Server/Informationsanbieter übernimmt die Verantwortung für die gelieferten Daten.

Diese Anforderungen haben auch die Betreiber von Servern und Informationsangeboten selbst, allerdings mit vertauschten Rollen. So werden die Authentisierung des Benutzers und die Verbindlichkeit der angestoßenen Aktionen wichtige Anforderungen, die aber eventuell auch etwas vernachlässigt werden können, wenn es sich nur um ein allgemeines Informationsangebot handelt.

Die Sicherheit des Benutzers wird natürlich auch durch andere Faktoren beeinflußt, z. B. durch die Bearbeitung seiner Kreditkartennummer bei dem Betreiber eines e-Commerce-Servers. Solche Bedrohungen liegen aber außerhalb des Bereiches, der durch die in diesem Kapitel behandelten kryptographischen Verfahren gesichert werden kann. Im Mittelpunkt steht die Sicherheit bei der Nutzung offener Netze für eine Kommunikation.

Kryptographische Verfahren

Wenn eine Person Informationen vor anderen „verstecken“ möchte, gibt es zwei Möglichkeiten:

1. **Kryptographie (geheimes Schreiben):** Die Daten werden so verändert, daß andere diese nicht mehr lesen können. Es bleibt jedoch erkennbar, das Daten vorhanden sind.
2. **Steganographie (verborgenes Schreiben):** Die Daten werden in eine Form gebracht, daß andere diese nicht mehr als Daten erkennen.

Während die Kryptographie lange Zeit eine Domäne des Militärs war, gibt es heute frei verfügbare Lösungen für jeden Anwender. Immer mehr werden heute auch steganographische Verfahren entwickelt und eingesetzt, d. h. Informationen werden z. B. als Pixel in GIF-Bildern kodiert. Dabei kann im allgemeinen ohne weitere Hinweise nicht entschieden werden, ob es sich tatsächlich um verborgene Daten, einen Übertragungs- bzw. Codierungsfehler oder einfach um das originale Bild handelt. Diese Verfahren gewinnen auch an Bedeutung, da auf politischer Ebene immer wieder ein Verbot starker kryptographischer Verfahren diskutiert wird.

Kryptographische Verfahren und ihre Anwendung sollen Gegenstand der folgenden Betrachtungen sein.

6.2 Kryptographie

Einen sehr großen Stellenwert beim Schutz von Informationen hat seit jeher der Einsatz von kryptographischen Verfahren. Bei der Benutzung von Computern lassen sich dabei grundsätzlich drei Verfahren unterscheiden:

- Verschlüsselungsverfahren
- Signaturverfahren
- Prüfsummenverfahren

6.2.1 Verschlüsselungsverfahren

Als **Verschlüsselung** (Chiffrierung) wird die Umwandlung eines lesbaren Textes (*Klartext*) in einen nicht-lesbaren Text (*Chiffretext*) bezeichnet; der entgegengesetzte Prozeß wird **Entschlüsselung** (Dechiffrierung) genannt [Pfleeger 1989].

Die Sicherheit eines Verschlüsselungs-Systems – also die Fähigkeit, möglichen Angriffen zu widerstehen – hängt in erster Linie vom verwendeten Algorithmus ab. Allerdings kommt auch der Qualität der gewählten Schlüssel eine entscheidende Rolle zu, da auch ein guter Algorithmus mit „schlechten“ (z. B. zu kurzen) Schlüsseln keine ausreichende Sicherheit gewährleisten kann. Ein vollständig sicheres Verschlüsselungs-System würde es einem potentiellen Angreifer unmöglich machen, aus einem erzeugten Chiffretext ohne den verwendeten Schlüssel den korrespondierenden Klartext zu erhalten [Denning 1982].

Es werden zwei grundsätzlich verschiedene Verschlüsselungs-Systeme unterschieden: die **symmetrische** und die **asymmetrische** Verschlüsselung.

Charakteristisch für symmetrische Verschlüsselungs-Systeme ist die Verwendung *eines* Schlüssels sowohl bei der Verschlüsselung als auch bei der Entschlüsselung (*single key* bzw. *secret key*): Eine Nachricht, welche mit einem beliebigen Schlüssel verschlüsselt wurde, kann nur mit dem identischen Schlüssel wieder in den Klartext umgewandelt werden. Zwei (oder mehrere) Kommunikationspartner, welche verschlüsselte Nachrichten austauschen wollen, müssen also vor der eigentlichen Kommunikation Maßnahmen einleiten, den Schlüssel auf sicherem Wege untereinander auszutauschen oder zu vereinbaren. Haben sich die Kommunikationspartner auf einen Schlüssel geeinigt, wird dieser Schlüssel anschließend sowohl zur Verschlüsselung als auch zur Entschlüsselung benutzt („Symmetrie“).

Bekannte Beispiele für symmetrische Verschlüsselungs-Systeme sind: der *Data Encryption Standard* (DES); Triple-DES, eine Variante von DES; der von der US-ameri-

kanischen Sicherheitsbehörde NSA entwickelte Algorithmus *Skipjack* [Brickell et al. 1993] sowie der *International Data Encryption Algorithm* (IDEA).

Im Gegensatz zu den symmetrischen werden bei den asymmetrischen Verschlüsselungs-Systemen *zwei* unterschiedliche – aber mathematisch voneinander abhängige – Schlüssel zum Ver- und Entschlüsseln benutzt, von denen ein Schlüssel veröffentlicht werden kann (*public key*), der andere Schlüssel jedoch geheimgehalten werden muß (*secret key* oder auch *private key*). Eine verschlüsselte Nachricht kann nur mit dem jeweils anderen Schlüssel entschlüsselt werden [Nechvatal 1991].

Üblicherweise wird der öffentliche Schlüssel eines Kommunikationspartners benutzt, um einen Klartext zu verschlüsseln; der Chiffretext kann dann nur mit dem korrespondierenden geheimen Schlüssel entschlüsselt werden.

Ein Vorteil der asymmetrischen gegenüber den symmetrischen Verschlüsselungs-Methoden ist der erheblich geringere Bedarf an Schlüsseln. In symmetrischen Systemen muß jeder Benutzer mit jedem potentiellen Kommunikationspartner einen geheimen Schlüssel vereinbaren: Ein System mit n Benutzern erfordert dann $n * (n-1)/2$ verschiedene Schlüssel. Die Generierung, Verteilung und Verwaltung dieser Schlüssel stellt in der Praxis ein großes Problem dar; für jeden neuen Benutzer des Systems steigt die Anzahl der benötigten Schlüssel sehr schnell.

In Public Key-Systemen spielt dieses Problem eine sehr viel geringere Rolle, da jeder Benutzer **genau einen** Schlüssel zum Entschlüsseln benötigt. Nachteilig ist allerdings die deutlich niedrigere Performanz gegenüber Secret Key-Systemen.

Bekannte Beispiele für asymmetrische Verschlüsselungs-Systeme sind: das von Whitfield Diffie und Martin Hellman vorgeschlagene Verfahren des *exponentiellen Schlüsselaustauschs* [Diffie & Hellman 1976]; das *RSA-Verfahren* nach Rivest, Shamir und Adleman [Rivest et al. 1978] sowie der Algorithmus von Taher ElGamal [ElGamal 1985].

Benutzung von Public Key-Systemen Generell können Public Key-Systeme auf zwei grundsätzlich verschiedene Arten benutzt werden, je nachdem, welcher Schlüssel zum Verschlüsseln verwendet wird.

1. Wird der öffentliche Schlüssel zum Verschlüsseln verwendet, kann jeder Benutzer an den Besitzer des korrespondierenden geheimen Schlüssels eine vertrauliche Nachricht senden. Nur der Empfänger kann mit seinem geheimen Schlüssel den Chiffretext entschlüsseln. Durch dieses Verfahren wird die **Vertraulichkeit** einer Nachricht gewährleistet.

2. Verschlüsselt ein Kommunikations-Teilnehmer einen Klartext mit seinem geheimen Schlüssel, kann jeder Empfänger des Chiffretextes die Nachricht mit dem öffentlichen Schlüssel entschlüsseln. Auf diese Weise ist die **Authentizität** der

Nachricht gewährleistet, da nur der Besitzer des geheimen Schlüssels den Chiffretext erzeugt haben kann.

Beide Verfahren lassen sich kombinieren, wie noch gezeigt werden wird.

6.2.2 Signaturverfahren

Elektronische Signaturverfahren stellen ein Pendant zu handschriftlichen Unterschriften dar und sollen die eindeutige Zuordnung eines elektronischen Dokumentes zu einem bestimmten Benutzer ermöglichen. Eine elektronisch erzeugte Unterschrift wird als *digitale Signatur* bzw. *elektronische Signatur* bezeichnet und soll wie eine handschriftliche Unterschrift die Echtheit (**Authentizität**) des unterschriebenen Dokumentes garantieren.

Das Konzept der digitalen Signatur wurde in [Diffie & Hellman 1976] im Zusammenhang mit der Public Key-Verschlüsselung erstmals beschrieben. Diffie und Hellman stellten drei Forderungen an Signaturverfahren:

1. Die digitale Unterschrift muß vom Dokument **und** vom Unterzeichner abhängig sein.

2. Die digitale Unterschrift kann **ausschließlich** vom autorisierten Benutzer erzeugt werden.

3. Die Korrektheit der digitalen Unterschrift muß von jedem Benutzer überprüft werden können.

Die Algorithmen, die den Signaturverfahren zugrundeliegen, sind den im vorigen Abschnitt beschriebenen asymmetrischen Verschlüsselungsalgorithmen ähnlich, da auch Signaturalgorithmen Zwei-Schlüssel-Systeme sind. Der Hauptunterschied liegt in der unterschiedlichen Verwendung des geheimen Schlüssels. Der geheime Schlüssel wird hier zur Erzeugung der Signatur verwendet; jeder Besitzer des korrespondierenden öffentlichen Schlüssels kann mit diesem die Gültigkeit der Unterschrift feststellen, wenn es ihm gelingt, den unterzeichneten Klartext zu erhalten. Liefert die Anwendung des öffentlichen Schlüssels nicht den ursprünglichen Klartext, wurde entweder der signierte Text nachträglich modifiziert oder versucht, die Unterschrift zu fälschen – die digitale Signatur wäre ungültig.

Der im Internet am häufigsten benutzte Signaturalgorithmus ist der RSA-Algorithmus.

6.2.3 Prüfsummenverfahren

Prüfsummenverfahren wurden entwickelt, um die **Integrität** (Unverfälschtheit) von Informationen garantieren zu können. Der Begriff der Integrität umfaßt dabei Merkmale wie Vollständigkeit sowie Korrektheit eines Textes.

„Einfache“ Prüfsummenverfahren wie das CRC-Verfahren (*Cyclic Redundancy Check*) berechnen anhand leicht zu implementierender mathematischer Funktionen einen Wert für einen vorgegebenen Text, welcher zu einem späteren Zeitpunkt erneut berechnet und verglichen werden kann. Wurde der Text modifiziert, ändert sich die Prüfsumme. Eine Beispiel-Implementation stellt das UNIX-Programm *sum* dar, welches eine 16 Bit lange Prüfsumme berechnet. Die Sicherheit solcher einfachen Verfahren ist allerdings sehr unzureichend: es ist für einen potentiellen Angreifer leicht möglich, den Klartext derart zu verändern, daß das Prüfsummenprogramm den gleichen Wert berechnet. Aus diesem Grund sollen im Verlauf dieser Arbeit lediglich *kryptographische* Prüfsummenverfahren, die sog. *Hash*-Funktionen, betrachtet werden.

Hash-Funktionen sind Einweg-Funktionen, welche einen Text variabler Länge als Funktionsargument übernehmen und anhand kryptographischer Algorithmen einen Wert fester Länge (den *Hash*-Wert) erzeugen. Die zugrundeliegenden Algorithmen können dabei ganz unterschiedlicher Natur sein. Die Idee hinter diesen Hash-Funktionen ist die Annahme, daß keine zwei verschiedenen Texte den selben Hash-Wert erzeugen können.

Verbreitete Hash-Funktion sind *MD2* (MD für *message digest*, eine andere Bezeichnung für eine Hash-Funktion) von Burt Kaliski, *MD4* und *MD5* von Ron Rivest [RFC 1319, RFC 1320, RFC 1321], der von der US-Normungsbehörde NIST bekanntgegebene *Secure Hash Algorithm* (SHA) sowie die europäische Entwicklung *RIPEMD-160* [Dobbertin et al. 1996, ISO/IEC 10118-3].

Prüfsummenverfahren werden häufig gemeinsam mit Signatur- und Verschlüsselungs-Verfahren eingesetzt. Dieser kombinierte Einsatz von verschiedenen Methoden wird daher *Hybrid-Kryptographie* genannt.

6.2.4 Hybridverfahren

Die in den vorigen Abschnitten vorgestellten kryptographischen Verfahren können in verschiedenen Kombinationen gemeinsam eingesetzt werden. Neben einer besseren Performanz soll solch ein kombinierter Einsatz vor allem die Gesamtsicherheit gegenüber der Verwendung lediglich einzelner Verfahren deutlich erhöhen.

Zu den typischen Implementationen hybrider Verfahren gehören: (1) die Kombination von Signatur- und Prüfsummenverfahren sowie (2) die gemeinsame Benutzung symmetrischer und asymmetrischer Verschlüsselungsalgorithmen.

Kombination von Signatur- und Prüfsummenverfahren Die Kombination dieser beiden Verfahren verbessert das Konzept der digitalen Signatur. In diesem Fall wird der Signaturalgorithmus nicht direkt auf den zu signierenden Text angewendet, sondern auf einen vorher erzeugten Hash-Wert des Textes:

- Der Absender einer Nachricht m erzeugt zunächst mit einem Prüfsummenverfahren (z. B. MD5) den Hash-Wert h dieser Nachricht, welcher anschließend mit einem Signaturalgorithmus (z. B. RSA) unter Verwendung des privaten Schlüssels verschlüsselt wird. Die Nachricht und der signierte Hash-Wert werden dann an den (oder die) vorgesehenen Empfänger weitergeleitet.

- Der Empfänger entschlüsselt den unterschriebenen Hash-Wert mit dem öffentlichen Schlüssel des Unterzeichners und erhält den Hash-Wert h. Anschließend bildet er mit demselben Prüfsummenverfahren wie der Unterzeichner einen Hash-Wert der Nachricht und erhält h'. Nur wenn h und h' übereinstimmen, ist die Signatur gültig.

Neben der Authentizität der Nachricht (nur eine Person mit Zugriff auf den privaten Schlüssel kann eine gültige Signatur erzeugt haben), läßt sich durch die Kombination von Signatur- und Prüfsummenverfahren auch die Integrität einer Nachricht dergestalt gewährleisten, daß jede Veränderung an der Nachricht festgestellt werden kann, da eine unbefugte Manipulation oder eine zufällige Veränderung auf dem Transportweg am Nachrichteninhalt einen abweichenden Hash-Wert zur Folge hätte.

Benutzung symmetrischer und asymmetrischer Verschlüsselungsalgorithmen Die gemeinsame Verwendung symmetrischer und asymmetrischer Verschlüsselungsalgorithmen garantiert die Vertraulichkeit einer Nachricht derart, daß ausschließlich der vorgesehene Empfänger den Inhalt der Nachricht lesen kann:

- Der Absender einer zu schützenden Nachricht generiert zunächst auf Zufallsbasis einen Nachrichten-Schlüssel k, der zum Verschlüsseln der Nachricht mittels eines symmetrischen Verfahrens (z. B. DES) verwendet wird. Anschließend wird der öffentliche Schlüssel des vorgesehenen Empfängers benutzt, um den Nachrichten-Schlüssel k durch ein asymmetrisches Verfahren (z. B. RSA) zu verschlüsseln. Sowohl der Chiffretext als auch der verschlüsselte Nachrichten-Schlüssel werden dann an den Empfänger übermittelt.

- Der Empfänger erhält nach Anwendung seines geheimen Schlüssels den ursprünglichen Nachrichten-Schlüssel k, welcher die Entschlüsselung des Chiffretextes ermöglicht.

Die Kombination der beiden Verschlüsselungsverfahren bringt einen deutlichen Performanzgewinn, da das langsamere asymmetrische Verfahren nur noch zur Verschlüsselung eines gegenüber der Nachricht erheblich *kleineren* Nachrichten-Schlüssels verwendet wird. Die eigentliche Nachricht wird dagegen mit sehr schnellen symmetrischen Verfahren verschlüsselt.

Die beiden beschriebenen Hybridverfahren lassen sich ihrerseits kombinieren, um sowohl die Vertraulichkeit als auch die Authentizität und Integrität einer Nachricht zu gewährleisten. Dabei führen unterschiedliche Reihenfolgen in der Anwendung der kryptographischen Routinen zu unterschiedlichen Ergebnissen.

6.3 Email: PEM, PGP und S/MIME

Wie bereits gezeigt (S. 81), stellen die konventionellen Email-Protokolle wie SMTP keine Sicherheitsdienste bereit. Sollen dennoch Sicherheitsmechanismen implementiert werden, bieten sich dafür zwei grundsätzlich verschiedene Verfahren an: (1) die Entwicklung eines **komplett neuen** Nachrichtenprotokolls, welches entsprechende Mechanismen auf einer für den Benutzer nicht unmittelbar sichtbaren Ebene implementiert und (2) die Integration von Sicherheitsmechanismen in Applikationen, welche oberhalb der Transport-Protokolle implementiert sind („peer-to-peer“).

Obwohl die Entwicklung eines neuen Nachrichten-Protokolls den konsequenteren und eleganteren Weg bedeuten würde, stellt die zweite Option eine akzeptable Lösung dar, da mit sicheren Applikationen die schon existierenden Protokolle benutzt werden könnten und der sofortige Einsatz ermöglicht werden würde. Diesen Ansatz verfolgen drei Verfahren bzw. Standards zum Versand von sicherer Email: PEM, PGP und S/MIME. Sie sollen in diesem Abschnitt vorgestellt und miteinander verglichen werden.[1]

6.3.1 Überblick

PEM (*Privacy Enhanced Mail*), PGP (*Pretty Good Privacy*) und S/MIME (*Secure/Multipurpose Internet Mail Extensions*) sind Verfahren, welche durch den Einsatz verschiedener kryptographischer Mechanismen den sicheren Versand von Email über das Internet ermöglichen.

Sie basieren beide auf grundsätzlich ähnlichen Verfahren, nämlich der Verschlüsselung und Signierung von Nachrichten vor dem Versand über ein unsicheres Netzwerk, unterscheiden sich jedoch entscheidend bei der Verwaltung der Schlüssel. Daher soll an

[1] Die in diesem Artikel verwendeten Abbildungen und Bildschirmausschnitte stellen lediglich Beispiele dar und sollen als Anhaltspunkt für die Benutzung der Programme gesehen werden.

dieser Stelle zunächst auf die allgemeinen Verfahren eingegangen werden, welche sowohl von PEM als auch von PGP und S/MIME unterstützt werden. Die Unterschiede – vor allem in Fragen der Schlüsselverwaltung – werden dann anschließend in gesonderten Abschnitten hervorgehoben.

PEM, PGP und S/MIME basieren maßgeblich auf asymmetrischen Kryptoverfahren. Jeder Benutzer solcher Verfahren benötigt zunächst ein Schlüsselpaar, welches er entweder selbständig generiert (Einzelheiten zur Schlüsselgenerierung sind in der jeweiligen Programmdokumentation zu finden) oder von einer vertrauenswürdigen Stelle generieren läßt. Einer der beiden Schlüssel – der *public key* – kann und sollte veröffentlicht werden. Er gibt anderen Benutzern im Netzwerk die Möglichkeit, verschlüsselte Nachrichten an den Schlüsselbesitzer zu senden. Der andere Schlüssel – der *private* oder auch *secret key* – darf unter keinen Umständen anderen Personen als dem Besitzer bekannt werden. Mit ihm werden verschlüsselte Nachrichten entschlüsselt sowie eigene Nachrichten signiert.

PEM, PGP und S/MIME benutzen die im Abschnitt 6.2.4 beschriebenen Kombinationen verschiedener kryptographischer Verfahren. Die für beide Programme *obligatorischen* Dienste Integrität und Authentizität werden erreicht, indem zunächst eine Hash-Summe über den vorgesehenen Text gebildet wird und dieser Hash-Wert dann anschließend mit dem eigenen geheimen Schlüssel signiert wird. Bei einer optionalen Verschlüsselung des Textes wird von der Software auf Zufallsbasis ein Nachrichten-Schlüssel (*session key*) gebildet, welcher nur für diese eine Nachricht gültig ist. Der Text wird dann mit einem symmetrischen Verschlüsselungsalgorithmus unter Zuhilfenahme dieses session keys verschlüsselt. Danach wird der session key mit dem öffentlichen Schlüssel des vorgesehenen Empfängers verschlüsselt und zusammen mit dem verschlüsselten Text übermittelt. Nur der vorgesehene Empfänger kann mit seinem geheimen Schlüssel den session key wieder herstellen und den Nachrichtentext entschlüsseln.

Soll eine Nachricht an mehrere Empfänger verschlüsselt gesendet werden, wird für jeden vorgesehenen Empfänger eine verschlüsselte Version des session keys erzeugt und mit der Nachricht versendet. Auf diese Weise ist es nicht notwendig, die Nachricht für jeden Empfänger einzeln zu verschlüsseln.

Sowohl zum Signieren wie auch zum Entschlüsseln einer Nachricht muß der Zugriff auf den eigenen secret key erfolgen können. Um diesen Schlüssel vor dem Mißbrauch durch Unbefugte zu schützen, wird er von Programmen, die PEM, PGP oder S/MIME implementieren, in aller Regel nur in *verschlüsselter* Form im Dateisystem – üblicherweise in einem für keinen anderen Benutzer zugreifbaren Verzeichnis – oder sogar auf einer SmartCard abgelegt. Erst nach Eingabe eines Passwortes – bei der SmartCard ist es meist eine Geheimzahl, die PIN (*personal identification number*) kann auf den secret key zugegriffen werden.

6.3.2 Privacy Enhanced Mail / PEM

PEM wurde bereits 1988 in drei Internet RFCs definiert. Diese RFCs wurden jedoch im Jahre 1993 komplett überarbeitet und neu veröffentlicht. Die überarbeiteten RFCs [RFC 1421, RFC 1422, RFC 1423, RFC 1424] umfassen die Spezifikation fast aller zur Zeit verfügbaren PEM-Implementationen.

Die einzige in Deutschland frei erhältliche PEM-Implementation („SecuDE“) wurde von der GMD (Darmstadt) entwickelt [Schneider 1994a, Schneider 1994b] [Schneider 1994c, Schneider 1994d]; die in den USA hergestellten Implementationen (z. B. von der Firma TIS) dürften bis Anfang 2000 aufgrund strenger Exportbestimmungen nicht von dort exportiert werden.

PEM identifiziert die einzelnen Benutzer auf Basis von sog. „Distinguished Names“ (DN). Dieses Konzept wurde aus dem *X.500 Directory Service* übernommen und soll eine einfache und einheitliche Namensgebung ermöglichen [CCITT 1988, Deutsch 1988]. Ein DN beinhaltet eine Folge von eindeutig kennzeichnenden Namensattributen. Diese umständlich zu handhabenden Namen lassen sich jedoch durch die Benutzung von Aliasnamen erheblich vereinfachen.

Schlüsselmanagement Um die Authentizität des öffentlichen Schlüssels zu gewährleisten, basiert PEM auf dem Konzept der Zertifizierungsinstanzen: Jeder Benutzer läßt seinen öffentlichen Schlüssel von einer übergeordneten vertrauenswürdigen Instanz signieren. Der so signierte öffentliche Schlüssel wird Zertifikat genannt und kann von jedem anderen Benutzer verifiziert werden, welcher den öffentlichen Signierschlüssel der Zertifizierungsinstanz kennt. Ein Zertifikat stellt eine *eindeutige Verbindung* zwischen der Identität des Benutzers und seinem öffentlichen Schlüssel her. Zertifizierungsinstanzen lassen sich ihrerseits durch übergeordnete Instanzen zertifizieren. Die Zertifizierungsinstanzen sind bei PEM hierarchisch (baumförmig) strukturiert; die Wurzel des Internet-Zertifizierungssystems bildet die IPRA (Internet Policy Registration Authority), welche von der Internet Society – bis heute allerdings nur testweise – betrieben wird. Jeder Benutzer kann also mit Hilfe *eines* Schlüssels – des IPRA public keys – alle darunterliegenden Instanzen und Benutzer verifizieren. [Kent 1993]

Ein Benutzer, der eine PEM-Software verwenden will, generiert sein eigenes Schlüsselpaar und sendet dann seinen öffentlichen Schlüssel an eine Zertifizierungsinstanz. Das von der Instanz zurückgesendete Zertifikat kann dann im Internet veröffentlicht werden, um anderen Benutzern den eigenen öffentlichen Schlüssel zugänglich zu machen, ohne Fälschungen am öffentlichen Schlüssel befürchten zu müssen. Die Veröffentlichung von Zertifikaten und „Schwarzen Listen“ (Sperrlisten, *Certification Revocation Lists, CRL*) kann über Datenbanken erfolgen, welche z. B. von den Zertifizierungsinstanzen betrieben werden; eine andere Möglichkeit besteht in der Verteilung über den X.500 Directory Service, welcher ein verteiltes Verzeichnis darstellt.

Jede Zertifizierungsinstanz hat bestimmte Richtlinien (eine sog. *Policy*) als Grundlage, welche beispielsweise den Vorgang der Zertifizierung beschreiben. Einzelheiten hierzu sind in [Roe 1992] beschrieben.

6.3.3 Pretty Good Privacy / PGP

PGP wurde 1991 von Phil Zimmermann in den USA entwickelt [Zimmermann 1993, **Garfinkel 1995**]. Aufgrund der bisherigen US-Exportbeschränkungen existieren mittlerweile internationale PGP-Versionen, welche sich von den Originalversionen durch abweichende kryptographische Routinen unterscheiden. Die Kompatibilität der verwendeten Schlüssel und Nachrichten zwischen den US- und den internationalen PGP-Versionen ist jedoch gewährleistet. (Interoperabilitätsprobleme kann es z. T. zwischen älteren PGP-Versionen (2.6.x) und neueren Versionen (5.x oder höher) geben.

Die Kryptoalgorithmen von PGP sind fest vorgegeben: implementiert sind in den PGP-Versionen bis 2.6.3 die Algorithmen IDEA, RSA und MD5. Alternative Verfahren waren in PGP zunächst nicht vorgesehen. Der OpenPGP-Standard [RFC 2440] läßt nun beliebige Verfahren zu, und die PGP-Versionen ab PGP 5.0 unterstützen (meist zusätzlich zu IDEA und RSA) u. a. Public-Key-Verschlüsselung nach ElGamal (von PGP als Diffie-Hellman bezeichnet) und Signaturen mit SHA-1 (*Secure Hash Algorithm*) nach dem US-*Digital Signature Standard* (DSS) [NIST 1994] sowie symmetrische Verschlüsselungsverfahren wie Triple-DES (3DES), CAST [RFC 2144] oder Blowfish [Schneier et al. 1994].

Auch PGP generiert bei der Bearbeitung von Klartexten einen Header, der aber – im Unterschied zu PEM – an die Nachricht angefügt wird. Außerdem existieren keine einzelnen Header-Einträge; der Header liegt in PGP als Block vor. Nachricht und Header werden in Begrenzungszeilen eingeschlossen, die allerdings je nach Nachrichtentyp unterschiedlich sind.

Das Überprüfen einer (gültigen) PGP-Signatur mit *PGP 2.6.2i* führt zu folgender Bildschirm-Meldung:

```
host> pgp test-message.asc

[...]

File has signature.  Public key is required to check signature.
Good signature from user "Stefan Kelm, DFN-PCA <kelm@pca.dfn.de>".
Signature made 2000/03/31 14:05 GMT
```

Die Identifizierung eines PGP-Benutzers geschieht durch eine frei wählbare *User-ID* (hier: der in Anführungszeichen stehende Text). Ein öffentlicher Schlüssel – welcher vor der Überprüfung vorhanden sein muß – wird durch seinen Hash-Wert als „Fingerprint“ eindeutig bestimmt und mittels einer numerischen *Key-ID* referenziert.

Schlüsselmanagement Auch PGP kennt das Prinzip der Public Key-Zertifikate: sie bestehen in erster Linie aus der User-ID, dem korrespondierenden öffentlichen Schlüssel und der Unterschrift des Zertifikat-Ausstellers (Unterzeichners). PGP speichert diese Zertifikate in einer Datei; dem sog. *(public) key ring*. Die Zertifikate lassen sich einzeln oder in Gruppen aus diesem „Schlüsselbund“ extrahieren und können an beliebige andere Benutzer verteilt werden. (Den kompletten key ring selbst sollte man nicht weitergeben, weil er neben den Zertifikaten auch noch persönliche, vom Inhaber des key rings vergebenen Vertrauensbewertungen der Inhaber aller dort vorhandenen öffentlichen Schlüssel enthält, also eine Information, die nicht unbedingt für Dritte bestimmt ist.)

Die Prüfung des öffentlichen Schlüssels eines anderen Benutzers erfolgt durch den Vergleich des Fingerprints, den man „out-of-band“, d. h. auf sicherem Wege (Telefon, Visitenkarte, etc.) direkt von der betreffenden Person erhalten hat. Außerdem existiert in PGP das Konzept der sog. „Introducer“. Dieses sind Personen (bzw. deren Schlüssel), welche man als vertrauenswürdig einstuft und von denen Public Key-Zertifikate für die Schlüssel anderer Benutzer akzeptiert werden. Der Zertifizierungsvorgang selbst ist bei PGP jedoch nicht geregelt.

Für die Verteilung der PGP-Zertifikate haben sich im Internet sog. *Public Key-Server* etabliert, bei denen jeder Benutzer seinen eigenen öffentlichen Schlüssel speichern lassen sowie die dort gespeicherten Schlüssel anderer Benutzer abrufen kann.[2] Die Informationen dieser Server sind durch einfache Kommandos per Email oder via WWW-Interface abrufbar (vgl. DIB-94:05 *PGP*[3]).

Da die einzelnen Server ihre lokalen Datenbanken in regelmäßigen Abständen untereinander abgleichen, reicht es aus, mit *einem* der Server zu kommunizieren.

Die Public Key-Server übernehmen lediglich eine reine *Speicher*funktion; die *Korrektheit* der gespeicherten Schlüssel wird nicht geprüft oder gewährleistet. Daher muß jeder Benutzer nach Erhalt eines Zertifikates durch einen Public Key-Server die Authentizität (die Echtheit) dieses Zertifikates selbst überprüfen. Nur wenn die Authentizität des Zertifikates feststeht, sollte es in den eigenen Public Key-Ring aufgenommen werden.

Soll eine verschlüsselte PGP-Nachricht entschlüsselt werden, ist die Freigabe des geheimen Schlüssels erforderlich, wie der folgende Dialog verdeutlicht (die Eingabe der „pass phrase“ ermöglicht den Zugriff auf den IDEA-verschlüsselten secret key):

```
host> pgp test-message.pgp

[...]
```

[2] Die DFN-PCA betreibt einen solchen PGP-Keyserver. Sein WWW-Interface ist unter der Adresse `http://blackhole.pca.dfn.de/` zugänglich.

[3] `http://www.cert.dfn.de/infoserv/dib/dib-9405.html`

```
File is encrypted.  Secret key is required to read it.
Key for user ID: Stefan Kelm, DFN-PCA <kelm@pca.dfn.de>
2048-bit key, Key ID 603F2D01, created 1998/04/20

You need a pass phrase to unlock your RSA secret key.
Enter pass phrase: Pass phrase is good.  Just a moment......
```

Warnmeldungen Soll ein PGP-Dokument entschlüsselt werden, welches für andere Empfänger vorgesehen ist, erscheint folgender Dialog:

```
host> pgp test-message.pgp

[...]

File is encrypted.  Secret key is required to read it.
This message can only be read by:
  Stephen A. Weeber <weeber@llnl.gov>
  David A. Curry <davy@ecn.purdue.edu>
  Eugene H. Spafford <spaf@cs.purdue.edu>

You do not have the secret key needed to decrypt this file.

For a usage summary, type:  pgp -h
For more detailed help, consult the PGP User's Guide.
```

Die signierten (und verschlüsselten) Nachrichten lassen sich sowohl von PEM als auch von PGP im ASCII-Format erzeugen, so daß sie anschließend in jede normale Internet-Email eingebunden und verschickt werden können.

6.3.4 Secure MIME / S/MIME

Aus dem Bestreben heraus, nicht nur Text-, sondern auch MIME-Mails, z. B. mit multimedialen Dokumenten als Anlage, „sicher“ versenden zu können, entstand S/MIME [RFC 2311, RFC 2312, RFC 2632, RFC 2633].

S/MIME verwendet für die Public-Key-Zertifikate eine fortentwickelte, flexible X.509v3-Version [ISO/IEC 9594-8], die die Erfahrungen berücksichtigt, die mit PEM und dem darin verwendeten älteren X.509*v1*-Zertifikat- und -Sperrlisten-Format [CCITT 1991] gemacht wurden. Das Zertifikatformat nach X.509v3 ist inzwischen als De-Facto-Standard für Public-Key-Zertifikate weltweit etabliert ist und wird auch von den meisten anderen Public-Key- oder Zertifikat-basierten Anwendungen unterstützt. (Eine der wenigen Ausnahmen stellt bislang PGP dar; in PGP-Version 6.5 gibt es inzwischen erstmalig auch eine Schnittstelle zum Import von X.509-Zertifikaten.)

S/MIME sieht, wie PEM, eine Vielzahl von kryptographischen Algorithmen vor, die bei Bedarf um neue Verfahren flexibel ergänzt werden können.

