

Überblick über Sicherheitsprobleme

Teil 1

Ahmet Yilmaz

27. Mai 2003

1 Einführung

Diese Seminararbeit soll einen Überblick über die Sicherheitsproblematik von Computersystemen und deren Vernetzung verschaffen. Der hohe Verbreitungsgrad von Computern und die parallel erfolgende Vernetzung hat neue Perspektiven in Wirtschaft, Medien, Wissenschaft und im privaten Bereich geschaffen. Aber hinter dieser innovativen Technologie verbirgt sich ein immer grösser und wichtiger werdender Bereich, der sich mit dem Thema Computersicherheit auseinandersetzt. Besondere Aufmerksamkeit und Sorge erregen die Angriffe auf die noch zum Teil insuffizient geschützten Systeme. Durch die Angriffe erfolgen Verletzungen und Beschädigungen von Privatsphären, Diebstählen, Spionageakte und Sabotagen u.s.w. Diese Angriffsarten bedienen sich genau, wie die dazu gehörige Technologie, immer ausgefeilterer und raffinierterer Mittel und bedürfen daher entsprechend angepasster Gegenmaßnahmen.

Folgende Umfrageergebnisse zeigen die Relevanz der Computersicherheit. Hierbei wurden 503 Institutionen in Bezug auf erkannte Angriffe abgefragt [1]:

- 90 % haben Computer Einbrüche festgestellt
- 80 % haben zugegeben, finanziell beschädigt worden zu sein
- 44 % konnten die Schäden quantifizieren (ca. \$ 455 848 000)
- 40 % haben ein Eindringen von Aussen festgestellt
- 40 % waren DoS Attacken ausgesetzt
- 78 % unterlagen dem Missbrauch ihrer Arbeitssysteme
- 85 % identifizierten Computer Viren

Um genauer zu wissen, wogegen man die Systeme schützen soll und welche Vorbeugungen man treffen kann, müssen erst einmal die Bedrohungsarten näher betrachtet werden. Anschließend erfolgt die Definition der "Sicherheit" im Rahmen der Computersicherheit. Hierbei werden Fragen zur Entstehung der Sicherheitsproblematik beantwortet. Weiterhin ergeht ein Überblick über die Schwachstellen in den Systemen und im Zusammenhang mit Hostsicherheit werden einige Beispiele aufgezeigt.

2 Bedrohungsarten

Die Gefahren, die es im Zusammenhang mit dem Betrieb von Computersystemen gibt, können in drei Hauptkategorien aufgeteilt werden:

- von der Umwelt ausgehende Bedrohung
- technischbedingte Bedrohung
- von Menschen ausgehende Bedrohung

2.1 Von der Umwelt ausgehende Bedrohung

Dieser Punkt kann nicht mit informatischen Lösungen bewältigt werden, dennoch erfordert er Beachtung, da erhebliche Schäden hieraus resultieren können. Dies können Feuer- und Wasserschäden sein. Gerade unachtsam im Keller installierte Server sind bei Wasser-, Rohr- und Flutschäden gefährdet. Genauso wichtig sind Brandschutzmaßnahmen und Blitzableitungsanlagen.

2.2 Technische Bedrohung

Heutige und zukünftige Netze sind gekennzeichnet durch einen hohen Grad an Komplexität und gleichzeitig dadurch, dass die einzelnen Komponenten bis zum Äussersten genutzt werden. Hierzu kommen die Probleme, die entstehen, wenn Produkte verschiedener Hersteller kombiniert werden. Jede einzelne Komponente ist ein Glied in einer Kette, und wenn eine Komponente irgendwann versagt, kann dies weitreichende Konsequenzen für den Rest des Systems haben. Komponenten, die unzureichende Standards einhalten oder die in Zusammenhängen verwendet werden, für die sie ursprünglich nicht gedacht waren, tragen zu neuen und in vielen Fällen unvorhersehbaren Fehlertypen bei. Softwarebedingte Betriebsstörungen als Folge von falscher oder mangelhafter Konfiguration, Inkompatibilität u.a.m. sind auch relevante Gefahren.

