

Honeypots

Einsatzmöglichkeiten beim Schutz von IT-Systemen *

Michael Krooß
5krooss@informatik.uni-hamburg.de

Fachbereich Informatik
Universität Hamburg

15. Mai 2003

Zusammenfassung

In dieser Arbeit sollen die Einsatzmöglichkeiten von Honeypots im Bereich des Schutzes von IT-Systemen betrachtet werden. Dazu wird zunächst ein Überblick über Honeypots, ihre Einsatzmöglichkeiten und die durch ihren Einsatz bedingten Risiken gegeben. Anschließend werden drei Beispiele für Honeypots betrachtet.

*Diese Arbeit wurde in Begleitung zu einem Vortrag im Seminar 18.415: „Sicherheit in vernetzten Systemen“ im Wintersemester 2002/2003 erstellt. Sie stellt in kurzer Form den im Vortrag behandelten Themenbereich vor.

Inhaltsverzeichnis

1	Überblick über Honeypots	3
1.1	Definition „Honeypot“	3
1.2	Implementationsmöglichkeiten	4
1.3	Einsatzgebiete	4
1.4	Grad der Interaktionsmöglichkeiten für Angreifer	5
1.5	Einsatz von Produktivitäts-Honeypots	5
1.6	Rechtliche Aspekte	7
1.7	Fazit	8
1.7.1	Vorteile von Honeypots	8
1.7.2	Nachteile von Honeypots	8
2	Beispiele für Honeypots	9
2.1	Beispiel 1: BackOfficer Friendly	9
2.2	Beispiel 2: Honeyd	11
2.2.1	Exkurs: ARP-Spoofing	11
2.3	Beispiel 3: Honeynets	15

Abbildungsverzeichnis

1	Aufzeichnung einer HTTP-Anfrage mit netcat	4
2	Möglichkeiten der Platzierung von Honeypots in Unternehmensnetzen	6
3	Einstellung von Optionen beim BackOfficer Friendly	10
4	Ausgabe von netstat nach der Konfigurierung von BackOfficer Friendly	10
5	Ausgabe des BackOfficer Friendly nach verschiedenen Anfragen	10
6	Funktionsweise des Address Resolution Protocol	12
7	ARP Request	12
8	ARP Reply	13
9	ARP-Spoofing in Broadcast-Netzen	13
10	ARP-Spoofing in geschichteten Netzen	14
11	Aufruf von arpd und honeyd	14
12	Architektur eines Honeynet	16

Tabellenverzeichnis

1	Von BOF überwachte Dienste bzw. Ports	9
---	---------------------------------------	---

1 Überblick über Honeybots

In diesem Abschnitt soll ein Überblick über Honeybots gegeben werden. Dazu erfolgt im Unterabschnitt 1.1 eine Definition des Begriffs Honeybot und eine Aufstellung von daraus folgenden Anforderungen für einen Honeybot. Im Unterabschnitt 1.2 werden einige Möglichkeiten zur Implementierung von Honeybots aufgezeigt. Im Unterabschnitt 1.3 werden Einsatzgebiete von Honeybots aufgezeigt und im Unterabschnitt 1.4 verschiedene Grade der Interaktion unterschieden, die Honeybots potentiellen Angreifern bieten können. Im Unterabschnitt 1.5 wird betrachtet, wie Honeybots zum Schutz von Rechnernetzen beitragen können. Im Unterabschnitt 1.6 werden rechtliche Aspekte aufgezeigt, die beim Einsatz von Honeybots beachtet werden müssen. Zum Abschluss dieses Abschnitts werden im Unterabschnitt 1.7 als Fazit dieses Überblicks die Vor- und Nachteile aufgezeigt, die mit dem Einsatz von Honeybots verbunden sind.

1.1 Definition „Honeybot“

Im Rahmen dieser Arbeit soll von der folgenden Definition für Honeybots ausgegangen werden (nach [Spi02, Seite 40]):

A security resource whose value lies in being probed, attacked or compromised.

Ein Honeybot trägt also erst dadurch zum Schutz bei, dass er (durch einen Angreifer) auf Schwachstellen untersucht, angegriffen oder sogar kompromittiert wird. Hieraus werden die folgenden Forderungen abgeleitet:

- Da Dienste, die als Honeybot angeboten werden, angegriffen werden sollen, sollten diese nicht als Produktionsdienste dienen.
- Da von einem Honeybot keine Produktionsdienste angeboten werden, gibt es grundsätzlich keine autorisierte Nutzung eines Honeybots.
- Da es keine autorisierte Nutzung gibt, ist ein Verbindungsaufbau zu einem Honeybot im Allgemeinen ein Scan (bzw. Probe)¹ oder ein Angriff. Ausnahmen sind Fehler, wie z.B. Eingabefehler bei der IP-Adresse, oder ein autorisierter Test der Funktionen des Honeybots.
- Erfolgt ein Verbindungsaufbau von einem Honeybot, so kann davon ausgegangen werden, dass das System kompromittiert wurde.

Es muss beachtet werden, dass durch den Einsatz von Honeybots keinen Verwundbarkeiten von IT-Systemen entgegengewirkt wird. Aus diesem Grund stellen sie keine Alternative sondern eine Ergänzung zu anderen Schutzmaßnahmen wie z.B. Firewalls, Intrusion Detection Systems (IDS) oder einem den Sicherheitsanforderungen angemessenen Nutzungsverhalten dar.