Schlüsselmanagement S/MIME setzt auf der von der IETF in [RFC 2459] „profilierten“ X.509-Zertifizierungsinfrastruktur auf. Sie ähnelt in ihren Grundzügen dem Ansatz einer hierarchischen Zertifizierungsstellen-Struktur, der auch in PEM verfolgt wurde, setzt also auch das Vorhandensein von Zertifizierungsinstanzen und von durch sie ausgestellten Zertifikaten für die Schlüssel der S/MIME-Nutzer voraus, ist aber u. a. dank des weiterentwickelten X.509-Zertifikatformates flexibler als die in PEM vorausgesetzte (und so nie wirklich in breitem Umfang realisierte) hierarchische Zertifizierungsstruktur. Unter anderem sind nun nicht mehr zwingend alle Zertifizierungsinstanzen in einer Hierarchie einer Wurzelinstanz untergeordnet, sondern es ist vielmehr möglich, daß mehrere hierarchische Zertifizierungsstrukturen losgelöst voneinander existieren. Interoperabilität wird dann durch Cross-Zertifizierung zwischen Instanzen aus unterschiedlichen Hierarchien erreicht.

Ferner sind nun nicht mehr bloß der X.500-Verzeichnisdienst oder das Versenden von Zertifikaten per E-Mail als Verteilungsmechanismen für Zertifizierungsinformationen vorgesehen. Auch die Verteilung von Zertifikaten oder Sperrinformationen z. B. via HTTP oder mittels LDAP [RFC 1777] sind nun möglich. Dadurch ist S/MIME auch ohne Vorhandensein eines Verzeichnisdienstes anwendbar.

6.3.5 Bewertung der Verfahren

PEM, PGP und S/MIME definieren Mechanismen zur Verschlüsselung und zur digitalen Signierung von beliebigen Dokumenten. Alle unterstützen ähnlich starke kryptographische Algorithmen. Einige S/MIME-Implementierungen (meist Exportversionen von US-Programmen) unterstützen allerdings nur schwächere Algorithmen bzw. starke Algorithmen mit künstlich herabgesetzter (und dadurch weniger sicherer) Schlüssellänge. Die erzeugten Nachrichten aller drei Verfahren lassen sich unabhängig vom darunterliegenden Transportprotokoll bearbeiten und können u. a. als ASCII-Texte innerhalb eines beliebigen Email-Dokumentes transportiert werden.

Manche PEM- und S/MIME-Implementierungen bietet die Möglichkeit, SmartCards statt verschlüsselter Dateien als Träger für die geheimen Schlüssel einzusetzen. Ein entsprechendes Feature ist bei PGP für Version 7.0 erst angekündigt.

Grundsätzlich unterscheiden sich PEM sowie S/MIME auf der einen und PGP auf der anderen Seite in den Funktionen des Schlüsselaustauschs. PEM- bzw. S/MIME-Schlüssel werden von Zertifizierungsinstanzen signiert; durch eine Hierarchie von Instanzen (oder die Überkreuz-Zertifizierung, Cross-Zertifizierung, zwischen Zertifizierungsinstanzen) entstehen formale Zertifizierungspfade, die eine einfache Überprü-

fung von fremden Schlüsseln ermöglichen. Ohne eine solche Infrastruktur ist eine sinnvolle Benutzung von PEM bzw. S/MIME jedoch nicht möglich.

PGP ist von derartigen Hierarchien unabhängig. Jeder Benutzer entscheidet selbst, welchen Schlüsseln er vertraut, und kann sofort mit anderen Benutzern kommunizieren, sobald er den public key der betroffenen Person kennt und als authentisch nachgewiesen hat. Schlüsselaustausch und -prüfung werden also nur von den Endbenutzern ausgeführt.

Während PEM keine Binärdateien bearbeiten kann, lassen sich sowohl PGP als auch S/MIME auf Text- wie auf Binärdateien anwenden. Sollen Binärdateien mit PEM bearbeitet werden, muß dagegen zuvor eine Umwandlung (z. B. mit dem SecuDE-Tool *encode* oder mit dem UNIX-Programm *uuencode*) in ein geeignetes Format vorgenommen werden.

Die Akzeptanz von Verfahren wie PEM, PGP oder S/MIME hängt nicht zuletzt auch von der Bedienerfreundlichkeit der Implementierungen ab. Die existierenden PEM- und viele der PGP-Programme sind als Filter implementiert; der eigentliche Versand als Email muß von externen Mail User Agents wie *elm* vorgenommen werden. Diese Trennung von Sicherheits- und Versand-Funktionen stellt eine – wenn auch geringe – Einschränkung dar. Es existieren allerdings bereits *elm*-Implementationen, welche die Funktionalitäten von PEM oder PGP integrieren.

Darüber hinaus sind im Zuge der Internet-Standardisierung benutzerfreundlichere (Open)PGP- bzw. S/MIME-konforme Applikationen verfügbar, welche die kryptographischen Mechanismen in fensterorientierte Tools integrieren oder Plugins für gängige fensterorientierte Mailprogramme mit grafischer Benutzeroberfläche bereitstellen. Ein Beispiel hierfür sind die neuesten Versionen von PGP oder die in Netscapes Mail-Programm *Messenger* integrierte S/MIME-Unterstützung.

PEM war von den drei betrachteten Verfahren als erster Standard definiert. Die geringe Verfügbarkeit von PEM-Implementierungen, Schwächen des PEM-Standards, die sich im Betrieb im Laufe der Jahre zeigten, und der infrastrukturelle Aufwand, der zur Benutzung von PEM erforderlich ist, führten dazu, daß PEM sich nicht breit im Internet als Verschlüsselungsstandard etablieren konnte.[4] Bei der in Europa testweise aufgebauten PEM-Zertifizierungs-Hierarchie konnte gezeigt werden, daß der Einsatz von PEM möglich ist [Grimm et al. 1994]. Problematisch war und ist bei PEM nach wie vor die Abhängigkeit von X.500-Verzeichnissen. Eine weltweit funktionierende Verzeichnis-Infrastruktur ist momentan nicht vorhanden, weshalb zur Verteilung von Zertifikaten andere Maßnahmen etabliert werden müssen.

S/MIME beseitigt oder umgeht die Schwächen von PEM und ist insofern schon viel besser für die Benutzung im Internet geeignet. S/MIME bietet sich vor allem dort

[4] So hat u. a. die geringe Nachfrage nach PEM-Zertifizierungen dazu geführt, daß die DFN-PCA mit Beginn des Jahres 1999 ihre Unterstützung für PEM aufgegeben hat und seitdem bis auf weiteres keine PEM-Zertifikate mehr anbietet.

an, wo in einem großen unüberschaubaren Netzwerk mit vorher unbekannten Teilnehmern kommuniziert werden soll. Das Vertrauen in eine zu etablierende Zertifizierungs-Infrastruktur ist eine wesentliche Voraussetzung beim Einsatz von S/MIME (und PEM). Ist diese Infrastruktur aufgebaut, stellt S/MIME ein gutes Werkzeug für den sicheren Versand von Email über ein unsicheres Netzwerk (wie das Internet) dar.

PGP wird vor allem dort zum Einsatz kommen, wo (relativ) kleine Benutzergruppen sichere Kommunikationskanäle aufbauen wollen. Es wird keine Infrastruktur vorausgesetzt; die Teilnehmer müssen das gegenseitige Vertrauen selbständig herstellen und keiner übergeordneten Instanz vertrauen. Dieser Umstand hat dazu geführt, daß PGP zur Zeit den wesentlich höheren Verbreitungsgrad im Internet gefunden hat.

Ein Problem stellt sich bei allen drei Verfahren, wenn in der jeweiligen Software eine *automatische* Überprüfung von Signaturen nicht vorgesehen ist. Erfahrungen aus der Vergangenheit haben gezeigt, daß eine Fälschung von Dokumenten oder Signaturen häufig nicht erkannt wird, weil die Empfänger keine Signaturprüfung vorgenommen und schon der reinen *Existenz* einer Signatur Vertrauen entgegen gebracht haben [Dippel 1994]. Auf die Überprüfung der Signatur darf daher **in keinem Fall** verzichtet werden.

Quintessenz PEM spielt zur Sicherung von E-Mail-Kommunikation keine wesentliche Rolle mehr. Sowohl S/MIME als auch PGP stellen hingegen verbreitete, grundsätzlich kryptographisch starke Verfahren[5] zur Sicherung von Electronic Mail zur Verfügung. Der Einsatz eines der beiden Verfahren muß deshalb unbedingt empfohlen werden. Für welchen Standard und welches Produkt letztendlich die Entscheidung getroffen wird, hängt vom jeweiligen Arbeitsumfeld und – womöglich der entscheidende Faktor – von den Verfahren bzw. Formaten ab, die die jeweiligen Kommunikationspartner unterstützen.

6.4 Fazit

Die Notwendigkeit der Verwendung kryptographischer Verfahren kann heute als allgemein akzeptiert angesehen werden. Ebenso gibt es genügend „starke“ Verfahren, so daß auch die technische Sicherheit nicht als Problem gilt. Kontroversen gibt es allerdings über die verschiedenen Anwendungen und die damit in Zusammenhang stehenden Infrastrukturen und Regularien.

[5] Ausnahme bisher: Export-Versionen von US-Software

6.4.1 Kryptographie im globalen Netz

Die Suche nach einem vielversprechenden technischen Verfahren löst nur einen Teil des Gesamtproblems. Eine wesentliche Verbesserung der allgemeinen Sicherheit ist daher nicht kurz- oder mittelfristig zu erwarten. Die Netz-Benutzer werden daher noch lange mit der Unsicherheit leben müssen.

Natürlich besteht potenziell immer die Möglichkeit, notwendige kryptographische Module durch nationale Anstrengungen entwickeln zu lassen. Praktisch gesehen ist eine solche Entwicklung jedoch mit erheblichen Schwierigkeiten und Kosten verbunden. Zunehmend konzentrieren sich daher sowohl Hersteller als auch Entwicklungsgruppen wie die IETF auf die Entwicklung generischer Schnittstellen und Protokolle, in die die kryptographischen Algorithmen „hineingesetzt“ werden können. So wird es möglich, die grundlegenden Bausteine international zu entwickeln und zu verteilen bzw. zwischen Implementierungen solcher Verfahren von unterschiedlichen Anbietern zu wählen und diese modul-artig in die eigentliche Sicherheitssoftware einzufügen (oder auch daraus zu entfernen und sie durch eine andere Implementierung zu ersetzen).

6.4.2 Zusammenfassung

Das Einsetzen verfügbarer Programme ist ein wichtiger Schritt hin zu mehr persönlicher Sicherheit, aber jede(r) Einzelne sollte sich bewußt machen, daß sie oder er sich damit auch zur Anwendung von Kryptographie in ihrer in Deutschland nicht beschränkten Form bekennt. Es gibt viele gute Argumente, die gegen eine Einschränkung oder Regulierung der Kryptographie sprechen. Doch dies muß von den Netz-Nutzern nachdrücklich vertreten werden, um entsprechenden Regulierungs-Bestrebungen oder -vorschlägen entgegenzuwirken, sollten sie wieder akut werden.

Bis zur Klärung aller offener Fragen muß auf pragmatische Maßnahmen durch jeden Einzelnen zurückgegriffen werden. Auch Vorhaben wie die DFN-PCA, die für eine große Benutzergruppe die Infrastruktur für eine den Anforderungen angemessene Zertifizierung von öffentlichen Schlüsseln aufbaut, weisen dabei in die richtige Richtung.

Der Einsatz kryptographischer Programme ist ein guter und sehr wichtiger Schritt zum Schutz der eigenen Daten und wird daher auch von den Datenschutzbeauftragten empfohlen. Jeder Benutzer sollte sich aber im Klaren darüber sein, daß auch gute Verschlüsselungswerkzeuge nur als Teil eines umfassenderen Sicherheitskonzeptes wirklich sinnvoll eingesetzt werden können.

Literaturverzeichnis

Die fettgedruckten Literaturangaben eignen sich aus unserer Sicht besonders gut als Einführung in die Themen Computersicherheit, PGP und Firewalls.

[Ackermann & Trapp 1996] Ackermann, Trapp: *Angriffe auf NIS-Systeme und Möglichkeiten einer kryptographischen Absicherung*, 3. DFN-CERT Workshop 1996[6]

[Aden & Weinand 1993] Jörg-Udo Aden und Marcel Weinand (Hrsg.): *Sicherheitseigenschaften von Unix System VRelease 4 und 4.1 ES*, Reihe: Sicherheit in der Informationstechnik, Oldenbourg, 1993

[Ahuja 1996] Vijay Ahuja, Dr.: *Network and Internet Security*, Academic Press, 1996.

[Albitz & Liu 1997] Paul Albitz and Cricket Liu: *DNS and BIND*, 2nd Edition, O'Reilly & Associates, Inc. 1997.

[Amoroso 1994] Edward Amoroso: *Fundamentals of Computer Security Technology*, Prentice-Hall, 1994.

[Arnold 1992] N. Derek Arnold: *UNIX Security: A Practical Tutorial*, McGraw-Hill, 1992.

[Barrett 1996] Daniel J. Barrett: *Bandits on the Information Superhighway*, O'Reilly, 1996.

[Bellovin 1990] Bellovin, Steven M.: *Pseudo-Network Drivers and Virtual Networks*, Murray Hill, NJ, 1990.

[Bernstein et al. 1996] Terry Bernstein, Anish Bhimani, Gene Schultz und Carol Siegel: *Internet Security for Business*, Wiley, 1996

[Brickell et al. 1993] Ernest F. Brickell, Dorothy E. Denning, Stephen T. Kent, David P. Maher und Walter Tuchman: *The SKIPJACK Algorithm: Interim Report*, Juli 1993.

[6] `http://www.cert.dfn.de/dfn/berichte/db081/nis/`

[BSI 1994a] Bundesamt für Sicherheit in der Informationstechnik: *Gefärdungen und Sicherheitsmaßnahmen beim Betrieb von digitalen Telekommunikationsanlagen*, Schriftenreihe zur IT-Sicherheit, Band 1. Bundesanzeige-Verlag, 1994

[BSI 1994b] Bundesamt für Sicherheit in der Informationstechnik: *Information zu Computer-Viren*, Schriftenreihe zur IT-Sicherheit, Band 2. Bundesanzeige-Verlag, 1994

[**BSI 1996] Bundesamt für Sicherheit in der Informationstechnik: *IT-Grundschutzhandbuch 1996***, **Schriftenreihe zur IT-Sicherheit, Band 3. Bundesanzeige-Verlag, 1996**

[BSI 1994c] Bundesamt für Sicherheit in der Informationstechnik: *Produktübersicht: Zertifizierte IT-Produkte, Zugelassene Hardware, Produkte für die materielle Sicherheit*, Schriftenreihe zur IT-Sicherheit, Band 4. Bundesanzeige-Verlag, 1994

[BSI 1995a] Bundesamt für Sicherheit in der Informationstechnik: *Chipkarten im Gesundheitswesen*, Schriftenreihe zur IT-Sicherheit, Band 5. Bundesanzeige-Verlag, 1995

[BSI 1995b] Bundesamt für Sicherheit in der Informationstechnik: *Informationstechnik zur Fahrerunterstützung im Straßenverkehr*, Schriftenreihe zur IT-Sicherheit, Band 6. Bundesanzeige-Verlag, 1995

[BSI 1993] Bundesamt für Sicherheit in der Informationstechnik: *Handbuch für die Bewertung der Sicherheit von Systemen der Informationstechnik (ITSEM)*, Schriftenreihe zur IT-Sicherheit, Band 7. Bundesanzeige-Verlag, 1993

[Carl-Mitchell & Quarterman 1992] Smoot Carl-Mitchell und John S. Quarterman: *Building Internet Firewalls*. In: UNIX World, (1992) 2, S. 93-102.

[CCITT 1988] CCITT, Melbourne: *Recommendation X.500: The Directory – Overview of concepts, models and services*, 1988. (ISO 9594-1).

[CCITT 1991] CCITT, Melbourne: *Rec. X.509 (1991), Information Technoloy – Open Systems Interconnection – The Directory: Authentication Framework*, 1991.

[**Chapman & Zwicky 1995] D. Brent Chapman, Elizabeth D. Zwicky: *Building Internet Firewalls***, **O'Reilly, 1995**

[Cheswick 1990] Cheswick, Bill: *The Design of a Secure Internet Gateway*[7]. Murray Hill, NJ, 1990.

[**Cheswick & Bellovin 1994] William R. Cheswick und Steven M. Bellovin: *Firewalls & Internet Security Repelling the Wily Hacker***, **Addison Wesley, 1994**

[7] `ftp://ftp.cert.dfn.de/pub/firewalls/att/gateway.ps.gz`

[Cheswick & Bellovin 1996] Cheswick, Bill & Bellovin, Steven M.: *A DNS Filter and Switch for Packet-filtering Gateways*. Proceedings of the 6th USENIX Security Symposium 1996.

[Cohen 1995] Frederick B. Cohen: *Protection and Security on the Information Superhighway*, Wiley, 1995

[Denning 1982] Dorothy E. Denning: *Cryptography and Data Security*, Addison-Wesley Publishing Company, 1982.

[Denning 1990] Peter J. Denning (Ed.): *Computers Under Attack: Intruders, Worms and Viruse*, ACM, 1990

[Deutsch 1988] Debra Deutsch: *An Introduction to the X.500 Series Network Directory Service*. BBN Laboratories Incorporated, Juni 1988.

[Diffie & Hellman 1976] Whitfield Diffie und Martin E. Hellman: *New Directions in Cryptography*, in *IEEE Transactions on Information Theory*, 22(6):644–654, November 1976.

[Dippel 1994] Kathrin Dippel: *Benutzungsoberflächen von Sicherungsfunktionen in Chipkarten*. Diplomarbeit, Technische Hochschule Darmstadt, 1994.

[Dobbertin et al. 1996] Hans Dobbertin, Antoon Bosselaers und Bart Preneel: *RIPEMD-160, a strengthened version of RIPEMD*[8], in D. Gollmann (Ed.): *Fast Software Encryption, LNCS 1039*, p71–82, Springer, 1996

[Eckstein 1994] Levona Eckstein: *STARCOS: Smartcard Application Package, Volume 2: Specification of the Smartcard Application Interface Version 1.1*, Gesellschaft für Mathematik und Datenverarbeitung (GMD), Januar 1994.

[ElGamal 1985] Taher ElGamal: *A public key cryptosystem and a signature scheme based on discrete logarithms*, in *IEEE Transactions on Information Theory*, 31(4):469–472, Juli 1985.

[Ellermann 1993] Uwe Ellermann: *Ein Firewall am Fachbereich Informatik*, Universität Hamburg, Dezember 1993.

[Ellermann 1994] Uwe Ellermann: *Firewalls: Isolations- und Audittechniken zum Schutz von lokalen Computer-Netzen*, DFN-Bericht Nr. 76, September 1994.

[Ellermann 1995] Uwe Ellermann: *Netzabsicherung durch Firewalls*, in *4. IT-Sicherheitskongreß, 8. bis 11. Mai 1995, Bonn*[9], Mai 1995.

[8] `http://www.esat.kuleuven.ac.be/ bosselae/ripemd160.html`

[9] `http://www.cert.dfn.de/team/ue/fw/fire/`

[Ellermann 1996] Uwe Ellermann: *IPv6 and Firewalls*, in *Proceedings of SECURICOM — 14th International Congress on Computer and Communications Security Protection, Paris*[10], Juni 1996.

[Ellermann & Benecke 1998] Uwe Ellermann, Carsten Benecke: *Firewalls für Hochgeschwindigkeitsnetze*, in *Deutscher Internet Kongress 1998, 5. und 6. Mai 1998, CongressCenter Messe Frankfurt*[11], Mai 1998.

[Essen & Felzmann 1991] U. van Essen, F. Felzmann: *Bedrohungsmodell in KI-Systemen*, Oldenbourg, 1991

[Essen 1995] U. van Essen: *Sicherheit des Betriebssystems VMS*, Oldenbourg, 1995

[Ferbrache & Shearer 1993] David Ferbrache und Gavin Shearer: *UNIX Installation Security and Integrity*, Prentice-Hall, 1993

[FIRST 1997] Members of FIRST: *Forum of Incident Response and Security Teams – Operational Framework*, Juni 1997. [FIRST WWW Server][12]

[Friedl 1997] Jeffrey Friedl: *Mastering Regular Expressions*. O'Reilly & Associates, Inc., Januar 1997.

[Fumy et al. 1995] Walter Fumy, Patrick Horster und Peter Kraaibeek: *Standards und Patente zur IT-Sicherheit*, Reihe: Sicherheit in der Informationstechnik, Oldenbourg, 1995

[Fumy & Rieß 1994] Walter Fumy und Hans Peter Rieß: *Kryptographie*, 2. Aufl., Reihe: Sicherheit in der Informationstechnik, Oldenbourg, 1994

[Garfinkel 1995] Simson Garfinkel: *PGP: Pretty Good Privacy*. O'Reilly & Associates, Inc., Sebastopol, CA, first edition, Januar 1995.

[Garfinkel & Spafford 1996] Simson Garfinkel and Gene Spafford: *Practical UNIX & Internet Security*, 2nd Ed., O'Reilly & Associates, Inc. 1996.

[Grimm et al. 1994] Rüdiger Grimm, Jan Lühe und Wolfgang Schneider: *Towards Trustworthy Communication Systems – Experiences with the Security Toolkit SecuDE*. in *Proceedings of the IFIP TC6/WG6.5 International Conference on Upper Layer Protocols, Architectures and Applications (ULPAA '94), Barcelona*, IFIP Transactions C-25, Seiten 107–120, North-Holland, 1994. Elsevier Science Publishers B.V.

[10] `http://www.fwl.dfn.de/eng/team/ue/fw/ipv6fw/`
[11] `http://www.fwl.dfn.de/team/ue/fw/fire-hsn/`
[12] `http://www.first.org/`

[GRIP 1997] Members of the IETF WG GRIP: *Guidelines and Recommendations for Incident Processing*, [Charter, Meeting Minutes und verschiedene Drafts][13]

[Hu 1995] Wei Hu: *DCE Security Programming*, O'Reilly, 1995

[IANA Numbers] Internet Assigned Numbers Authority, Protocol Numbers and Assignment Services[14]

[Icove et al. 1995] David Icove, Karl Seger, William VonStorch (Consulting Editor Eugene H. Spafford): *Computer Crime*, O'Reilly, 1995

[ISO/IEC 10118-3] International Organization for Standardization: *ISO/IEC 10118-3: Information Technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions*, Geneva, Switzerland, 1998

[ISO/IEC 9594-8] ITU/ISO/IEC: *ITU-T Recommendation X.509 (1997:E) The Directory: Authentication Framework*, 1997

[Kaufman et al. 1995] Charles Kaufman, Radia Perlman und Michael Speciner: *Network Security: Private Communication in a Public World*, 1/e Prentice-Hall, 1995

[Kent 1993] Stephen T. Kent: *Internet Privacy Enhanced Mail*. in *Communications of the ACM*, 36(8):48–60, August 1993.

[Kersten 1991] Heinrich Kersten: *Einfürung in die Computersicherheit*, Oldenbourg, 1991

[Kersten 1995] Heinrich Kersten: *Sicherheit in der Informationstechnik*, 2. Aufl., Oldenbourg, 1995

[Kersten & Kreutz 1991] H. Kersten, H. Kreutz: *Sicherheit unter dem Betriebssystem Unix*, Reihe: Sicherheit in der Informationstechnik, Oldenbourg, 1991

[Kersten & Weinand 1991a] H. Kersten, M. Weinand: *Sicherheitsaspekte bei der Vernetzung von 386/ix und SCO-Unix*, Reihe: Sicherheit in der Informationstechnik, Oldenbourg, 1991

[Kersten & Weinand 1991b] H. Kersten, M. Weinand: *Sicherheitsaspekte bei der Vernetzung von Unix-Systemen*, Reihe: Sicherheit in der Informationstechnik, Oldenbourg, 1991

[Klaus 1996] Christopher Klaus: *How to Set up a Secure Anonymous FTP Site*, Anonymous FTP FAQ[15]

[13] `http://www.cert.dfn.de/eng/resource/ietf/grip/`
[14] `http://www.iana.org/numbers.htm`
[15] `http://xforce.iss.net/security_library/faqs/anonftp.php`

[Koblas & Koblas 1992] David Koblas, Michelle R. Koblas: *SOCKS*[16], 1992.

[Kossakowski 1994] Klaus-Peter Kossakowski: *The DFN-CERT Experience – Building up a new CERT within Europe*, DFN-CERT, Hamburg, Mai 1994. [Vortrag auf der INET'94/JENC5 1994 in Prag][17]

[Kossakowski 1996a] Klaus-Peter Kossakowski: *Coordination of Incident Response Teams*, DFN-CERT, Hamburg, Juni 1996 [Vortrag auf dem 28. I-4 Meeting in Oslo]

[Kossakowski 1996b] Klaus-Peter Kossakowski: *Providing Incident Response Services*, DFN-CERT, Hamburg, Februar 1996. [Tutorium auf der Open System Security Europe in London]

[Kossakowski 1997] Klaus-Peter Kossakowski: *From Incident Response to Incident Control Management*, DFN-CERT, Hamburg, 1997. [Vortrag auf dem 9. FIRST Workshop in Bristol]

[Kossakowski 1998] Klaus-Peter Kossakowski: *Virtuelle Private Netzwerke: Einsatzmöglichkeiten und Techniken*, in: *5. DFN-CERT Workshop 1998 „Sicherheit in vernetzten Systemen", 4. und 5. März 1998, Hamburg*, DFN-Bericht Nr. 85, Mai 1998.

[McCormac 1996] John McCormac: *European Scrambling Systems 5 - The Black Book*, 1996

[Malik 1996] Imtiaz Malik: *Computer Hacking: Detection and Protection*, Sigma, 1996

[Nechvatal 1991] James Nechvatal: *Public-Key Cryptography*. National Institute of Standards and Technology, Gaithersburg, MD, April 1991. (NIST Special Publication 800-2).

[Pfleeger 1989] Charles P. Pfleeger: *Security in Computing*. Prentice-Hall International, Inc., Englewood Cliffs, NJ, 1989.

[Pohl & Weck 1995a] Hartmut Pohl und Gerhard Weck (Hrsg.): *Beiträge zur Informationssicherheit*, Reihe: Sicherheit in der Informationstechnik, Oldenbourg, 1995

[Pohl & Weck 1995b] Hartmut Pohl und Gerhard Weck (Hrsg.): *Einführung in die Informationssicherheit*, Reihe: Sicherheit in der Informationstechnik, Oldenbourg, 1995

[16] `ftp://ftp.cert.dfn.de/pub/tools/net/socks/socks_usenix_paper.ps.gz`
[17] `http://www.cert.dfn.de/eng/pre99papers/jenc5.html`

[Pohl & Weck 1995c] Hartmut Pohl und Gerhard Weck (Hrsg.): *Managementaufgaben im Bereich der Informationssicherheit*, Reihe: Sicherheit in der Informationstechnik, Oldenbourg, 1995

[Ranum 1992] Ranum, Marcus J.: Thinking about Firewalls[18], Greenbelt, MD, 1992.

[Rivest et al. 1978] R. L. Rivest, A. Shamir und L. Adleman: *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*. in *Communications of the ACM*, 21(2):120–126, Februar 1978.

[Roe 1992] Michael Roe: *PASSWORD R2.5: Certification Authority Requirements*. Cambridge University Computer Laboratory, November 1992.

[Rosenoer 1996] J, Rosenoer: *Cyberlaw: The Law of the Internet* Springer, 1996

[Russell & Gangemi 1991] Deborah Russell, G. T. Gangemi, Sr.: *Computer Security Basics*, O'Reilly, 1991

[Shaffer 1993] Steven L. Shaffer *Network Security*, Academic Press, 1993

[Smith 1994] Danny Smith: *Forming an Incident Response Team*, AUSCERT, University of Queensland, Brisbane, Qld, Juli 1994. [Vortrag auf dem 6. Computer Security Incident Handling Workshop 1994 in Boston, Mass.]

[Schneier et al. 1994] Bruce Schneier: *Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)*, in *Fast Software Encryption, Cambridge Security Workshop Proceedings*, p191–204, Springer, December 1993

[Schneider 1994a] Wolfgang Schneider: *SecuDE Overview Version 4.4*. Gesellschaft für Mathematik und Datenverarbeitung (GMD), Oktober 1994.

[Schneider 1994b] Wolfgang Schneider: *SecuDE Vol. 1: Principles of Security Operations Version 4.4*. Gesellschaft für Mathematik und Datenverarbeitung (GMD), Oktober 1994.

[Schneider 1994c] Wolfgang Schneider: *SecuDE Vol. 2: Security Commands, Functions and Interfaces Version 4.4*. Gesellschaft für Mathematik und Datenverarbeitung (GMD), Oktober 1994.

[Schneider 1994d] Wolfgang Schneider: *SecuDE Vol. 3: Security Application's Guide Version 4.4*. Gesellschaft für Mathematik und Datenverarbeitung (GMD), Oktober 1994.

[Stallings 1994] William Stallings: *Network and Internetwork Security: Principles and Practice*, Prentice-Hall, 1994

[18] `ftp://ftp.cert.dfn.de/pub/firewalls/dec/thinking.about.firewalls.ps.gz`

[Stallings 1995] William Stallings: *Protect Your Privacy: The PGP User's Guide*, Prentice-Hall, 1994

[Stigliani 1995] Joan Stigliani *The Computer User's Survival Guide*, O'Reilly, 1995

[NIST 1994] United States National Institute of Standards and Technology: *Digital Signature Standard DSS, Federal Information Processing Standards Publication (FIPS PUB) 186*[19], 19/1994

[Venema 1992] Venema, Wietse: *TCP Wrapper, Network monitoring, access control, and booby traps*[20], Eindhoven, 1992.

[Wack 1991] John Wack: *Establishing a Computer Security Incident Response Capability*, US National Institute of Standards and Technology, Gaithersburg, MD, NIST Special Publication 800-3, November 1991.

[Zimmermann 1993] Philip Zimmermann: *PGP v2.6.1: Pretty Good Privacy – User's Guide*, Juni 1993.

Sicherheitsrelevante RFCs

[RFC 821] Jonathan B. Postel: *Simple Mail Transfer Protocol*, RFC 821, August 1982.

[RFC 822] David H. Crocker: *Standard for the format of ARPA Internet text messages*, RFC 822, August 1982.

[RFC 854] J. Postel, J. Reynolds: *Telnet Protocol Specification*, RFC 854, Mai 1983.

[RFC 959] J. Postel, J. Reynolds: *File Transfer Protocol (FTP)*, RFC 959, Oktober 1985.

[RFC 1004] D. L. Mills: *Distributed-protocol authentication scheme*, RFC 1004, April 1987

[RFC 1108] S. Kent: *U. S. Department of Defense Security Options for the Internet Protocol*, RFC 1108, November 1991.

[RFC 1125] J. K. Reynolds: *Helminthiasis of the Internet*, RFC 1135, December 1989.

[RFC 1166] S. Kirkpatrick, M. K. Stahl, M. Recker: *Internet numbers*, RFC 1166, July 1990.

[19] `http://csrc.nist.gov/fips/fips186.pdf`
[20] `ftp://ftp.cert.dfn.de/pub/tools/net/TCP-Wrapper/tcp_wrapper.ps.gz`

[RFC 1170] R. B. Fougner: *Public key standards and licenses*, RFC 1170, January 1991.

[RFC 1186] R. L. Rivest: *MD4 Message Digest Algorithm*, RFC 1186, October 1990.

[RFC 1272] C. Mills, D. Hirsh, G. R. Ruth: *Internet Accounting*, RFC 1272, November 1991.

[RFC 1281] R. Pethia, S. Crocker, B. Fraser: *Guidelines for the Secure Operation of the Internet*, RFC 1281, November 1991.

[RFC 1319] Burton S. Kaliski: *The MD2 Message-Digest Algorithm*, RFC 1319, April 1992.

[RFC 1320] Ronald L. Rivest: *The MD4 Message-Digest Algorithm*, RFC 1320, April 1992.

[RFC 1321] Ronald L. Rivest: *The MD5 Message-Digest Algorithm*, RFC 1321, April 1992.

[RFC 1352] J. Galvin, K. McCloghrie, J. Davin: *SNMP Security Protocols*, RFC 1352, July 1992.

[RFC 1355] J. Curran, A. Marine: *Privacy and Accuracy Issues in Network Information Center Databases*, RFC 1355, August 1992.

[RFC 1411] D. Borman, Editor: *Telnet Authentication: Kerberos Version 4*, RFC 1411, January 1993.

[RFC 1412] K. Alagappan: *Telnet Authentication: SPX*, RFC 1412, January 1993.

[RFC 1413] M. StJohns: *Identification Protocol*, RFC 1413, January 1993.

[RFC 1414] M. StJohns, M. Rose: *Identification MIB*, RFC 1414, January 1993.

[RFC 1416] D. Borman, Editor: *Telnet Authentication Option*, RFC 1416, February 1993.

[RFC 1421] John Linn: *Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures*, RFC 1421, Februar 1993.

[RFC 1422] Steve Kent: *Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management*, RFC 1422, Februar 1993.

[RFC 1423] David Balenson: *Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers*, RFC 1423, Februar 1993.

[RFC 1424] Burton S. Kaliski: *Privacy Enhancement for Internet Electronic Mail: Part IV: Key Certification and Related Services*, RFC 1424, Februar 1993.

[RFC 1446] J. Galvin, K. McCloghrie: *Security Protocols for version 2 of the Simple Network Management Protocol (SNMPv2)*, RFC 1446, April 1993.

[RFC 1457] R. Housley: *Security Label Framework for the Internet*, RFC 1457, May 1993.

[RFC 1472] F. Kastenholz: *The Definitions of Managed Objects for the Security Protocols of the Point-to-Point Protocol*, RFC 1472, June 1993.

[RFC 1492] C. Finseth: *An Access Control Protocol, Sometimes Called TACACS*, RFC 1492, July 1993.

[RFC 1507] C. Kaufman: *DASS - Distributed Authentication Security Service*, RFC 1507, September 1993.