2.3 Von Menschen ausgehende Bedrohung

Eine grobe Einteilung dieser Bedrohung kann wie folgt aussehen:

- Unbeabsichtigt
 - Implementierungsfehler
 - Nutzungsfehler
- Beabsichtigt
 - Attacken
 - Absichtlich eingebaute böswillige Codes

2.3.1 Unbeabsichtigt

Die unbeabsichtigten Bedrohungen können in Form von Anwenderfehlern erfolgen. Dies sind beispielsweise das Löschen oder Überschreiben von Dateien. Ein weiterer Punkt ist eine ungeeignete Passworauswahl durch unzureichend sensibilisierte Anwender. Ein unüberdachtes Passwort bietet einem Eindringling ein leichtes Spiel, um ins System zu gelangen. Weiterhin stellen schlecht ausgebildete Systemadministratoren ein Risiko dar, denn sie durchschauen die Komplexität häufig nicht. Dadurch werden unnötige Dienste zugelassen, die eine weitere Angriffsfläche darstellen können.

Da die Implementierung von Algorithmen ein komplexer Prozess sein kann, können sich bei der Programmierung Fehler einschleichen (sogenannte Bugs), die zum grössten Teil bei den Attacks ausgenutzt werden können. Die meisten implementierten Algorithmen sind nur unter eingeschränkten Rahmenbedingungen zu testen. Dies lässt eine vollständige Vorhersage über die Korrektheit nicht zu. Zudem verhindert der Mangel an formalen Beweisen den hundertprozentigen Ausschluss von Bugs.

2.3.2 Beabsichtigt

Die Diversität der Personen dieser Rubrik ist vielfältig. Sie reicht von dem unzufriedenen Mitarbeiter bis hin zu Geheimdiensten. Sie setzen unterschiedliche Mittel ein und haben unterschiedliches Wissen. In der Literatur und im Volksmund ist die Bezeichnung "Hacker" als Stereotyp geläufig. Der Angreifer versucht unter anderem die obengenannten Schwachstellen auszunutzen. Desweiteren existieren Codes z.B. Viren, Würmer, Trojanische Pferde, Trap door, Aktiv content, die konzipiert worden sind, um Schäden anzurichten.

Folgende Schäden können hieraus resultieren:

- Geldverlust
- Imageverlust
- Verletzung gesetzlicher Vorschriften
- Gestiegene Betriebskosten
- Verlorene Geschäftsmöglichkeiten
- Nachteile gegenüber der Konkurrenz

3 Definition von Sicherheit

Ich möchte zunächst zwei Definitionen der Sicherheit anführen.

- Verhindern unbefugter Aktivität an, mit oder durch das eigene System [2]
- Gewährleisten gewollter Aktivitäten an, mit oder durch das System [2]

Sicherheit bedeutet, gewisse Ziele trotz gewisser Bedrohungen zu erreichen [3].

Für eine erfolgreiche Umsetzung eines Sicherheitskonzeptes müssen zunächst die folgenden Eigenschaften etabliert werden:

Verfügbarkeit: Nutzungsmöglichkeit der Betriebsmittel für gewollte Aktivitäten

Integrität: Datenintegrität bezeichnet den Umstand, dass die gespeicherten Daten die Realität exakt widerspiegeln sollen.

Vertraulichkeit: Die Daten des Systems können nur von den Personen gelesen und benutzt werden, die dazu berechtigt sind.

Authentizität: Die Identität des Verfassers entspricht die Realität.

Nicht Leugbarkeit: Die Identität ist eindeutig, kann nicht bestritten werden.

4 Historische Entwicklung von Sicherheitsproblemen

Bis Ende der achtziger Jahre stand der Sicherheitsaspekt von Computern und Netzwerken eher im Hintergrund des Interesses. Die Gründe dafür liegen in deren Entwicklung. Bedingt durch die Komplexität der Systeme stand anfangs die einfache Funktionstüchtigkeit im Vordergrund. Damals lieferten die Anlagen ihre Dienste an auserwählte Personen. So lange man ausreichend dicke Wände und robuste Schlösser hatte und keiner der Auserwählten die Lager wechselte, brauchte man sich keine Sorgen machen. Im Laufe der Zeit hat die Vernetzung und die Nutzung von verteilten Ressourcen zugenommen. Parallel dazu sind die Sicherheitsbedürfnisse gestiegen, aber es wurde mehr Wert auf die Funktionalität gelegt und somit wurde der Sicherheitsaspekt vernachlässigt. Da heutige Systeme auf Paradigmen, bei deren Entwicklung die Sicherheit keine große Rolle gespielt hatte, von gestern aufbauen, erweist sich die Integration von Sicherheit als problematisch.