¹Probe – Gezielte Überprüfung auf Existenz einzelner Lücken und Schwachstellen; Scan – Überprüfung einer Menge von Rechnern nach Sicherheitslücken und Schwachstellen (nach [Kos00, Seite 9])

1.2 Implementationsmöglichkeiten

Honeypots können auf verschiedene Weise implementiert sein. So können einfach Ports² überwacht, emulierte Dienste angeboten oder eingeschränkte Betriebssysteme bzw. vollständige Systeme als Honeypot eingerichtet werden. Durch das Überwachen von Ports können Versuche eines Verbindungsaufbaus und evtl. auch eingehende Anfragen auf diesen Ports aufgezeichnet werden. Dies kann z.B. mit dem Befehl `'netcat -l -v -p 80'` geschehen. Dadurch wird ein Dienst gestartet, der bei einem Verbindungsaufbau auf Port 80, an den generell Anfragen mit dem „Hypertext Transfer Protocol“ (HTTP) gestellt werden, die gestellten Anfragen ausgibt. In Abbildung 1 wird die Aufzeichnung einer HTTP-Anfrage auf dem Rechner `groover` (IP-Adresse 192.168.16.80) aufgezeigt.

```
# netcat -l -v -p 80
listening on [any] 80 ...
connect to [192.168.16.80] from elmo [192.168.16.79] 32805
GET / HTTP/1.0
```

Abbildung 1: Aufzeichnung einer HTTP-Anfrage mit `netcat`

Als Erweiterung hierzu können emulierte Dienste eingesetzt werden, die in einem eingeschränkten Rahmen auf Anfragen reagieren. Es ist sogar denkbar, dass diese auch Verwundbarkeiten des zu emulierenden Dienstes (z.B. eines bestimmten HTTP-Servers) vortäuschen, um somit mehr Informationen darüber zu erlangen, ob es sich bei der Anfrage um einen Angriff handelt und von welcher Art dieser ist. Ferner können Betriebssysteme mit eingeschränktem Funktionsumfang, die z.B. mit der `chroot`-Umgebung unter Unix realisiert werden können, oder einzelne Rechner bzw. ganze Rechnernetze als Honeypot eingesetzt werden.

1.3 Einsatzgebiete

Marty Roesch, der Entwickler von `Snort`, einem Open Source Intrusion Detection System, unterscheidet generell die folgenden Einsatzgebiete von Honeypots (vgl. [Spi02, Seite 44]):

- Produktivitäts-Honeypots (production honeypots)
Diese stellen in dem Sinne einen Produktivitätsfaktor dar, als dass sie zur Erhöhung der Sicherheit eines Rechnernetzes und damit zum Schutz der das Rechnernetz betreibenden Organisation beitragen.
- Forschungs-Honeypots (research honeypots)
Diese sollen zum Sammeln von Informationen z.B. über Verwundbarkeiten, Bedrohungen und Angreifer u.a. ihrer Werkzeuge, Motive und Taktiken dienen.

In dieser Arbeit wird vorwiegend auf Produktivitäts-Honeypots und nur am Rande auf Forschungs-Honeypots eingegangen werden.

²Zu Grundlagen von Internet-Protokollen vgl. [GS03]

1.4 Grad der Interaktionsmöglichkeiten für Angreifer

Im Unterabschnitt 1.2 wurden verschiedene Implementationsmöglichkeiten für Honeybots aufgezeigt. Die verschiedenen Varianten erlauben einem Angreifer unterschiedliche Grade der Interaktion mit dem Honeybot. Grundsätzlich kann gesagt werden, dass je höher die Interaktionsmöglichkeiten eines Angreifers mit einem Honeybot sind, umso mehr über Angriffe bzw. über den Angreifer gelernt werden kann. Es muss allerdings festgestellt werden, dass mit dem Grad der Interaktion auch die mit dem Einsatz des Honeybots verbundenen Risiken steigen. So können kompromittierte Honeybots von einem Angreifer z.B. genutzt werden, um andere Systeme anzugreifen. Ferner steigt mit dem Grad der Interaktion auch der mit dem Betrieb verbundene Aufwand. Aus diesen Gründen kann angenommen werden, dass Produktivitäts-Honeybots generell eher niedrige Interaktionsmöglichkeiten bieten, um das Risiko und den Aufwand zu begrenzen, während Forschungs-Honeybots in der Regel eher hohe Interaktionsmöglichkeiten bieten, um mehr Informationen sammeln zu können.

1.5 Einsatz von Produktivitäts-Honeybots

Maßnahmen zum Schutz von IT-Systemen können grundsätzlich in vorbeugende (prevention), erkennende (detection) und reagierende (response) Schutzmaßnahmen unterteilt werden (vgl. [Sch00]). Im Folgenden soll aufgezeigt werden, auf welche Weise Honeybots zur Vorbeugung, Erkennung oder zum Reagieren auf Angriffe eingesetzt werden können und in welchem Grad sie hierbei wirken.

Um Angriffen vorzubeugen sind Honeybots nur bedingt geeignet. Grundsätzlich kann durch Honeybots auf folgende Weisen einem Angriff entgegengewirkt werden:

- Sie können Angreifer täuschen, so dass diese Ressourcen für einen Angriff auf ein vermeintliches Produktionssystem aufwenden, das in Wirklichkeit ein Honeybot ist.
- Sie können eine abschreckende Wirkung haben, wenn Angreifer eine Gefahr darin sehen, dass sie bei einem Angriff auf das Rechnernetz einer Organisation u.U. einen Honeybot angreifen.

Beides zeigt aber nur bei gezielten Angriffen durch menschliche Angreifer eine nennenswerte Wirkung. Häufig werden von Angreifern aber automatisierte Tools eingesetzt und die angegriffenen Adressbereiche zufällig ausgewählt.

Zur Erkennung von Angriffen sind Honeybots sehr gut geeignet. Da es keine autorisierte Nutzung eines Honeybots gibt, stellt jeder Verbindungsaufbau grundsätzlich einen Angriff dar. Auf diese Weise kann z.B. festgestellt werden, ob es einem Angreifer gelungen ist, andere Schutzmaßnahmen zu überwinden, die eine direkte Verbindung zum Honeybot (und andere Teile des Rechnernetzes) verhindern sollten.

Fand ein erfolgreicher Angriff statt, so können Honeybots u.U. genutzt werden, um festzustellen, was genau vorgefallen ist. Hat ein Angreifer z.B. Zugriff auf verschiedene Teile des IT-Systems und auch auf einen Honeybot erlangt, so kann u.U. anhand der Log-Daten auf dem Honeybot, festgestellt werden, auf welche Weise es einem Angreifer gelungen ist,

sich Zugriff auf die Systeme zu verschaffen. Da auf einem Honeypot keine Produktionsdienste laufen sollten, kann davon ausgegangen werden, dass diese Log-Daten oft keine, in jedem Fall aber weniger Daten enthalten, die nicht mit dem Angriff in Verbindung stehen, was eine Rekonstruktion der Vorgänge vereinfacht. Ferner können Honeypots zum Training von Incident Response Fähigkeiten eingesetzt werden. In beiden Fällen bieten Honeypots den Vorteil, dass sie auch über einen Zeitraum von mehreren Stunden oder Tagen vom übrigen Rechnernetz getrennt werden können, ohne dass sich eine Beeinträchtigung der Geschäftsprozesse der das Rechnernetz betreibenden Organisation ergibt.