[RFC 1510] J. Kohl, C. Neuman: *The Kerberos Network Authentication Service (V5)*, RFC 1510, September 1993.

[RFC 1511] J. Linn: *Common Authentication Technology Overview*, RFC 1511, September 1993.

[RFC 1535] E. Gavron: *A Security Problem and Proposed Correction With Widely Deployed DNS Software*, RFC 1535, October 1993.

[RFC 1636] R. Braden, D. Clark, S. Crocker, C. Huitema: *Report of IAB Workshop on Security in the Internet Architecture - February 8-10, 1994*, RFC 1636, June 1994.

[RFC 1675] S. Bellovin: *Security Concerns for IPng*, RFC 1675, August 1994.

[RFC 1700] J. Reynolds, J. Postel: *Assigned Numbers*, RFC 1700, October 1994.

[RFC 1704] N. Haller, R. Atkinson: *On Internet Authentication*, RFC 1704, October 1994.

[RFC 1710] R. Hinden: *On Simple Internet Protocol Plus White Paper*, RFC 1710, October 1994.

[RFC 1731] J. Myers: *IMAP4 Authentication Mechanisms*, RFC 1731, December 1994.

[RFC 1734] J. Myers: *POP3 AUTHentication command*, RFC 1734, December 1994.

[RFC 1750] D. Eastlake, 3rd, S. Crocker, J. Schiller: *Randomness Recommendations for Security*, RFC 1750, December 1994.

[RFC 1751] D. McDonald: *A Convention for Human-Readable 128-bit Keys*, RFC 1751, December 1994.

[RFC 1760] N. Haller: *The S/KEY One-Time Password System*, RFC 1760, February 1995.

[RFC 1777] W. Yeong, T. Howes, S. Kille: *Lightweight Directory Access Protocol*, RFC 1777, March 1995.

[RFC 1805] A. Rubin: *Location-Independent Data/Software Integrity Protocol*, RFC 1805, June 1995.

[RFC 1810] J. Touch: *Report on MD5 Performance*, RFC 1810, June 1995.

[RFC 1824] H. Danisch: *The Exponential Security System TESS: An Identity-Based Cryptographic Protocol for Authenticated Key-Exchange (E.I.S.S.-Report 1995/4)*, RFC 1824, August 1995.

[RFC 1828] P. Metzger, W. Simpson: *IP Authentication using Keyed MD5*, RFC 1828, August 1995.

[RFC 1829] P. Karn, P. Metzger, W. Simpson: *The ESP DES-CBC Transform*, RFC 1829, August 1995.

[RFC 1847] J. Galvin, S. Murphy, S. Crocker, N. Freed: *Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted*, RFC 1847, October 1995.

[RFC 1848] S. Crocker, N. Freed, J. Galvin: *MIME Object Security Services*, RFC 1848, October 1995.

[RFC 1851] P. Karn, P. Metzger, W. Simpson: *The ESP Triple DES Transform*, RFC 1851, September 1995.

[RFC 1852] P. Metzger, W. Simpson: *IP Authentication using Keyed SHA*, RFC 1852, September 1995.

[RFC 1853] W. Simpson: *IP in IP Tunneling*, RFC 1853, October 1995.

[RFC 1858] G. Ziemba, D. Reed, P. Traina: *Security Considerations for IP Fragment Filtering*, RFC 1858, October 1995.

[RFC 1864] J. Myers, M. Rose: *The Content-MD5 Header Field*, RFC 1864, October 1995.

[RFC 1875] N. Berge: *UNINETT PCA Policy Statements*, RFC 1875, December 1995.

[RFC 1910] G. Waters: *User-based Security Model for SNMPv2*, RFC 1910, February 1996.

[RFC 1928] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas & L. Jones: *SOCKS Protocol Version 5*, RFC 1928, April 1996.

[RFC 1929] M. Leech: *Username/Password Authentication for SOCKS V5*, RFC 1929, April 1996.

[RFC 1948] S. Bellovin: *Defending Against Sequence Number Attacks*, RFC 1948, May 1996.

[RFC 1949] A. Ballardie: *Scalable Multicast Key Distribution*, RFC 1949, May 1996.

[RFC 1961] P. McMahon: *GSS-API Authentication Method for SOCKS Version 5*, RFC 1961, June 1996.

[RFC 1964] J. Linn: *The Kerberos Version 5 GSS-API Mechanism*, RFC 1964, June 1996.

[RFC 1968] G. Meyer: *The PPP Encryption Control Protocol (ECP)*, RFC 1968, June 1996.

[RFC 1984] IAB & IESG: *IAB and IESG Statement on Cryptographic Technology and the Internet*, RFC 1984, August 1996.

[RFC 1991] D. Atkins, W. Stallings & P. Zimmermann: *PGP Message Exchange Formats*, RFC 1991, August 1996.

[RFC 1994] W. Simpson: *PPP Challenge Handshake Authentication Protocol (CHAP)*, RFC 1994, August 1996.

[RFC 2015] M. Elkins: *MIME Security with Pretty Good Privacy (PGP)*, RFC 2015, October 1996.

[RFC 2025] C. Adams: *The Simple Public-Key GSS-API Mechanism (SPKM)*, RFC 2025, October 1996.

[RFC 2040] R. Baldwin, R. Rivest: *The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS Algorithms*, RFC 2040, October 1996.

[RFC 2045] N. Freed & N. Borenstein: *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*, RFC 2045, November 1996.

[RFC 2046] N. Freed & N. Borenstein: *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*, RFC 2046, November 1996.

[RFC 2047] K. Moore: *MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text*, RFC 2047, November 1996.

[RFC 2048] N. Freed, J. Klensin & J. Postel: *Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures*, RFC 2048, November 1996.

[RFC 2049] N. Freed & N. Borenstein: *Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples*, RFC 2049, November 1996.

[RFC 2057] S. Bradner: *Source Directed Access Control on the Internet*, RFC 2057, November 1996.

[RFC 2078] J. Linn: *Generic Security Service Application Program Interface, Version 2*, RFC 2078, January 1997.

[RFC 2082] F. Baker, R. Atkinson: *RIP-2 MD5 Authentication*, January 1997.

[RFC 2084] G. Bossert, S. Cooper, W. Drummond: *Considerations for Web Transaction Security*, RFC 2084, January 1997.

[RFC 2085] M. Oehler, R. Glenn: *HMAC-MD5 IP Authentication with Replay Prevention*, RFC 2085, February 1997.

[RFC 2086] J. Myers: *IMAP4 ACL extension*, RFC 2086, January 1997.

[RFC 2093] H. Harney, C. Muckenhirn: *Group Key Management Protocol (GKMP) Specification*, RFC 2093, July 1997.

[RFC 2094] H. Harney, C. Muckenhirn: *Group Key Management Protocol (GKMP) Architecture*, RFC 2094, July 1997.

[RFC 2104] H. Krawczyk, M. Bellare, R. Canetti: *HMAC: Keyed-Hashing for Message Authentication*, RFC 2104, February 1997.

[RFC 2137] D. Eastlake: *Secure Domain Name System Dynamic Update*, RFC 2137, April 1997.

[RFC 2138] C. Rigney, A. Rubens, W. Simpson, S. Willens: *Remote Authentication Dial In User Service (RADIUS)*, RFC 2138, April 1997.

[RFC 2139] C. Rigney: *RADIUS Accounting*, RFC 2139, April 1997.

[RFC 2144] C. Adams: *The CAST-128 Encryption Algorithm*, RFC 2144, May 1997.

[RFC 2154] S. Murphy, M. Badger, B. Wellington: *OSPF with Digital Signatures*, RFC 2154, June 1997.

[RFC 2179] A. Gwinn: *Network Security For Trade Shows*, RFC 2179, July 1997.

[RFC 2196] B. Fraser: *Site Security Handbook*, RFC 2196, September 1997.

[RFC 2202] P. Cheng, R. Glenn: *Test Cases for HMAC-MD5 and HMAC-SHA-1*, RFC 2202, September 1997.

[RFC 2203] M. Eisler, A. Chiu, L. Ling: *RPCSEC_GSS Protocol Specification*, RFC 2203, September 1997.

[RFC 2222] J. Myers: *Simple Authentication and Security Layer (SASL)*, RFC 2222, October 1997.

[RFC 2228] M. Horowitz, S. Lunt: *FTP Security Extensions*, RFC 2228, October 1997.

[RFC 2230] R. Atkinson: *Key Exchange Delegation Record for the DNS*, RFC 2230, October 1997.

[RFC 2231] N. Freed, K. Moore: *MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations*, RFC 2231, November 1997.

[RFC 2245] C. Newman: *Anonymous SASL Mechanism*, RFC 2245, November 1997.

[RFC 2267] P. Ferguson, D. Senie: *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*, RFC 2267, January 1998.

[RFC 2268] R. Rivest: *A Description of the RC2(r) Encryption Algorithm*, RFC 2268, January 1998.

[RFC 2284] L. Blunk, J. Vollbrecht: *PPP Extensible Authentication Protocol (EAP)*, RFC 2284, March 1998.

[RFC 2286] J. Kapp: *Test Cases for HMAC-RIPEMD160 and HMAC-RIPEMD128*, RFC 2286 February 1998.

[RFC 2289] N. Haller, C. Metz, P. Nesser, M. Straw: *A One-Time Password System*, RFC 2289, February 1998.

[RFC 2311] S. Dusse, P. Hoffman, B. Ramsdell, L. Lundblade, L. Repka: *S/MIME Version 2 Message Specification*, RFC 2311, March 1998.

[RFC 2312] S. Dusse, P. Hoffman, B. Ramsdell, J. Weinstein: *S/MIME Version 2 Certificate Handling*, RFC 2312, March 1998.

[RFC 2313] B. Kaliski: *PKCS 1: RSA Encryption Version 1.5*, RFC 2313, March 1998.

[RFC 2314] B. Kaliski: *PKCS 10: Certification Request Syntax Version 1.5*, RFC 2314, March 1998.

[RFC 2315] B. Kaliski: *PKCS 7: Cryptographic Message Syntax Version 1.5*, RFC 2315, March 1998.

[RFC 2316] S. Bellovin: *Report of the IAB Security Architecture Workshop*, RFC 2316, April 1998.

[RFC 2350] N. Brownlee, E. Guttman: *Expectations for Computer Security Incident Response*, RFC 2350, June 1998.

[RFC 2356] G. Montenegro, V. Gupta: *Sun's SKIP Firewall Traversal for Mobile IP*, RFC 2356, June 1998.

[RFC 2401] S. Kent, R. Atkinson: *Security Architecture for the Internet Protocol*, RFC 2401, November 1998.

[RFC 2402] S. Kent, R. Atkinson: *IP Authentication Header*, RFC 2402, November 1998.

[RFC 2406] S. Kent, R. Atkinson: *IP Encapsulating Security Payload (ESP)*, RFC 2406, November 1998.

[RFC 2419] K. Sklower, G. Meyer: *The PPP DES Encryption Protocol, Version 2 (DESE-bis)*, RFC 2419, September 1998.

[RFC 2440] J. Callas, L. Donnerhacke, H. Finney, R. Thayer: *OpenPGP Message Format*, RFC 2440, November 1998.

[RFC 2444] C. Newman: *The One-Time-Password SASL Mechanism*, RFC 2444, October 1998.

[RFC 2459] R. Housley, W. Ford, W. Polk, D. Solo: *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, RFC 2459, January 1999.

[RFC 2474] K. Nichols, S. Blake, F. Baker, D. Black: *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*, RFC 2474, December 1998.

[RFC 2484] G. Zorn: *PPP LCP Internationalization Configuration Option*, RFC 2484, January 1999.

[RFC 2504] E. Guttman, L. Leong, G. Malkin: *Users' Security Handbook*, RFC 2504, February 1999.

[RFC 2535] D. Eastlake: *Domain Name System Security Extensions*, RFC 2535, March 1999.

[RFC 2536] D. Eastlake: *DSA KEYs and SIGs in the Domain Name System (DNS)*, RFC 2536, March 1999.

[RFC 2537] D. Eastlake: *RSA/MD5 KEYs and SIGs in the Domain Name System (DNS)*, RFC 2537, March 1999.

[RFC 2538] D. Eastlake, O. Gudmundsson: *Storing Certificates in the Domain Name System (DNS)*, RFC 2538, March 1999.

[RFC 2539] D. Eastlake: *Storage of Diffie-Hellman Keys in the Domain Name System (DNS)*, RFC 2539, March 1999.

[RFC 2541] D. Eastlake: *DNS Security Operational Considerations*, RFC 2541, March 1999.

[RFC 2574] U. Blumenthal, B. Wijnen: *User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*, RFC 2574, April 1999.

[RFC 2575] B. Wijnen, R. Presuhn, K. McCloghrie: *View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)*, RFC 2575, April 1999.

[RFC 2577] M. Allman, S. Ostermann: *FTP Security Considerations*, RFC 2577 May 1999.

[RFC 2617] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart: *HTTP Authentication: Basic and Digest Access Authentication*, RFC 2617, June 1999.

[RFC 2623] M. Eisler: *NFS Version 2 and Version 3 Security Issues and the NFS Protocol's Use of RPCSEC_GSS and Kerberos V5*, RFC 2623, June 1999.

[RFC 2632] B. Ramsdell, Ed.: *S/MIME Version 3 Certificate Handling*, RFC 2632, June 1999.

[RFC 2633] B. Ramsdell, Ed.: *S/MIME Version 3 Message Specification*, RFC 2633, June 1999.

[RFC 2634] P. Hoffman, Ed.: *Enhanced Security Services for S/MIME*, RFC 2634, June 1999.

[RFC 2646] R. Gellens: *The Text/Plain Format Parameter*, RFC 2646, August 1999.

[RFC 2659] E. Rescorla, A. Schiffman: *Security Extensions For HTML*, RFC 2659, August 1999.

[RFC 2660] E. Rescorla, A. Schiffman: *The Secure HyperText Transfer Protocol*, RFC 2660, August 1999.

[RFC 2709] P. Srisuresh: *Security Model with Tunnel-mode IPsec for NAT Domains*, RFC 2709, October 1999.

[RFC 2712] A. Medvinsky, M. Hur: *Addition of Kerberos Cipher Suites to Transport Layer Security (TLS)*, RFC 2712, October 1999.

[RFC 2725] C. Villamizar, C. Alaettinoglu, D. Meyer, S. Murphy: *Routing Policy System Security*, RFC 2725, December 1999.

[RFC 2743] J. Linn: *Generic Security Service Application Program Interface Version 2, Update 1*, RFC 2743, January 2000.

[RFC 2744] J. Wray: *Generic Security Service API Version 2 : C-bindings*, RFC 2744, January 2000.

[RFC 2827] P. Ferguson, D. Senie: *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*, RFC 2827, May 2000.

[RFC 2828] R. Shirey: *Internet Security Glossary*, RFC 2828 May 2000.

Anhang A

Tools für UNIX Systeme

A.1 Das *logdaemon*-Paket

Bei dem *logdaemon*-Paket von Wietse Venema handelt es sich um eine Reihe von Systemprogrammen, welche die gängigsten Netzwerk-Dienste ersetzen. Die Verwendung dieser Systemprogramme bringt Vorteile, da sie um Sicherheits-Routinen erweitert wurden, die mehr oder weniger relevante Sicherheitslücken der Standardinstallation des UNIX Betriebssystems schließen. Ersetzt werden die Programme *login*, *rshd*, *ftpd* und *rexecd*. Der Vollständigkeit halber wird auch noch ein *telnetd* mitgeliefert, welcher allerdings nur notwendig ist, falls der herstellerspezifische *telnetd* das *login*-Programm in einer besonderen Art und Weise aufruft.

Die neuen Programme bieten folgende Verbesserungen:

- Logging
  Die stark erweiterten Log-Meldungen der einzelnen Programme haben dem Paket seinen Namen gegeben. Bei dem Einsatz dieser Programme melden alle Programme fehlerhafte Login-Versuche bereits nach dem ersten Fehlversuch, auf den nicht direkt ein gültiger Login folgt. Bei den Original-Programmen werden nur Fehlversuche geloggt, welche eine bestimmte Anzahl (in der Regel 3 oder 5) überschreiten. So kann dann ein Cracker zwei Paßworte ausprobieren und dann die Verbindung abbrechen und neu aufbauen. Auf diese Weise können beliebig viele Paßworte unbemerkt ausprobiert werden. Der Original-*rexecd* meldet selbst wiederholte Fehlversuche nicht.

  Zusätzlich kann man auch alle korrekten Logins protokollieren lassen.

- Zugangskontrolle
  Die Zugangskontrolle findet auf Basis von Hostnamen bzw. -adressen statt. Der Zugang zum Server kann auf diese Weise auf bestimmte Hosts oder Netze beschränkt werden. Die einzelnen Produkte benutzen die *libwrap* des TCP-Wrappers und bieten somit die Möglichkeit, über die beiden Konfigurationsdateien `/etc/hosts.allow` und `/etc/hosts.deny` das Verhalten jederzeit frei einzustellen. Der Aufbau und die Wirkungsweise dieser Dateien wird bei der Vorstellung des TCP-Wrappers in Abschnitt A.2 beschrieben.

  Zusätzlich bietet das *login*-Programm eine weitere Zugangskontrolle für einzelne Benutzer oder Gruppen. In der Datei `/etc/login.access` kann detailliert geregelt werden, wer sich von wo einloggen darf bzw. nicht. So kann man dort z.B. einstellen, daß sich Mitglieder der Gruppe *wheel* nur an der Konsole (aber nicht über das Netzwerk) einloggen können. Oder der Benutzer *info* darf sich nicht an der Konsole einloggen...

- Einwegpaßworte
  Die Programme *login* und *ftpd* unterstützen das Programm S/Key (s. DIB-94:04-

S/Key[1]) und bieten die Möglichkeit, Einwegpaßworte zu benutzen. Die notwendigen Programme zur Erzeugung der Paßworte *keyinit* und *key* werden ebenfalls mitgeliefert. Die Benutzung von Einwegpaßworten kann alternativ oder verbindlich angeboten werden. Je nach Ursprung einer aufgebauten Verbindung kann dies variabel eingestellt werden. So ist es z.B. möglich, innerhalb des eigenen Netzwerkes sowohl das normale Login-Paßwort als auch ein Einwegpaßwort zu akzeptieren, während bei einem Login von außerhalb dieses Netzwerkes die Verwendung von Einwegpaßworten zwingend vorgeschrieben ist.

Das *logdaemon*-Paket richtet sich an Systemadministratoren, die die Sicherheit eines am Internet angeschlossenen Hosts bzw. Netzes erhöhen wollen und setzt die *libwrap* voraus, welche mit dem TCP-Wrapper mitgeliefert wird (siehe Abschnitt A.2).

Das Logdaemon[2]-Paket kann vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-94:03-Logdaemon[3] beschrieben.

A.2 TCP-Wrapper *tcpd*

In der Standardinstallation des UNIX - Betriebssystems gibt es kaum eine Möglichkeit, häufig benutzte Netzwerktools wie *finger*, *ftp* oder *telnet* zu kontrollieren. Mit dem TCP - Wrapper ist es möglich, die Benutzung dieser Programme zu überwachen und gegebenenfalls auf bestimmte Rechner zu beschränken.

Viele Netzwerk-Applikationen basieren auf dem sog. „Client-Server“ - Modell: startet ein Benutzer z.B. das Programm *telnet* (den Client), baut dieses eine Verbindung zum telnet - Server (telnetd) auf dem Zielrechner auf. Die Kontrolle über diese Serverprogramme hat in der Regel der Internetdaemon (*inetd*), welcher auf alle Arten von Netzwerkverbindungen wartet und bei Bedarf den gewünschten Server - Prozeß startet. Es wird allerdings nicht überprüft, von welchen Rechnern die Netzwerkverbindungen aufgebaut werden und welche Klienten vom *inetd* gestartet werden.

Der TCP-Wrapper ist einerseits ein Paket zur Zugangskontrolle für Programme, welche von dem *inetd* gestartet werden, andererseits bietet die mitgelieferte Library *libwrap* eine universelle Möglichkeit, Zugriffsbeschränkungen in eigenen Programmen abzufragen. Andere Pakete, wie das in Abschnitt A.1 beschriebene *logdaemon*-Paket, benutzen diese Library.

Bei dem Einsatz zur Kontrolle von Programmen, welche von dem *inetd* gestartet werden, wird statt des Original-Programms der TCP-Wrapper gestartet, welcher nach der

[1] `http://www.cert.dfn.de/infoserv/dib/dib-9404.html`
[2] `ftp://ftp.cert.dfn.de/pub/tools/net/logdaemon/`
[3] `http://www.cert.dfn.de/infoserv/dib/dib-9403.html`

Überprüfung der Zugangskontrolle seinerseits das Original-Programm startet. In der Konfigurationsdatei `/etc/inetd.conf` wird aus dem Eintrag

```
ftp stream tcp nowait root /usr/etc/ftpd ftpd
```

beispielsweise die Zeile

```
ftp stream tcp nowait root /usr/etc/tcpd ftpd
```

Dadurch startet der *inetd* bei einer FTP-Anfrage nicht den *ftpd*, sondern den neuen *tcpd*.

Den eigentlichen Kern dieses TCP-Wrappers bildet die Bibliothek *libwrap.a*, welche alle wesentlichen Funktionen zur Zugangskontrolle enthält. Die Bibliothek läßt sich durch die beiden Dateien `/etc/hosts.allow` und `/etc/hosts.deny` konfigurieren. Dort kann für jeden Service (Original-Programmname) der Zugang für bestimmte Rechner erlaubt oder verboten werden. Dort läßt sich auch einstellen, ob und wo der Aufbau dieser Verbindung mitgeloggt werden soll. Bei dem Mitloggen kann man optional versuchen, die Benutzerkennung des aufrufenden Benutzers über einen *ident-lookup* zu erfahren. Voraussetzung für das Gelingen dieser Abfrage ist ein laufender *identd* auf der Gegenseite (siehe Abschnitt A.6). Zusätzlich können diverse weitere Aktionen ausgelöst werden.

Dienste wie *rlogin* oder *rsh* überprüfen den Systemzugang mittels der Dateien `/etc/hosts.equiv` und `~/.rhosts`. In diesen Dateien befinden sich Listen von Rechnernamen, denen der Zugang zum lokalen System gewährt werden soll. Wird nun eine Verbindung von einem anderen Rechner aufgebaut, so fragt das lokale System (der Zielrechner der Verbindung) einen Nameserver nach dem Hostnamen zu der beim Verbindungsaufbau übermittelten IP-Adresse. Ist der vom Nameserver zurückgelieferte Hostname in einer der beiden o.g. Listen enthalten, wird der Zugang zum System erlaubt.

Sollte ein potentieller Angreifer die Kontrolle über diesen Nameserver haben, so könnte er sich einen bestimmten Namen zurückgeben lassen, um sich den unbefugten Zugang zu erschleichen. Die Library *libwrap.a* macht aus Sicherheitsgründen daher eine weitere Anfrage, in der sie den erhaltenen Hostnamen wieder in eine IP-Adresse wandeln läßt. Sollte die so erhaltene Adresse nicht mit der ursprünglich übermittelten Adresse übereinstimmen, so wird die Verbindung unterbrochen.

Zusätzlich zum Logging bietet der TCP - Wrapper die Möglichkeit der Zugangskontrolle, so daß bestimmten Rechner die Benutzung der lokalen Netzwerkdienste untersagt werden kann, während anderen Rechnern der Zugang erlaubt wird.

Der *tcpd* kann alle TCP- und UDP-Programme kontrollieren, die vom Internetdaemon gestartet werden, mit Ausnahme der Programme, die in der Datei `/etc/inetd.conf` als „*rpc/tcp*“ oder „*wait*“ gekennzeichnet sind (z.B. *rexd*).

Der TCP - Wrapper[4] kann vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-93:07-TCP-Wrapper[5] beschrieben.

A.3 Secure Portmapper

Ein weiteres Programm, welches eine Zugangskontrolle mittels der in Abschnitt A.2 vorgestellten Library *libwrap.a* ermöglicht, ist der „secure Portmapper“ von Wietse Venema. Zusätzlich nimmt er keine Anfragen zum Registrieren bzw. Löschen von Portmapper-Informationen von anderen Rechnern an. Änderungswünsche für Programme, welche mit reservierten Portnummern (d.h. kleiner 1024) registriert sind, müssen ebenfalls von privilegierten Portnummern kommen. So wird sichergestellt, daß Systemservices, welche sich mit reservierten Portnummern anmelden, nur von anderen Programmen mit *root*-Berechtigung wieder gelöscht werden können.

Aufforderungen zur Weitergabe von NFS-Anfragen werden vom Portmapper ignoriert (siehe Kapitel 4.11).

Desweiteren gibt das Programm nach dem Öffnen des reservierten Ports 111 seine *Root*-Rechte auf und läuft als Benutzer *daemon* weiter. Dies minimiert den Schaden, falls es doch noch weitere Lücken in der Portmapper-Konzeption geben sollte.

Der Portmapper[6] kann vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-96:03-Portmapper[7] beschrieben.

A.4 SSH

Bei *SSH* handelt es sich um ein Software-Paket, welches sowohl Server- als auch Klienten-Programme enthält. SSH kann als vollwertiger Ersatz für *rlogin, rcp* und *rsh* eingesetzt werden.

Im Gegensatz zu *rlogin* und *rsh* führt *SSH* keine Authentisierung anhand von IP-Adressen durch, sondern nutzt das RSA-Verfahren. Der Zugriff wird nur dem Rechner gewährt, der im Besitz des geheimen RSA-Schlüssels ist. Zusätzlich werden neben

[4] `ftp://ftp.cert.dfn.de/pub/tools/net/TCP-Wrapper/`
[5] `http://www.cert.dfn.de/infoserv/dib/dib-9307.html`
[6] `ftp://ftp.cert.dfn.de/pub/tools/net/portmapper/`
[7] `http://www.cert.dfn.de/infoserv/dib/dib-9603.html`

RSA-Keys für Hosts („Host-Key“) auch individuelle Schlüsselpaare für einzelne Benutzer unterstützt. Nach einer erfolgreichen Authentisierung beim Verbindungsaufbau werden alle folgenden Daten für die Übertragung verschlüsselt – ein Abhören ist somit ebenfalls nicht möglich. Auch Verbindungen von X11-Programmen zum X11-Server über das Netzwerk können über die sichere ssh Verbindung weitergeleitet werden. Das Abhören von „MIT-MAGIC-COOKIES“ wird dadurch verhindert.

Im Gegensatz zu den zu rlogin und rsh gehörigen Servern *rlogind* und *rshd*, die in der Regel durch den *inetd* gestartet werden, muß der SSH-Server (*sshd*) als Dämon beim Booten gestartet werden. Dies wird erreicht, indem folgender Eintrag in der Startup-Datei `/etc/rc.local` ergänzt wird:

```
if [ -x /usr/local/sbin/sshd -a -f /etc/ssh_host_key
  -a -f /etc/sshd_config ]; then
        /usr/local/sbin/sshd
        echo -n " sshd"
fi
```

Die genannten Pfade können sich bei einigen UNIX-Varianten leicht unterscheiden. So sind die Startup-Dateien bei Sun Solaris 2.x/7 im Verzeichnis `/etc/init.d/` zu finden; der SSH-Server sollte unter Solaris durch die Angabe `--prefix=/opt` beim „configure“ als `/opt/sbin/sshd` installiert sein. Die Pfade sind entsprechend anzupassen.

Über das bei der Installation erzeugte Schlüsselpaar, dem Host-Key, werden sich später die Hosts authentisieren. Für eine erfolgreiche Authentisierung müssen beiden Hosts die öffentlichen Schlüssel des jeweiligen Kommunikationspartners vorliegen. Der öffentliche Schlüssel des Hosts wird bei der Installation in der Datei `/etc/ssh_host_key.pub` abgespeichert. Die Liste von Host-Keys anderer Rechner muß vom Administrator in der Datei `/etc/ssh_known_hosts` gepflegt werden.

Da die manuelle Erzeugung der `ssh_known-hosts` Datei durch Zusammenfügen der `ssh_host_key.pub` Dateien mehrerer Rechner bei größeren Netzen sehr aufwendig werden kann, ist bei *SSH* ein Perl-Script enthalten, daß die öffentlichen Schlüssel aller Rechner einer (Sub-)Domain zusammensammelt. Die Datei `ssh_known_hosts` mit den Host-Keys aller Rechner der Domain „ihre.domain.de“ kann durch folgenden Aufruf erzeugt werden:

```
make-ssh-known-hosts ihre.domain.de > ssh_known_hosts
```

Die erzeugte Datei muß vom Administrator als `/etc/ssh_known_hosts` auf alle Rechner der Domain verteilt werden. Da diese Schritte noch ungeschützt durchgeführt werden müssen, stellt sich hier das Problem eines sicheren Bootstraps.

Nach der erfolgreichen Installation sollten im letzten Schritt die Einträge für die *rlogin*- und *rsh*-Dämonen aus der Konfigurationsdatei `/etc/inetd.conf` entfernt werden. Falls der Zugriff auf die beiden Server über den TCP-Wrapper auf wenige IP-Adressen eingeschränkt war, empfiehlt es sich, die selben Adressen auch in die Konfiguration von *SSH* zu übernehmen. Die Zeilen „AllowHosts“ und „DenyHosts“ haben die gleiche Funktion wie eine Eintragung in `/etc/hosts.{allow|deny}` beim TCP-Wrapper.

Die Programme verhalten sich wie normale *rlogin*-, *rsh*- und *rcp*-Klienten. Der Benutzer braucht keine neuen Kommandos zu lernen. *SSH* bietet dabei eine große Vielfalt von Konfigurationsmöglichkeiten. Die detaillierte Darstellung würde den Rahmen dieses Dokuments überschreiten. Eine genaue Beschreibung findet sich in der Dokumentation zu SSH.

SSH[8] kann vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-96:01-SSH[9] beschrieben.

A.5 Crack

Die Paßwort - Schutzmechanismen bilden bekanntermaßen eine Schwachstelle in der Standardinstallation von UNIX - Systemen. Benutzeraccounts, die nicht oder nur unzureichend geschützt sind, bieten einen möglichen Angriffspunkt in das System. Das Programm Crack versucht, mit verschiedenen Methoden, Paßwörter zu raten und gibt bei Erfolg Warnmeldungen aus.

Dem Programm werden als Eingabe ein oder mehrere Paßwortdateien, sowie Wörterbücher (z.B. `/usr/dict/words`) übergeben. Crack sortiert die Wörter aus den Wörterbüchern, verschlüsselt sie nach dem Standardverfahren (crypt) und vergleicht die so erhaltenen Wörter mit den Einträgen aus den Paßwortdateien.

Crack versucht, Paßworte zu raten, indem es vorgegebene Wörter auf verschiedene Art und Weise modifiziert und ausprobiert. In einem ersten Durchgang werden Benutzerinformationen wie Benutzerkennung, voller Name etc. sowie Kombinationen der einzelnen Informationen ausprobiert. In einem zweiten Durchgang werden die Wörter aus einem bereitgestellten Wörterbuch ausprobiert. Zusätzlich wird jedes Wort aus dem Wörterbuch nach vorgebbaren Regeln (z.B. Wortkombinationen, etc.) modifiziert. Dazu bietet Crack eine eigene vollständige Sprache, welche in einer Konfigurationsdatei benutzt werden kann. Die mitgelieferten Regeln probieren unterschiedliche Groß- und Kleinschreibung aus, hängen Ziffern an Wörter, schreiben Ziffern vor Wörter, Wörter werden rückwärts geschrieben, etc. Selbst auf langsameren Rechnern werden ca. 1000 Paßworte pro Sekunde ausprobiert!

[8] `ftp://ftp.cert.dfn.de/pub/tools/net/ssh/`
[9] `http://www.cert.dfn.de/infoserv/dib/dib-9601.html`

Crack versucht nicht, in Benutzeraccounts einzubrechen! Es hält auch keinen Benutzer davon ab, „leicht“ zu ratende Paßwörter zu benutzen! Es ist ein Programm für Systemadministratoren, die regelmäßig die Paßwortdatei(en) überprüfen wollen. Es kann allerdings von jedem Benutzer gestartet werden, der die Paßwortdatei lesen kann (z.B. die Datei `/etc/passwd`, auf die im Normalfall jeder Benutzer lesend zugreifen darf).

Crack bietet die Möglichkeit des verteilten Rechnens auf mehreren Workstations, die automatische (Mail-)Benachrichtigung an Benutzer, deren Paßwort geraten wurde, sowie die Möglichkeit, jederzeit Crack abzubrechen und zu einem späteren Zeitpunkt an der gleichen Stelle wieder aufzusetzen und weiterrechnen zu lassen.

Da das Programm eine relativ langsame Verschlüsselungsroutine benutzt, ist es sinnvoll, eine andere Implementation zu benutzen. Crack 4.1 bietet eine Schnittstelle zum Programm *UFC-crypt* an (s. DIB-93:02), welches in das Verzeichnis *ufc-crypt* im Hauptverzeichnis kopiert werden muß. Crack erkennt bei der Installation, ob UFC-crypt vorhanden ist.

„Crack“[10] kann vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-93:01-Crack[11] beschrieben.
„Cracklib“[12] kann vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-93:03-Cracklib[13] beschrieben.
„UFC-crypt“[14] kann vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-93:02-UFC-crypt[15] beschrieben.

A.6 Ident-Server *identd*

Das „Identification Protocol“, welches in RFC 1413 spezifiziert ist, identifiziert den Benutzer einer aktiven TCP-Verbindung. Das Programm *pidentd* implementiert den Identification – Server (kurz: Ident – Server), der es anderen Rechnern ermöglicht, bei Netzwerkverbindungen die aufrufende User-ID zu ermitteln.

