

# Analyse und Implementierung eines Honeypot-Systems

## STUDIENARBEIT

für die Prüfung zum  
Bachelor of Engineering  
des Studienganges Informationstechnik  
an der  
Dualen Hochschule Baden-Württemberg Karlsruhe  
von  
**Steffen Kurstak und Julian Kühn**

Abgabedatum 1. April 2013

Bearbeitungszeitraum  
Matrikelnummer  
Kurs  
Gutachter der Studienakademie

12 Wochen  
Matrikelnummer  
TINF11B3  
Dr. Ralf Brune

## **Erklärung**

Gemäß §16 (3) der „Studien- und Prüfungsordnung für den Studienbereich Technik“ vom 1.11.2007.

Ich habe die vorliegende Studienarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

---

Ort      Datum

---

Unterschrift

## **Zusammenfassung**

Hier steht eine Kurzzusammenfassung.

# Sperrvermerk

Die vorliegende Arbeit beinhaltet interne vertrauliche Informationen der z.B. Firma EDEKA Handelsgesellschaft Südwest mbH.

Die Weitergabe des Inhaltes der Arbeit und eventuell beiliegender Zeichnungen und Daten im Gesamten oder in Teilen ist grundsätzlich untersagt.

Es dürfen keinerlei Kopien oder Abschriften – auch in digitaler Form - gefertigt werden. Ausnahmen bedürfen der schriftlichen Genehmigung der Firma EDEKA Handelsgesellschaft Südwest mbH in Abstimmung mit dem/der Verfasser/in.

Die vorliegende Arbeit ist nur den Korrektoren sowie ggf. den Mitgliedern des Prüfungsausschusses zugänglich zu machen.

(Stempel)

(Ort, Datum)

(Unterschrift des Betreuers bzw. Ausbildungsleiters)

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>7</b>
1.1	Ziele der Arbeit . . . . .	7
1.2	Stand der Technik . . . . .	7
1.3	Inhalt . . . . .	7
1.4	Umgebung der Arbeit . . . . .	7
<b>2</b>	<b>Honeypots</b>	<b>9</b>
2.1	Definition . . . . .	9
2.2	Ziele . . . . .	9
2.2.1	Research . . . . .	9
2.2.2	Production . . . . .	11
2.3	Rechtliche Grundlage . . . . .	13
2.3.1	Zivil- und Strafrecht . . . . .	13
2.3.2	Datenschutz . . . . .	14
<b>3</b>	<b>Honeynets</b>	<b>15</b>
3.1	Anforderungen eines Honeynets . . . . .	15
3.2	Architektur eines Honeynets . . . . .	16
3.2.1	GenI Honeynet . . . . .	16
3.2.2	GenII und GenIII Honeynets . . . . .	17
<b>4</b>	<b>Implementierungsarten</b>	<b>19</b>
4.1	Typen von Honeypots . . . . .	19
4.1.1	High-Interactive . . . . .	19
4.1.2	Low-Interactive . . . . .	19
4.1.3	Virtualisierung . . . . .	20
4.2	Risiken . . . . .	21
<b>5</b>	<b>Hacking</b>	<b>22</b>
5.1	Definition von Hacking . . . . .	22
5.2	Grundlagen von Netzwerkhacks . . . . .	23
5.2.1	Netzkommunikation . . . . .	23
5.2.2	Netzwerkprotokolle . . . . .	24
5.3	Angriffsszenarien . . . . .	26
5.4	Datenanalyse . . . . .	26
5.4.1	Dateien sichern . . . . .	28
5.4.2	Netzwerkanalyse . . . . .	30

<b>6</b>	<b>Tools und Anwendungen</b>	<b>31</b>
6.1	Honeypot und Honeynet Tools . . . . .	31
6.1.1	Sebek . . . . .	31
6.1.2	Intrusion Detection System . . . . .	33
6.2	Honeypot Software . . . . .	34
6.2.1	Honeyd . . . . .	35
6.2.2	Tiny oder Tarpit . . . . .	36
6.2.3	Roo . . . . .	36
6.2.4	High-Interactive? . . . . .	36
6.3	Anwendungen zur Datenanalyse . . . . .	36
6.3.1	Honey View . . . . .	36
6.3.2	... . . . .	36
<b>7</b>	<b>Implementierung des Honeypots</b>	<b>37</b>
7.1	Systemspezifikationen . . . . .	37
7.2	Honeyd Implementierung . . . . .	38
7.2.1	Installation . . . . .	38
7.2.2	Konfiguration und Inbetriebnahme . . . . .	38
7.2.3	Datenanalyse . . . . .	40
7.3	Kommerzielle Implementierung . . . . .	40
7.3.1	Installation . . . . .	40
7.3.2	Konfiguration . . . . .	40
7.3.3	Datenanalyse . . . . .	40
7.4	Open Source Implementierung . . . . .	40
7.4.1	Installation . . . . .	40
7.4.2	Konfiguration . . . . .	40
7.4.3	Datenanalyse . . . . .	40
<b>8</b>	<b>Auswertung der gesammelten Daten</b>	<b>41</b>
<b>9</b>	<b>Fazit</b>	<b>42</b>
	<b>Literaturverzeichnis</b>	<b>43</b>