Generell muss festgestellt werden, dass Honeypots keine alleinige Lösung zur Vermeidung von Sicherheitsvorfällen darstellen. Statt dessen müssen sie als ein Baustein in einem umfassenden Sicherheitskonzept gesehen werden. Dabei können Honeypots an verschiedenen Stellen in einem Rechnernetz platziert werden (vgl. Abbildung 2):

- Honeypot A: Platzierung in einem nicht von der Firewall geschützten Bereich
Da der Honeypot in diesem Fall völlig ungeschützt möglichen Angriffen aus dem Internet ausgesetzt ist, muss er sehr stark überwacht werden, um z.B. zu verhindern, dass dieser bei einer Kompromittierung von einem Angreifer genutzt wird, um Dritte anzugreifen. Der Aufwand, der durch die Überwachung entsteht, kann im Allgemeinen nicht gerechtfertigt werden. Daher ist diese Platzierung generell nicht über einen längeren Zeitraum sinnvoll. Für einen kurzen Zeitraum kann sie aber sinnvoll sein,

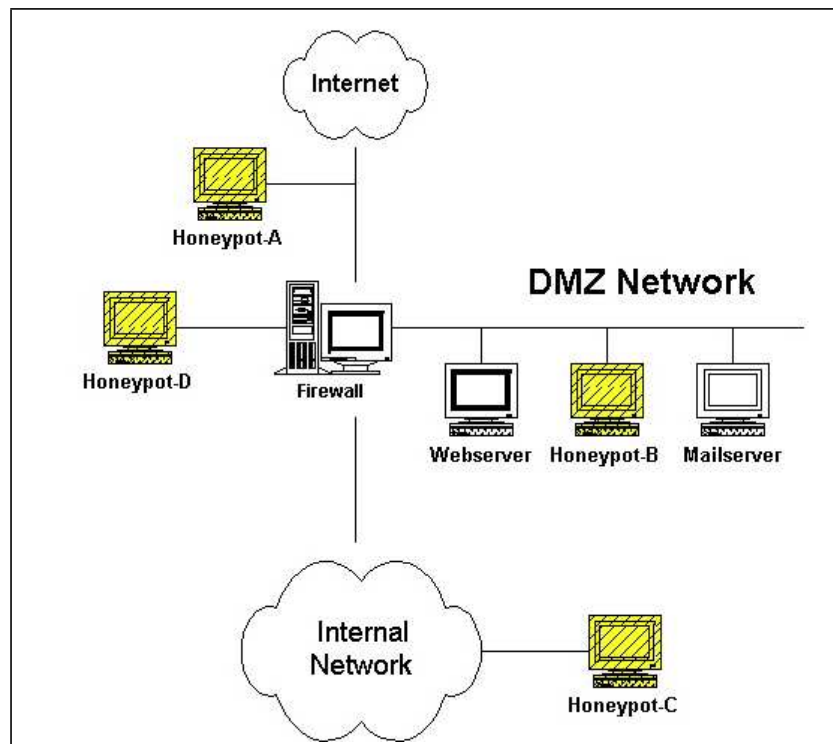


Abbildung 2: Möglichkeiten der Platzierung von Honeypots in Unternehmensnetzen (vgl. [Spi02, Seite 287])

um grundsätzlich feststellen zu können, ob, welchen und wie vielen Angriffen ein so platzierter Honeypot und daher im Prinzip auch das Rechnernetz ausgesetzt ist.

- Honeypot B: Platzierung in der sog. Demilitarisierten Zone (DMZ)
Auch bei dieser Platzierung sollte der Honeypot nicht als Produktionssystem dienen. Daher sollte z.B. kein DNS(Domain Name System)-Eintrag für den Honeypot eingerichtet werden, der dazu führen könnte, dass legitime Nutzer (z.B. Kunden) aus dem Internet Verbindungen zum Honeypot aufbauen. Statt dessen sollte es die Aufgabe eines in der DMZ platzierten Honeypots sein, Scans über den Adressbereich der DMZ und u.U. auch die Art eines Angriffs (z.B. angegriffener Dienst) festzustellen.
- Honeypot C: Platzierung im internen Rechnernetz
Eine Platzierung im internen Rechnernetz kann sowohl zur Erkennung von Angriffen aus dem externen als auch aus dem internen Netz dienen. Bei Angriffen aus dem externen Rechnernetz kann festgestellt werden, dass ein Angreifer es geschafft hat, andere Sicherheitsmechanismen wie z.B. eine Firewall zu umgehen. Bei Angriffen aus dem internen Rechnernetz kann entweder ein Angriff von autorisierten Nutzern vorliegen, der unter Missbrauch ihrer Rechte durchgeführt wird, oder ein Angriff von einem Externen, der Zugriff auf einen Rechner im internen Rechnernetz erlangt hat.
- Honeypot D: Platzierung in einem eigenen Netz
Eine Platzierung in einem gesonderten Netz ist insbesondere für Forschungs-Honeypots interessant. Im Rahmen von Produktionssystemen kann sie höchstens dazu dienen, die Incident Response Fähigkeiten im Unternehmen zu verbessern. Dabei kann nach einem erfolgten Angriff das gesamte gesonderte Netz zur Untersuchung vom übrigen Rechnernetz getrennt werden, ohne dass dabei eine Beeinträchtigung der Geschäftsprozesse erfolgt.

1.6 Rechtliche Aspekte

Beim Einsatz von Honeypots können rechtliche Probleme auftreten. Im Folgenden sollen einige dieser Probleme kurz aufgezeigt werden. Eine detaillierte Betrachtung ist im Rahmen dieser Arbeit allerdings nicht möglich.

- Datenschutz
Dürfen von einem Angreifer übertragene oder gespeicherte Daten eingesehen und ausgewertet werden?
- Verleitung, Anstiftung
Stellt die Bereitstellung eines angreifbaren Systems eine Verleitung bzw. Anstiftung dar?
- Haftung
Bestehen Haftungsansprüche Dritter, wenn diese von einem kompromittierten Honeypot aus angegriffen wurden?