Ein Identification – Server bietet externen Rechnern, zu denen eine aktive TCP - Verbindung besteht, die Möglichkeit, den lokalen Benutzernamen zu dieser Verbindung zu erhalten. Jede TCP-Verbindung läßt sich eindeutig identifizieren durch folgendes Quadrupel:

<lokale Adresse> <entfernte Adresse> <lokaler Port> <entfernter Port>

[10] `ftp://ftp.cert.dfn.de/pub/tools/password/Crack/`
[11] `http://www.cert.dfn.de/infoserv/dib/dib-9301.html`
[12] `ftp://ftp.cert.dfn.de/pub/tools/password/Crack/`
[13] `http://www.cert.dfn.de/infoserv/dib/dib-9303.html`
[14] `ftp://ftp.cert.dfn.de/pub/tools/crypt/ufc-crypt/`
[15] `http://www.cert.dfn.de/infoserv/dib/dib-9302.html`

Der Ident – Server ist eine TCP-basierte Applikation, welche am Port 113 auf Anfragen wartet. Diese haben folgende Form:

<Server - Portnummer> <Client - Portnummer>

Stellt ein Client eine Anfrage, überprüft der Ident – Server zunächst, ob die Verbindung existiert, und liefert bei positivem Resultat den zu der Verbindung gehörenden Benutzernamen. Beispiel:

6193, 23 : USERID : UNIX : joe

Das Programm richtet sich an Systemadministratoren. Es unterstützt keine verbindungslosen Protokolle wie UDP.

Keinesfalls sollte die Auskunft eines Ident – Servers zur Authentisierung benutzt werden! So können Benutzer von „nicht-Unix"-Systemen in der Regel die zurückzugebende Information selber einstellen. Trotz dieser Einschränkung kann die so gelieferte Information dazu dienen, bestimmte Vorfälle genauer zurückzuverfolgen. Eine komplette Beschreibung über die Implementierung und Zielsetzung dieses Dienstes ist in RFC 1413 zu finden.

„Pident"[16] kann vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-94:Pident[17] beschrieben.

A.7 COPS und Tiger

In UNIX - Betriebssystemen gibt es eine Reihe von Systembereichen, die für die Sicherheit des Systems relevant sein können. Hierzu gehören beispielsweise SUID - Programme, Paßworte, Dateizugriffsrechte, etc. Das Überprüfen dieser Bereiche ist in der Regel unproblematisch, aufgrund der Vielzahl von möglichen Problemen jedoch relativ zeitaufwendig. Die Programmpakete *Cops* und *Tiger* beinhalten eine große Zahl von Shellskripts, welche die einzelnen Problembereiche auf bekannte Sicherheitslücken hin untersuchen und eventuell Warnmeldungen ausgeben. Dabei sind sämtliche Programmteile frei konfigurierbar. Dazu werden verschiedene Punkte überprüft:

- Zugriffsrechte
  Dateien oder Verzeichnisse, welche für die Allgemeinheit oder für bestimmte Gruppenmitglieder schreibbar sind, werden aufgelistet.

- Programme mit erweiterten Rechten
  Programme mit einem S-Bit werden aufgelistet. Tiger überprüft zusätzlich die

[16] `ftp://ftp.cert.dfn.de/pub/tools/audit/pidentd/`
[17] `http://www.cert.dfn.de/infoserv/dib/dib-9401.html`

Programme, welche bei dem Systemstart ausgeführt werden, oder welche sich in der *root*-crontab befinden.

- Verdächtige Datei- bzw. Verzeichnisnamen
  Um einzelne Programme oder ganze Verzeichnisse zu verstecken, benutzen Cracker Namen, welche oft überlesen werden. Ein Beispiel hierfür ist ein Verzeichnis mit dem Namen „`...`“, welches wegen der vorhandenen Einträge für „`.`“ und „`..`“ leicht übersehen wird. Solche Einträge werden besonders angezeigt.

- Paßworte
  Benutzer mit Paßworten, welche in dem Systemwörterbuch `/usr/dicts/words` stehen oder sich aus ihren Benutzerinformationen ableiten lassen, werden erkannt. Desweiteren werden Benutzer ohne Paßwort oder Inkonsistenzen, wie doppelt vergebene numerische Benutzerkennungen, angezeigt.

- Programmversionen
  Zu dem Programm Tiger gibt es eine Liste mit Checksummen von Systemprogrammen sowie deren aktuellen Patches. Es wird gewarnt, falls ein Programm nicht bekannt ist oder es eine neuere Version eines Patches für dieses Programm gibt.

- Einbruchversuch
  Cops bietet ein zusätzliches Modul, welches nach gewissen Regeln versucht, die erkannten Schwachstellen in einem System auszunutzen, um *root*-Privilegien zu erlangen.

- Anonymous-FTP
  Die Konfiguration von „anonymous FTP“ wird kritisch überprüft. Fehler werden aufgezeigt und Lösungen angeboten.

- Export von Dateisystemen
  Dateisysteme, welche ohne Einschränkungen exportiert werden, werden aufgelistet.

- Sonstiges
  Cops bietet ein zusätzliches Programm, mit dem die Benutzer die Sicherheit ihrer eigenen Benutzerkennung automatisch überprüfen können. Das Programm analysiert die Startup-Dateien der Benutzer, die Zugriffsrechte ihrer Daten und die Einträge in der `~/.rhosts`-Datei.

Generell sind diese Programme nur für die Entdeckung von Schwachstellen konzipiert, nicht jedoch für die Behebung von gefundenen Fehlern. Der so entstandene Report muß danach von dem Systemadministrator analysiert werden. Beide Programme enthalten eine ausführliche Dokumentation, welche auf gefundene Problemfälle näher eingeht und Lösungsmöglichkeiten vorschlägt.

Beide Programme bieten die Option einer dauerhaften Installation, wobei in einem solchen Falle nur Veränderungen gegenüber dem letzten Report aufgelistet werden.

Da die vorgestellten Programme unterschiedliche Überprüfungsmethoden integriert haben, ist die Installation beider Programme zu empfehlen.

„Cops“[18] und „Tiger“[19] können vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-93:05-Cops[20] beschrieben.

A.8 Tripwire

Das Erkennen von unerwünschten Programmänderungen, welche z.B. von Hackern oder Computerviren verursacht werden können, wird mit zunehmender Systemgröße immer schwieriger. Tripwire überprüft jegliche Arten von Dateien auf Modifikationen und gibt entsprechende Warnmeldungen aus.

Tripwire ist ein Programm, welches die Integrität von Dateisystemen überprüft. Bei jedem Aufruf werden beliebige, vom Benutzer spezifizierte Dateien oder Verzeichnisse mit Informationen verglichen, welche in einer vorher generierten Datenbank gespeichert sind. Diese Datenbank kann jederzeit vom Benutzer verändert bzw. ergänzt werden, so daß gewollte Programmänderungen (z.B. neue Version) nicht zu Warnmeldungen führen. Tripwire benutzt diverse Signaturalgorithmen, um Veränderungen in Dateien festzustellen. Außerdem kann Tripwire in Verzeichnissen feststellen, ob Dateien gelöscht bzw. hinzugefügt wurden.

Das Programm erkennt nur Veränderungen am Dateisystem, die nach der ersten Installation aufgetreten sind. Vor dem Generieren der Datenbank muß man also sicher sein, daß das System „sauber“ ist. Sinnvollerweise sollten sowohl das Programm als auch die Datenbank auf einem schreibgeschützten Medium installiert werden.

Das Programm richtet sich an Systemadministratoren, kann aber von jedem Benutzer verwendet werden.

Um Veränderungen an Programmen oder wichtigen Daten zu erkennen, kann man Prüfsummen der einzelnen Dateien erzeugen, speichern und vergleichen. Diese Aufgabe übernimmt das Programm *tripwire*. Um einen möglichst guten Schutz vor unbemerkten Veränderungen zu bieten, werden mehrere Prüfsummen (jeweils mit unterschiedlichen Verfahren erzeugt) gespeichert. Das Programm unterstützt ganze Netzwerke und einzelne Rechner, indem es verschiedene Datenbanken für netzwerkweite

[18] `ftp://ftp.cert.dfn.de/pub/tools/admin/Cops/`
[19] `ftp://ftp.cert.dfn.de/pub/tools/admin/Tiger/`
[20] `http://www.cert.dfn.de/infoserv/dib/dib-9305.html`

und lokale Einträge verwaltet. Damit ein Cracker die Vergleichsdaten nicht modifizieren kann, sollten diese auf einem Dateisystem liegen, welches physikalisch schreibgeschützt ist. Für jede überprüfte Datei werden ca. 150 Bytes benötigt, um die verschiedenen Prüfsummen zu speichern. Auf einer normalen Diskette lassen sich also bis zu 10000 Programminformationen speichern.

Nachdem die Datenbank erzeugt wurde, wird mit dem Befehl „tripwire“ die Integrität des angegebenen Dateisystems geprüft. Änderungen werden sofort auf dem Bildschirm ausgegeben.

Tripwire kann mit unterschiedlichen Optionen aufgerufen werden, deren wichtigste die folgenden sind:

-update DATEI: aktualisiert den Datenbank-Eintrag für die angegebene Datei.

-interactive: überprüft das Dateisystem und fragt bei Änderungen nach, ob die Datenbank aktualisiert werden soll.

-i: ignoriert bei der Überprüfung die gespeicherten Checksummen und gibt Informationen über gelöschte bzw. neue Dateien aus. Der entscheidende Vorteil dieses Modus ist die Schnelligkeit, mit der die Überprüfung durchgeführt wird.

„Tripwire“[21] kann vom DFN-CERT FTP-Server bezogen werden und ist in einem gesonderten Informations-Bulletin DIB-93:04-Tripwire[22] beschrieben.

[21] `ftp://ftp.cert.dfn.de/pub/tools/admin/Tripwire/`
[22] `http://www.cert.dfn.de/infoserv/dib/dib-9304.html`

Anhang B

Tools für Windows NT 4 Systeme

B.1 Microsoft Management Console

Die *Microsoft Management Console*[1] (MMC) ist ein frei verfügbares Tool, das die Verwaltung einer Windows NT 4.0 Domäne vereinfacht.
Das Programm an sich bietet eine einheitliche Oberfläche, um sog. *Snap-In*s darzustellen. Diese Snap-Ins sind Verwaltungsprogramme, wie die Systemsteuerung, Monitorprogramme und Sicherheitstools.

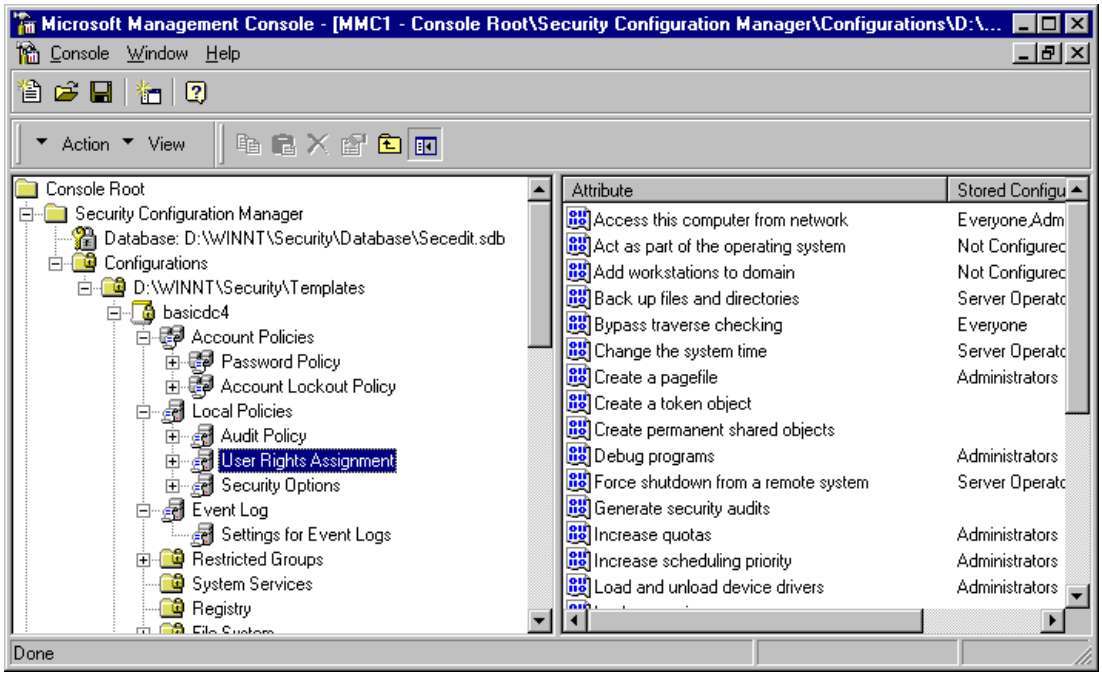


Abbildung B.1: Die Microsoft Management Console

Ein mit der MMC geliefertes Tool ist besonders beachtenswert:
Der *Security Configuration Manager* ermöglicht, Schablonen (Templates) für die sicherheitsrelevante Konfiguration der Windows NT Domäne zu erstellen und anzuwenden. In diesen Schablonen können Einstellungen für die Bereiche:

- Account Policies (Passwort- und Account-Lockout-Einstellungen)
- Local Policies (Audit, Benutzerrechte)
- Eventlog-Einstellungen
- Systemdienste

[1] `ftp://ftp.microsoft.com/bussys/winnt/winnt-public/tools/SCM/`

- Registry-Zugriffsrechte
- Dateisystem-Zugriffsrechte

vorgenommen werden. Dabei müssen nicht immer alle Bereiche konfiguriert werden. Alle Einstellung mit der Option „Not Configured“ ändern nichts an den aktuellen Einstellungen.

Als eine weitere Neuheit können nach der Installation der MMC im Dateisystem von Windows NT 4.0 zusätzliche Rechte gesetzt werden. Dadurch besteht die Möglichkeit, den Zugriffsschutz noch feiner zu granulieren. Diese Rechte sind schon immer im Betriebssystem integriert gewesen, können aber ohne die Installation von MMC nicht angewendet werden. Die zusätzlichen Rechte sind in Abbildung B.2 dargestellt.

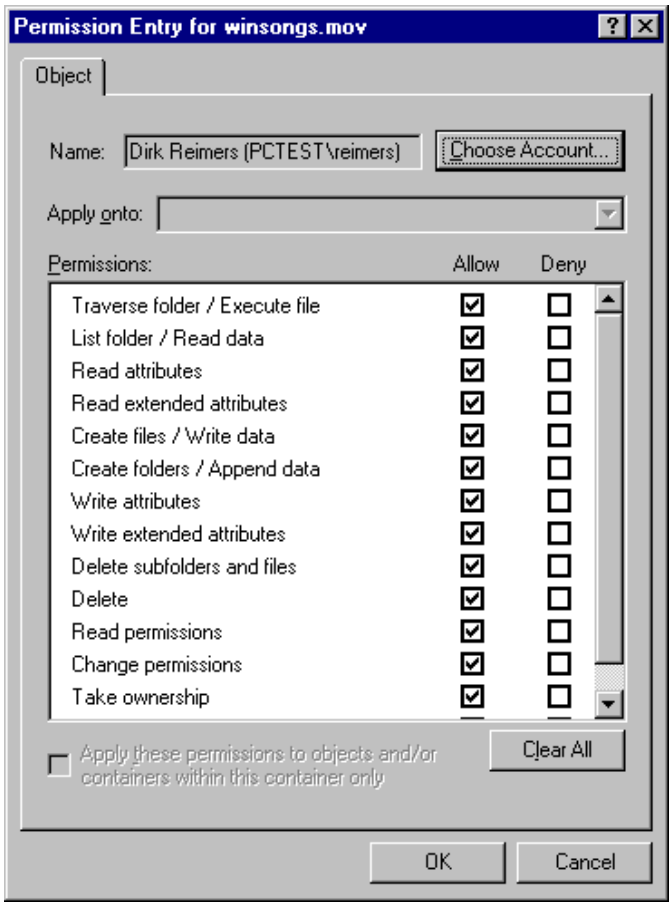


Abbildung B.2: Erweiterte Rechte nach Installation der MMC

Die neuen Rechte werden auch bei Zugriffen über das Netzwerk durchgesetzt. Ein Klient, der diese neuen Rechte nicht kennt, erhält eine Information, daß die betreffende

Datei mit Rechten versehen wurde, die den gewünschten Zugriff nicht erlauben. Die neuen Rechte können jedoch nicht angezeigt werden.

B.2 ElWiz - Der Eventlog Wizard

ElWiz[2] ist ein Tool, das die Verwaltung von Eventlog-Daten eines oder mehrerer Windows NT Netzwerke übernimmt. Auf allen Rechnern, die überwacht werden sollen, wird vom Administrator ein spezieller Dienst installiert. Dieser kommuniziert über eine gesicherte Verbindung mit dem Rechner, der die Daten zur Auswertung bereitstellt.

Durch die Verwendung von ElWiz wird die Möglichkeit geschaffen, Logdaten zentral zu speichern und somit die Gefahr gebannt, daß relevante Log-Information bei einem Systemmißbrauch gelöscht oder verändert werden.

Das Programm bietet die folgende Funktionalität an:

- Ereignisanzeige ähnlich dem EventViewer, nur etwas ergonomischer
- Bequeme Filtermöglichkeiten für alle überwachten Ereignisprotokolle. Es können beim Start alle Ereignisse, die sich seit dem letzten Beenden von Elwiz ereignet haben, automatisch durchsucht werden. Das funktioniert unabhängig davon, wie lange der letzte Lauf von Elwiz zurückliegt.
- Speichermöglichkeiten für die Ereignisprotokolle sowie die Möglichkeit, das momentan geöffnete Protokoll als Textdatei zu speichern. Falls ein Anzeigefilter aktiv ist, werden nur die gefilterten Sätze in die Textdatei übernommen.
- Der Eventwatcher meldet sofort an der Konsole, wenn ein Ereignis protokolliert wurde.
- Grafische Darstellung von interessanten Netzwerk-Informationen, z.B. Daten über gegenwärtig angemeldete Nutzer, Differenzen der Rechneruhren, Uptime, freien Speicher usw.
- Die Überwachungsrichtlinien können für alle zu überwachenden Rechner getrennt festgelegt werden. Aus beliebig vielen Vorgabe-Richtlinien kann eine Liste angelegt werden, mit deren Hilfe es einfach möglich ist, Richtlinien mit Rechnern zu assoziieren.

[2] `http://www.heysoft.de/nt/eventlog/dp-elwiz.htm`

B.3 NTsu

Mit dem Tool NTsu[3] kann der Sicherheitskontext eines beliebigen Benutzers angenommen werden, ohne sich neu ab- und anmelden zu müssen, vergleichbar mit dem Unix-Befehl „su“.

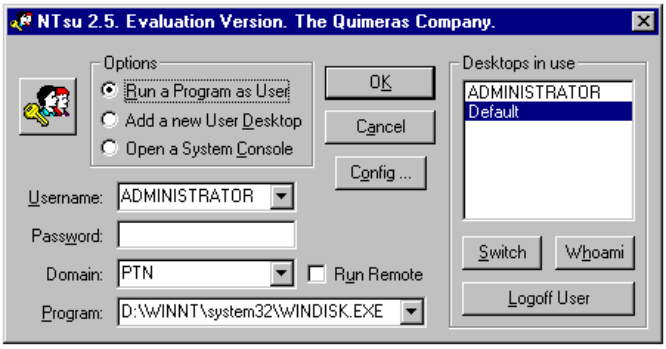


Abbildung B.3: Bedienkonsole von NTsu

NTsu beinhaltet folgende Funktionen:

- Programme im Sicherheitskontext eines anderen Benutzers starten, ohne sich neu ab- und anmelden zu müssen
- Self Impersonated Files (SID): Programme im Sicherheitskontext eines anderen Benutzers starten, ohne Benutzername und Passwort eingeben zu müssen, vergleichbar mit SUID-Programmen unter Unix.
- Virtuelle Desktops, die unter verschiedenen Sicherheitskontexten laufen.

B.4 DumpSec (Dump ACL)

DumpSec[4], früher bekannt als DumpACL, bietet eine komfortable Möglichkeit, Zugriffsrechte auf Dateien, Verzeichnisse u.ä. in einer übersichtlichen Art und Weise darzustellen.

Dieses Programm zeigt eine Aufstellung der Zugriffsrechte für:

- das Dateisystem

[3] http://www.quimeras.com/
[4] http://www.systemtools.com/somarsoft/

- die Systemregistrierung
- Drucker
- freigegebene Verzeichnisse (shares)

Außerdem werden Informationen angeboten über:

- Benutzer
- Gruppen
- Policies
- Benutzerrechte
- Services

Die Darstellung der Rechte für Dateien/Verzeichnisse ist insbesondere sehr geschickt gelöst, da die Möglichkeit besteht, Dateien/Unterverzeichnisse nur dann aufzulisten, wenn ihre Rechte von denen der übergeordneten Verzeichnisse abweichen.

Alle Aufstellungen können sowohl gedruckt als auch zur weiteren Verarbeitung als Textdatei gespeichert werden.

Das Programm Dump-ACL ist Freeware. Eine aktuelle Version[5] kann vom DFN-CERT FTP-Server bezogen werden.

B.5 Dump REG

Da Windows NT keine Möglichkeit bietet, die Zugriffsrechte auf die Registry in einer übersichtlichen Art und Weise darzustellen, wurde das Programm Dump-REG entwickelt.
Mit Dump-REG können die Schlüssel in der Registry incl. Zugriffsrechte und dem letzten Änderungsdatum übersichtlich dargestellt werden. Über eine Export-Schnittstelle ist es außerdem möglich, verschiedene Registries zu vergleichen.

Das Programm Dump-REG ist Freeware. Eine aktuelle Version[6] kann vom DFN-CERT FTP-Server bezogen werden.

[5] `ftp://ftp.cert.dfn.de/pub/tools/admin/dumpacl`
[6] `ftp://ftp.cert.dfn.de/pub/tools/admin/dumpreg`

B.6 NT-Filemon

Mit dem Programm NT-Filemon können Zugriffe auf lokalen Festplatte in Echtzeit überwacht werden.

Konkret werden folgende Informationen gegeben:

- wann erfolgte der Zugriff
- welcher Prozeß löste den Zugriff aus
- welcher Art war der Zugriff
- welche Datei wurde zugegriffen
- war der Zugriff erfolgreich

Die erhaltenen Informationen können gleich gefiltert oder zur weiteren Verarbeitung exportiert werden.

Das Programm NT-Filemon ist Freeware. Eine aktuelle Version[7] kann vom DFN-CERT FTP-Server bezogen werden. Dort befindet sich auch eine aktuelle Windows 95/98 Version.

B.7 NT-Regmon

Mit dem Programm NT-Filemon können Zugriffe auf die Registry in Echtzeit überwacht werden.

Konkret werden folgende Informationen gegeben:

- welcher Prozeß löste den Zugriff aus
- welcher Art war der Zugriff
- auf welchen Key wurde zugegriffen
- war der Zugriff erfolgreich

Die erhaltenen Informationen können gleich gefiltert oder zur weiteren Verarbeitung exportiert werden.

Das Programm NT-Regmon ist Freeware. Eine aktuelle Version[8] kann vom DFN-CERT FTP-Server bezogen werden. Dort befindet sich auch eine aktuelle Windows 95/98 Version.

[7] `ftp://ftp.cert.dfn.de/pub/tools/admin/ntfilmon`
[8] `ftp://ftp.cert.dfn.de/pub/tools/admin/ntregmon`

B.8 SSH

Bei *SSH* handelt es sich um ein Software-Paket, welches sowohl Server- als auch Klienten-Programme enthält. SSH wurde in der Unix-Umgebung entwickelt und stellt einen vollwertigen Ersatz für die (unsicheren) Programme *rlogin*, *rcp* und *rsh* dar. Später wurden auch Versionen auf Windows-Umgebungen portiert.

Im Gegensatz zu *rlogin* und *rsh* führt *SSH* keine Authentisierung anhand von IP-Adressen durch, sondern nutzt das RSA-Verfahren. Der Zugriff wird nur dem Rechner gewährt, der im Besitz des geheimen RSA-Schlüssels ist. Zusätzlich werden neben RSA-Keys für Hosts („Host-Key“) auch individuelle Schlüsselpaare für einzelne Benutzer unterstützt. Nach einer erfolgreichen Authentisierung beim Verbindungsaufbau werden alle folgenden Daten für die Übertragung verschlüsselt — ein Abhören ist somit ebenfalls nicht möglich. Auch Verbindungen von X11-Programmen zum X11-Server über das Netzwerk können über die sichere ssh Verbindung weitergeleitet werden. Das Abhören von „MIT-MAGIC-COOKIES“ wird dadurch verhindert.

Für Windows NT sind mehrere SSH-Klienten frei verfügbar. Die Funktionalität dieser Klienten beinhaltet aber zumeist nur eine verschlüsselte *telnet* Verbindung, mit der Befehle auf dem entfernten Rechner ausgeführt werden können.
Einer dieser Klienten wird auf dem FTP-Server des DFN-CERTs vorgehalten[9].
Die Funktionalität dieses Programmes umfaßt neben *rlogin* auch X11[10]- und Port[11]-Forwarding.

SSH-Server für Windows NT sind z.Zt. nur kommerziell[12] zu erhalten.

[9] `ftp://ftp.cert.dfn.de/pub/tools/net/ssh-win/`
[10] Weiterleiten einer X11-Verbindung über einen verschlüsselten Kanal
[11] Weiterleiten von Ports (z.B. Port 25 für Mail) über einen verschlüsselten Kanal
[12] z.B. bei `http://www.datafellows.com`

Anhang C

UNIX-Systemsicherheit

C.1 Sichere Installation von UNIX Systemen

Die sichere Installation eines Betriebssystems kann in mehrere Schritte aufgeteilt werden. Diese Schritte werden im Folgenden dargestellt und weiter erklärt.

C.1.1 Planung der Installation

Der erste Schritt zu einem sicheren System beginnt schon vor der Installation. Bereits hier müssen wichtige Entscheidungen zur Konfiguration des zukünftigen Systems getroffen werden. Diese Entscheidungen betreffen:

- **Auswahl der verwendeten Hardware**
  Unter UNIX-Systemen ist zu beachten, daß die verwendete Rechnerhardware einen Schutz vor unberechtigtem Zugriff auch ohne Funktion des Betriebssystems bietet. Dazu eignet sich ein EEPROM-Paßwort, ohne dessen Kenntnis ein Rechner nicht gebootet werden kann. Alternativ benutzen einige Hardware-Hersteller ein physikalisches Schloß, das über einen Anschluß an das Mainboard ein Booten der Maschine verhindern kann. Dabei ist zu beachten, daß ein solcher Schutz nicht effektiv ist, wenn das Gehäuse des betreffenden Systems einen direkten Zugang zum Mainboard erlaubt.
  Außerdem bedeutet jede zusätzliche Peripherie wie Diskettenlaufwerke oder CD-ROM-Laufwerke ein zusätzliches potentielles Sicherheitsrisiko, da über diese Medien ein eigenes Betriebssystem gestartet werden kann, das einen direkten Zugriff auf ansonsten geschützte Bereiche des Systems bieten kann.

- **Partitionierung der Festplatte**
  Unter UNIX-Betriebssystemen werden die einzelnen Dateisysteme in einem globalen Verzeichnis-Baum zusammengefaßt (*gemountet*). UNIX richtet Platten mit Partitionen fester Größe ein, deren Größe und Lage beim Formatieren der Platte festgelegt werden.[1] Die Partitionen werden als logisch unabhängige Geräte aufgefaßt, die jeweils ein eigenes Dateisystem enthalten und auf die wie auf physikalisch separate Platten zugegriffen wird.

  Das Partitionieren der Festplatte wird dazu benutzt, den Plattenplatz von gewissen Verzeichnissen im Dateisystem festzulegen. Damit kann z. B. verhindert werden, daß die temporären und Spool-Verzeichnisse den gesamten zur Verfügung stehenden Plattenplatz belegen. Das Platten-Layout unterscheidet sich zum Teil beträchtlich bei den unterschiedlichen UNIX Versionen. Ganz generell läßt sich jedoch sagen, daß System-Verzeichnisse getrennt von Benutzer-Verzeichnissen angelegt werden sollten und beide wiederum getrennt von den

[1] Virtuelle Partitionen, wie z. B. unter AIX oder mit SUN Solstice, seien hier nicht weiter betrachtet.

variablen und temporären Verzeichnissen angelegt werden sollten. Damit wird verhindert, daß Benutzer den gesamten Plattenplatz beanspruchen und z.B. Spool-Services fehlschlagen oder Accounting- bzw. Logging-Prozesse beendet werden. Gleichzeitig wird aber auch verhindert, daß Spool-Prozesse, z. B. Mail, den gesamten Speicherplatz belegen und das System lahmlegen. Des weiteren sollten auch grundsätzlich alle Verzeichnisse, die über NFS exportiert werden sollen, jeweils in eigenen Partitionen liegen.

Das Anlegen mehrerer Partitionen erhöht die Performanz des Systems und ermöglicht die Trennung zwischen schreibbaren (z. B. `/var, /tmp` und `home`) und nicht-schreibbaren (z. B. `/` und `/usr`) Verzeichnissen. Da ein hardwaremäßiger Schreibschutz bei Festplatten in der Regel schwer durchzusetzen ist, bietet die Partitionierung zwar nur eine bedingte Sicherheit, reduziert jedoch die potentiellen Gefahren. Ein nur-lesbar (*read-only*) angelegtes Verzeichnis schützt nicht die Daten, da z. B. über das *raw device*-Interface darauf geschrieben werden kann, es wird jedoch der schreibende *block device*-Zugriff verhindert. Hinzu kommt, daß eine *read-only* Partition jederzeit schreibbar gemountet werden kann, wozu allerdings *Root*-Rechte benötigt werden. Trotzdem empfiehlt es sich, z. B. die Verzeichnisse `/` und `/usr` *read-only* anzulegen, um die Angriffsmöglichkeiten zu minimieren. Die Trennung von schreibbaren Verzeichnissen und System-Verzeichnissen erleichtert die Pflege des Systems und erschwert damit z. B. das Einfügen von Viren und Trojanern.

Durch die Partitionierung besteht die Möglichkeit, Verzeichnisse *nosuid*[2] anzulegen, und damit keinen SUID Zugriff (vgl. 2.3) auf diese Verzeichnisse bzw. die auf der Partition befindlichen Programme zu erlauben. Dieses sollte unbedingt bei allen Verzeichnissen erfolgen, die für Benutzer schreibbar sind (also z. B. `/var, /tmp` und `/home`).

C.1.2 Installation des Betriebssystems

Die Installation des Betriebssystems ist selbstverständlich der zentrale Teil bei der Einrichtung eines neuen Rechners. Bei der Installation ist darauf zu achten, daß die zu installierenden Programme mit den entsprechend richtigen Rechten durchgeführt werden. Dabei ist darauf zu achten, daß nur die für eine Installation notwendigen Rechte verwendet werden. Ein System, bei dem permanent mit dem Systemverwalter-Zugang gearbeitet wird, ist für Angriffe von Programmen ungewollter Funktionalität offener, da die gestarteten Programme mit den Rechten des Benutzers gestartet werden. In diesem Beispiel also mit den Rechten eines Systemverwalters. Auf diese Art können dann Dateien verändert werden, die normalerweise vor der Veränderung durch einen Benutzer ohne besondere Privilegien geschützt sind.

[2]Per Default werden diese Rechte auf *suid* gesetzt, so daß standardmäßig ein SUID Zugriff erlaubt ist

Auf die eigentliche Installation, also das Aufspielen des betreffenden UNIX Systems, soll hier nicht weiter eingegangen werden. Es sei auf die Installationsanweisungen der entsprechenden UNIX Version verwiesen. Grundsätzlich sollten nur die benötigten Programme installiert werden.

C.1.3 Überprüfung der Standardkonfiguration

Nach der Installation von UNIX sollte diese Standardkonfiguration zunächst einmal überprüft und auf bekannte Sicherheitslücken getestet werden. Dafür gibt es die frei verfügbaren Programme *cops* und *tiger* (siehe Anhang A.7), die Pfadeinstellungen, Dateizugriffsrechte und SUID-Programme etc. überprüfen. Bei *Solaris* kann diese Aufgabe das standardmäßig mitgelieferte Programm *aset* (Automated Security Enhancement Tool) übernehmen, das zusätzlich zur Überprüfung der Konfiguration auch eine automatische Korrektur der Konfiguration ermöglicht.

Zusätzlich sollte unbedingt geprüft werden, ob für den regulären Betrieb des Rechners nur die Programme bzw. Services gestartet werden, die wirklich benötigt werden. Es empfiehlt sich daher eine Überprüfung der Startup-Dateien (`/etc/rc*` oder `/etc/init.d/*`) sowie der Datei `/etc/inetd.conf`.

Normalerweise startet der Kernel beim Systemstart das Shell-Skript `/etc/rc`. Abhängig von der eingesetzten UNIX Version werden von `/etc/rc` weitere Shell-Skripte gestartet, die alle mit `/etc/rc*` beginnen (z. B. `/etc/rc.local` oder `/etc/rc.network`) oder die im Verzeichnis `/etc/init.d` (oder `/etc/rc?.d`) liegen. System V Systeme benutzen zusätzlich die Datei `/etc/inittab`, um zu kontrollieren, was bei den unterschiedlichen *Run Levels* ausgeführt werden muß.
Die `/etc/rc` Skripte versetzen UNIX in ein Mehrbenutzer-System und führen eine Anzahl von weiteren Diensten aus. Dazu gehören zum Beispiel:

- Entfernen von temporären Dateien aus dem `/tmp` Verzeichnis
- Entfernen von Lock-Dateien.
- Überprüfung der Konsistenz des Filesystems und Einbindung (Mounten) zusätzlicher Dateisysteme
- Aktivierung von Accounting und Quota Überprüfung
- Starten von Netzwerk-Services (teilweise indirekt über den Internet Daemon *inetd*)

Alle beim Systemstart von `/etc/rc*` gestarteten Programme laufen unter *Root*, sollten also auch dem Benutzer *Root* gehören. Dadurch ist gewährleistet, daß nur *Root* diese Dateien verändern kann. Andernfalls kann der Besitzer des Programms dieses durch ein anderes ersetzen, welches beim nächsten Bootvorgang unter *Root*-Privilegien ausgeführt wird, da alle Programme, die beim Systemstart ausgeführt werden, mit den Rechten des Benutzers *Root* ausgeführt werden.
Es müssen also unbedingt die Besitzrechte der dort gestarteten Programme überprüft und gegebenenfalls geändert werden.