5 Hostsicherheit

Ein System ist so sicher, wie sein schwächstes Glied. Ein Angriff auf die Hostsicherheit kann auf zwei Ebenen erfolgen.

- Physikalische Ebene
- Software Ebene

5.1 Physikalische Ebene

Alle physikalisch zugänglichen Systemkomponenten wie Kabel, Hub, Switch, Router, Rechner u.s.w. können einem potenziellen Angriff ausgesetzt sein. Ein System mit einem ausgezeichneten Passwortmechanismus wäre nutzlos, wenn entsprechende physikalische Sicherheitsmassnahmen, wie Zutrittskontrolle, Datenträgerkontrolle u.s.w. nicht gewährleistet sind. Durch physikalische Sicherheit sollen folgende Angriffe verhindert werden:

- Hardware Ausbau
- Reboot von Systemen
- Anschluss eines Rechners an das Netz
- Umkonfiguration
- Account Manipulation

5.2 Software Ebene

Nach einer erfolgreichen Etablierung von physikalischen Sicherheitsmassnahmen müssen Softwaretechnische Sicherheitsmassnahmen eingeleitet werden. Dies betrifft insbesondere die Betriebssysteme, Anwendungen und die Protokolle.

Wesentliche Schwachstellen sind:

- Fehler und Sicherheitslücken (aus oben benannten Gründen) im Betriebssystem und den Serverprogrammen
Ein typischer Betriebssystemfehler, der ein Eindringen ermöglicht, ist der Buffer-Overflow.
- Schwach implementierte Passwort Mechanismen
- Anerkennung von System-Aufrufen, Diensten oder ausführbare Codes enthaltene Daten als vertrauensvoll. Darunter fallen R-Kommandos bei UNIX-Systemen, Makrosprachen in Verbindung mit Excel- und Word-Applikationen.
- Netzprotokolle besitzen oft Lücken, die "historisch" bedingt sind. (Siehe Teil 2)

5.3 Angriffsmöglichkeiten auf die Hostsicherheit

Das Ziel bei einem Angriff ist es, den Zugang zum System und genügend Kontrolle zu erlangen, um das System zu kompromittieren oder mit Malware zu infizieren. Ausserdem kann ein extremer Verbrauch von Ressourcen, bedingt durch unzählige Inanspruchnahme von regulären Diensten, zu einem Zusammenbruch des Systems führen. Anschliessend werden die Spuren beseitigt. Je effektiver die Spuren gelöscht werden, desto schwieriger ist es, den Hack aufzudecken und passende Gegenmassnahmen zu ergreifen.

Eine mögliche Systempenetration ist durch folgende Schritte gekennzeichnet:

1. Erlangung des Zugangs zum System
2. Etablieren eines Supervisor- oder Superuser-Status
3. Einrichtung einer Hintertür zum System
4. Löschen aller Spuren

Beispiele einiger häufig vorkommender Angriffsarten [4]:

- Hintertüren: Eine versteckte Funktionalität in einem System, welche es erlaubt, die Sicherheitsfunktionen zu umgehen. Hintertüren werden oft von System-Entwicklern eingebaut, um Programme während der Entwicklung einfacher testen zu können oder sich später auf einfachem Wege Zugriff auf ein System verschaffen zu können.
- denial-of-service: Eine Aktion oder Serie von Aktionen, die dazu führt, dass ein System oder eine seiner Ressourcen nicht mehr effizient und verlässlich arbeitet (z. B. durch hohe eigene Inanspruchnahme von Dienstleistungen).
- Trojanische Pferde: Ein eigenständiges Programm, welches vordergründig eine vorgesehene Aufgabe verrichtet, zusätzlich jedoch eine oder mehrere nicht-autorisierte Funktionen enthält.
- Würmer: Ein eigenständiges Programm, welches sich selbst (meist unter Ausnutzung von Schwachstellen) durch Kopieren von einem System zum nächsten fortpflanzt, üblicherweise über ein Netzwerk.
- Viren: Ein Programmteil (Code Fragment, kein eigenständiges Programm!), welches sich durch das Anhängen an oder Einnisten in andere Programme reproduziert.
- Logische Bomben: Eine versteckte Funktionalität (z. B. in einem Trojanischen Pferd), welche eine nicht-autorisierte Aktion (meist eine Schadenfunktion) auslöst, wenn ein bestimmter Zustand des Systems erreicht wird.
- Spoofing: (engl. "Spoon" = "Parodi") Spezielle Form der Maskerade. Täuschung eines Subjekts (z. B. eines Benutzers) oder einer System-Instanz durch einen Angreifer, welche eine Verletzung der Systemsicherheit oder die Herausgabe von Informationen an den Angreifer zur Folge hat.
- Aktiv content (HTML, JavaScript, Java, VBA, VBS, AktivX, PHP, Pearl, usw): Sind ausführbare Programmteile, die in ein Dokument oder eine Internetseite integriert sind. Diese Programmteile werden bei der Öffnung des Dokuments oder Internetseite ausgeführt.