1.7 Fazit

Zum Abschluss dieses Abschnitts sollen die Vor- und Nachteile, die mit einem Einsatz von Honeypots verbunden sind, aufgezeigt werden.

1.7.1 Vorteile von Honeypots

- Die gesammelten Daten haben einen hohen Informationswert
Da es keine autorisierte Nutzung von Honeypots gibt und Verbindungen daher im Allgemeinen einen Angriff darstellen, haben die mit einem Honeypot gesammelten Daten einen hohen Informationswert bzgl. Sicherheitsvorfällen. Dadurch treten auf der einen Seite wenige „False Positives“ auf (Alarmierung, obwohl kein Angriff erfolgt). Auf der anderen Seite ist es durch das relativ geringe Datenaufkommen möglich, alle entstehenden Daten zu sammeln bzw. bei jedem Verbindungsaufbau zu alarmieren. Dadurch gibt es wenige False Negatives (keine Alarmierung, obwohl ein Angriff erfolgt).
- Einfachheit des Konzepts
Im Gegensatz zu anderen Sicherheitsmechanismen, wie z.B. Firewalls oder IDS, sind beim Einsatz von Honeypots keine Filterregeln oder Signaturen notwendig, deren Güte die Wirksamkeit der Maßnahme unmittelbar bestimmt.
- Betrieb ist relativ wenig ressourcenintensiv
Beim Einsatz von Honeypots müssen keine im Rahmen der Geschäftsprozesse anfallenden Daten verarbeitet werden. Daher ist auch beim Einsatz von wenig leistungsfähigen Systemen keine Überlastung durch zu hohes Datenaufkommen zu erwarten.
- Return of Investment
Bei vielen, insbesondere vorbeugenden Maßnahmen ist es schwierig zu erkennen, welchen Wert diese haben. Beim Einsatz von Honeypots kann unmittelbar festgestellt werden, wie viele Angriffe erfolgreich waren bzw. gewesen wären und u.U. auch welches Ziel mit ihnen verfolgt wurde.

1.7.2 Nachteile von Honeypots

- Eingeschränkte Sicht
Findet ein Angriff statt und erfolgt dabei kein Zugriff auf den Honeypot, so besteht keine Möglichkeit, diesen Angriff mithilfe des Honeypots festzustellen.
- Es werden keine verwundbaren Systeme geschützt
Durch den Einsatz von Honeypots wird keinen Verwundbarkeiten entgegengewirkt.
- Beim Einsatz entstehende Risiken
Zunächst muss beachtet werden, dass auch Honeypots grundsätzlich von Angreifern kompromittiert werden und dann zur Durchführung weiterer Angriffe auf Dritte oder

andere Systeme des eigenen Rechnernetzes genutzt werden können. Ferner besteht beim Einsatz von Honey pots grundsätzlich das Risiko, dass diese von Angreifern z.B. durch Fingerprinting³ als solche erkannt werden können.

Es soll zum Abschluss dieses Abschnitts noch einmal explizit darauf hingewiesen werden, dass trotz aller Vorteile des Konzepts von Honey pots das grundsätzliche Problem besteht, dass durch den Einsatz von Honey pots keine unsicheren Systeme geschützt werden!

2 Beispiele für Honey pots

In diesem Abschnitt sollen drei Beispiele für Honey pots vorgestellt werden. Im Unterabschnitt 2.1 soll als ein sehr einfaches Beispiel BackOfficer Friendly von Marcus Ranum, im Unterabschnitt 2.2 Honeyd von Niels Provos und im Unterabschnitt 2.3 das Konzept von Honey nets vorgestellt werden.

2.1 Beispiel 1: BackOfficer Friendly

BackOfficer Friendly (BOF) wurde von Marcus Ranum (beteiligt an der Entwicklung des Network Flight Recorder, einem Netzwerk-IDS) entwickelt und diente ursprünglich zur Erkennung von und zum Antworten auf Verbindungsversuchen des BackOrifice-Client (zu BackOrifice vgl. z.B. [CER98]). BOF kann unter fast allen Windows-Plattformen von Microsoft oder mit einem eingeschränkten Funktionsumfang unter Unix-Plattformen betrieben werden. BOF kann bis zu sieben festgelegte Ports beobachten (vgl. Tabelle 1) und bei entsprechender Einstellung im eingeschränkten Maße auf eingehende Anfragen antworten.

Port	21/TCP	23/TCP	25/TCP	80/TCP	110/TCP	143/TCP	31337/UDP
Dienst	FTP	Telnet	SMTP	HTTP	POP3	IMAP	BackOrifice

Tabelle 1: Von BOF überwachte Dienste bzw. Ports

Die Windows-Version besitzt eine graphische Oberfläche und ist einfach konfigurierbar (vgl. Abbildung 3). In Abbildung 4 wird die Ausgabe des Befehls `'netstat -an'` bei gestarteten BOF aufgezeigt. Dabei kann festgestellt werden, dass bei den von BOF überwachten Ports der Status LISTENING besteht, also auf eingehende Verbindungen gewartet wird. In Abbildung 5 wird die Ausgabe von BOF nach einer HTTP-Anfrage, einer FTP-Anfrage und einem Aufbau einer Telnet-Verbindung aufgezeigt. Bei der Telnet-Verbindung wurde von BOF eine Login-Eingabeaufforderung vorgetäuscht und als Reaktion vom Initiator der Verbindung das Login „root“ mit dem Passwort „secret“ eingegeben.

³Beim Fingerprinting wird versucht, ein System anhand von Besonderheiten z.B. im Netzwerkprotokollstapel zu identifizieren.