In der Datei `/etc/inetd.conf` muß überprüft werden, welche Programme bzw. Services dort gestartet werden. Die Standard-Konfiguration stellt eine Menge von Services bereit, von denen nicht alle auf jedem System benötigt werden, und welche, die aus Sicherheitsgründen nicht gestartet werden sollten. Dazu gehören *systat, tftp, rexd* und *link*, unter Umständen aber auch die r-Services, der *finger*-Daemon und in besonders sicherheitsrelevanten Umgebungen die *ftp* und *telnet* Daemons.

Zur Überwachung des Systems ist der Einsatz des TCP-Wrappers (vgl. Anhang A.2) zu empfehlen. Dazu wird in der Datei `/etc/inetd.conf` der Pfad zu jedem Netzwerk-Service durch den Pfad des TCP-Wrappers ersetzt, so daß zunächst der *tcpd* gestartet wird, bevor der eigentliche Netzwerk-Service gestartet wird. Z. B. muß in `/etc/inetd.conf` der Eintrag für den *finger*-Daemon

```
finger  stream  tcp  nowait  nobody  /usr/etc/in.fingerd  in.fingerd
```

durch

```
finger  stream  tcp  nowait  nobody  /usr/etc/tcpd  /usr/etc/in.fingerd
```

ersetzt werden.
Erhält der *inetd* nun einen Request für den *fingerd*, wird der *tcpd* gestartet. Dieser protokolliert die *fingerd* Anfrage, überprüft die Zugriffskontroll-Informationen und startet, falls erlaubt, den eigentlichen *fingerd*.

Bestimmte Routinearbeiten werden von dem System durch den Start von Serviceprogrammen automatisch erledigt. Dazu gehören insbesondere Programme zur Auswertung von Log-Informationen oder zum Aufräumen von Dateisystemen. Jeder Benutzer hat die Möglichkeit, solche regelmäßig ablaufenden Programme und deren Startzeiten zu definieren. Die Steuerdateien für diesen Service liegen unter `/var/spool/cron/crontabs/`*userid* und können mit dem Kommando

```
crontab -e
```

ediert werden. Hierbei sollten die gleichen Sicherheitsvorkehrungen wie bei Programmen, welche zum Systemstart ausgeführt werden, angewandt werden. Wenn möglich sollte das Programm unter einer nichtprivilegierten Userid laufen.

C.1.4 Einspielen von Patches

Programme haben Fehler. Sicherheitskritische Software hat daher auch sicherheitskritische Fehler. Ein wichtiger Schritt ist die Korrektur der vorhandenen Software mit Hilfe von Patches, um die bekannten Probleme zu beheben. Direkt nach der Installation sollten daher alle vom Hersteller verfügbaren Patches (zumindest die Sicherheits-Patches) installiert werden. Nach der abgeschlossenen Installation muß sichergestellt werden, daß diese Prozedur (Einspielen von Patches) in regelmäßigen Abständen wiederholt wird, um das System gegen bekannte Sicherheitslücken abzusichern.

Um eine aktuelle Liste der verfügbaren Patches und die Patches selber zu erhalten, wenden Sie sich an den Hersteller Ihres Betriebssystems. In der Regel sind diese Informationen von frei zugänglichen Informationsservern (WWW oder FTP) verfügbar, z. B.

- DEC: Compaq Services Software Patches[3]
- HP: IT Resource Center[4]
- IBM: AIX General Software Fixes[5]
- SGI: Supportfolio Online[6]
- SUN: Recommended & Security Patches[7]

C.1.5 Überprüfung der Integrität des Systems

Zum Abschluß der Installation, bevor die ersten Benutzer Zugriff auf das System erhalten, empfiehlt sich der Einsatz des Programms *Tripwire* (vgl. Anhang A.8), welches die Integrität von Dateisystemen überprüft. Bei jedem Aufruf von *Tripwire* werden, mit Hilfe eines anzugebenden Signaturalgorithmus, die Checksummen der vorher spezifizierten Dateien und Verzeichnisse erzeugt und in einer Datenbank[8] gespeichert. Diese

[3] `http://www.support.compaq.com/patches/index.html`
[4] `http://europe-support.external.hp.com/`
[5] `http://techsupport.services.ibm.com/rs6k/fixes.html`
[6] `http://support-europe.sgi.com/colls/patches/tools/browse/`
[7] `http://sunsolve.sun.com/pub-cgi/show.pl`
[8] Zu beachten ist, daß pro Rechner eine Tripwire-Datenbank angelegt werden muß.

Checksummen müssen nach dem ersten Aufruf von *Tripwire* zusätzlich noch sicher und nicht veränderbar gespeichert werden. Am Besten auf einem Medium, das mit einem Hardware-mäßigen Schreibschutz versehen ist. Sollte dies nicht möglich sein, bietet sich als Kompromiß-Lösung die Verwendung eines Software-Schreibschutzes an, so daß z. B. ein Zip-Drive verwendet werden kann (eine Floppy hat in den meisten Fällen nicht genügend Speicherplatz). Durch regelmäßiges Generieren der Checksummen mit anschließendem Vergleich mit den gesicherten Datenbankeinträgen, lassen sich Änderungen am System leicht feststellen. Dies bedeutet aber auch, daß *Tripwire* im laufenden Betrieb regelmäßig zur Überprüfung angewendet werden muß und daß nach jedem eingespielten Patch und nach Änderungen am System die Prozedur der Checksummen-Generierung und Sicherung wiederholt werden muß.

C.2 Zugangskontrolle über Paßworte

Ein Zugang zu einem Computersystem ist zumeist an einen sog. Account gebunden. Durch diesen Account werden die Rechte eines Benutzers (z.B. Plattennutzung, Prozessorzeit, Zugriffsrechte) spezifiziert. Um bessere Strukturierungsmöglichkeiten zu erhalten, werden neben den Accounts auch noch Gruppen benutzt. Durch die Verwendung von Gruppen ist es möglich einer Menge von Benutzern die gleichen Rechte zuzuweisen. Die Rechte eines Benutzers ergeben sich somit aus der Addition der Accountrechte und der Rechte aller Gruppen, in denen der Benutzer Mitglied ist.
Damit nicht jeder Benutzer einen beliebigen Account verwenden kann, sind die Accounts durch ein Paßwort geschützt. Die Paßworte bilden damit die Grundlage der Authentisierung von Benutzern gegenüber Computersystemen.

Bei der Verwendung von Paßworten zur Authentisierung können - bedingt durch die Art der Implementierung - gewisse Sicherheitsprobleme auftreten (vgl. 2.1)

C.2.1 Wahl der Paßworte

Neben der Speicherung ist die Wahl eines „genügend sicheren“ Paßwortes wichtig. Da dem Benutzer eines Rechnersystems bei der Wahl seines Paßwortes große Verantwortung übertragen wird, ist eine umfassende Aufklärung der Benutzer zu diesem Thema wichtig.
Bei der Auswahl eines Paßwortes sollte darauf geachtet werden, daß es sich dabei nicht um ein Paßwort handelt, das leicht zu erraten ist. Problematisch sind alle Paßworte, die von einem Angreifer ausprobiert werden, z.B.:

- Worte aus dem Sprachschatz (div. Wörterbücher)
- Worte aus anderen Sprachen (ebenfalls div. Wörterbücher)
- alle Arten von Namen (Personen, Städte, Gebäude, Comic-Figuren, ...)
- Rechnernamen, Benutzerkennungen
- Geburtsdaten, Telefonnummern
- Abkürzungen
- Tastaturfolgen (z.B. „qwerty“ oder „asdfgh“)
- Anfangsbuchstaben von bekannten Sprichworten, Liedern, etc.
  (z.B. amesads = „alle meine entchen schwimmen auf dem see“,
  wrssdnuw = „wer reitet so spät durch nacht und wind“, fdhdgg, ...)

Ebenso unbrauchbar sind Modifikationen dieser Worte durch z.B.:

- Anhängen oder Voranstellen einer Zahl (peter09, 7peter, ...)
- Rückwärtsschreibung (retep, reteP, ...)
- Anhängen oder Voranstellen eines beliebigen Zeichens (&peter, ∗peter, ...)
- ...

Obwohl die Aufklärung der Benutzer an erster Stelle stehen sollte, kann es sinnvoll sein, die Auswahl der möglichen Paßworte durch weitere technische Maßnahmen einzuschränken.

C.2.2 Aging

Wird ein Paßwort längere Zeit verwendet, so steigt die Wahrscheinlichkeit, daß das Paßwort (oder Teile davon) anderen Benutzern bekannt wird (z.B. durch ein „über die Schulter gucken“ bei der Eingabe des Paßwortes).

Die Notwendigkeit der gelegentlichen Paßwort-Änderung sollte dem Benutzer deutlich gemacht werden. Moderne Betriebssysteme bieten die Möglichkeit, den Benutzer nach einer vorgebbaren Zeit automatisch zu der Wahl eines neuen Paßwortes aufzufordern. Diese Möglichkeit, die Gültigkeit eines Paßwortes auf eine bestimmte Zeit einzuschränken, wird **aging** genannt. Dem Administrator sollte allerdings bewußt sein, daß dieser Mechanismus nur eine Aufforderung zum Paßwortwechsel darstellt. In der Regel wird nicht überprüft, ob das Paßwort wirklich geändert wurde oder ob der Benutzer kurze Zeit später wieder sein altes Paßwort wählt. Ein wichtiger Schritt bei der Aktivierung des Aging-Verfahrens ist also auch die Aufklärung der Benutzer.

Ein weiteres Problem stellt die Wahl der maximalen Gültigkeitsdauer eines Paßwortes dar. Während zu kurze Intervalle dazu führen können, daß sich die Benutzer ein Schema zur Generierung neuer Paßworte überlegen (etwa die Verwendung der Monatsnummer bei einem monatlichen Wechsel), können zu lange Intervalle den Nutzen des Aging nachhaltig beeinflussen. In diesem Zusammenhang hat es sich als zweckmäßig erwiesen, Paßworte mindestens jedes halbe Jahr zu wechseln.

C.2.3 Speicherung der Paßworte

Unix-Betriebssysteme verschlüsseln die Paßworte mit einem Verfahren, mit dem die Verschlüsselung nicht rückgängig gemacht werden kann (siehe Abschnitt C.2.4). Das so erhaltene verschlüsselte Paßwort wird zusammen mit anderen Benutzer-Informationen

in der Datei `/etc/passwd` gespeichert. Viele Informationen in dieser Paßwort-Datei müssen für alle Benutzer und Programme lesbar sein (um z.B. eine Zuordnung von numerischen Benutzerids (UID's) zu Benutzernamen durchführen zu können). Dadurch können auch potentielle Angreifer (Cracker) darauf zugreifen. Sind in dieser Datei auch die verschlüsselten Paßworte enthalten, kann ein *password cracking* Programm darauf angesetzt werden und schlecht gewählte Paßworte herausfinden. Der für Paßworte benutzte Verschlüsselungsalgorithmus wurde bislang noch nicht „geknackt". Somit kann auch kein Eindringling das verschlüsselte Paßwort in das Klartext-Paßwort entschlüsseln. Aber verschlüsselte Paßworte können mit verschlüsselten Wörterbüchern[9] verglichen werden. Dadurch können schlechte Paßworte (siehe dazu auch C.2.1) leicht geraten werden. Aus diesem Grund unterstützen alle gängigen UNIX-Implementationen *shadow password files*, bei denen die Paßwortdatei in zwei verschiedene Dateien aufgeteilt wird:
Die Datei `/etc/passwd` enthält nur allgemeine Informationen und ist für alle Benutzer lesbar, während die Datei mit den verschlüsselten Paßworten (`/etc/shadow`) vor allen Zugriffen durch Nicht-Root User geschützt ist. Damit können nur Programme, die unter `root`-Privilegien laufen, auf die Paßworte zugreifen, was ausreichend ist.

Leider ist die Verwendung von *shadow password files* nach einer Erstinstallation nicht bei allen Herstellern automatisch aktiviert. Da die genaue Konfiguration von shadow-Paßworten je nach Hersteller sehr unterschiedlich ist, müssen wir hier auf die mitgelieferte Dokumentation Ihres Betriebssystems verweisen. Oft finden sich die Hinweise zu diesem Setup unter anderen Rubriken wie „C2 Security".
Wird ein altes Betriebssystem benutzt, welches nicht über die Möglichkeit der shadow-Paßworte verfügt, so kann das Public-Domain Programm „shadow" eingesetzt werden. Diese Software ist über anonymous FTP vom DFN-CERT FTP-Server[10] erhältlich.

C.2.4 Details zum Paßwortalgorithmus

Bei der Anmeldung am System wird das vom Benutzer eingegebene Paßwort mit dem Paßwort aus der lokalen Paßwortdatenbank verglichen. Die einfachste Möglichkeit wäre nun, diese Paßworte in einer Datenbank zu speichern und direkt mit der Eingabe des Benutzers zu vergleichen. Dies ist aber wenig zweckmäßig, wenn ein Benutzer eben diese Datenbank mit den Paßworten im Klartext lesen kann.
Der Paßwortalgorithmus beschreibt das Verfahren, mit dem verhindert wird, daß aus dem in der Datenbank gespeicherten Eintrag auf das Paßwort zurückgeschlossen werden kann.

[9] Ganz so einfach ist das jedoch nicht, da das verschlüsselte Paßwort noch zwei extra Bits, den sogenannten *Salt* (vgl. Abschnitt C.2.4) vorangestellt haben. Dieser muß natürlich auch berücksichtigt werden und erschwert das *Cracking*.

[10] `ftp://ftp.cert.dfn.de/pub/tools/password/shadow/`

Unix verwendet zur sicheren Speicherung der Paßworte ein Verschlüsselungsverfahren, das gewährleistet, daß durch die Kenntnis des verschlüsselten Paßwortes nicht auf das ursprüngliche Paßwort geschlossen werden kann.
Die Paßworte werden mit der frei verfügbaren Funktion *crypt()* verschlüsselt. Crypt benutzt eine leicht modifizierte Version des bekannten DES Algorithmus'. Konkret wird ein 64-bit Block von Nullen mit dem Paßwort des Benutzers als Schlüssel verschlüsselt. Die Modifikation des DES-Algorithmus besteht darin, daß nicht nur der zu verschlüsselnde Block und der Schlüssel als Parameter übergeben werden, sondern auch ein zufälliger 12-bit Wert, der sog. *Salt*. Die 12-bit werden durch zwei Zeichen aus dem Bereich [A..Z], [a..z], [1..9] und [.,], die vor das eigentliche Paßwort in die Paßwortdatei eingefügt werden, repräsentiert. Der Salt geht direkt in die Verschlüsselung mit ein und sorgt dafür, daß ein Benutzerpaßwort den 64-bit Block von Nullen auf 4096 verschiedene Arten verschlüsseln kann. Dieses hat zwei Effekte: Zum einen werden Angriffe auf die Paßwortdatenbank schwieriger, da es nicht mehr ausreicht, ein Wörterbuch mit z.B. 100.000 Wörtern zu verschlüsseln, sondern es müssen, um die gleiche Menge von Paßworten testen zu können, 409.600.000 Einträge in dem Wörterbuch erzeugt werden, da jedes Wort auf 4096 verschiedene Arten verschlüsselt werden muß. Zum anderen werden für zwei Benutzer mit dem gleichen Paßwort mit großer Wahrscheinlichkeit unterschiedliche Einträge in der Paßwortdatei erzeugt.

C.2.5 Länge der Paßworte

Wie oben beschrieben werden die Paßworte nicht im Klartext gespeichert. Ist der Verschlüsselungsalgorithmus für die Paßworte hinreichend gut, so kann ein Paßwort aus einem gegebenen verschlüsselten Paßwort nur bestimmt werden, indem auf ein mögliches Paßwort das Verschlüsselungsverfahren angewendet wird. Ist das Ergebnis dieses Verschlüsselungsverfahrens das gegebene verschlüsselte Paßwort, so war das gewählte mögliche Paßwort korrekt.
Angriffe auf Paßwortdatenbanken (mit Einträgen von verschlüsselten Paßworten) werden häufig über zwei verschiedene Arten vorgenommen:

- Wörterbuchattacken
  Hierbei wird ein Wörterbuch mit wahrscheinlichen Worten benutzt und davon ausgehend versucht, das Paßwort zu finden.

- Brute-Force-Attacken
  Hierbei werden alle möglichen Kombinationen über einem gegebenen Alphabet erzeugt und probiert.

Damit ist ein Wörterbuchangriff für eine gegebene Paßwortlänge schneller als ein Brute-Force-Angriff, wobei letzterer aber auch sämtliche möglichen Paßworte findet.

Um sich vor einem Brute-Force-Angriff zu schützen, müssen die Benutzer davon überzeugt werden, ein möglichst langes Paßwort zu benutzen.
Eine Sun Ultra mit 170 MHz kann beispielsweise pro Sekunde ca. 10.000 Paßworte verschlüsseln. Daraus ergibt sich der in Tabelle C.1 dargestellte Aufwand[11] :

Länge der Paßworte	**Kombinationen**	**Zeit**
1	96	< 1 Sek
2	9.216	$\approx 1{,}0$ Sek
3	884.736	$\approx 89{,}0$ Sek
4	84.934.656	$\approx 2{,}3$ Std
5	8.153.726.976	$\approx 9{,}4$ Tage
6	$7{,}8276*10^{11}$	$\approx 2{,}5$ Jahre
7	$7{,}5145*10^{13}$	$\approx 238{,}1$ Jahre
8	$7{,}2139*10^{15}$	$\approx 22.859{,}5$ Jahre

Tabelle C.1: Zeitabschätzung für einen Brute-Force-Angriff mit einer Sun Ultra

Bei den meisten Unix-Derivaten ist die Länge eines Paßwortes auf 8 Zeichen beschränkt. Die Zeichen können aus dem gesamten Zeichenvorrat, der mit einer Terminaltastatur erzeugt werden kann, gewählt werden. Als realistische Schätzung wurde für die Zeitabschätzung in Tabelle C.1 angenommen, daß der Zeichenvorrat aus 96 verschiedenen Zeichen besteht. Wird vom Benutzer ein längeres Paßwort gewählt, so werden nur die ersten 8 Zeichen dieses Paßwortes zur Authentisierung benutzt. Sollte ein kurzes Paßwort gewählt werden, so kann dieses eventuell durch vollständige Suche (d.h. automatisiertes Ausprobieren aller möglichen Kombinationen) erraten werden. Eine solche vollständige Suche wird auch Brute-Force-Attack genannt. Es werden dabei systematisch alle Buchstabenkombinationen verschlüsselt und mit dem gesuchten verschlüsselten Paßwort verglichen.

Ein Paßwort sollte daher mindestens 7 Zeichen lang sein und den möglichen Zeichenvorrat nutzen. Es sollte möglichst immer aus einer Kombination von Klein-, Groß-Buchstaben, Ziffern und Sonderzeichen bestehen.

[11] Zu beachten ist, daß diese Abschätzung für einen Rechner gilt. Werden Brute-Force-Angriffe durchgeführt, so werden meist mehrere Rechner parallel benutzt und die zu berechnenden Kombinationen aufgeteilt.

C.2.6 Erkennung schwacher Paßworte

C.2.6.1 Nachträgliche Erkennung ungeeigneter Paßworte

Da Paßworte den Zugang zu einem Rechnersystem sichern, sind schwache, also leicht zu ratende Paßworte der erste Punkt, den ein Angreifer testen wird. Bei einem bestehenden System sollte in regelmäßigen Abständen überprüft werden, ob die Benutzer schwache Paßworte verwendet haben.

Relativ bekannt ist die Methode, eine gegebene Paßwort-Datei mit Hilfe eines Programmes zum Ausprobieren von Paßworten zu überprüfen. Das bekannteste Programm für UNIX ist „Crack“[12], das in Kapitel A.5 beschrieben ist.

Diese Methode hat diverse Nachteile. Da nur die verschlüsselten Paßwort-Einträge zur Verfügung stehen, muß jedes zu testende Wort ebenfalls verschlüsselt und anschließend mit dem gespeicherten Eintrag verglichen werden. Dieses erfordert ein großes Maß an Rechenzeit. Ein weiteres Problem ergibt sich aus der Zeitspanne zwischen der Wahl des Paßwortes und der Erkennung als „schwaches“ Paßwort.

C.2.6.2 Erkennung „schwacher“ Paßworte bei der Wahl

Eine effizientere Methode schwache Paßworte zu finden ist, direkt bei der Eingabe zu prüfen, ob die Paßworte bestimmten Regeln genügen. Damit kann dann eine langwierige Suche, wie sie bei der nachträglichen Erkennung nötig ist, umgangen werden.

Um den Benutzer bei der Wahl eines Paßwortes zu unterstützen, kann man unter UNIX das `passwd`-Programm durch das `npasswd`- Programm ersetzen, welches ein gewünschtes neues Paßwort nach vorgebbaren Regeln überprüft und dem Benutzer eventuell weitere Hilfestellungen gibt. Vorteilhaft an dieser Methode ist, daß die Überprüfung des Paßwortes anhand des Klartext-Paßwortes sehr effizient und zügig erfolgen kann.

C.2.6.3 Alternative Authentisierungsverfahren

Wie oben beschrieben (vgl. Tabelle C.1), erfordert das Anmelden an einem System üblicherweise die Angabe eines geheimen Paßwortes, welches am Zielrechner verschlüsselt und mit den Einträgen in der Paßwort-Datei (`/etc/passwd`) verglichen wird. Problematisch ist dieses Verfahren beim Anmelden über ein Netzwerk, da das eingegebene Paßwort unverschlüsselt über das Netz übertragen wird. Angreifer könnten die übertragenen Paßworte abhören und mißbrauchen. Es gibt zwei Möglichkeiten, dieses Problem zu lösen:

[12] `ftp://ftp.cert.dfn.de/pub/tools/password/Crack/`

- Verschlüsselung bei einer Verbindung über das Netzwerk
  Werden die Daten schon vor der Übertragung des Paßwortes verschlüsselt, so können die Daten zwar abgehört werden, es ist jedoch nicht möglich, aus den erlauschten Daten ein Paßwort zu erkennen.
  Eine Verschlüsselung bedingt aber, daß auf dem Zielrechner ein sog. Dämon installiert ist, der einen verschlüsselten Verbindungsaufbau erkennt und entsprechend reagiert.

- Einsatz von Einmal-Paßwortsystemen
  Hierbei wird für jeden Benutzer eine Liste von Paßworten erzeugt, die in einer vorher definierten Reihenfolge zum Anmelden benutzt werden. Jedes Paßwort auf dieser Liste wird genau einmal verwendet. Somit kann ein Angreifer auch nicht mit einem abgehörten Paßwort einen Zugang zum System erhalten.

Das Programmpaket S/Key unterstützt die Benutzung von Einmal-Paßworten bei Unix-Systemen. Durch dieses Programmpaket ist gewährleistet, daß das geheime Paßwort des Benutzers niemals über das Netzwerk übertragen wird.

Das S/Key - System schützt den Benutzer zwar nicht vor dem unerwünschten Abhören seines geheimen Paßwortes, da ein Paßwort jedoch nur einmal benutzt wird, ist eine abgehörtes Paßwort in dem Moment des Abhörens veraltet.
Beim Anmelden an einem entfernten Rechner (z.B. über rlogin, telnet, ftp, etc.) wird statt des geheimen Paßwortes ein Einmal-Paßwort zur Authentisierung benutzt. Dieses Paßwort wird nie wieder verwendet, so daß das Abhören des Paßwortes durch Dritte keinen Missbrauch ermöglicht.

Das S/Key - System besteht aus zwei Seiten: Der Benutzer (Klient) muß das Einmal-Paßwort erzeugen; das entfernte System (Server) muß dieses Paßwort verifizieren.

Die Generierung des Paßwortes erfolgt am lokalen System durch das Programm `key`. Dieses Programm erwartet als Eingabe ein geheimes Paßwort und liefert ein oder mehrere Einmal-Paßworte. Nur das Einmal-Paßwort wird anschließend an den entfernten Rechner (den Server) übermittelt. Alternativ ist es dem Benutzer auch möglich, eine bestimmte Zahl von Einmal-Paßworten im Voraus berechnen zu lassen und beispielsweise auf einem Zettel zu verwahren.

C.3 Zugriffskontrolle

Nachdem sich ein Benutzer am System durch die Eingabe seines Paßwortes authentisiert hat, muß das Betriebssystem dafür sorgen, daß dem Benutzer entsprechend seiner Rechte nur gewisse System-Privilegien und System-Zugriffe gewährt werden. Dies wird zum Großteil über Zugriffe auf Dateien kontrolliert.

C.3.1 Zugriffskontrolle auf Dateien

Daten werden häufig auf speziellen Netzwerkservern gespeichert. So können die Benutzer von allen Workstations im Netzwerk auf diese Dateien zugreifen. Insbesondere ist es auch möglich, daß mehrere Benutzer die gleichen Dateien benutzen können. So zum Beispiel Anwendungsprogramme, die auf einem Server im Netz installiert sind und über das Netzwerk gestartet werden. Der Vorteil dieses gemeinsamen Zugriffs besteht darin, daß Speicherplatz gespart wird und sich der Verwaltungsaufwand reduziert. Bei einem Programm-Update muß nicht jede einzelne Workstation mit dem neuen Programm versehen werden. Bei Datenbeständen liegen alle Daten an einer zentralen Stelle vor und es existieren keine lokalen Kopien, deren Aktualität nur schwer zu bestimmen ist.
Diese zentralen Daten sollen aber nicht ohne Einschränkung für alle Benutzer zugänglich sein. Insbesondere ist ein Schreibrecht auf installierte Programme nicht notwendig. Daher sollte dem Benutzer nur Zugang zu den von ihm benötigten Programmen und Daten erlaubt werden. Dies wird dadurch erreicht, daß die Zugriffsrechte auf die Dateien an einen Besitzer und eine Gruppe geknüpft werden, die unabhängig voneinander sind.

C.3.1.1 Rechte unter UNIX

Im UNIX-Dateisystem wird zwischen drei Typen des Zugriffsrechts auf Dateien[13] und Verzeichnisse unterschieden: read, write und execute (vgl. Tabelle C.2). Zusätzlich gibt es die drei Klassen **Besitzer**, **Gruppe** und **Andere** (das sind alle anderen Benutzer des Systems) für die die Zugriffsrechte separat spezifiziert werden.

Rechte können mit dem Befehl `ls -l` angezeigt werden. Dabei wird das Leserecht mit `r`, das Schreibrecht mit `w` und das Ausführungsrecht mit `x` gekennzeichnet. Faßt man jeweils die Rechte für Besitzer, Gruppe und Andere zusammen, so können diese einfach als oktale Zahl gespeichert werden (siehe Tabelle C.2).

[13] Außerdem können Rechte auch für Geräte, Sockets und Pipes vergeben werden. Die Rechte sind dann äquivalent zu den Rechten für Dateien.

Recht	Verzeichnis	Datei	Oktalzahl
Read	Lesen des Inhalts	Lesen des Inhalts	4
Write	Erstellen und Löschen von Einträgen	Speichern einer vorhandenen Datei	2
E**x**ecute	Wechseln in das Verzeichnis	Ausführen der Datei	1

Tabelle C.2: Rechte unter UNIX

Ein mögliche Ausgabe des `ls -l` Befehls ist unten dargestellt:

```
-rw-r--r--   1 reimers  dfncert  8841568 Jun 13  1997 Bericht.ps
-rwxr-x---   1 reimers  dfncert      116 Sep 12 13:01 noum-
laut
-rw-------   1 reimers  dfncert      969 Jun  3  1997 script
drwxr-x---   3 reimers  dfncert      512 Sep 18 14:25 sour-
ce
```

Das erste Zeichen in diesem Listing gibt an, ob der betreffende Eintrag ein Link `l`, ein Verzeichnis `d` oder eine Datei `-` ist. Die Anordnung der Rechte erfolgt in Tripeln (von links nach rechts) für Besitzer, Gruppe und Andere.

Um die Rechte auf eine Datei zu setzen, muß die Datei dem jeweiligen Benutzer gehören. Ist das der Fall, so kann das Programm `chmod` benutzt werden[14].

Wird ein neues Verzeichnis oder eine neue Datei erstellt, so werden die Rechte entsprechend der sog. umask (das ist das Kürzel für *user file-creation mode mask*) gesetzt. Mit dem Befehl `umask` kann die umask jederzeit neu gesetzt werden, dabei sind die Rechte, die eine neu erstellte Datei **nicht** haben soll, oktal als Parameter zu übergeben. Diese standardmäßige Rechtedefinition macht die Arbeit mit einem System sicherer, da ansonsten eine neu erstellte Datei die Rechte 666 (rw-rw-rw) und ein neu erstelltes Verzeichnis die Rechte 777 (rwxrwxrwx) erhalten würde. Ohne umask wäre es also möglich, daß auf ein neu erstelltes Verzeichnis oder eine neu erstellte Datei direkt nach der Erstellung durch einen unberechtigten Benutzer zugegriffen wird. Daher muß die umask als erster Befehl beim Starten des Betriebssystems bearbeitet werden, damit z.B. temporäre Dateien von anschließend gestarteten Dämonen mit den richtigen Rechten erzeugt werden.

Als weitere Möglichkeit zur Vergabe von Zugriffsrechten können auch Access Control Listen (ACLs) benutzt werden. Mit diesen ACLs können die gleichen Rechte wie oben beschrieben vergeben werden, es ist jedoch möglich, diese Rechte diffiziler zu

[14]Die genaue Syntax dieses Befehls kann den man-Seiten entnommen werden.

definieren. So können durch eine ACL einer Menge von Benutzern, die nicht notwendigerweise der gleichen Gruppe angehören müssen, Zugriffe auf ein Verzeichnis bzw. eine Datei gewährt werden. Dabei können den Benutzern dieser Menge auch durchaus unterschiedliche Zugriffsrechte zugewiesen werden.
Die konkrete Implementation dieser ACLs ist allerdings sehr stark herstellerabhängig. Ein Beispiel wäre SunOS 5.x (Solaris 2.x/7) mit den Befehlen `getfacl` zur Anzeige und `setfacl` zur Änderung der ACLs.

Besonderheiten von Zugriffsrechten:

- Es besteht die Möglichkeit eine Datei auszuführen, ohne sie lesen zu können. Dies kann sinnvoll sein, falls der binäre Code des Programms vor Benutzern verborgen werden soll. Auch wenn das Kopieren einer ausführbaren Datei verhindert werden soll, ist diese Einstellung zweckmäßig.
- Neben den bisher besprochenen Rechten sollen noch kurz zwei Besonderheiten beschrieben werden:
  - set-UserID (SUID) bzw. set-GroupID-Bits SGID) auf Dateien, die in Abschnitt C.3.2 besprochen werden
  - set-GroupID-Bit auf Verzeichnisse
    Wird das SGID-Bit (siehe Abschnitt C.3.2) auf ein Verzeichnis gesetzt, so erhalten alle in diesem Verzeichnis angelegten Dateien den gleichen Gruppen-Besitzer, wie das Verzeichnis selbst (unabhängig von der Gruppenzugehörigkeit des Verzeichnis-Besitzers). Dabei ist zu beachten, daß bei den angelegten Unterverzeichnissen automatisch das SUID-Bit gesetzt wird.
  - Sticky-Bit
    Wird das Sticky-Bit auf ein Verzeichnis gesetzt, bewirkt es, daß die Benutzer nur die eigenen Dateien in dem Verzeichnis löschen dürfen bzw. die Dateien, auf denen sie Schreibrechte haben. (Findet z.B. Verwendung bei /tmp-Verzeichnis)

Nachdem die Rechte unter UNIX-Systemen erklärt wurden, können die Änderungen der Rechte aufgezählt werden, die nach der Installation des Betriebssystems durchgeführt werden müssen. Auch hier kann leider kein installationsunabhängiges Verfahren vorgestellt werden, da in unterschiedlichen Distributionen verschiedene Schwachstellen durch falsche Zugriffsrechte entstehen. Auf der Suche nach solchen Schwachstellen sollte die folgende Anleitung trotzdem grundsätzlich helfen:

Zunächst muß festgestellt werden, welche Dateien nach der Installation noch für alle Benutzer schreibbar sind:

```
find / -perm -2 \! -type l -ls
```

Mit dem Befehl

```
find / -perm -20 \! -type l -ls
```

wird angezeigt, welche Dateien für welche Gruppen schreibbar sind. Mit den Ergebnissen beider Befehle sollte dann für den Einzelfall bestimmt werden, welche Rechte unbedingt notwendig sind und welche entfernt werden können. Hierzu siehe auch das folgende Kapitel C.3.2

C.3.2 Privilegierte Programme

Im laufenden Rechnerbetrieb müssen die normalen Benutzer hin und wieder Programme benutzen, die besondere Privilegien benötigen, die der betreffende Benutzer nicht hat. Beispielsweise benötigt das UNIX-Programm `passwd` besondere Privilegien, damit Eintragungen in die Paßwortdatenbank vorgenommen werden können. Wird ein Programm ausgeführt, so geschieht diese Ausführung mit den Rechten des jeweiligen Benutzers. Am Beispiel des `passwd`-Programms würde das bedeuten, daß eine Änderung des Paßworteintrags in der Paßwortdatenbank nicht vorgenommen werden kann, da einem normalen Benutzer die Schreibrechte auf eben diese Datei fehlen. Damit die Benutzer trotzdem ihre Paßworte ändern können, gibt es besondere sog. privilegierte Programme. Diese Programme haben die Möglichkeit, das Recht, mit dem sie ausgeführt werden, zu ändern:
Das Programm `passwd` unter UNIX schaltet sich zur Änderung des Eintrages in der Paßwortdatenbank auf die Rechte des Benutzers *root* um und kann dann in die entsprechende Datei schreiben.
Bei der Verwendung solcher Programme ist allerdings mit großer Vorsicht zu verfahren, damit keine Sicherheitslücken in einem bestehenden System geschaffen werden (siehe unten).