5.3.1 Angriffe auf Passwort-Mechanismen

In den meisten Systemen wird die Identifikation und Autorisierung des Benutzers durch einen Passwort-Mechanismus geregelt. Das System-Gedächtnis über den Benutzer, sein Passwort und weitere Informationen wie Benutzername, Gruppenidentifikationsnummer, Beschreibung, Angabe des Heimatverzeichnis werden in eine Datei z.B. `/etc/passwd` unter Unix-Systemen gespeichert. Dabei ist das Passwort z.B. mit dem System-Aufruf `crypt` verschlüsselt. Die Passwort-Datei ist aber für jeden Benutzer lesbar. Dies kann von einem Angreifer genutzt werden. Indem der Angreifer ein mögliches Passwort auswählt, mit dem Systemaufruf `crypt` verschlüsselt und das Ergebnis mit den Passwortfeldern der `/etc/passwd` vergleicht. Ist das ausgewählte Wort nach dem Verschlüsseln identisch mit dem Eintrag in dem Passwortfeld, so ist das Passwort gefunden. Hieraus folgt, dass man aus einem verschlüsselten Passwort das Passwort

selbst nicht zurück gewinnen kann, daher werden zum Erlangen des Passwortes Passwort-Cracker eingesetzt. Diese automatisieren den oben beschriebenen Vorgang.

Passwort-Cracker sind nichts anderes als Programme, die mittels der Kombination von Wörterbuch-Attacken und der Brute-Force-Methode versuchen, Passwörter zu knacken. Auf der Basis eines umfangreichen Wörterbuches testen sie mit hoher Geschwindigkeit Wort für Wort, Zeichenkette für Zeichenkette auf Passwörter. Eine andere Methode basiert auf einer umfangreichen Wörterbuchdatei. Hier wird jeder Wörterbucheintrag mit der vom Betriebssystem benutzten Verschlüsselung verschlüsselt und mit einem verschlüsselten Zielpasswort verglichen. Wenn es zu einer Übereinstimmung kommt, ist das Passwort geknackt. Gute Passwort-Knacker gehen aber noch weiter, indem sie Regeln verwenden, z. B.:

- jedes Wort auch rückwärts testen
- Groß- und Kleinschreibung kombinieren
- Wörter zusammenhängen (z. B. Wort vorwärts + Wort rückwärts)
- Einfügen von Zahlen am Anfang, Ende oder im Wort

Einige Namen von Passwort-Cracker:

UNIX-Passwort-Cracker:

- Crack
- John the Ripper
- Hades
- CrackerJack

Windows NT- Passwort-Cracker:

- 10phtCrack 2.0,
- ScanNT
- NTCrack

Für die Auswahl eines Passwortes gilt:

- keine Login-Namen
- keine anderen Namen
- keine Wörter oder Abkürzungen, die im Wörterbuch zu finden sind
- keine persönlichen Informationen (Geburtstag, Telefonnummer usw.)

- keine Variante der oben genannten schlechten Passwörter (z.B. rückwärts oder Grossbuchstaben usw.)
- nicht nur Ziffern nutzen

Gut sind Passwörter mit einer Mischung aus Gross- und Kleinbuchstaben, die mindestens sechs Zeichen lang sind und aus einer zufälligen Auswahl von Ziffern und Buchstaben bestehen.

5.3.2 Angriffe auf Systemaufrufe

Ein Parade-Beispiel für einen Angriff ist der Pufferüberlauf. Mit diesem Beispiel kann sehr gut gezeigt werden, wie eine Systemlücke ausgenutzt wird, um Administratorrechte zu erlangen. Aus diesem Grund wird dieser Sachverhalt näher betrachtet.

Egal ob Solaris, HP/UX oder Windows 2000: Zu jedem Betriebssystem kommen regelmässig neue Meldungen über Sicherheitslücken und nicht selten steht in der Erklärung, dass es sich um einen Pufferüberlauf handelt, der einem Angreifer letztlich sogar Systemverwalter-Berechtigung auf der Zielmaschine verschaffen kann.