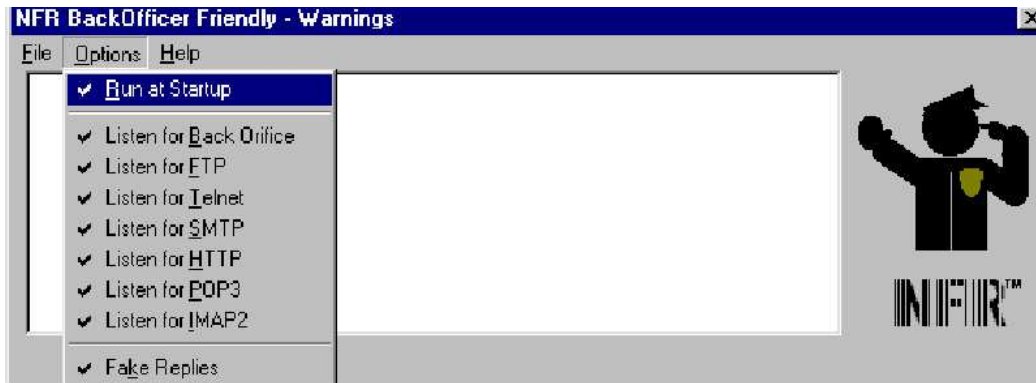


Abbildung 3: Einstellung von Optionen beim BackOfficer Friendly

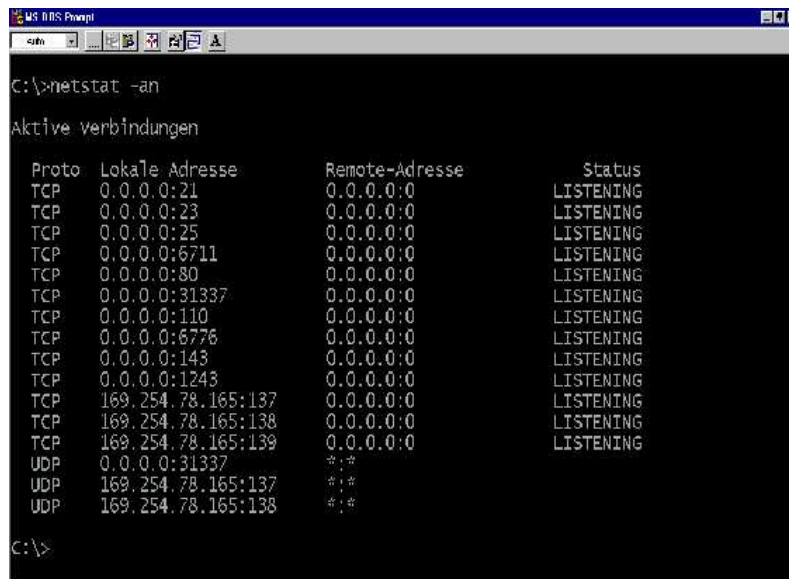


Abbildung 4: Ausgabe von netstat nach der Konfigurierung von BackOfficer Friendly

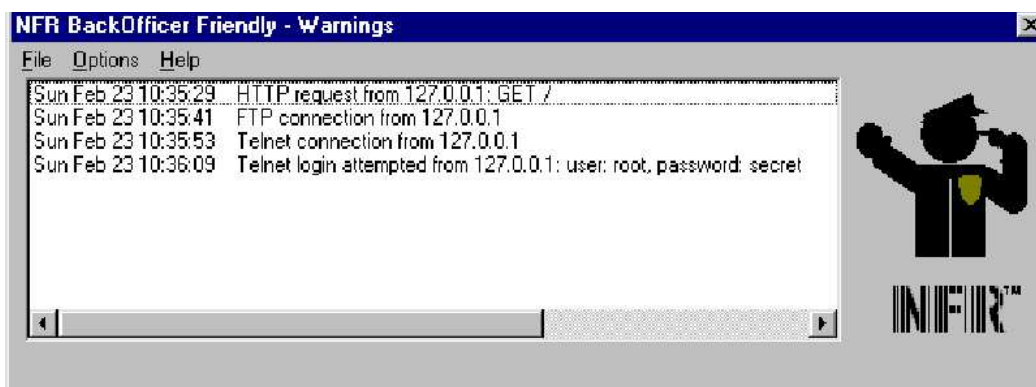


Abbildung 5: Ausgabe des BackOfficer Friendly nach verschiedenen Anfragen

2.2 Beispiel 2: Honeyd

Honeyd wurde von Niels Provos (Universität Michigan) für den Betrieb auf Unix-Plattformen entwickelt (zu Honeyd vgl. [Pro03]). Im Gegensatz zu BOF stellt Honeyd keinen physikalischen, sondern einen virtuellen Honeypot dar. Dies bedeutet, dass auf einem Rechner ein Honeypot installiert wird, der für beliebige virtuelle Systeme mit unterschiedlichen, in einem Subnetz nicht vergebenen IP-Adressen eingehende Netzwerkanfragen bearbeiten kann. Dazu wird von Honeyd lediglich der Netzwerk-Protokollstapel bereitgestellt. Ein emuliertes Betriebssystem o.ä. wird nicht zur Verfügung gestellt, wodurch es einem Angreifer nicht möglich ist, Zugriff auf die virtuellen Systeme zu erlangen.

Honeyd ermöglicht die Beobachtung von Verbindungsversuchen zu allen Ports dieser virtuellen Systeme. Ferner können für TCP, UDP und ICMP emulierte Dienste als Plugins eingebunden bzw. Anfragen durch Proxying an andere Rechner weitergeleitet werden. Honeyd kann so konfiguriert werden, dass für verschiedene virtuelle Systeme die IP-Stapel verschiedener Betriebssysteme emuliert werden, wodurch eine Erkennung des Honeypots erschwert wird (dies gilt z.B. für Fälle, in denen durch ein Honeypot ein anderes Betriebssystem emuliert wird als das, auf dem die Honeypot-Software installiert wurde). Schließlich ist es möglich, mit Honeyd virtuelle Netzwerk-Topologien zu konfigurieren, wobei auch Aspekte wie Verzögerung und Paketverlust emuliert werden.

Eine Voraussetzung zum Betrieb von Honeyd ist, dass die Netzwerk-Anfragen, die an IP-Adressen der zu emulierenden Systeme gerichtet sind, an den Rechner auf dem Honeyd installiert wurde, umgeleitet werden. Dabei bestehen grundsätzlich die folgenden beiden Möglichkeiten:

- Blackholing: Router werden so konfiguriert, dass der Verkehr, für die IP-Adressen eines ganzen Subnetzes zum Rechner auf dem Honeyd installiert ist, umgeleitet wird.
- ARP-Spoofing: in einem Subnetz nicht vergebene IP-Adressen werden statisch durch Konfiguration der Router bzw. dynamisch gespoofed (vgl. Unterabschnitt 2.2.1 für einen Exkurs zum Thema ARP-Spoofing).