UNIX-Systeme verwenden privilegierte Programme für die verschiedensten Zwecke. Ein Beispiel wurde oben beschrieben, das Programm `passwd`.
In UNIX-Systemen wird ein privilegiertes Programm dadurch gekennzeichnet, daß das sog. S-Bit (für set-UserID bzw. set-GroupID) für eine Datei gesetzt ist. Durch das Setzen des set-UserID-Bits auf eine Datei (z.B. durch den Befehl `chmod u+s <Dateiname>`) wird das Programm mit den Rechten des Besitzers dieses Programmes gestartet. Wird das set-GroupID-Bit gesetzt, (z.B. durch den Befehl `chmod g+s <Dateiname>`), so wird das Programm mit den entsprechenden Rechten der Gruppe, der das Programm gehört, ausgeführt.

Privilegierte Programme bergen aber auch Gefahren:
Besondere Gefahren bestehen bei Programmen, die das Starten von weiteren Program-

men (z.B. einen Kommandointerpreter) ermöglichen. Hat so z.B. der Kommandointerpreter `csh` das set-UserID-Bit gesetzt, so haben alle Benutzer, die diesen Kommandointerpreter starten, alle Rechte, die der Benutzer *root* hat[15].
Neben diesen Programmen sind alle anderen Programme, die ein S-Bit gesetzt haben, eine potentielle Gefahr. Diese Gefahr besteht, da zumeist nicht bekannt ist, ob ein betreffendes Programm über eine besondere Option oder einen Programmierfehler verfügt, der die Ausführung eines Kommandos aus dem Programm heraus ermöglicht. Durch einen solchen Befehl kann dann z.B. ein neuer Kommandointerpreter (z.B. mit *root*-Privilegien) gestartet werden.
Nach der Installation eines UNIX-Systems haben viele Programme bereits ein S-Bit gesetzt. Bevor jedoch weiter auf diese Programme eingegangen werden kann, müssen sie zunächst gefunden werden. Dies kann mit dem Befehl

`find / -type f -perm -4000 -ls` für set-UserID bzw.

`find / -type f -perm -2000 -ls` für set-GroupID

erreicht werden.
Die Entscheidung, ob dieses S-Bit tatsächlich benötigt wird oder nicht, ist jedoch nicht trivial und kann an dieser Stelle nicht umfassend für alle Betriebssysteme geklärt werden. Dies ist insofern besonders schwierig, da verschiedene Systemanforderungen verschiedene Programmkonfigurationen benötigen. Um aber trotzdem einen Eindruck zu vermitteln, welchen Programmen in vielen Systemkonfigurationen das S-Bit entzogen werden kann, werden im folgenden einige Beispiele kurz aufgeführt[16].

- `/usr/bin/chkey`
  Um die secure RPC Schlüssel in der Datei `/etc/publickey` zu ändern, wird (analog zum `passwd` Programm und der `/etc/passwd` Datei) das set-UserID Bit benötigt.
- `/usr/bin/eject`
  Wenn ein auszuwerfendes Medium noch gemountete Bereich enthält, so wird von dem `eject` Kommando aus das `umount` Kommando gestartet, um diese Partitionen zu unmounten. Dieses unmounten benötigt natürlich *root* Berechtigungen.
- `/usr/bin/fdformat`
  Hier wird eine ähnliche Strategie wie bei dem `eject` Kommando verfolgt. Das

[15] Standardmäßig besitzt *root* diesen Kommandointerpreter
[16] Diese Beispiele beziehen sich auf eine Sun Solaris Installation.

set-UserID-Bit ist nur für die `-U` Option (unmount) notwendig. Mit der gleichen Begründung wie bei dem `eject` Kommando wird auch hier das S-Bit entfernt.

- `/usr/bin/login`
  Dieses Programm authentisiert den Benutzer (z.B. über Abgleich des eingegebenen Paßwortes mit den Daten in `/etc/shadow`). Nach einer erfolgreichen Authentisierung werden die entsprechenden Rechte für diesen Benutzer benötigt (also `setuid(username)` etc.). Für diese Aufgaben werden selbstverständlich *root* Rechte benötigt. Andererseits wird das `login` Programm eigentlich nie direkt gestartet, sondern immer von Programmen aus aufgerufen, die sowieso schon mit *root* Berechtigungen laufen. Von daher wird das S-Bit nur dann gebraucht, wenn ein bereits eingeloggter Benutzer sich nochmals unter einer anderen Userid einloggen möchte. Dafür kann aber das `su` Programm verwendet werden und das S-Bit kann von dem `login` Programm gelöscht werden.

- `/usr/bin/passwd, /usr/bin/yppasswd, /usr/bin/nispasswd`
  Die *root* Berechtigungen werden benötigt, um das neue Paßwort in die `/etc/shadow` Datei einzutragen (und natürlich auch, um das alte Paßwort zu verifizieren). Da eine Änderung der Paßworte im allgemeinen erlaubt sein soll, ändert sich hier nichts.

Allein diese kleine Auswahl zeigt, daß es sehr schwierig ist, ohne genaue Anforderungsdefinitionen zu bestimmen, welche Programme unbedingt mit dem S-Bit gestartet werden müssen. Wenn Unklarheiten bestehen, sollte ein fachkundiger UNIX-Administrator konsultiert werden, der mit Kenntnis der betreffenden Konfiguration und der Systemanforderungen genauere Aussagen treffen kann.

C.3.3 Zugriffskontrolle auf Ressourcen

Ähnlich den Daten muß auch die Nutzung der Systemressourcen kontrolliert werden. Sonst könnte z.B. ein Benutzer durch einen übermäßigen Verbrauch von Rechnerzeit den ordnungsgemäßen Betrieb verhindern.
Bevor an dieser Stelle auf den Schutz dieser Systemressourcen eingegangen werden kann, muß noch kurz beschrieben werden, was wir an dieser Stelle unter den Systemressourcen verstehen wollen:

- Prozessorzeit
- Hauptspeicher
- Anzahl der Prozesse

- Anzahl der offenen Dateien
- Genutzter Platz auf den Festplatten
- Peripherie

Die Ressourcen lassen sich in die folgenden Kategorien einteilen:

- Systemkern-Ressourcen
  - Prozessorzeit
  - Hauptspeicher
  - Anzahl der Prozesse
  - Anzahl der offenen Dateien
- Speichergeräte
  - Genutzter Platz auf den Festplatten
- externe Geräte

Systemkern-Ressourcen
In UNIX-Systemen muß der Systemkern (*Kernel*) permanent im Speicher geladen sein. Im Gegensatz dazu werden die Prozesse der Benutzer auf Verlangen auf andere Geräte (z.B. einen besonderen Bereich einer Festplatte) ausgelagert (*geswappt*). Viele Parameter werden zur Zeit der Kompilierung des Kernels fest eingestellt. So auch die Maximalgröße verschiedener Parameter, die unter anderem die Anzahl der geöffneten Dateien und die aktuell laufenden Prozesse beinhalten. Wird nun durch einen Prozeß, der extrem viele Dateien öffnet, oder viele Kind-Prozesse startet, die Grenze eines dieser Parameter erreicht, so können keine weiteren Einträge vorgenommen werden. Die Folge wäre, daß keine weiteren Dateien geöffnet werden können bzw. keine weiteren Prozesse gestartet werden können.

Um dieses zu verhindern, werden besondere Restriktionen vergeben. So kann ein Prozeß nur eine bestimmte maximale Anzahl von Dateien öffnen und nur eine bestimmte Anzahl von Kind-Prozessen starten. Außerdem wird ein Mindestmaß an Freiraum in den Parametern für den Systemadministrator reserviert, so daß dieser sich auf jeden Fall auf einem Rechner einloggen kann und notfalls durch direktes Löschen einiger Prozesse Platz in beiden Tabellen schaffen kann. Somit kann ein Denial-Of-Service Angriff verhindert werden. Die Einschränkungen für:

- Rechnerzeit

- Dateigröße
- Datengröße
- Größe für Coredump Datei
- Größe des virtuellen Speichers
- Anzahl der gleichzeitig geöffneten Dateien

können mit dem Befehl `limit` eingestellt werden. Dies kann in den Dateien `\etc\profile` und `\etc\.login` eingetragen werden.

Speichergeräte
Unter UNIX-Systemen können viele unterschiedliche Speichergeräte wie Festplatten, Floppy-Disks, Magnetlaufbänder oder CD-ROM-Laufwerke angeschlossen werden. Diese Geräte werden für die Benutzer transparent im Dateisystem gemountet.
Damit die Benutzer nicht, wie bei anderen Betriebssystemen, z.B. bei Windows NT, den kompletten Platz auf einem Device in Anspruch nehmen können, werden neben Rechten auch sog. *Quotas* für die schreibbaren Verzeichnisse des Dateisystems eingerichtet[17]. Eine Quota stellt eine Begrenzung des maximal nutzbaren Speicherplatzes dar. Es wird zwischen einer Soft-Quota und einer Hard-Quota unterschieden. Bei einer Soft-Quota wird dem Benutzer eine Warnmeldung zugestellt, die den Benutzer daran erinnert, daß er dabei ist, den ihm zur Verfügung gestellten Plattenplatz zu überschreiten. Eine Soft-Quota kann (für einen vorher vom Administrator festgelegten Zeitraum) von dem Benutzer überschritten werden. Die Hard-Quota stellt eine tatsächliche Grenze dar, die vom Benutzer nicht überschritten werden kann. Über die eingestellte Quota kann sich ein Benutzer mit dem Befehl `quota -v` informieren. Der *r*oot-Benutzer kann die Quota mit dem Befehl `edquota` bearbeiten.

Externe Geräte
Unter externen Geräten seien hier Geräte wie die Tastatur oder der Bildschirm verstanden. Auch diese Geräte müssen geschützt sein, damit nicht ein Benutzer, der sich über eine Netzwerkverbindung auf einen Rechner eingeloggt hat, die Eingaben der Tastatur lesen kann und dann z.B. ein an der Tastatur eingegebenes Paßwort abhören kann. Ebenso sollte es nicht möglich sein, direkt auf den Bildschirminhalt eines entfernten Computers zugreifen zu können. Unter UNIX wird das dadurch geregelt, daß die Rechte dieser Geräte direkt nach dem Login auf der Konsole geändert werden. Genauer wird der Besitz dieser Geräte auf den jeweils lokal eingeloggten Benutzer geändert wird:

[17] Ein Benutzer könnte versuchen, die Platte, in der die Audit- und Log-Daten gespeichert werden mit Datenmüll zu füllen, so daß es nicht möglich ist, weitere Daten in die Audit- oder Log-Dateien zu schreiben. Danach würden keine weiteren Aktionen in die Log- und Audit Dateien eingetragen werden können; u.a. auch keine Meldungen über erfolgte oder versuchte Angriffe auf das System.

```
                          crw-------
  1 reimers  dfncert   16,  0 Mar 18  1996 conskbd@0:kbd
```

C.4 Überwachung des Systems

Die Überwachung des Systemverhaltens ist ein wichtiger Vorgang, um die Sicherheit, Verfügbarkeit und Integrität von Systemen zu gewährleisten. Die alleinige Installation von Schutzmechanismen ist nicht ausreichend, da diese in der Regel Ergebnisse – in Form von Meldungen – produzieren, die aufgezeichnet und vor allem auch ausgewertet werden müssen. Zum einen, um sicherzugehen, daß die Mechanismen korrekt funktionieren, zum anderen, um auf Aktionen, die nicht den eigenen Sicherheitspolitiken entsprechen, richtig reagieren zu können. Dabei ist zu beachten, daß es oft nicht möglich ist, basierend auf den Ergebnissen eines einzigen Sicherheitsmechanismus, die richtigen Schlüsse zu ziehen. Oft kann nur durch die Kombination der verschiedenen Protokolle auf die Art des Problems geschlossen werden.

Die von den Schutzmechanismen produzierten Meldungen werden zumeist in Dateien gespeichert, den sog. Log-Dateien. Die dort gespeicherten Informationen sind wichtig, um ein System zu sichern, da sie eine Art Vorgangsgeschichte beschreiben (z. B. **welche** Programme **wann** von **wem** gestartet wurden). [Diese Vorgangsgeschichte wird auch *Audit Trail* genannt.] Dadurch ist es möglich, bereits geschehene Ereignisse zu verfolgen und die Ursache für das Fehlverhalten zu finden. Auch wenn durch die Log-Dateien kein Fehlverhalten verhindert wird, so können die Verantwortlichen (Prozesse, Personen) für eben diesen Fehler gefunden werden.

Log-Dateien haben jedoch eine grundsätzliche verwundbare Stelle: Da die Log-Dateien meist auf der Maschine gespeichert werden, auf der die Daten produziert werden, kann ein erfolgreicher Angreifer die Daten löschen, bevor sie ausgewertet werden konnten. Ein sinnvolles Vorgehen ist daher, die Log-Daten auf einer anderen Maschine aufzuzeichnen oder, falls dies nicht möglich sein sollte, die Dateien in regelmäßigen Intervallen an einen anderen Rechner zu senden.

UNIX-Systeme bieten eine Vielzahl von Log-Daten an, die von unterschiedlichen Programmen erzeugt werden. Neben individuellen Logs einzelner Programme gibt es ein Standard-Verfahren, um Informationen zu Logging-Zwecken zu verschicken, diese zu kategorisieren und je nach Erzeuger und Priorität weiterzuverarbeiten. Dieses zentrale Logging-Konzept wird *syslog* genannt und wird in Kapitel C.4.1 vorgestellt.

Das Überwachen des kernelbezogenen Systemverhaltens wird auch *Audit* genannt und im Abschnitt C.4.2 vorgestellt. Beim Auditing werden, im Gegensatz zum *Accounting* (vgl. C.4.3), nicht nur Daten, wie die Prozessorauslastung, Festplattennutzung oder Online-Zeit erfaßt, die zur Kostenberechnung dienen, sondern detaillierte Auskünfte über die gestarteten Prozesse und deren Aktivitäten protokolliert.

C.4.1 Syslog

Betriebssystem, System-Programme, Informationsdienste und Anwendungen liefern zahlreiche Meldungen über den aktuellen Zustand der Systeme, Zustandsänderungen, erkannte Fehler oder benutzte Ressourcen.

Fehlermeldungen des Betriebssystems (z. B. „kein Speicher mehr frei“ oder „Festplatte voll“) sind meistens Informationen an den Administrator, der dann regulierend in den Ablauf eingreifen muß, um einen fehlerfreien Betrieb des Systems zu gewährleisten. Meldungen von Informationssystemen werden häufig für Statistiken oder zur Fehlersuche (z. B. bei Zugriffen auf nicht existierende HTML-Dokumente) genutzt und die Meldungen von Anwendungsprogrammen sind als Hilfestellung für den Benutzer gedacht. Von dem Standpunkt der Sicherheit aus, sind neben den Meldungen des Betriebssystems insbesondere die Informationen der einzelnen System-Programme sehr wichtig. Fehlerhafte oder ungewöhnliche Ereignisse bei den angebotenen Diensten lassen eventuell Aktionen erkennen, bei denen versucht wird, Sicherheitsmechanismen oder Policies zu umgehen.

Im einfachsten Fall schreibt ein Programm seine Log-Informationen in eine Datei. Das ist zwar für die Entwickler schnell und einfach zu realisieren, hat jedoch einige entscheidende Nachteile:

- Jedes Programm schreibt in eine eigene Datei. Die Folge davon ist eine große Anzahl von Dateien, in denen die Informationen des Gesamtsystems verstreut sind.
- Die Daten werden für jeden Rechner einzeln gespeichert. In einem größeren Netzwerk von Rechnern hat man dadurch kaum noch einen Gesamtüberblick.
- Bei Programmen, die (auch nur teilweise) unter Benutzerberechtigungen laufen, gibt es Probleme mit dem Schreibzugriff, falls man nur eine Logdatei anlegen möchte.
- Es werden genau die Informationen gespeichert, die das Programm liefert bzw. die bei Programmstart aktiviert werden. Eine Kontrolle über die zu speichernden Nachrichten zur Laufzeit existiert nicht.

Um diese Probleme zu lösen, wurde das *syslog*-Protokoll entwickelt. Durch ein eigenes Client-Server Konzept wird dabei die Generierung einer Nachricht von der Speicherung getrennt. Jede Nachricht wird von dem erzeugenden Programm klassifiziert und priorisiert. Unter UNIX stehen dabei üblicherweise 8 System-Klassen und 8 weitere, vom Benutzer frei zu belegende, Klassen zur Verfügung. Für die Prioritäten stehen 8 Abstufungen (von „debug“ bis „emergency“) zur Verfügung. Durch die Verwendung einer eingeschränkten Menge von Klassen erhält der Systemadministrator

die Möglichkeit, die Log-Meldungen zu strukturieren, wodurch die Übersichtlichkeit erhöht wird.
Programme schicken Textnachrichten an den `syslogd`, der anhand einer Konfigurationsdatei und den vorgegebenen Merkmalen der Nachricht (Klasse und Priorität) entscheidet, wie die Information gespeichert und/oder weitergeleitet werden soll. Folgende Aktion sind in der Datei *syslog.conf* konfigurierbar:

- Abspeicherung der Nachricht in einer Datei.
- Weiterleitung der Nachricht an einen anderen Rechner.
- Weiterleitung an bestimmte Benutzer, falls diese eingeloggt sind (z.B. an Operateure)
- Weiterleitung der Nachricht an alle eingeloggten Benutzer.

Alle Aktionen können beliebig kombiniert werden.

Durch den Einsatz des *syslog*-Protokolls können die Informationen zentral gesammelt werden. Mit Hilfe der Unterteilung in einzelne Prioritätsklassen können bereits erste Entscheidungen getroffen werden, welche Art von Meldungen man speichern möchte. So brauchen im Normalfall „debug“-Meldungen nicht aufgezeichnet zu werden. Leider sind nicht alle möglichen Nachrichten und deren Priorität bekannt. In der Regel werden also mehr Daten anfallen, als man tatsächlich für den unmittelbaren Bedarf benötigt. Es stellt sich also die Frage, wie mit den Informationen umgegangen werden muß.

Zusätzlich zu einer lokalen Speicherung der Logdaten sollten diese auf jeden Fall an einen anderen Rechner weitergeleitet werden, damit in dem Falle eines Mißbrauchs eine Kopie der Daten vorhanden ist, die von dem Angreifer auf dem anderen Rechner nicht ohne weiteres gelöscht werden kann (es sei denn, er hat ebenfalls einen Zugang zu dem Loghost).

Durch die Zeile

```
*.debug    @loghost
```

in der Datei `/etc/syslog.conf` werden alle Syslog-Nachrichten an den Rechner `loghost` weitergeleitet.

Von dieser Möglichkeit der Konfiguration der Systemmeldungen sollte unbedingt Gebrauch gemacht werden. Natürlich reicht es nicht aus, wenn die Meldungen nur in einer Datei gespeichert werden. Die Loginformationen sollten regelmäßig gelesen

werden. Hier finden sich auch Informationen über fehlerhafte Loginversuche oder andere Mißbrauchversuche.

Eine Möglichkeit zur automatischen Logdaten-Analyse wird in Kapitel C.4.4 vorgestellt.

C.4.2 Audit

Sicherheitsrelevante Aktionen (z. B. das Anmelden am System) sollten vom Betriebssystem überwacht und protokolliert werden. Erfolgreiches Audit, d. h. Audit, durch das im Verdachtsfalle die verursachende Person identifiziert werden kann, hängt von zwei Merkmalen ab:

- Identifikation und
- Authentisierung

Beim Anmelden am System werden diese beiden Merkmale vom Benutzer erfüllt. Die Identifikation wird durch den Accountnamen erreicht. Die Authentisierung erfolgt mit der Eingabe des Paßwortes.
Beim Anmelden wird dem Benutzer eine eindeutige Audit-ID zugewiesen, mit der alle folgenden Aktionen im Audit-System protokolliert werden. Selbst wenn ein Benutzer seine Identität ändert (z. B. mit dem `su` Kommando), bleibt die Audit-ID erhalten. Durch Audit wird ermöglicht, daß

- alle sicherheitsrelevanten Aktionen auf dem System protokolliert werden können
- diese Aktionen im Audit-Trail aufgezeichnet werden und
- Mißbrauch oder unberechtigter Gebrauch des Systems durch die Analyse des Audit-Trails entdeckt werden kann.

Audit-Daten sind mit mehr Informationen versehen als Accounting-Daten und sollten nach Möglichkeit verwendet werden. Zunächst muß darauf hingewiesen werden, daß Audit unter UNIX herstellerabhängig ist. D. h. daß Unterschiede zwischen den überwachbaren Ereignissen in den verschiedenen UNIX-Varianten bestehen. Audit wird im folgenden am Beispiel von Solaris dargestellt:

Aktivierung des Audits
Für die Aktivierung des Audits bietet Solaris ein Skript namens `/etc/security/bsmconv`. Dieses Skript sorgt dafür, daß die entsprechende Audit-Startup-Datei unter `/etc/init.d/audit` aktiviert wird und in der Datei

`/etc/system` Kernel-Parameter gesetzt werden, die dafür sorgen, daß das Audit im Kernel aktiviert wird.

Speicherung der Auditdaten
Die Auditdaten werden, soweit nicht anders konfiguriert, unter `/var/audit/` gespeichert. Dieses kann aber durch Ändern der Datei `/etc/security/audit_control` angepaßt werden. Hierbei ist zu beachten, daß mehrere Verzeichnisse angegeben werden können, die der Reihe nach solange mit Auditdaten gefüllt werden, bis keine Daten mehr gespeichert werden können.

Konfiguration der Auditdaten
Die Konfiguration der Auditdaten kann in vier Kategorien vorgenommen werden:

- Audit-Klassen (Eintrag *flags* in der Datei `/etc/security/audit_control`):
  Die Auditklassen beschreiben die verschiedenen Aktionen, die überwacht werden können. Diese Aktionen sind z. B. das Lesen von Dateien, das Ändern von Dateiattributen, An- und Abmelden am System. Für eine komplette Liste der überwachbaren Aktionen kann die `audit_class`(4) man-Seite zu Rate gezogen werden.

- Audit-Ereignisse (Eintrag +/- vor den einzelnen Audit-Klassen):
  Die Audit-Ereignisse beschreiben, ob die erfolgreiche Ausführung einer Aktion überwacht werden soll, die nicht erfolgreiche Ausführung oder beides.

- Benutzereinstellungen (Einträge in der Datei `/etc/security/audit_user`):
  Es kann notwendig sein, für verschiedene Benutzer unterschiedliche Audit-Strategien (welche Aktionen sollen überwacht werden) anzuwenden. Die Konfiguration für die einzelnen Benutzer kann über die Datei `/etc/security/audit_control` erfolgen.

- Audit-Policies:
  Mit den Audit-Policies wird das Verhalten eines Benutzers beschrieben, das nicht überwacht werden muß. Weicht der Benutzer in seinem Verhalten davon ab, so werden diese Abweichungen aufgezeichnet. Dies ist für die Reduzierung der Audit-Daten zweckmäßig.

Es ist zu beachten, daß alle Einstellungen lokal auf den einzelnen Rechnern vorgenommen werden müssen, da das Auditing von der lokalen Maschine durchgeführt wird. Ein grundsätzlicher Schwachpunkt des Auditing ergibt sich daraus, daß Audit im Systemkern (Kernel) implementiert ist. Beim Ersetzen bestimmter Systemprogramme

(z. B. `login`-Programm) wird die Audit-Funktionalität stark eingeschränkt. Kommt z. B. `ssh` zum Einsatz, ist kein Audit mehr möglich, da keine Audit-ID mehr zugewiesen wird.

C.4.3 Accounting

Accounting-Daten werden hauptsächlich zur Kostenabrechnung mit dem Benutzer über die verbrauchten Ressourcen verwendet. Sie bieten aber gleichzeitig eine einfache Möglichkeit, die Nutzung bestimmter Programme benutzerbezogen zu protokollieren. Da Audit im Systemkern implementiert sein muß, kann beim Ersetzen bestimmter Systemprogramme, wie dem `login`-Programm oder einem `ftp`-Klienten, die Funktionalität stark eingeschränkt werden. Durch den parallelen Einsatz von Audit- und Accountingsystemen kann z. B. die Benutzung eines eigenen FTP-Klienten, der nicht unbedingt Audit-Einstellungen tätigt, trotzdem überwacht werden.

Zum Aktivieren (Deaktivieren) von Accounting unter UNIX-Systemen (hier am Beispiel von Solaris), werden die Systemstartdateien unter `/etc/init.d/` benutzt. Dazu wird das Skript (`/etc/init.d/acct`) mit dem Parameter *start* (bzw. *stop*) aufgerufen. Beim Aktivieren des Accountings wird eine Meldung in die `wtmp`-Datei eingetragen, das Accounting gestartet und alle Accountingdaten vom vorherigen Tag gelöscht. Da dieses Skript nur beim Neustart des Systems ausgeführt wird, sollte noch ein *cron-job* aktiv sein, der in regelmäßigen Abständen (am besten täglich) die Accountingdaten sichert[18].

Die erfaßten Accountingdaten können mit den Befehlen `lastcomm` und `acctcom` betrachtet werden. `lastcomm` wird benutzt, um festzustellen, welche Befehle wann zuletzt benutzt wurden, `acctcom` wird benutzt, um festzustellen, welcher Benutzer wann welche Befehle ausgeführt hat. Wie an den Darstellungen unten zu erkennen ist, bietet `acctcom` etwas mehr Informationen. Zu beachten ist außerdem, daß

[18] Zusätzlich sollten die Daten auf das tägliche Backup gespeichert werden

`lastcomm` die Befehle in der umgekehrten Reihenfolge (d. h. die zuletzt eingegebenen Befehle zuerst) darstellt.

```
>lastcomm  reimers |grep 11:40
scan        reimers  __             0.04 secs Tue Dec 16 11:40
folder      reimers  __             0.03 secs Tue Dec 16 11:40
rmdir       reimers  __             0.01 secs Tue Dec 16 11:40
rm          reimers  __             0.01 secs Tue Dec 16 11:40
inc         reimers  __             0.03 secs Tue Dec 16 11:40
folder      reimers  __             0.03 secs Tue Dec 16 11:40
rmdir       reimers  __             0.02 secs Tue Dec 16 11:40
rm          reimers  __             0.02 secs Tue Dec 16 11:40
inc         reimers  __             0.04 secs Tue Dec 16 11:40

>acctcom -u reimers -s 11:40 -E 11:41
END AFTER: Tue Dec 16 11:40:00 1997
END BEFOR: Tue Dec 16 11:41:00 1997
COMMAND                          START    END          REAL     CPU    MEAN
NAME       USER     TTYNAME      TIME     TIME       (SECS)  (SECS) SIZE(K)
inc        reimers   ?           11:40:12 11:40:12     0.04    0.04 1472.00
rm         reimers   ?           11:40:12 11:40:12     0.02    0.02 2264.00
rmdir      reimers   ?           11:40:12 11:40:12     0.02    0.02 2280.00
folder     reimers   ?           11:40:13 11:40:13     0.03    0.03 2445.33
inc        reimers   ?           11:40:43 11:40:43     0.03    0.03 1736.00
rm         reimers   ?           11:40:43 11:40:43     0.02    0.01 3504.00
rmdir      reimers   ?           11:40:43 11:40:43     0.02    0.01 3560.00
folder     reimers   ?           11:40:43 11:40:43     0.03    0.03 2514.67
scan       reimers   ?           11:40:43 11:40:43     0.05    0.04 2070.00
```

Den protokollierten Programmnamen in den Accountingdaten kann allerdings nur sehr bedingt vertraut werden. Zum einen werden weder die Parameter der aufgerufenen Programme aufgeführt, noch wird aufgezeichnet, aus welchem Verzeichnis heraus das Kommando aufgerufen wurde. Zum anderen kann ein Angreifer ein kritisches Kommando (z. B. `cc`) unter einen harmlosen Namen kopieren (z. B. `vi`) und dann diese Kopie ausführen. In den Accountingdaten wird dann der Name der Kopie aufgezeichnet.
Trotzdem stellen die Accountingdaten eine wertvolle Informationsquelle dar.

C.4.4 Auswertung der Sicherheitslogs

Leider sind die Log-, Audit- und Accounting-Daten bei den verschiedenen Betriebssystemen nicht identisch. Die Auswertung der Daten muß daher betriebssystemspezifisch erfolgen.

Audit

Audit-Daten können mit dem vom Hersteller mitgelieferten Tool gefiltert und angezeigt werden. Dieses Kommando ist zum Beispiel als `praudit` oder `auditpr` verfügbar.

Beispiel:

```
% praudit /var/audit/19971217150610.19971217151620.rzdspc79
[...]
header,81,2,login - local,,Wed Dec 17 16:15:43 1997, + 870009500 msec
subject,root,root,other,root,other,494,494,0 0
                                    rzdspc79.informatik.uni-
hamburg.de
text,successful login
return,success,0
[...]
```

Audit Informationen dieser Art müssen allerdings für den Benutzer weiter aufbereitet werden. Leider gibt es hierfür keine Standard-Programme für UNIX.

Accounting

Neben der reinen Anzeige der Accounting Informationen existieren je nach Hersteller weitere Programme zur Aufbereitung der Daten. In der Regel wird in einer solchen Zusammenfassung aufgelistet, welcher Benutzer wieviel Ressourcen (Loginzeit, Festplattenplatz, Programmausführung) genutzt hat. Das Hauptziel des Accounting ist die Abrechnung der verbrauchten Ressourcen, so daß die Zusammenfassungen aus Sicherheitssicht nicht weiter interessant sind.

Die Accounting-Daten können allerdings Hinweise auf eine ungewöhnliche Nutzung enthalten (z. B. wenn von einem Benutzer, der sonst nur tagsüber arbeitet, plötzlich nachts sehr viele Programme gestartet werden). Auf die Anzeige der Prozeß-Informationen aus den Audit-Logs wurde bereits in Kapitel C.4.3 eingegangen.

Syslog und Logs generell

Häufig werden die gesammelten Informationen während des „normalen Betriebes“ nicht weiter beachtet. Man geht davon aus, daß der Betrieb problemlos läuft, und braucht daher keine Ressourcen für die Analyse der anfallenden Informationen zu verwenden. Erst dann, wenn etwas nicht mehr planmäßig läuft, wird versucht, die Ursache des Problems mit Hilfe der Logdaten zu finden.

Obwohl diese Vorgehensweise auf den ersten Blick relativ effizient aussieht, hat sie einige Nachteile. So erkennt man beispielsweise Probleme erst dann, wenn sie sich im täglichen Betrieb äußern oder wenn man von anderen auf das Problem hingewiesen wird. Nicht selten kann ein Angreifer monatelang fremde Rechnersysteme nur deswegen unbemerkt benutzen, weil die vorhandenen Meldungen über die Aktivitäten des Angreifers einfach nicht angesehen werden. Werden die Logdaten dann tatsächlich benötigt, so stellt man häufig fest, daß die mit der Auswertung beschäftigen Personen keine Kenntnis über die verschiedenen Log-Informationen, deren Ursprung oder Bedeutung haben. Durch fehlende Routine werden die Analysen langwierig und schwierig.

Die vermeintliche Arbeitsersparnis kann sich sehr schnell in das genaue Gegenteil umkehren. Eine Vogel-Strauß-Politik („Kopf in den Sand und mir passiert schon nichts“) ist angesichts der steigenden Zahlen ernst zu nehmender Sicherheitsvorfälle grob fahrlässig. Was werden Sie sagen, wenn Sie gefragt werden, warum Sie denn nichts gegen den Angreifer unternommen haben, um Schaden von Ihnen und Dritten abzuwenden, obwohl Sie anhand Ihrer Logdaten seit Monaten wissen müßten, daß eine unberechtigte Nutzung vorliegt?

Aus dem vorherigen Absatz den Schluß zu ziehen, jede Meldung zu lesen und darauf zu reagieren, hilft auch nicht weiter. Man muß sehr schnell erkennen, daß die anfallenden Informationen viel zu umfangreich sind, als daß sie noch manuell bearbeitet werden könnten. Ein einzelner Rechner generiert problemlos mehrere Megabytes an Logdaten pro Woche.

Viele der festgehaltenen Logdaten beschreiben ganz gewöhnliche Aktivitäten, wie das Anmelden eines Studenten an einer Konsole in einem CIP-Pool. Da (hoffentlich) die meisten Meldungen keiner weiteren Beachtung bedürfen, stören diese nur bei der Analyse der Gesamtdaten. Man muß sich also Gedanken machen, wie man die Datenflut reduzieren kann. Dies führt zum nächsten Schritt, bei dem Programme zur Reduzierung der Datenmengen eingesetzt werden.

Um die Bearbeitung der anfallenden Daten überhaupt erst zu ermöglichen, müssen diese auf ein vertretbares Maß reduziert werden. Wenn statt mehreren Megabytes nur noch einige Kilobytes übrigbleiben, so können diese einfacher gesichtet werden. Das hört sich zwar relativ logisch an, ist aber nicht einfach umzusetzen:

- Welche Logmeldungen gibt es überhaupt?

  In der Regel hat man keinen Überblick über alle möglichen Meldungen der unterschiedlichen Programme, sondern kennt lediglich einige Meldungen, die häufig vorkommen. Beispiel:
  Die Meldung des Programmes `/bin/login` „repeated login failures“ kennt fast jeder — die Meldung des gleichen Programmes „/etc/fbtab: bad entry: /dev/....“ kennt dagegen kaum jemand.

- Welche Meldungen sind wichtig, welche unwichtig?