Wie kommt es eigentlich zu einem Pufferüberlauf und wie lässt er sich für einen Angriff ausnutzen? Wie der Name schon sagt, wird bei einem solchen Programmierfehler ein Puffer zum Überlaufen gebracht. Meist ist das ein Speicherbereich, in dem Eingaben für die weitere Verarbeitung abgelegt werden. Diese Eingaben kommen entweder aus Übergabeparametern beim Aufruf eines Programms von der Kommandozeile, aus Dialogeingaben oder Netzwerkprotokollen. Im Fehlerfall ist ein Eingabewert länger, als es das Programm erwartet. Er überschreibt damit den Speicherbereich einer Variablen und die im Speicher darauf folgenden Werte. Voraussetzung hierfür ist natürlich, dass der Programmierer vergessen hat, die maximal zulässige Länge der Eingabewerte zu überprüfen.

Dies allein liefert jedoch noch keine Einbruchmöglichkeit, da bei fast jeder Prozessorarchitektur das ausführbare Programm getrennt von den Variablen gespeichert wird. Um in die Maschine einzubrechen, möchte der Angreifer jedoch nicht einfach den Wert von Variablen überschreiben, sondern eigenen Programmcode einschleusen, der anschließend auch ausgeführt wird.

Risikante Überlauf-Probleme treten bei lokalen Variablen auf. Lokale Variablen sind nur innerhalb einer Unterroutine oder Funktion gültig und werden daher zusammen mit der Rücksprungadresse der Funktion auf dem so genannten Stack gespeichert. Der Stack ist ein Speicherbereich, der wie ein Stapel wächst. Werte können "oben" auf den Stapel gelegt und wieder vom Stapel "herunter" genommen werden. Dadurch lassen sich die Rücksprungadressen vieler hierarchisch verschachtelter Unterprogramme sehr elegant verwalten: Der Prozessor legt beim Aufruf eines Unterprogramms die aktuelle Adresse als Rücksprungadresse auf den Stack. Ruft das Unterprogramm seinerseits ein weiteres Unterprogramm auf, so wird wieder die aktuelle Adresse auf den Stack gelegt, der Stack wächst. Nachdem ein Unterprogramm beendet ist, liegt die Adresse, an der es im übergeordneten Programm weitergeht, oben auf dem Stack. "Oben" auf dem Stack bedeutet dabei eine niedrigere Speicheradresse, da der Stack von höheren Speicheradressen zu niedrigeren Adressen hin wächst.

Lokale Variable, die nur innerhalb eines Unterprogramms gültig sind, können ebenfalls elegant auf dem Stack gespeichert werden. Beim Verlassen des Unter-

programms gibt der Prozessor ihren Speicherplatz automatisch wieder frei. Falls nun die Länge einer Zeichenkette, die in einer lokalen Variable gespeichert werden soll, grösser ist als der dafür vorgesehene Platz, überschreibt eine ungeprüfte Speicherung dieses Wertes nicht nur andere lokale Variablen, sondern eventuell auch die Rücksprungadresse der aktuellen Unterroutine.

Bestimmte Funktionen der Programmiersprache C begünstigen dies; sie kommen in unzähligen Programmen zum Einsatz. Es handelt sich dabei vor allem um die Zeichenkettenverarbeitungsfunktion `strcpy()`. Sie hat zwei Parameter: eine Quell- und eine Ziel-Adresse, an der die Funktion jeweils Speicherplatz für einen String erwartet. Sie kopiert Zeichen für Zeichen des Quell-Strings in den Ziel-String und hört erst dann auf, wenn das Zeichen `\ 0` (der ASCII-Wert 0) den Quell-String beendet.

Das Null-Zeichen als Zeichenkettenende ist in C durchaus üblich. Wenn ein Programmierer davon ausgeht, dass bestimmte Eingabewerte maximal 80 Zeichen lang sind, und er diesen Wert mit `strcpy()` in einen Puffer kopiert, für den er 100 Bytes reserviert hat, dann prüft die Funktion `strcpy()` dieses Maximum nicht. Falls der Quell-String erst nach 200 Zeichen eine Null enthält, dann überschreibt die Funktion eben weitere Speicherbereiche. Ist der Ziel-String eine lokale Variable, dann kann dieses Prozedere auch die Rücksprungadresse der aktuellen Funktion auf dem Stack überschreiben.