2.2.1 Exkurs: ARP-Spoofing

In diesem Exkurs soll in kurzer Form aufgezeigt werden, durch welche Mechanismen es möglich ist, nicht vergebene IP-Adressen dynamisch zu spoofen. Dies soll hier am Beispiel von IP und Ethernet durchgeführt werden. In Abbildung 6 wird die Funktionsweise des Address Resolution Protocol (ARP) anhand eines Beispiels aufgezeigt. Dabei möchte ein Rechner mit der IP-Adresse `192.168.16.79` Daten an einen (sich im selben IP-Subnetz befindlichen) Rechner mit der IP-Adresse `192.168.16.80` senden, kennt die hierfür notwendige Hardware-Adresse⁴ aber nicht. Um diese zu ermitteln, wird eine ARP-Anfrage mit den eigenen Adressinformationen und der IP-Adresse des Ziel-Rechners an alle Rechner in dem Subnetz gesandt (Broadcast). Der Rechner mit der nachgefragten IP-Adresse antwortet dann mit der eigenen Hardware-Adresse direkt an den anfragenden Rechner.

⁴Der Netzwerkverkehr wird bei Ethernet ausschliesslich über die Hardware-Adressen abgewickelt.

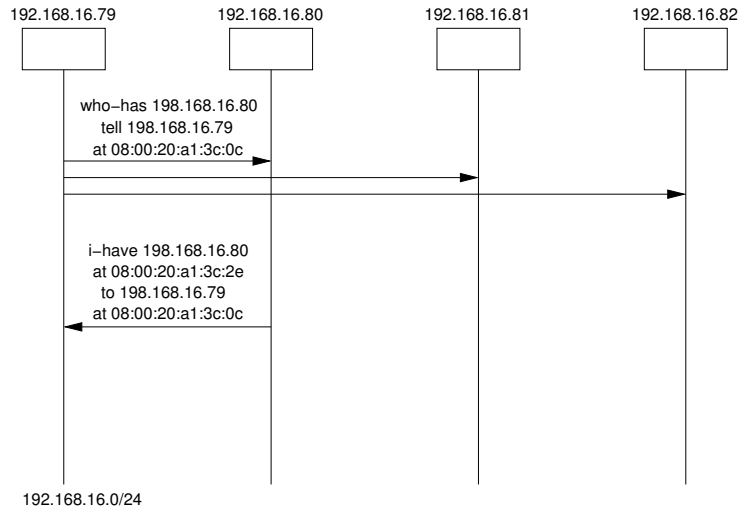


Abbildung 6: Funktionsweise des Address Resolution Protocol

In der Abbildung 7 wird die mit `snoop` abgehörte ARP-Anfrage aufgezeigt. Dabei kann insbesondere gesehen werden, dass im Ethernet Header als Ziel die Broadcast-Adresse und als Typ des Pakets 0806 für ARP angegeben ist. Im ARP-Frame ist der Opcode mit 1 für eine Anfrage angegeben. Die Adressinformation für den Sender ist vollständig angegeben, für das Ziel ist nur die IP-Adresse angegeben, während das Feld für die Hardware-Adresse freigelassen wurde, da diese nachgefragt wird.

In der Abbildung 8 wird die entsprechende ARP-Antwort aufgezeigt. Bei der Antwort ist im Ethernet-Header als Zieladresse nicht die Broadcast-Adresse, sondern die Hardware-Adresse des nachfragenden Rechners angegeben. Im ARP-Frame ist der Opcode 2 für eine

```

ETHER:  ----- Ether Header -----
ETHER:
ETHER:  Packet 15 arrived at 10:22:4.23
ETHER:  Packet size = 60 bytes
ETHER:  Destination = ff:ff:ff:ff:ff:ff, (broadcast)
ETHER:  Source      = 8:0:20:a1:3c:c, Sun
ETHER:  Ethertype = 0806 (ARP)
ETHER:
ARP:    ----- ARP/RARP Frame -----
ARP:
ARP:  Hardware type = 1
ARP:  Protocol type = 0800 (IP)
ARP:  Length of hardware address = 6 bytes
ARP:  Length of protocol address = 4 bytes
ARP:  Opcode 1 (ARP Request)
ARP:  Sender's hardware address = 8:0:20:a1:3c:c
ARP:  Sender's protocol address = 192.168.16.79, elmo
ARP:  Target hardware address = ?
ARP:  Target protocol address = 192.168.16.80, groover
ARP:
  
```

Abbildung 7: ARP Request

```

ETHER: ----- Ether Header -----
ETHER:
ETHER: Packet 16 arrived at 10:22:4.23
ETHER: Packet size = 42 bytes
ETHER: Destination = 8:0:20:a1:3c:c, Sun
ETHER: Source      = 8:0:20:a1:3c:2e, Sun
ETHER: Ethertype = 0806 (ARP)
ETHER:
ARP: ----- ARP/RARP Frame -----
ARP:
ARP: Hardware type = 1
ARP: Protocol type = 0800 (IP)
ARP: Length of hardware address = 6 bytes
ARP: Length of protocol address = 4 bytes
ARP: Opcode 2 (ARP Reply)
ARP: Sender's hardware address = 8:0:20:a1:3c:2e
ARP: Sender's protocol address = 192.168.16.80, groover
ARP: Target hardware address = 8:0:20:a1:3c:c
ARP: Target protocol address = 192.168.16.79, elmo
ARP:

```

Abbildung 8: ARP Reply

Antwort angegeben. Ferner wurden die Adressinformationen des nachfragenden Rechners als Zieladressen eingetragen und die eigene Hardware-Adresse vom angefragten Rechner bei den Adressen des Senders ergänzt.