  Eine Entscheidung darüber, welche Nachrichten als wichtig erachtet werden müssen, hängt sehr stark von dem Einsatzgebiet der Rechner und der benutzten Programme ab.

Es muß ein Mechanismus entwickelt werden, der ermöglicht, Wissen über die Relevanz einzelner Meldungen zu erlangen. Da nicht alle möglichen Meldungen bekannt sind, muß man zunächst alle Meldungen als kritisch einstufen. Ausgehend von dieser Datenmenge müssen nun einzelne Meldungen gefunden werden, die für den laufenden Betrieb als normal erachtet werden und daher keine besondere Aufmerksamkeit benötigen. Im einfachsten Falle betrachtet man dazu jede Zeile einzeln.

Für einen Erkennungsmechanismus bietet sich die Beschreibung der jeweiligen Meldungen durch einen regulären Ausdruck an. Dabei können mit Hilfe von Platzhaltern (Wildcards) die variablen Teile einer Nachricht beschrieben werden. So steht z. B. der Ausdruck `[0-9]` für eine beliebige Ziffer zwischen „`0`“ und „`9`“. Eine genauere Beschreibung der Syntax von regulären Ausdrücken sprengt den Rahmen dieses Leitfadens. Weitere Informationen befinden sich in den UNIX `man`-Seiten der Programme *egrep(1), ed(1)* sowie mit einer sehr ausführlichen Beschreibung in [Friedl 1997]. Auch wenn diese Art der Beschreibung etwas komplex und unübersichtlich erscheint, so sollte man sich unbedingt mit regulären Ausdrücken beschäftigen. Alle Programme zur automatischen Auswertung benötigen diese Form der Beschreibung.

Beispiel:

Auf dem Rechner `fred` läuft ein anonymous FTP-Server. Normalerweise sind die FTP-Logins zu diesem Rechner (als Benutzer „anonymous“) nicht relevant:

```
fred ftpd[8488]: ANONYMOUS FTP LOGIN FROM rechner.irgendwo.de [123.45.6
```

Natürlich will man nicht nur diese eine Verbindung als „weniger wichtig“ einstufen, sondern alle anonymous-FTP Verbindungen zu dem Rechner `fred`. Dazu benutzt man nun die erwähnten regulären Ausdrücke, um die obige Zeile zu beschreiben:

```
fred ftpd\[[0-9]*\]: ANONYMOUS FTP LOGIN FROM .*
```

Eine Möglichkeit, das Datenvolumen zu verringern, besteht darin, mit Hilfe dieser regulären Ausdrücke und dem `egrep`-Befehl ein Skript zu schreiben, das aus einer Vielzahl von `grep -v` Kommandos besteht. Die verbleibenden Restdaten müssen dann manuell gesichtet und bewertet werden.

Ein regelmäßiges Ausführen dieser Auswertescripte kann die Menge der gesammelten Daten reduzieren. Leider hat dieses Vorgehen immer noch einige Nachteile:

- Eine Auswertung findet zu festgelegten Zeiten (wöchentlich, monatlich etc.) statt. Damit kann nicht unmittelbar auf ein Ereignis reagiert werden.
- Die Restdaten müssen immer noch manuell bearbeitet werden.

Eine komplette vollautomatische Auswertung wird es nicht geben können. Dazu müßte man alle möglichen Ereignisse kennen und die sich daraus ergebenden Konsequenzen vorher vorgeben. Selbst wenn man dieses für alle möglichen Meldungen schaffen würde, so ergeben sich aus den beliebigen Kombinationen immer neue Konsequenzen, die vorher nicht definierbar sind. Trotzdem ist es sinnvoll, in vielen Situationen automatische Reaktionen anstoßen zu können. Wenn zum Beispiel jemand außerhalb des lokalen Netzwerkes den NIS- oder NFS-Server anspricht, so möchte man unter Umständen mit einem `finger` nachsehen, wer auf der anderen Seite eingeloggt sein könnte. Bei der vorgestellten Methode der Datenreduktion würde der `finger`-Aufruf nur zum Auswertezeitpunkt (z. B. am Wochenende) möglich sein, was kaum Sinn machen würde.

Es wird also nicht nur eine einmalige automatische Auswertung benötigt, sondern eine Möglichkeit, zeitgleich mit dem Eingang der Meldung bestimmte Aktionen auslösen zu können.

Frei verfügbare Software-Pakete, die eine automatische online Auswertung der Logdaten übernehmen, sind z. B. *Swatch* oder *Logsurfer*. Auf diese Programme im Detail einzugehen würde an dieser Stelle allerdings zu weit führen. Entsprechende Informationen zu diesen Tools sind z. B. vom DFN-CERT Webserver auf der *Logsurfer Homepage*[19] verfügbar.

[19] `http://www.cert.dfn.de/eng/logsurf/`

Anhang D

Windows NT 4 Systemsicherheit

D.1 Sichere Installation von Windows NT Systemen

Die sichere Installation eines Betriebssystems kann in mehrere Schritte aufgeteilt werden. Diese Schritte werden im Folgenden dargestellt und weiter erklärt.

D.1.1 Planung der Installation

Der erste Schritt zu einem sicheren System beginnt schon vor der Installation. Bereits hier müssen wichtige Entscheidungen zur Konfiguration des zukünftigen Systems getroffen werden. Diese Entscheidungen beinhalten:

- **Auswahl der verwendeten Hardware**
  Da Windows NT meist auf einem preislich günstigen PC-System eingesetzt wird, auf dem auch andere Betriebssysteme funktionsfähig sind, muß insbesondere daran gedacht werden, daß ein vorhandenes Floppy- oder CD-ROM-Laufwerk die Gefahr erhöht, daß von diesem Medium ein Betriebssystem gestartet wird. Durch Einstellungen im sog. CMOS-Setup kann zwar verhindert werden, daß diese Geräte als bootfähig erkannt werden, diese Einstellungen können aber umgangen werden. Dies ist insbesondere gefährlich, da die meisten PC-Mainboards zwar ein Paßwort benutzen, um das CMOS-Setup gegen unberechtigten Zugriff zu schützen, das CMOS selbst aber, durch das Setzen eines Jumpers gelöscht werden kann.
  Ist ein PC mit einer Floppy zu starten, so kann ein anderes Betriebssystem, z.B. ein UNIX-Derivat (z.B. Linux) oder DOS gestartet werden. Dieses Betriebssystem kann dann, die entsprechenden Treiber vorausgesetzt, auf ansonsten geschützte Bereiche des Betriebssystems, z.B. die Paßwort-Datenbank, zugreifen.

- **Partitionierung der Festplatte**
  Nachdem die Entscheidung über die zu verwendende Hardware getroffen wurde, müssen die Festplatten partitioniert werden. Hier müssen insbesondere Informationen über die zukünftige Nutzung des Systems einfließen, da Windows NT 4.x keine Begrenzung des für die einzelnen Benutzer verbrauchbaren Festplattenplatzes vorsieht. D.h., daß ein Benutzer den von ihm benutzten Platz auf der Festplatte bis zu den physikalischen Grenzen der Partition ausnutzen kann. Werden die Log- und Auditdaten in die gleiche Partition gespeichert, so kann dies durch mangelnden Speicherplatz zu Problemen in der kontinuierlichen Aufzeichnung von Audit-Daten führen.
  So ist es also wichtig, zu wissen, ob der Rechner zukünftig als File-Server oder Workstation verwendet werden soll.

D.1.2 Installation des Betriebssystems

Die Installation von *Windows NT* stellt den Administrator schon vor etwas größere Anforderungen. Da auf dem betrachteten PC-System nur noch *MS-DOS* installiert sein soll, ist ein Upgrade von einem Windows 3.11 oder 95 System unmöglich. Das bedeutet, daß zunächst einmal 3 Startdisketten angelegt werden müssen, die eine 32-Bit Installationsumgebung für *Windows NT* schaffen.
Ausgehend von dieser 32-Bit-Umgebung kann dann die eigentliche Installation vorgenommen werden. Zunächst wird der Administrator aufgefordert, die Zielpartition für die Installation zu wählen.

Achtung: Soll ein Rechner Domänen-Server werden, ist dieses bereits bei der Installation anzugeben. An späterer Stelle läßt sich diese Einstellung nicht mehr ändern. Ggf. muß das System komplett neu installiert werden.

Windows NT kann auf unterschiedlichen Datei-Systemen eingesetzt werden. Bei der Installation muß sich der Benutzer für ein adäquates Dateisystem entscheiden. Es ist empfehlenswert, das Windows NT eigene Dateisystem **NTFS** zu benutzen, da nur NTFS optionale Zugriffs-Kontrolle und Auditing unterstützt. Nur dadurch kann definiert werden, auf welche Dateien/Verzeichnisse welcher Benutzer zugreifen darf und welche Zugriffe protokolliert werden.

D.1.3 Einspielung aktueller Patches

Die neuesten Patches können direkt von Microsoft[1] bezogen werden. Dabei ist zu beachten, daß es genügt, den Service-Pack mit der jeweils höchsten Nummer zu installieren, da alle übrigen Service-Packs eingeschlossen sind. Nach der Installation des Service-Packs sollte nachgesehen werden, ob sog. Post Hot-Fixes vorhanden sind. Diese Hot-Fixes enthalten die aktuell aufgetretenen Sicherheits-Updates, die noch nicht in einem Service-Pack veröffentlicht wurden.

D.1.4 Anpassung der Systemkonfiguration

Nachdem das Basissystem eingerichtet wurde und die aktuellen Service-Packs installiert sind, müssen noch einige wichtige Anpassungen vorgenommen werden:

- *Ersetzen der Berechtigungen für den Benutzer* J*eder durch den Benutzer* D*omänen-Benutzer*

[1] `http://www.microsoft.com/downloads/`

Der Benutzer *Jeder* ist ein Sicherheitsrisiko, da selbst ein unauthentisierter Netzzugriff die Rechte des Benutzers *Jeder* hat. Daher ist es sinnvoll, alle Rechte von *Jeder* auf *Domänen-Benutzer* umzusetzen. Dies kann am einfachsten mit dem Tool everyone2user[2] durchgeführt werden. Allerdings konnte ein Test im DFN-CERT keine eindeutige Funktion des Programmes feststellen[3].

- *Auswahl der benutzten Services*
  Wie bei allen anderen Betriebssystem-Installationen gilt auch bei Windows NT, daß nur die Dienste installiert werden sollten, die unbedingt benötigt werden.

- *Anpassung der Zugriffsrechte für die Systemdaten und -Verzeichnisse*
  (wird in Kapitel D.8 beschrieben).

- *Anpassung der Systemregistrierungsschlüssel*
  Einige Schlüssel[4] sind in der Windows NT Systemregistrierung direkt nach der Installation entweder noch nicht eingetragen, oder sie haben einen falschen Wert (falsch im Sinne von *nicht den Sicherheitsanforderungen angepaßt*.).
  Außerdem ist es möglich, die einzelnen Schlüssel mit Zugriffsrechten zu versehen. Diese Möglichkeit sollte genutzt werden, um die Schlüssel gegen ungewollte Veränderung zu schützen.
  Wie diese Rechte gesetzt und welche Werte geändert/ eingetragen werden sollten, wird im Folgenden beschrieben:

  1. **Aktivieren einer Anzeige beim Login**
     Es besteht die Möglichkeit, bei Windows NT eine Meldung direkt vor dem Login-Prompt anzuzeigen. Dies kann aus rechtlichen Gründen (z.B. Hinweis, daß die Benutzung dieses Systems nur für autorisierte Personen erlaubt ist) oder zur Information (z.B. bei einem öffentlichen Informationssystem) geschehen.
  2. **Ausschalten des letzten Logins**
     Windows NT zeigt standardmäßig den zuletzt angemeldeten Benutzer beim nächsten Login an. Um diese Anzeige zu unterbinden, kann der in Tabelle D.2 beschriebene Schlüssel eingetragen werden:
  3. **Schutz der Systemregistrierung vor Veränderung über das Netzwerk**
     Direkt nach der Installation ist eingestellt, daß die Systemregistrierung mit

[2] `ftp://ftp.gwdg.de/pub/misc/iss/tools/everyone2user.exe`

[3] Microsoft empfiehlt außerdem, den *Administrator*-Account umzubenennen, damit ein etwaiger Angreifer neben dem Paßwort auch noch den Benutzer-Namen für den System-Verwalter raten muß. Diese Maßnahme bringt aber keine wirkliche Sicherheit, da sie nur den Account verschleiert und mit einfachen Mitteln umgangen werden kann.

[4] Da der Begriff Systemregistrierungsschlüssel in der Literatur verwendet wird, soll auch hier von einem Schlüssel gesprochen werden. Der Begriff Schlüssel bezieht sich auf ein Element einer Datenbank und ist nicht mit dem bei der Verschlüsselung verwendeten Begriff zu verwechseln.

Key-Name	HKEY_LOCAL_MACHINE
Key	Software\Microsoft\Windows NT\Current Version\Winlogon
Name	LegalNoticeCaption
Typ	REG_SZ
Wert	*Hier den Text für die Überschrift der Dialog-Box eintragen*

Key-Name	HKEY_LOCAL_MACHINE
Key	Software\Microsoft\Windows NT\Current Version\Winlogon
Name	LegalNoticeText
Typ	REG_SZ
Wert	*Hier den Text der Dialog-Box eintragen*

Tabelle D.1: Anzeige einer Nachricht beim Einloggen

Key-Name	HKEY_LOCAL_MACHINE
Key	Software\Microsoft\Windows NT\Current Version\Winlogon
Name	DontDisplayLastUserName
Typ	REG_SZ
Wert	1

Tabelle D.2: Anzeige des letzten Benutzers ausschalten

dem Programm `regedt32` auch über das Netzwerk verändert werden kann. Um zu überprüfen, welche Benutzer die Systemregistrierung verändern dürfen, sollten die **Berechtigungen** des folgenden Schlüssels überprüft werden. Dazu wird der unten aufgeführte Schlüssel angezeigt und im Menü `Security/ Permissions` die entsprechenden Rechte entfernt (vgl. Tabelle D.3).

Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Control\SecurePipeServers
Name	\winreg

Tabelle D.3: Berechtigungen prüfen

4. **Verhinderung, daß Benutzer-Namen, Gruppen und freigegebene Ressourcen bekannt gegeben werden.**
   Standardmäßig dürfen anonyme Verbindungen Informationen über die eingetragenen Benutzer, die verfügbaren Gruppen und die freigegebenen Ressourcen abfragen. Dies kann ab Service Pack 3 mit dem in Tabelle D.4 dargestellten Schlüssel unterbunden werden. Wichtig wird dieses, wenn nicht bekannt werden soll, welche Benutzer in welche Gruppen eingetragen sind (Schutz vor Bekanntwerden des Administratoraccounts)

Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Control\Lsa
Name	RestrictAnonymous
Typ	REG_DWORD
Wert	1

Tabelle D.4: Anonyme Abfrage

5. **Schutz der Systemregistrierung**
   In der Systemregistrierung werden die unterschiedlichsten Informationen gespeichert. Insbesondere speichern viele installierte Programme ihre Daten (wie Arbeitsverzeichnis, Liste der zuletzt geöffneten Dateien etc.) in der Systemregistrierung. Diese Daten müssen den Programmen zugänglich sein. Ein Lese-Schutz ist daher nicht möglich. Auch müssen viele der Werte änderbar sein (z.B. die Liste der zuletzt geöffneten Dateien), so daß es zumeist nicht möglich ist, die komplette Systemregistrierung mit einem Schreibschutz zu versehen. Unten ist eine Liste der Schlüssel angegeben, bei denen die folgenden Änderungen durchgeführt werden sollten (**Achtung!** Die Änderungen sollten, sofern nicht anders beschrieben, nur für den jeweils betrachteten Schlüssel durchgeführt werden und nicht für den gesamten Teil-Baum):

   1. Benutzer: *Jeder* — alle Rechte löschen
   2. Benutzer: *Domänen-Benutzer* — Query Value, Enumerate Subkeys, Notify, Read Control

   – Im Registrierungsbaum **HKEY_LOCAL_MACHINE on Local Machine** sind unterschiedliche Einstellungen des Windows-Kern-Systems eingetragen. Alle folgenden Schlüssel sollten geändert werden. Diese Liste kann jedoch nicht alle Schlüssel umfassen, da häufig zusätzlich installierte Programme in diesen Bereich Werte eintragen und an dieser Stelle nicht bestimmt werden kann, welche Programme beim Benutzer eingesetzt werden. Es sollte am Besten nach jeder Installation geprüft werden, welche Einträge neu in die Registry (besonders im Bereich Software) eingetragen wurden und dann geprüft werden, ob diese Schlüssel durch den Benutzer *Jeder* verändert werden können. Ist das der Fall, so sollte getestet werden, ob die Rechte entsprechend der oberen Tabelle geändert werden können.
     * \Software [5]
     * \Software\Microsoft\Windows NT\CurrentVersion

[5] Verhindert, daß normale Benutzer weitere Software installieren können.

- \Software\Microsoft\Windows NT\CurrentVersion\Profile List
- \Software\Microsoft\Windows NT\CurrentVersion\AeDebug
- \Software\Microsoft\Windows NT\CurrentVersion\Compatibility
- \Software\Microsoft\Windows NT\CurrentVersion\Drivers
- \Software\Microsoft\Windows NT\CurrentVersion\Embedding
- \Software\Microsoft\Windows NT\CurrentVersion\Fonts
- \Software\Microsoft\Windows NT\CurrentVersion\FontSubstitutes
- \Software\Microsoft\Windows NT\CurrentVersion\Font Drivers
- \Software\Microsoft\Windows NT\CurrentVersion\Font Mapper
- \Software\Microsoft\Windows NT\CurrentVersion\Font Cache
- \Software\Microsoft\Windows NT\CurrentVersion\GRE_Initialize
- \Software\Microsoft\Windows NT\CurrentVersion\MCI
- \Software\Microsoft\Windows NT\CurrentVersion\MCI Extensions
- \Software\Microsoft\Windows NT\CurrentVersion\PerfLib
- \Software\Microsoft\Windows NT\CurrentVersion\Ports
  inklusive aller untergeordneten Schlüssel!
- \Software\Microsoft\Windows NT\CurrentVersion\Type 1 Installer
- \Software\Microsoft\Windows NT\CurrentVersion\WOW
  inklusive aller untergeordneten Schlüssel!
- \Software\Windows3.1MigrationStatus
  inklusive aller untergeordneten Schlüssel!
- \System\CurrentControlSet\Services\LanmanServer\Shares
- \System\CurrentControlSet\Services\UPS

– Im Registrierungsbaum **HKEY_CLASSES_ROOT on Local Machine** werden die Assoziationen von Dateien zu den Programmen definiert, insbesondere auch, welche Programme standardmäßig geladen werden, wenn eine Datei gestartet wird (entweder durch Doppel-Klick oder `START <Dateiname>` in der Kommandozeile). Diese Informationen sollten vor der Veränderung durch normale Benutzer geschützt werden. Daher müssen die Rechte des Schlüssels \HKEY_CLASSES_ROOT und alle untergeordneten Schlüssel geändert werden.

– Im Registrierungsbaum **HKEY_USERS on Local Machine** werden Benutzer-Einstellungen, einschließlich dem aktuellen Benutzer und dem Standard-Benutzer, gespeichert. Die Standardeinstellungen im Schlüssel \.DEFAULT sollten gegen Veränderung geschützt werden.

6. **Anpassung der erlaubten Aktionen**
   Einige Aktionen, wie das Einloggen auf der lokalen Maschine, sind bei der Installation noch nicht optimal gesetzt, da Zugriffe von Benutzern erlaubt sind, die für die tägliche Arbeit dieser Benutzer nicht notwendig sind. In

Tabelle D.5 werden die durchzuführenden Änderungen beschrieben. Dabei zeigt die erste Spalte die betreffende Aktion, die Spalten 2 und 4 die standardmäßigen Rechte auf einem Einzelplatzrechner bzw. einem Domänenkontroller und die Spalten 3 und 5 die durchzuführenden Änderungen. Diese Änderungen können mit dem Benutzer-Manager (im Menüpunkt `Benutzer\Benutzerrechte`) durchgeführt werden.

Aktion	**Standardmäßig installiert für Workstation** & **Einzelplatz-rechner**	**Änderung**	**Standardmäßig installiert für Domänenkontroller**	**Änderung**
Lokal einloggen	Administratoren, Jeder, Gast, Benutzer, Power-Benutzer	Jeder und Gast entfernen	Account Operateure, Administratoren, Backup Operateure, Server Operateure, Druck Operateure	keine
System herunter-fahren	Administratoren, Jeder, Gast, Benutzer, Power-Benutzer	Jeder und Gast entfernen	Account Operateure, Administratoren, Backup Operateure, Server Operateure, Druck Operateure	keine
Zugang auf diesen Rechner vom Netz	Administratoren, Jeder, Gast, Power-Benutzer	Jeder und Gast entfernen, Benutzer hinzufügen	Administratoren, Jeder	Administratoren, Backup Operateure, Server Operateure, Druck Operateure, Benutzer, evtl. Gast hinzufügen

Tabelle D.5: erlaubte Aktionen anpassen

7. **Schutz der Log-Daten vor Zugriff von unberechtigten Accounts**
   In der Standard-Installation dürfen Gast-Benutzer auf die System- und Applikations-Logdaten lesend zugreifen. Mit den in Tabelle D.6 und Tabelle D.7 beschriebenen Einstellungen kann dieser Zugriff verhindert werden.

8. **Einschalten der Nachrichtensignierung im Smb-Protokoll**
   Mit Service Pack 3 wurde die Signierung von Nachrichten im Smb-Protokoll implementiert. Das Smb-Protokoll wird von Windows NT benutzt, um basierend auf einer TCP-Verbindung, Daten auszutauschen. Dadurch werden Man-In-The-Middle-Attacken verhindert. Aber es ist möglich, daß ältere

Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\EventLog\Application
Name	RestrictGuestAccount
Typ	REG_DWORD
Wert	1

Tabelle D.6: Gast-Zugriff für Application-Log sperren

Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\Eventlog\System
Name	RestrictGuestAccount
Typ	REG_DWORD
Wert	1

Tabelle D.7: Gast-Zugriff für System-Log sperren

Klienten[6] das Signieren von Nachrichten noch nicht implementiert haben und nicht mehr mit Rechnern kommunizieren können, die die Signierung von Nachrichten fordern. Um sicherzustellen, daß der Server nur noch mit Klienten Kontakt aufnimmt, die das Signieren unterstützen, muß folgender Schlüssel eingefügt werden:

Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\LanManServer\Parameters
Name	RequireSecuritySignature
Typ	REG_DWORD
Wert	1

Tabelle D.8: Signierung von Nachrichten am Server

Entsprechend wird für Klienten, die sich nur noch mit Servern verbinden sollen, die das Signieren beherrschen, eingetragen:

Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\Rdr\Parameters
Name	RequireSecuritySignature
Typ	REG_DWORD
Wert	1

Tabelle D.9: Signierung von Nachrichten am Klienten

[6] z.B. Windows 95, Windows 3.11 und OS/2

9. **Herunterfahren des Rechners ohne Einloggen**
   Sind Reset- und Power-Schalter am Computer nicht zugänglich, so kann es zweckmäßig sein, die Möglichkeit, einen Rechner herunterzufahren, auf eingeloggte Benutzer zu beschränken. Damit kann durch die Log-Dateien ausgewertet werden, wer wann welchen Rechner ausgeschaltet hat. In Tabelle D.10 ist der Schlüssel angegeben, der dafür eingetragen werden muß.

Key-Name	HKEY_LOCAL_MACHINE
Key	Software\Microsoft\Windows NT\Current Version\Winlogon
Name	ShutdownWithoutLogon
Typ	REG_SZ
Wert	0

Tabelle D.10: Herunterfahren des Rechners nur nach Einloggen

10. **Hinzufügen von Netzwerk-Druckern verhindern**
    Durch das Setzen des Wertes 1 im in Tabelle D.11 beschriebenen Schlüssel können nur noch die Benutzer *Administrator* und *Printer-Operator* auf dem Server einen neuen Drucker hinzufügen. Auf der Workstation dürfen nur noch *Power-User* Drucker hinzufügen. Das ist sinnvoll, da durch Einfügen eines neuen Drucker-Devices z.B. Daten abgehört werden können[7].

Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Control\Print\Providers_ \LanMan Print Services\Servers
Name	AddPrintDrivers
Typ	REG_DWORD
Wert	1

Tabelle D.11: Hinzufügen von Druckern sperren

11. **Sicheres Löschen der Auslagerungsdatei**
    Die Auslagerungsdatei wird während einer Sitzung benutzt, um Hauptspeicher auf die Festplatte auszulagern und so mehr physikalischen Speicher zu simulieren als tatsächlich vorhanden ist. Wird die Auslagerungsdatei nicht beim Abschalten des Computers gelöscht, so bleiben sensitive Daten in der Auslagerungsdatei gespeichert, die nach dem Starten des Computer mit einem anderen Betriebssystem gelesen werden können. Mit dem Eintrag aus

[7] z.B. indem ein neu eingefügter Druckertreiber die Daten nicht nur direkt an einen Drucker sendet, sondern auch in eine Datei schreibt.

Tabelle D.12 kann sichergestellt werden, daß die Auslagerungsdatei nach dem ordnungsgemäßen Herunterfahren des Systems komplett gelöscht ist. Und zwar so, daß auf die Daten nicht mehr zugegriffen werden kann, d.h. die Datei wird mit zufälligen Daten gefüllt.

Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Control\SessionManager\MemoryManagement
Name	ClearPageFileAtShutdown
Typ	REG_DWORD
Wert	1

Tabelle D.12: Auslagerungsdatei beim Herunterfahren löschen

D.1.5 Maßnahmen zur sicheren Einrichtung von Windows 95 / 3.x auf einem PC:

Windows 95 und Windows 3.x bieten keine Schutz-Möglichkeiten, die bei oder nach der Installation benutzt werden können. Es besteht allerdings die Möglichkeit, bei Microsoft[8] einen Service-Pack für Windows 95 zu bekommen, der einige Updates von System-Programmen enthält und das Desktoppaßwort aufwendiger verschlüsselt als unter Windows 95 ohne Service-Packs. Dieser Service-Pack sollte installiert werden, da Windows 95 die Paßworte für weitere Netzwerkressourcen[9] in der Datei `<BENUTZERNAME>.pwl` speichert. Die Kodierung, die dabei verwendet wird, wird durch die Verwendung des Service-Packs um den Faktor 2^{96} schwerer zu decodieren.[10] Wird dies nicht durchgeführt, so können mit wenig Aufwand und Kenntnis der entsprechenden `*.pwl` Datei die Paßworte anderer Netzwerkressourcen bestimmt werden, auf die ein Benutzer Zugriff gehabt hat.

[8] `http://www.microsoft.com/downloads/`

[9] z.B. Kennwörter für Mailboxen, oder speziell freigegebene Netzwerklaufwerke

[10] Nach den etwas schwammigen Aussagen von Microsoft.

D.2 Zugangskontrolle über Paßworte

Ein Zugang zu einem Computersystem ist zumeist an einen sog. Account gebunden. An diesen Account sind die Rechte eines Benutzers (z.B. Plattennutzung, Prozessorzeit, Zugriffsrechte) gebunden. Um bessere Strukturierungsmöglichkeiten zu erhalten, werden neben den Accounts auch noch sog. Gruppen benutzt. Durch Gruppen ist es möglich, einer Menge von Benutzern, nämlich den Mitgliedern der Gruppe, Rechte zuzuweisen. Die Rechte eines Benutzers ergeben sich aus der Addition der Accountrechte und der Rechte aller Gruppen, in denen der Benutzer Mitglied ist.
Damit nicht jeder Benutzer einen beliebigen Account verwenden kann, sind die Accounts durch ein Paßwort geschützt. Die Paßworte bilden damit die Grundlage der Authentisierung von Benutzern gegenüber Computersystemen.

Bei der Verwendung von Paßworten zur Authentisierung können — bedingt durch die Art der Implementierung — gewisse Sicherheitsprobleme auftreten (vgl. 2.1)

D.2.1 Wahl der Paßworte

Neben der Speicherung ist die Wahl eines „genügend sicheren“ Paßwortes wichtig. Da dem Benutzer eines Rechnersystems bei der Wahl seines Paßwortes große Verantwortung übertragen wird, ist eine umfassende Aufklärung der Benutzer zu diesem Thema wichtig.
Bei der Auswahl eines Paßwortes sollte darauf geachtet werden, daß es sich dabei nicht um ein Paßwort handelt, das leicht zu erraten ist. Problematisch sind alle Paßworte, die von einem Angreifer ausprobiert werden, z.B.:

- Worte aus dem Sprachschatz (div. Wörterbücher)
- Worte aus anderen Sprachen (ebenfalls div. Wörterbücher)
- alle Arten von Namen (Personen, Städte, Gebäude, Comic-Figuren, ...)
- Rechnernamen, Benutzerkennungen
- Geburtsdaten, Telefonnummern
- Abkürzungen
- Tastaturfolgen (z.B. „qwerty“ oder „asdfgh“)
- Anfangsbuchstaben von bekannten Sprichworten, Liedern, etc.
  (z.B. amesads = „alle meine entchen schwimmen auf dem see“,
  wrssdnuw = „wer reitet so spät durch nacht und wind“, fdhdgg, ...)

Ebenso unbrauchbar sind Modifikationen dieser Worte durch z.B.:

- Anhängen oder Voranstellen einer Zahl (peter09, 7peter, ...)
- Rückwärtsschreibung (retep, reteP, ...)
- Anhängen oder Voranstellen eines beliebigen Zeichens (&peter, ∗peter, ...)
- usw.

Obwohl die Aufklärung der Benutzer an erster Stelle stehen sollte, kann es sinnvoll sein, die Auswahl der möglichen Paßworte durch weitere technische Maßnahmen einzuschränken.

D.2.2 Aging

Wird ein Paßwort längere Zeit verwendet, so steigt die Wahrscheinlichkeit, daß das Paßwort (oder Teile davon) anderen Benutzern bekannt wird (z.B. durch ein „über die Schulter gucken“ bei der Eingabe des Paßwortes).

Die Notwendigkeit der gelegentlichen Paßwort-Änderung sollte dem Benutzer deutlich gemacht werden. Moderne Betriebssysteme bieten die Möglichkeit den Benutzer nach einer vorgebbaren Zeit automatisch zu der Wahl eines neuen Paßwortes aufzufordern. Diese Möglichkeit, die Gültigkeit eines Paßwortes auf eine bestimmte Zeit einzuschränken, wird **aging** genannt. Dem Administrator sollte allerdings bewußt sein, daß dieser Mechanismus nur eine Aufforderung zum Paßwortwechsel darstellt. Bei Windows NT-Systemen kann eine Liste der maximal 9 letzten verwendeten Paßworte gespeichert werden, so daß zumindest 9 verschiedene Paßworte verwendet werden müssen, bis das System ein bereits benutztes Paßwort akzeptiert.
Ein wichtiger Schritt bei der Aktivierung des Aging-Verfahrens ist also auch die Aufklärung der Benutzer.

Ein weiteres Problem stellt die Wahl der maximalen Gültigkeitsdauer eines Paßwortes dar. Während zu kurze Intervalle dazu führen können, daß sich die Benutzer ein Schema zur Generierung neuer Paßworte überlegen (etwa die Verwendung der Monatsnummer bei einem monatlichen Wechsel), können zu lange Intervalle den Nutzen des Aging nachhaltig beeinflussen. In diesem Zusammenhang hat es sich als zweckmäßig erwiesen, Paßworte mindestens jedes halbe Jahr zu wechseln.

D.2.3 Speicherung der Paßworte

Bei Windows NT-Systemen wird ein verschlüsselter Wert des Benutzerpaßwortes (siehe Kapitel D.2.4) in einem besonders geschützten Bereich der Registry-Datenbank auf

dem Primären-Domänenkontroller, auf allen vorhandenen Backup-Domänenkontrollern und Sicherheitsdisketten abgelegt. Auf den Paßwort-Bereich der Registry-Datenbank kann nur der System-Prozeß zugreifen. Also kann auch kein Administrator diesen Bereich direkt lesen.
Angriffe auf die Paßwortdatenbank eines NT-Systems werden immer in zwei Schritten durchgeführt:

1. Die verschlüsselten Paßwortdaten müssen gewonnen werden.
2. Ein Paßwort-Cracker wird auf diese Daten angesetzt. Dieser Paßwort-Cracker versucht dann, ähnlich dem Programm *Crack* (vgl. A.5) für UNIX anhand einer Wortliste oder durch systematisches Testen alle möglichen Buchstabenkombinationen die Paßworte zu bestimmen.

Die Paßwortdaten können auf vier Arten gewonnen werden:

- Angriff als Administrator
  Bei diesem Angriff, muß bereits ein Administrator-Zugang vorhanden sein (z.B. aufgrund eines schwachen Administrator-Paßwortes).
  Um die geschützten Daten der Registry lesen zu können, wird zunächst der Schedule-Service gestartet. Dies ist nur einem Administrator erlaubt. Alle Prozesse, die durch den Schedule-Dienst ausgeführt werden, werden mit der Berechtigung des Systemprozesses ausgeführt. Nachdem der Schedule-Dienst gestartet ist, wird durch den Scheduler der Registry-Editors gestartet. Das hat den Effekt, daß der Registry-Editor mit den Rechten des System-Prozesses gestartet wird. Damit können dann auch die Bereiche der Registry, die für den Administrator normalerweise nicht zugänglich sind, bearbeitet werden. Aus diesen Bereichen können dann die verschlüsselten Paßwortinformationen extrahiert und zur Entschlüsselung in ein externes Programm importiert werden.
- Angriff auf ein Backup-Medium
  Auf Notfalldisketten, die mit einem Windows NT Systemprogramm erzeugt werden können, sind wichtige Systeminformationen gespeichert, mit denen ein Windows NT-System nach einem Absturz wiederhergestellt werden kann. Neben Informationen über die Hardware werden auch die Paßwortinformationen der Benutzer auf dieser Diskette gespeichert. Diese Informationen sind nicht weiter geschützt und können in jedem Computer angezeigt und mit einem Paßwort-Cracker ausgewertet werden. Daher ist es besonders wichtig, diese Notfalldisketten besonders zu schützen.
- Angriff per Sniffer im Netzwerk
  Während der Authentisierung eines Benutzers am Server werden die Paßwortinformationen über das Netzwerk übertragen. Diese Informationen können von

einem Netzwerksniffer abgehört und für einen Paßwort-Cracker verwendet werden. Um einen Netzwerksniffer unter Windows NT installieren zu können, werden Rechte als Administrator (des lokalen Rechners) benötigt. Diese können jedoch zumeist leichter erreicht werden als Administratorrechte auf einem Server.