Wenn der Prozessor nach dem Beenden der aktuellen Funktion mit dem aufrufenden Programm fortfahren will, dann liest er aus dem Speicher der Rücksprungadresse einen Teil des kopierten Textes und interpretiert ihn als Adresse. Da an dieser Adresse bei unbeabsichtigten Überläufen meist kein sinnvolles Programm steht, führt dieser Zustand häufig zu einem Absturz mit der Meldung "Segmentation Fault" (Speicherzugriffsschutz-Fehler). Falls an der neuen Adresse jedoch ein funktionsfähiger Maschinencode steht, so wird dieser ausgeführt.

Genau dies ist das Ziel des Angreifers. Er sucht nach Stellen in Programmen oder Netzwerkdiensten, die Eingabewerte mit Funktionen wie `strcpy()` bearbeiten, und versucht dann über den Eingabewert Maschinencode einzuschleusen und gleichzeitig die Rücksprungadresse derart zu überschreiben, dass sie auf den gerade eingeschleusten Code zeigt.

Wenn der Maschinencode entsprechend ausgewählt ist, wird bei dem Rücksprung ein Shell mit Administratorrechten gestartet. Damit erlangt der Angreifer die absolute Kontrolle über das System [5] [6].

5.3.3 Angriffe durch Anwendungen und Dienste

Eine weitere Angriffsmöglichkeit ist die Einbettung von schädlichen Funktionen verpackt in Anwendungsprogrammen (Trojanisierung) oder in Office-Dateien, die über eine Makrosprache (wie: VBS, VBA) verfügen. Dies stellt eine Kommandosprache dar, die beim Öffnen der Datei automatisch ausgeführt wird. Sie ist destruktiv instruiert und richtet Schäden an, wie z.B. durch Löschen, Umbenennen von Dateien.

Ausserdem existieren in Webseiten eingebettete ausführbare Codes wie: JavaScript, ActiveX, Java, diese können auf der Seite des Klienten Schäden anrichten.

Bei der Systemkonfiguration werden meist Dienste zugelassen, die nicht unbedingt erforderlich sind. Dieses Zulassen stellt eine potentielle Gefahrenquelle dar. Z.B.: remote Dienste, da sie eine Kommandoausführung von fremden Rech-

nern aus ermöglichen, wobei die Verbindung nicht verschlüsselt ist. Damit kann der Angreifer die übertragenen Daten und die darunter befindlichen Passwörter mitlesen. Eine Gegenmassnahme wäre das Abschalten von remote Diensten oder das Ersetzen durch ihre sichereren Äquivalente ssh.

Literatur

- [1] Computer Security Institute, San Francisco 2002:
<http://www.gocsi.com/press/20020407.html>
- [2] Hans-Joachim Mück, Carsten Benecke, Stefan Kelm (2000):
"Sicherheit in vernetzten Systemen. Uni-HH", Seminar Bericht 224.
- [3] Universität Saarbrücken, Arbeitsgruppe Sicherheit und Kryptographie:
<http://www-krypt.cs.uni-sb.de>
- [4] Dipl. Inf. Thomas Hungenberg:
www.demonium.de
- [5] Stephan Kallnik, Daniel Pape, Daniel Schröter, Stefan Strobel:
"c't 23/2001, S. 216: Sicherheit: Buffer-Overflows".
<http://www.heise.de/ct/01/23/216/>
- [6] Zeitschrift für Kommunikations- und EDV-Sicherheit:
<http://www.kes.info>
- [7] Prof. Jürgen Plate. Dipl.-Ing. Jörg Holzmann:
"Sicherheit in Netzen"
<http://www.netzmafia.de/skripten/sicherheit/index.html>
- [8] Petra Eilfeld:
<http://www.lrz-muenchen.de/services/security/motivation/>
- [9] Wikipedia, die freie Enzyklopädie:
<http://de.wikipedia.org/wiki/Computersicherheit>
- [10] Bruce Schneier:
"Secrets & Lies." "IT-Sicherheit in einer vernetzten Welt".
- [11] Georg Erwin Thaller:
"Computersicherheit".

- [12] Anonymous:
Hacker's Guide ."Sicherheit im Internet und im lokalen Netz".

- [13] Anonymous:
Linux Hacker's Guide ."Sicherheit für Linux-Server und -Netze".

- [14] <http://www.theo2.physik.uni-stuttgart.de/jtb/overflow/tutorial.html>