In Abbildung 9 wird aufgezeigt, wie ein dynamisches ARP-Spoofing von nicht vergebenen IP-Adressen in Broadcast-Netzen funktionieren kann. In diesem Beispiel soll der Rechner mit der IP-Adresse 192.168.16.82 alle nicht vergebenen IP-Adressen spoofen. Die zugrunde liegende Idee ist dabei, dass dieser Rechner auf alle Anfragen nach nicht vergebenen IP-Adressen mit seiner Hardware-Adresse antwortet. Dazu muss der Rechner einerseits feststellen, für welche IP-Adressen ARP-Anfragen gesendet werden, und andererseits überprüfen, ob eine dazugehörige ARP-Antwort folgt. Wird eine nicht vergebene

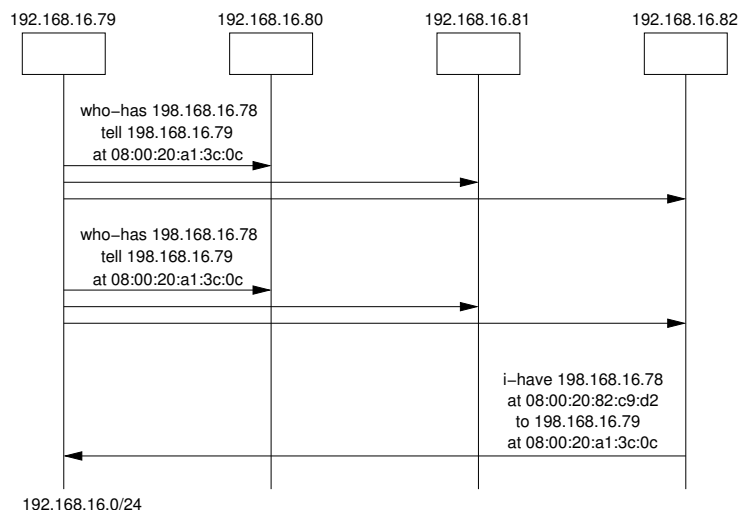


Abbildung 9: ARP-Spoofing in Broadcast-Netzen

IP-Adresse (in diesem Beispiel 192.168.16.78) nachgefragt, so wird diese Anfrage an alle Rechner des Subnetzes gesendet. Das Feststellen der ARP-Anfragen ist daher einfach möglich. Da eine evtl. folgende ARP-Antwort aber direkt an den nachfragenden Rechner adressiert ist, kann diese von anderen Rechnern nur empfangen werden, wenn sämtlicher über das Broadcast-Medium übertragener Datenverkehr abgehört wird. Dazu ist es notwendig, die Schnittstellenkarte in den sog. Promiscuous-Mode umzuschalten.

In Abbildung 10 wird aufgezeigt, wie ein dynamisches ARP-Spoofing von nicht vergebenen IP-Adressen in geschwichten Netzen funktionieren kann. In geschwichten Netzen kann nicht davon ausgegangen werden, dass ARP-Antworten von jedem Rechner empfangen werden können. Aus diesem Grund muss ein Rechner, der ARP-Spoofing von nicht vergebenen Adressen durchführen will, für jede eingehende ARP-Anfrage eine eigene ARP-Anfrage stellen, wobei davon ausgegangen wird, dass nur dann eine ARP-Antwort folgt, wenn auch die ursprüngliche Anfrage beantwortet wurde bzw. in einem Fall, in dem keine Antwort folgt, auch die ursprüngliche Anfrage nicht beantwortet wurde und die IP-Adresse somit nicht vergeben ist.

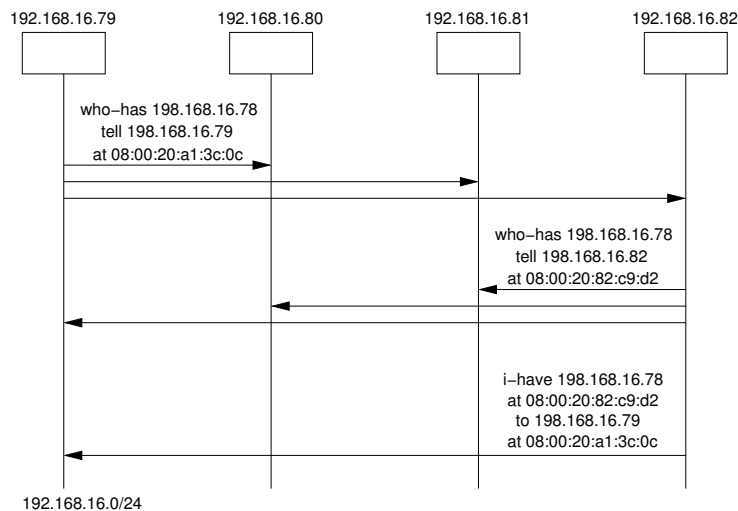


Abbildung 10: ARP-Spoofing in geschwichten Netzen

Beim Einsatz von Honeyd werden von einem zusätzlichen Programm `arpd` unabhängig von Honeyd alle im Subnetz nicht vergebenen IP-Adressen dynamisch gespoofed. In Abbildung 11 wird der Aufruf von `arpd` und `honeyd` für das in diesem Exkurs verwendete Beispiel aufgezeigt. Dabei wird `arpd` für das Ethernet-Interface `hme0` und mit der Aufgabe aufgerufen, alle nicht verwendeten IP-Adressen im Subnetz 192.168.16.0/24 zu spoofen. Dem folgend wird `honeyd` mit der Konfigurationsdatei `/etc/honeyd.conf`

```
arpd -i hme0 192.168.16.0/24
honeyd -f /etc/honeyd.conf 192.168.16.0/24
```

Abbildung 11: Aufruf von `arpd` und `honeyd`

und der Anweisung gestartet, alle eingehenden IP-Pakete für IP-Adressen aus dem Subnetz 192.168.16.0/24 zu bearbeiten.

2.3 Beispiel 3: Honeynets

Honeynets sind Netzwerke aus Standard-Systemen. Sie stellen damit eine Architektur und kein Produkt oder eine Software dar. Sie werden im Allgemeinen nur zur Forschung eingesetzt, da sie ein sehr hohes Maß an Interaktionsmöglichkeiten für Angreifer bieten und daher mit ihrem Einsatz sowohl ein hoher Aufwand als auch ein hohes Risiko verbunden ist. Beim Aufbau eines Honeynets werden insbesondere Anforderungen an die Kontrolle des Datenflusses und die Erfassung von Daten gestellt. Diese werden im Folgenden ausgeführt.