- Angriff auf die Registry, wenn das System nicht gestartet ist.
  Die Registry eines Windows NT-Rechners ist, wie bei den meisten anderen Systemen auch, nur zur Laufzeit des Windows NT Betriebssystems geschützt. Wird der Rechner mit einem anderen Betriebssystem gestartet, so können die Daten der Registry mit einem Programm, daß NT-Partitionen auslesen kann, gelesen und mit einem Paßwort-Cracker verarbeitet werden.

D.2.4 Details zum Paßwortalgorithmus

Bei der Anmeldung am System wird das vom Benutzer eingegebene Paßwort mit dem Paßwort aus der lokalen Paßwortdatenbank verglichen. Die einfachste Möglichkeit wäre nun, diese Paßworte in einer Datenbank zu speichern und direkt mit der Eingabe des Benutzers zu vergleichen. Dies ist aber wenig zweckmäßig, wenn ein Benutzer eben diese Datenbank mit den Paßworten im Klartext lesen kann.
Der Paßwortalgorithmus beschreibt das Verfahren, mit dem verhindert wird, daß aus dem in der Datenbank gespeicherten Eintrag auf das Paßwort zurückgeschlossen werden kann.

Paßworte werden in der Standardinstallation von Windows NT in zwei Arten gespeichert:

1. LAN-Manager (NT-LM)
   Beim NT-LM Verfahren wird das bekannte DES-Verfahren benutzt, um die Paßworte zu verschlüsseln. Dabei wird das maximal 14 Zeichen lange Paßwort in zwei 7 Zeichen lange Blöcke gespalten. Diese beiden Blöcke werden dann jeweils als Schlüssel für das DES-Verfahren benutzt, um einen Block von 0-en zu verschlüsseln. Das Ergebnis dieser beiden Verschlüsselungen wird dann in der Registry gespeichert.
   Das NT-LM Paßwort verwendet 14 Zeichen aus der Menge {A..Z, 1..9, [einige Sonderzeichen]}.

2. NT-MD4
   Beim NT-MD4-Verfahren wird aus dem Paßwort mittels dem Verfahren MD4 ein Hashwert gebildet. Ausgehend von diesem Hashwert kann nicht mehr aus auf ursprüngliche Paßwort geschlossen werden.
   Theoretisch kann das NT-MD4 Paßwort 250 Zeichen lang sein. Es können aber

aufgrund einer Restriktion im Eingabefeld des Paßwortes nur 14 Zeichen aus der Menge {A..Z, a..z, 1..9, [einige Sonderzeichen]} verwendet werden.

Die Ergebnisse dieser beiden Verfahren werden dann zu einem Wert zusammengefaßt und mit einer reversiblen Funktion verschleiert.

Angriffe auf Windows NT Paßworte werden zumeist auf das NT-LM Paßwort durchgeführt. Dieses ist effektiver, da die Menge der verwendeten Zeichen kleiner ist und außerdem nur zwei 7 Zeichen lange Paßworte und nicht ein 14 Zeichen langes Paßwort getestet werden müssen.
Durch Setzen eines Wertes in der Registry kann verhindert werden, daß die NT-LM Paßworte benutzt werden (vgl. D.6). Damit kann die Sicherheit bei einem Angriff auf die Paßwortdatenbank erhöht werden. Jedoch können sich einige Klienten, die das neue NT-MD4 Verfahren nicht beherrschen, nicht mehr im Netzwerk anmelden.

D.2.5 Länge der Paßworte

Wie oben beschrieben werden die Paßworte nicht im Klartext gespeichert. Ist der Verschlüsselungsalgorithmus für die Paßworte hinreichend gut, so kann ein Paßwort aus einem gegebenen verschlüsselten Paßwort nur bestimmt werden, indem das Verschlüsselungsverfahren auf ein mögliches Paßwort angewendet wird. Ist das Ergebnis dieses Verschlüsselungsverfahrens das gesuchte verschlüsselte Paßwort, so war das gewählte mögliche Paßwort korrekt.
Angriffe auf Paßwortdatenbanken (mit Einträgen von verschlüsselten Paßworten) werden häufig über zwei verschiedene Arten vorgenommen:

- Wörterbuchattacken
  Hierbei wird ein Wörterbuch mit wahrscheinlichen Worten benutzt und davon ausgehend versucht, das Paßwort zu finden.

- Brute-Force-Attacken
  Hierbei werden alle möglichen Kombinationen über einem gegebenen Alphabet erzeugt und probiert.

Es ist offensichtlich, daß ein Wörterbuchangriff für eine gegebene Paßwortlänge schneller ist als ein Brute-Force-Angriff. Wobei der letztere aber auch sämtliche möglichen Paßworte findet.
Um sich vor einem Brute-Force-Angriff zu schützen, müssen die Benutzer davon überzeugt werden, ein möglichst langes Paßwort zu benutzen.
Eine Sun Ultra mit 170 MHz kann beispielsweise pro Sekunde ca. 10.000 Paßworte

verschlüsseln. Daraus ergibt sich der in Tabelle D.13 folgende Aufwand[11] :

Länge der Paßworte	Kombinationen	Zeit
1	96	< 1 Sek
2	9.216	$\approx 1{,}0$ Sek
3	884.736	$\approx 89{,}0$ Sek
4	84.934.656	$\approx 2{,}3$ Std
5	8.153.726.976	$\approx 9{,}4$ Tage
6	$7{,}8276*10^{11}$	$\approx 2{,}5$ Jahre
7	$7{,}5145*10^{13}$	$\approx 238{,}1$ Jahre
8	$7{,}2139*10^{15}$	$\approx 22.859{,}5$ Jahre
14	$5{,}6473*10^{27}$	$\approx 1{,}70*10^{16}$ Jahre

Tabelle D.13: Zeitabschätzung für einen Brute-Force-Angriff mit einer Sun Ultra

Ein Paßwort sollte daher mindestens 7 Zeichen lang sein und den möglichen Zeichenvorrat nutzen. Es sollte möglichst immer aus einer Kombination von Klein-, Groß-Buchstaben, Ziffern und Sonderzeichen bestehen.

D.2.6 Erkennung schwacher Paßworte

D.2.6.1 Nachträgliche Erkennung ungeeigneter Paßworte

Da Paßworte den Zugang zu einem Rechnersystem sichern, sind schwache, also leicht zu ratende Paßworte, die erste Schwachstelle, die ein Angreifer testen wird. Bei einem bestehenden System sollte in regelmäßigen Abständen überprüft werden, ob die Benutzer schwache Paßworte verwendet haben.

Relativ bekannt ist die Methode, eine gegebene Paßwort-Datei mit Hilfe eines Programmes zum Ausprobieren von Paßworten zu überprüfen. Das bekannteste Programm für Windows NT ist „L0phtcrack“[12]

Diese Methode hat diverse Nachteile. Da nur die verschlüsselten Paßwort-Einträge zur Verfügung stehen, muß jedes zu testende Wort ebenfalls verschlüsselt und anschließend mit dem gespeicherten Eintrag verglichen werden. Dieses erfordert ein großes

[11] Zu beachten ist, daß diese Abschätzung für einen Rechner gilt. Werden Brute-Force-Angriffe durchgeführt, so werden meist mehrere Rechner parallel benutzt und die zu berechnenden Kombinationen aufgeteilt.

[12] `ftp://ftp.cert.dfn.de/pub/tools/password/l0pht/`

Maß an Rechenzeit. Ein weiteres Problem dieses Verfahrens ist die Zeitspanne zwischen der Wahl des Paßwortes und der Erkennung als „schwaches“ Paßwort.

D.2.6.2 Erkennung „schwacher“ Paßworte bei der Wahl

Eine effizientere Methode schwache Paßworte zu finden ist, direkt bei der Eingabe zu prüfen, ob die Paßworte bestimmten Regeln genügen. Damit kann eine langwierige Suche, wie sie bei der nachträglichen Erkennung nötig ist, umgangen werden.

Unter Windows NT ist diese Möglichkeit ab Service Pack 3 mit der Bibliothek `passfilt.dll` realisiert. Diese Bibliothek realisiert eine Funktion, die vor der Änderung eines Paßwortes überprüft:

- ist das Paßwort mindestens 6 Zeichen lang?
- enthält das Paßwort den Benutzernamen, den realen Namen des Benutzers?
- enthält das Paßwort mindestens Zeichen aus drei von vier möglichen Kategorien :
  1. {A..Z}
  2. {a..z}
  3. {1..9}
  4. {Sonderzeichen wie !,?}

Vorteilhaft an diesen Methoden ist, daß die Überprüfung des Paßwortes anhand des Klartext-Paßwortes sehr effizient und zügig erfolgen kann.

D.2.6.3 Alternative Authentisierungsverfahren

Wie schon oben beschrieben erfordert das Anmelden an einem System üblicherweise die Angabe eines geheimen Paßwortes, welches am Zielrechner verschlüsselt und mit den Einträgen in der Paßwort-Datei verglichen wird. Problematisch ist dieses Verfahren beim Anmelden über ein Netzwerk, da das eingegebene Paßwort unverschlüsselt über das Netz übertragen wird. Angreifer könnten die übertragenen Paßworte abhören und mißbrauchen. Es gibt zwei Möglichkeiten, diese Problem zu lösen:

- Verschlüsselung bei einer Verbindung über das Netzwerk
  Werden die Daten schon vor der Übertragung des Paßwortes verschlüsselt, so können die Daten zwar abgehört werden, es ist jedoch nicht möglich, aus den

erlauschten Daten ein Paßwort zu erkennen.
Eine Verschlüsselung bedingt aber, daß auf dem Zielrechner ein sog. Dämon installiert ist, der einen verschlüsselten Verbindungsaufbau erkennt und entsprechend reagiert.

- Einsatz von Einmal-Paßwortsystemen
  Hierbei wird für jeden Benutzer eine Liste von Paßworten erzeugt, die in einer vorher definierten Reihenfolge zum Anmelden benutzt werden. Jedes Paßwort auf dieser Liste wird genau einmal verwendet. Somit kann ein Angreifer auch nicht mit einem abgehörten Paßwort einen Zugang zum System erhalten.

Programmpakete wie *S/Key* der Firma Bellcore unterstützen die Benutzung von Einmal-Paßworten. Durch diese Programmpakete ist gewährleistet, daß das geheime Paßwort des Benutzers niemals über das Netzwerk übertragen wird.

Das S/Key - System schützt den Benutzer vor dem unerwünschten Abhören seines geheimen Paßwortes. Beim Anmelden an einem entfernten Rechner (z.B. über rlogin, telnet, ftp, etc.) wird statt des geheimen Paßwortes ein Einmal-Paßwort zur Authentisierung benutzt. Dieses Paßwort wird nie wieder verwendet, so daß das potentielle Abhören des Paßwortes durch Dritte keinen Mißbrauch ermöglicht.

Das S/Key - System besteht aus zwei Seiten: Der Benutzer (Klient) muß das Einmal-Paßwort erzeugen; das entfernte System (Server) muß dieses Paßwort verifizieren.

Die Generierung des Paßwortes erfolgt am lokalen System. Ein Erzeugerprogramm erwartet als Eingabe ein geheimes Paßwort und liefert ein oder mehrere Einmal-Paßworte. Nur das Einmal-Paßwort wird anschließend an den entfernten Rechner (den Server) übermittelt. Alternativ ist es dem Benutzer auch möglich, eine bestimmte Zahl von Einmal-Paßworten im Voraus berechnen zu lassen und beispielsweise auf einem Zettel zu verwahren.

D.3 Access Control Listen

Windows NT verwendet Access Control Listen (ACL), um die Zugriffsrechte auf ein Objekt (Datei, Verzeichnis, Drucker, etc.) zu speichern (vgl. D.4.1.1). Einträge in ACLs sind lokale Benutzer, globale Benutzer, lokale Gruppen und globale Gruppen. Lokale Benutzer / Gruppen sind nur auf dem betreffenden Rechner definiert. Globale Benutzer / Gruppen sind in der ganzen Domäne definiert.
Jedes Objekt enthält im sog. Sicherheitsdescriptor eine ACL, die wiederum 3 Access Control Entries (ACE) enthält:

1. Access Denied
   Alle Benutzer / Gruppen, die in dieser ACE eingetragen sind, haben keinen Zugriff auf das Objekt.

2. Access Allowed
   Alle Benutzer / Gruppen haben den in dieser ACL definierten Zugriff. Da ein Benutzer in mehreren Gruppen Mitglied sein kann, erhält der Benutzer die Summe der Zugriffsrechte aller Gruppen.

3. System
   In der System ACE werden die zu überwachenden (auditierenden) Aktionen mit einem Objekt für Benutzer / Gruppen eingetragen.

Bei einem Zugriff auf ein Objekt werden anhand des Zugriffstokens, das für jeden Benutzer beim Login erzeugt wird, die Gruppen festgestellt, in denen ein Benutzer Mitglied ist. Danach wird überprüft, ob der Benutzer (oder eine Gruppe, in der er Mitglied ist, in der Access Denied ACE eingetragen ist. Wenn ja, wird jeglicher Zugriff auf das Objekt verweigert. Wenn nein, wird überprüft, ob der Benutzer (oder eine Gruppe, in der er Mitglied ist) über die Access Allowed ACE ein Zugriffsrecht erhalten hat. Wenn nein, wird der Zugriff auf das Objekt verweigert. Ansonsten wird der Zugriff gewährt.

D.4 Zugriffskontrolle

Nachdem sich ein Benutzer am System durch die Eingabe seines Paßwortes authentisiert hat, muß das Betriebssystem dafür sorgen, daß der Benutzer entsprechend seinen Rechten nur auf bestimmte Bereiche des Filesystems zugreifen und nur eine bestimmte Art und Menge von Systemressourcen benutzen kann (siehe unten).

D.4.1 Zugriffskontrolle auf Objekte

Objekte (Verzeichnisse, Dateien, Drucker) werden häufig über Netzwerkserver einer Gruppe von Benutzern zugänglich gemacht. Insbesondere ist es somit möglich, daß mehrere Benutzer die gleichen Objekte benutzen können. So zum Beispiel Anwendungsprogramme, die auf einem Server im Netz installiert sind und über das Netzwerk gestartet werden. Der Vorteil dieses gemeinsamen Zugriffs besteht darin, daß Speicherplatz gespart und die Verwaltung einfacher gemacht wird.
Nun sollten die Objekte aber nicht ohne Einschränkung für alle Benutzer zugänglich gemacht werden. Ein Schreibrecht auf die installierten Programme ist z.B. in der Regel nicht notwendig. Benutzern sollte daher nur Zugang zu den von ihnen benötigten Programmen und Daten erlaubt werden. Dies kann durchgeführt werden, indem die Zugriffsrechte eines Objektes an einen Benutzer/Gruppen-Account geknüpft werden.

D.4.1.1 Rechte unter Windows NT

Bevor über die Vergabe der Rechte unter Windows NT gesprochen werden kann, sollen die Rechte kurz vorgestellt werden. Sie werden nach Rechten für Verzeichnisse und Rechten für Dateien unterschieden. In Tabelle D.14 findet sich ein Überblick der Rechte, die für Verzeichnisse gesetzt werden können.

Recht	Erlaubt
Read	Lesen der Dateien und Unterverzeichnisse
Write	Hinzufügen von Dateien und Unterverzeichnissen
E**x**ecute	Wechseln in Unterverzeichnisse
Delete	Löschen des Verzeichnisses
Change **P**ermission	Ändern der Rechte des Verzeichnisses
Take **O**wnership	Übernehmen des Besitzes

Tabelle D.14: Rechte für Verzeichnisse

Für die Verzeichnisse lassen sich stets zwei Kategorien von Rechten setzen:

- Rechte für das Verzeichnis
- Rechte für die Dateien in diesem Verzeichnis (sowohl für die existierenden, als auch für die neu erstellten Dateien).

Beim Ändern der Rechte kann dann entsprechend entschieden werden, ob die Änderungen nur für neu erstellte Dateien gelten sollen, oder ob die Rechte bereits existierender Dateien geändert werden sollen. Außerdem gibt es die Möglichkeit, die Rechte bereits existierender Unterverzeichnisse und deren Inhalte im gleichen Schritt zu setzen.

Neben den Rechten für Verzeichnisse gibt es auch Rechte für Dateien. Diese unterscheiden sich in einigen Details von den Rechten für Verzeichnisse. Tabelle D.15 zeigt im Überblick die Rechte für Dateien:

Recht	Erlaubt
Read	Lesen des Dateiinhalts
Write	Ändern des Dateiinhalts
E**x**ecute	Ausführen der Datei
Delete	Löschen der Datei
Change **P**ermission	Ändern der Rechte der Datei
Take **O**wnership	Übernehmen des Besitzes

Tabelle D.15: Rechte für Dateien

Die meisten dieser Rechte entsprechen den Rechten aus UNIX-Umgebungen. Allein das Recht Take **O**wnership wird etwas anders genutzt als unter UNIX, da nicht nur der Administrator das Recht hat, den Besitz einer Datei zu ändern, sondern alle Benutzer / Gruppen, die das entsprechende Recht haben.

Die Sicherheitsattribute der Dateien werden im sog. Sicherheitsdescriptor gespeichert. Dieser enthält die Angaben:

- die *Sicherheits-ID des Benutzers*, die beschreibt, wer der Besitzer eines Objektes ist,
- die *Sicherheits-ID der Gruppe*, die nur vom POSIX-Subsystem benutzt und von Windows NT ignoriert wird,
- die *Zugriffskontrolliste (ACL = Access Control List)* (vgl. D.3)

D.4.1.2 Beschränkung des nutzbaren Platzes im Benutzerverzeichnis

Es ist unter Windows NT nicht möglich, dem Benutzer vorzuschreiben, wieviel Platz in den Benutzerverzeichnissen in Anspruch genommen werden darf. Dies ist nur mit Programmen von Drittanbietern definierbar.

D.4.2 Zugriffskontrolle auf Ressourcen

Ähnlich den Daten muß auch die Nutzung der Systemressourcen kontrolliert werden. Sonst könnte z.B. ein Benutzer durch einen übermäßigen Verbrauch von Festplattenplatz auf der Systempartition den ordnungsgemäßen Betrieb verhindern.
Bevor an dieser Stelle auf den Schutz dieser Systemressourcen eingegangen werden kann, muß noch kurz beschrieben werden, was wir an dieser Stelle unter den Systemressourcen verstehen wollen:

- Prozessorzeit
- Hauptspeicher
- Anzahl der Prozesse
- Anzahl der offenen Dateien
- Genutzter Platz auf den Festplatten
- Peripherie

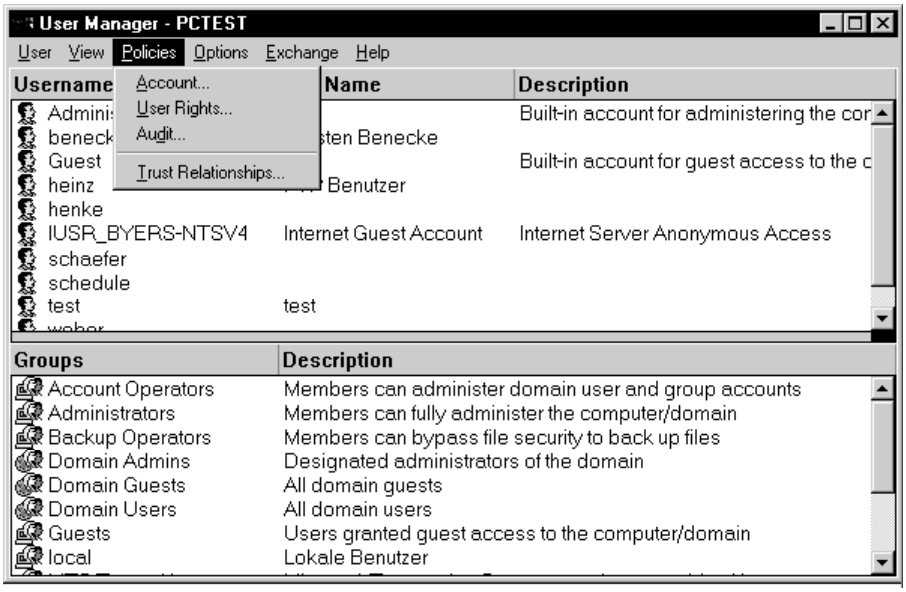
D.4.2.1 Ressourcenkontrolle unter Windows NT

Unter Windows NT gibt es keine Ressourcenkontrolle im Sinne von UNIX. Die einzige, in einem gewissen Sinne kontrollierbare Ressource, ist der Drucker. Ein Drucker kann unter Windows NT an bestimmte Benutzer(-Gruppen) gebunden werden, so daß nur diese Benutzer den Drucker benutzen können. Alle anderen Benutzer können nicht auf diesen Drucker zugreifen.

Eine Beschränkung der Prozessorzeit ist nicht möglich. Zwar kann die Priorität der einzelnen Prozesse interaktiv bestimmt werden, es ist aber nicht möglich zu definieren, daß ein Benutzer z.B. nur maximal 50% der Prozessorzeit benutzen darf. Gleiches gilt auch für die Nutzung des Hauptspeichers.

D.5 Aktivierung von Audit

Um unter Windows NT Audit Funktionalität zu erhalten, muß am Domänenserver über den Benutzermanager für Domänen (bzw. über den Benutzermanager bei Workstations) im Menü Policies Audit aktiviert werden. Dies ist eine notwendige Voraussetzung für alle weiteren Einstellungen.



Das Windows NT Auditprogramm erfaßt, je nach Einstellung, Daten in drei Kategorien:

- **Security-Audit** (Logmeldungen, die die Sicherheit des Systems betreffen)
  Die Auswahl der zu überwachenden Auditereignisse wird im Benutzermanager für Domänen vorgenommen. Im einzelnen sind folgende Ereignisse auditierbar:
  - Logon / Logoff
    An- und Abmelden im Netzwerk, sowie nicht anonyme Netzwerkverbindungen werden protokolliert.
  - Datei- und Objektzugriffe
    Zugriffe auf Dateien, Verzeichnisse und Drucker, die für Audit konfiguriert sind, werden protokolliert. Zugriffe auf Dateien werden jedoch erst protokolliert, wenn die Eigenschaften der betreffenden Dateien im Menü `Audit` entsprechend eingestellt wurden. In den Eigenschaften kann definiert werden, ob Audit für einzelne Benutzer oder ganze Benutzergruppen aktiviert werden soll.
    Beim Datei-Audit sind die folgenden Aktionen protokollierbar (jeweils erfolgreich oder nicht erfolgreich):

    - Lesezugriff
    - Schreibzugriff
    - Ausführen
    - Löschen
    - Ändern der Rechte
    - Übernehmen des Besitzes der Datei /des Verzeichnisses
  - Benutzer und Gruppenverwaltung
    Änderungen von Benutzern und Gruppen (Löschen, Hinzufügen, Namen ändern), sowie das Eintragen und Ändern von Paßworten werden protokolliert.
  - Richtlinienänderung
    Änderungen der Benutzerrechte, im Audit oder den Vertrauensbeziehungen zu anderen Domänen werden protokolliert.
  - Systemereignisse
    Systemstart und Herunterfahren werden protokolliert.
  - Detaillierte Verfolgung
    Es werden detaillierte Informationen zu bestimmten Abläufen, wie dem Start von Programmen (die aus mehreren Prozessen bestehen), und das Ende von Prozessen protokolliert.

- **System-Audit** (Ereignisse beim Starten von systemnahen Komponenten)
  Hier werden Informationen über den Status von geladenen Treibern (für Grafikkarten, Netzwerkkarten, etc), Systemprogrammen und Diensten (DNS, WINS, etc.), die beim Hochfahren des Systems gestartet werden, festgehalten.

- **Applikation-Audit** (Ereignisse von anderen Programmen)
  Hier werden Informationen eingetragen, die von Applikationen geloggt werden. Applikationen unterscheiden sich von Systemprogrammen dadurch, daß sie nach dem Hochfahren des Systems auf Veranlassung des Benutzers gestartet werden. Die Applikationen, die benutzt werden, unterscheiden sich, im Gegensatz zu den Systemprogrammen, von Benutzer zu Benutzer[13].
  Diese Informationen der Applikationsprogramme werden direkt von den Anwendungsprogrammen in das Log-Programm übergeben und können nur in den betreffenden Programmen konfiguriert werden.

[13] betrachtet man den gleichen Rechner mit verschiedenen Benutzern

D.6 Anpassung der Registrykeys

Anzeigen einer Informationsmeldung vor dem Login (Titel)	
Key-Name	HKEY_LOCAL_MACHINE
Key	Software\Microsoft\Windows NT\Current Version\Winlogon
Name	LegalNoticeCaption
Typ	REG_SZ
Wert	*Hier den Text für die Überschrift der Dialog-Box eintragen*

Anzeigen einer Informationsmeldung vor dem Login (Text)	
Key-Name	HKEY_LOCAL_MACHINE
Key	Software\Microsoft\Windows NT\Current Version\Winlogon
Name	LegalNoticeText
Typ	REG_SZ
Wert	*Hier den Text der Dialog-Box eintragen*

Letzten eingeloggten Benutzer nicht anzeigen	
Key-Name	HKEY_LOCAL_MACHINE
Key	Software\Microsoft\Windows NT\Current Version\Winlogon
Name	DontDisplayLastUserName
Typ	REG_SZ
Wert	**1**

Schutz der Registry vor Veränderung über das Netzwerk	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Control\SecurePipeServers
Name	\winreg
Aktion	Berechtigungen prüfen

Benutzernamen nicht an anonyme Netzwerkverbindungen exportieren	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Control\Lsa
Name	RestrictAnonymous
Typ	REG_DWORD
Wert	**1**

Zugriff auf Applikations-Logdaten verhindern	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\Eventlog\Application
Name	RestrictGuestAccount
Typ	REG_DWORD
Wert	1

Zugriff auf System-Logdaten verhindern	
Key	System\CurrentControlSet\Services\Eventlog\System
Name	RestrictGuestAccount
Typ	REG_DWORD
Wert	1

Herunterfahren des Rechners nur durch eingeloggte Benutzer	
Key-Name	HKEY_LOCAL_MACHINE
Key	Software\Microsoft\Windows NT\Current Version\Winlogon
Name	ShutdownWithoutLogon
Typ	REG_SZ
Wert	0

Sicheres Löschen der Auslagerungsdatei	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Control\SessionManager\MemoryManagement
Name	ClearPageFileAtShutdown
Typ	REG_DWORD
Wert	1

Nachrichtensignierung aktivieren (NT-Server)	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\LanManServer\Parameters
Name	EnableSecuritySignature
Typ	REG_DWORD
Wert	1 (enable)
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\LanManServer\Parameters
Name	RequireSecuritySignature
Typ	REG_DWORD
Wert	1 (enable)

Nachrichtensignierung aktivieren (NT-Workstation)	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\Rdr\Parameters
Name	EnableSecuritySignature
Typ	REG_DWORD
Wert	1 (enable)
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\Rdr\Parameters
Name	RequireSecuritySignature
Typ	REG_DWORD
Wert	1 (enable)

Verwendung des Paßwortfilters `passfilt.dll`	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\LSA
Name	NotificationPackages
Typ	REG_SZ
Wert	**passfilt.dll**

Entfernen der administrativen Shares (C$, D$, etc.) - Server	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\LanmanServer\Parameters
Name	AutoShareServer
Typ	REG_DWORD
Wert	0 (aus) / 1 (an)

Entfernen der administrativen Shares (C$, D$, etc.) - Workstation	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\LanmanServer\Parameters
Name	AutoShareWks
Typ	REG_DWORD
Wert	0 (aus) / 1 (an)

Bestimmung des benutzten Authentisierungsverfahrens	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Control\LSA
Name	LMCompatibilityLevel
Typ	REG_DWORD
Wert	0 = NTLanMan und NTMD4 Authentisierung 1 = NTLanMan Authentisierung nur wenn der Server es verlangt 2 = keine NTLanMan Authentisierung

Eintragen des Knotentyps (Windows 95)	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\VxD\MSTCP
Name	NodeType
Typ	REG_DWORD
Wert	8 (= Hybrid-Node)

Eintragen des Knotentyps (Windows NT)	
Key-Name	HKEY_LOCAL_MACHINE
Key	System\CurrentControlSet\Services\VxD\MSTCP
Name	NodeType
Typ	REG_DWORD
Wert	8 (= Hybrid-Node)

D.7 Anpassung der Zugriffsrechte in der Registry

Für die im folgenden aufgeführten Benutzer sollten die unten beschriebenen Änderungen im H_KEY_LOCAL_MACHINE Zweig der Registry durchgeführt werden:

1. Benutzer: *Jeder* — alle Rechte löschen
2. Benutzer: *Domänen-Benutzer* — Query Value, Enumerate Subkeys, Notify, Read Control

- \Software [14]
- \Software\Microsoft\Windows NT\CurrentVersion
- \Software\Microsoft\Windows NT\CurrentVersion\Profile List
- \Software\Microsoft\Windows NT\CurrentVersion\AeDebug
- \Software\Microsoft\Windows NT\CurrentVersion\Compatibility
- \Software\Microsoft\Windows NT\CurrentVersion\Drivers
- \Software\Microsoft\Windows NT\CurrentVersion\Embedding
- \Software\Microsoft\Windows NT\CurrentVersion\Fonts
- \Software\Microsoft\Windows NT\CurrentVersion\FontSubstitutes
- \Software\Microsoft\Windows NT\CurrentVersion\Font Drivers
- \Software\Microsoft\Windows NT\CurrentVersion\Font Mapper
- \Software\Microsoft\Windows NT\CurrentVersion\Font Cache
- \Software\Microsoft\Windows NT\CurrentVersion\GRE_Initialize
- \Software\Microsoft\Windows NT\CurrentVersion\MCI
- \Software\Microsoft\Windows NT\CurrentVersion\MCI Extensions
- \Software\Microsoft\Windows NT\CurrentVersion\PerfLib
- \Software\Microsoft\Windows NT\CurrentVersion\Port

inklusive aller untergeordneten Schlüssel!

[14] Verhindert, daß normale Benutzer weitere Software installieren können

- \Software\Microsoft\Windows NT\CurrentVersion\Type 1 Installer
- \Software\Microsoft\Windows NT\CurrentVersion\WOW

  inklusive aller untergeordneten Schlüssel!
- \Software\Microsoft\Windows NT\CurrentVersion\Windows3.1MigrationStatus

  inklusive aller untergeordneten Schlüssel!
- \System\MicrosoftCurrentControlSet\Services\

  CurrentVersionLanmanagerServer\Shares
- \System\MicrosoftCurrentControlSet\Services\UPS

- Im Registrierungsbaum **HKEY_CLASSES_ROOT on Local Machine** werden die Assoziationen von Dateien zu den Programmen definiert, insbesondere auch, welche Programme standardmäßig geladen werden, wenn eine Datei gestartet wird (entweder durch `START <Dateiname>` in der Kommandozeile oder Doppel-Klick). Diese Informationen sollten vor der Veränderung durch normale Benutzer geschützt werden. Daher müssen die Rechte des Schlüssels \HKEY_CLASSES_ROOT und alle untergeordneten Schlüssel geändert werden.
- Im Registrierungsbaum **HKEY_USERS on Local Machine** werden Benutzer-Einstellungen, einschließlich dem aktuellen Benutzer und dem Standard-Benutzer, gespeichert. Die Standardeinstellungen im Schlüssel \.DEFAULT sollten gegen Veränderung geschützt werden.

D.8 Änderungen der Standard-Verzeichnisrechte

In der unten aufgeführten Tabelle sind die Rechteänderung (ggf. existierende Rechte ersetzen) dargestellt. Es ist hierbei wichtig, in welcher Reihenfolge die Rechte Änderung durchgeführt werden.

Verzeichnis	Art	Benutzer	Rechte
%Systemdrive%	Rekursiv	Domänen-Benutzer	Read
		Administrator	Full Control
		Domänen-Admin	Full Control
		System, Ersteller	Full Control
\Temp	nur Dateien	Domänen-Benutzer	Add & Read
		Administrator	Full Control
		Domänen-Admin	Full Control
		System, Ersteller	Full Control
\Winnt\Help	nur Dateien	Domänen-Benutzer	Add & Read
		Administrator	Full Control
		Domänen-Admin	Full Control
		System, Ersteller	Full Control
\Winnt\Config	nur Dateien	Domänen-Benutzer	List
		Administrator	Full Control
		Domänen-Admin	Full Control
		System, Ersteller	Full Control
\Winnt\Repair	Verzeichnis	Administrator	Full Control
		Domänen-Admin	Full Control
		System	Full Control
\Winnt\System32\DHCP\	Verz. löschen	*wenn möglich*	
\Winnt\System32\RAS\	Verz. löschen	*wenn möglich*	
\Winnt\System32\WINS\	Verz. löschen	*wenn möglich*	
\BOOT.INI	Datei	Administrator	Full Control
		Domänen-Admin	Full Control
		System	Full Control
\NTDETECT.COM	Datei	Administrator	Full Control
		Domänen-Admin	Full Control
		System	Full Control
\NTLDR	Datei	Administrator	Full Control
		Domänen-Admin	Full Control
		System	Full Control
\Winnt\	Rekursiv	Domänen-Benutzer	Add & Read
%Benutzer%	Rekursiv	User	Full Control
		Domänen-Admin	List