Eine Kontrolle des Datenflusses ist insbesondere wichtig, um zu verhindern, dass Angreifer, die ein System im Honeynet kompromittiert haben, dieses nutzen, um Dritte anzugreifen. Da ein Honeynet im Allgemeinen nicht rund um die Uhr überwacht werden kann und unter Umständen ein sehr schnelles Eingreifen notwendig ist, sollte die Datenflusskontrolle automatisiert und, um der Möglichkeit der Unzulänglichkeit oder des Ausfalls einer Maßnahme Rechnung zu tragen, auf mehreren Ebenen stattfinden. Auf der anderen Seite sollte eine Datenflusskontrolle auf eine Weise geschehen, dass ein Angreifer nicht erkennt, dass seine Aktionen auf diese Weise kontrolliert werden. Ferner sollte eine Alarmgebung erfolgen, die einerseits darauf hinweist, wenn das Honeynet angegriffen wird, andererseits mit höherer Priorität meldet, wenn das Honeynet kompromittiert wurde. Schließlich ist es notwendig, Möglichkeiten für ein manuelles Eingreifen zu haben, um einerseits die automatische Datenflusskontrolle deaktivieren zu können, wenn keine unmittelbare Gefahr für Dritte besteht, und andererseits manuell alle Verbindungen schließen zu können, wenn die automatische Datenflusskontrolle nicht ausreichend ist.

Ferner ist eine umfassende Datenerfassung wichtig. Es wurde bereits aufgezeigt, dass Honeynets vorwiegend in der Forschung eingesetzt werden. Hierbei besteht oft das Ziel, mehr über Angreifer und ihre Vorgehensweisen zu lernen. Um diese Informationen zu erlangen, ist es notwendig, ausreichend viele Daten zu sammeln, um eine spätere Rekonstruktion der Vorgänge zu ermöglichen. Ferner ist es notwendig, diese Daten gegen Modifizierung und Zerstörung zu schützen.

In Abbildung 12 wird ein möglicher Aufbau für ein Honeynet aufgezeigt. Dabei sind im Honeynet vier Rechner (A: Syslog-Server, B: Sparc-Station, C: Linux-PC und D: Windows-NT-PC) angeschlossen. Das Honeynet ist über einen Router und einer Firewall mit dem Internet verbunden. Diese sollen die Datenflusskontrolle sicherstellen. Ferner wurde ein Administrationsnetzwerk eingerichtet, in dem sowohl ein Log/Alert-Server als auch ein Server, auf dem ein Intrusion Detection System konfiguriert wurde, angeschlossen sind. Die gesonderte Verbindung des IDS mit dem Honeynet besteht ausschließlich auf der Schicht 2 des OSI-Modells, es wurde der entsprechenden Netzwerk-Schnittstelle also keine IP-Adresse zugewiesen. Damit soll sichergestellt werden, dass das IDS zwar die auf dem Honeynet übertragenen Daten abhören kann, vom Honeynet aber keine Verbindungen zum IDS aufgebaut werden können (ein Verbindungsaufbau vom Honeynet in das Administrative Netzwerk über die Firewall wird von dieser nicht erlaubt).

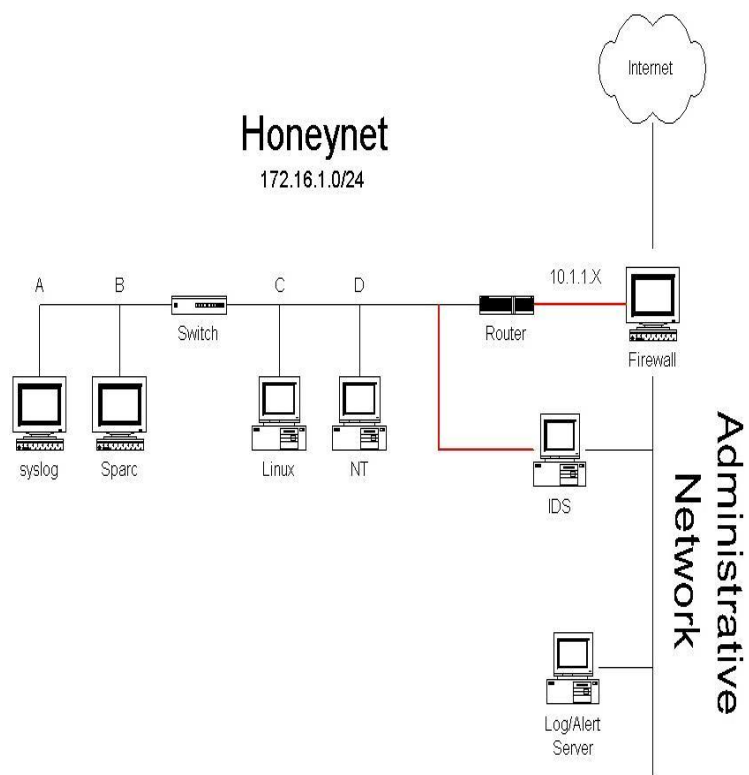


Abbildung 12: Architektur eines Honeynet (vgl. [Spi02, Seite 244])

Literatur

- [CER98] CERT / CC, COMPUTER EMERGENCY RESPONSE TEAM COORDINATION CENTER: *CERT Vulnerability Note VN-98.07 Back Orifice*, October 1998. http://www.cert.org/vul_notes/.
- [GS03] GAITZSCH, MARTIN und TORSTEN SORGER: *Grundlagen von Internet-Protokollen*. Seminar „18.415 Sicherheit in vernetzten Systemen“. Fachbereich Informatik der Universität Hamburg, Wintersemester 2002/2003. http://www.informatik.uni-hamburg.de/RZ/lehre/18.415/seminararbeit/1_Internet-Protokolle.pdf.
- [Kos00] KOSSAKOWSKI, KLAUS-PETER: *Information Technology Incident Response Capabilities*. Doktorarbeit, Universität Hamburg, 2000.
- [Pro03] PROVOS, NIELS: *Honeyd: A Virtual Honeypot Daemon (Extended Abstract)*. In: SCHAUMBURG, ROLF und MARCO THORBRÜGGE (Herausgeber): *10. Workshop Sicherheit in vernetzten Systemen*, Hamburg, 25./26. Februar 2003. DFN CERT. <http://www.citi.umich.edu/u/provos/papers/honeyd-eabstract.pdf>.
- [Sch00] SCHNEIER, BRUCE: *Secrets and Lies*. Wiley, 2000.
- [Spi02] SPITZNER, LANCE: *Honeypots – Tracking Hackers*. Addison-Wesley, 2002.