# Abbildungsverzeichnis

2.1	Research Honeypot - Variante I . . . . .	10
2.2	Research Honeypot - Variante II . . . . .	10
2.3	Production Honeypot . . . . .	11
3.1	GenI Honeynet . . . . .	17
3.2	GenII Honeynet . . . . .	18
5.1	Kommunikation über das OSI-Schichtenmodell . . . . .	24
5.2	Drei-Wege-Handshake mit TCP . . . . .	25
5.3	Einstellungen für RAM-Seicher Dmp-File . . . . .	29
5.4	Erstellung eines Speicherabbilds mit dd . . . . .	30
6.1	Abfangen der Daten durch Sebek [4] . . . . .	32
6.2	Zugriff auf Netzwerk Driver [4] . . . . .	32
6.3	Ablauf einer Sebek-Überwachung [4] . . . . .	33
6.4	Beispielkonfiguration einer Honeyd Config-Datei . . . . .	35
7.1	Erst Testkonfiguration mit Honeyd . . . . .	39
7.2	Zweite Testkonfiguration mit Honeyd . . . . .	39

# Tabellenverzeichnis



# Abkürzungsverzeichnis

<b>DMZ</b>	Demilitarisierte Zone
<b>IDS</b>	Intrusion Detection System

# Kapitel 1

## Einleitung

### 1.1 Ziele der Arbeit

Im Rahmen dieser Arbeit soll ein Honeypot implementiert werden und über das Internet als potentielltes Angriffsziel für Hacker dienen. Anschließend sollen die Angriffe ausgewertet und diverse Statistiken erstellt werden. Zudem sollen ein oder zwei Angriffe, mit Hilfe von Tools, detailliert aufgearbeitet werden. Für die Implementierung des Honeypots ist auch ein Softwarevergleich nötig, um eine passende Software zu finden. Möglich ist auch der Einsatz eines kommerziellen Honeypots. Für die Analyse der Angriffe ist ebenfalls der Einsatz mehrere Tools möglich und erwünscht.

### 1.2 Stand der Technik

Zwischen 2000 und 2006 war der Honeypot am populärsten. Neben Firmen haben auch Privatanutzer mit Hilfe der Honeypots angefangen Hacker zu studieren. Da ein Honeypot allerdings gerade in der Industrie nur Geld kostet und oftmals keinen messbaren Mehrwert bringt, ist der Einsatz von Honeypots zurückgegangen. Zudem sind auch die Communitys zurückgegangen und der damit einhergehend Support von OpenSource Projekten. Die Honeyport-Software Honeyd ist hierfür ein passendes Beispiel, da das letzte Release der Software bereits aus dem Jahre 2007 ist. Somit ist bei dem Einsatz von OpenSource Produkten in dieser Studienarbeit, mit Problemen bezüglich Dokumentation und Support zu rechnen.

### 1.3 Inhalt

### 1.4 Umgebung der Arbeit

Die Studienarbeit „Analyse und Implementierung eines Honeypots-Systems“ wird im Rahmen des Bachelor Studiengangs Informationstechnik an der DHBW-Karlsruhe durchgeführt. Ausgearbeitet wird das Thema von den Studenten Julian Kühn und Steffen Kurstak. Für die Durchführung dieser Arbeit steht ein Server mit zwei Netzwerk-Ports zur Verfügung sowie zwei öffentliche IP-Adressen der DHBW-Karlsruhe. Das Betriebssystem des Servers wird voraussichtlich auf Linux basieren.

Neben dem Server wird auch der Kauf eines vorinstallierten Honeypots angestrebt, welcher damit ebenfalls zur Verfügung stehen wird

Die Dokumentation dieser Arbeit wird in  $\text{\LaTeX}$  geschrieben. Für einen vereinfachten Umgang mit  $\text{\LaTeX}$  wird die Software TeXstudio eingesetzt. Ein ausschlaggebendes Kriterium für den Einsatz von  $\text{\LaTeX}$ , ist der unkomplizierte Umgang bei einem parallelen Arbeiten der Autoren. Durch die Untergliederung in einzelne Dateien werden Konflikte vermieden. Mit Hilfe des Repositorys GitHub wird die Versionsverwaltung vorgenommen.

# Kapitel 2

## Honeypots

### 2.1 Definition

Einleitend und noch oberflächlich erklärt ist die Aufgabe eines Honeypots, einen Angreifer von einem bedeutsamen Ziel abzulenken. In der Informationstechnik erfüllen Honeypots, je nach Einsatzgebiet und Ansicht des Entwicklers, verschiedene Aufgaben.

Einige Unternehmen sehen den Einsatz eines Honeypots als ein Intrusion Detection System. Für andere ist es nur eine Täuschung für Hacker, welche dadurch von den produktiven Systemen abgelenkt werden sollen. Ein weiterer Nutzen eines Honeypots ist es, die angreifenden Hacker analysieren zu können und somit neue Vorgehensweisen und Trends der Hacker zu erkennen. Diese Informationen sind vor allem für Sicherheitsfirmen relevant, da sie dabei neue Viren, Trojaner oder weitere Schadsoftware erkennen können.

Trotz dieser verschiedenen Einsatzmöglichkeiten hat Lance Spitzner eine passende Definition gefunden: „A honeypot is security resource whose value lies in being probed, attacked, or compromised.“ [6]

Der Honeypot ist nach Spitzner eine Sicherheits-Ressource deren Wert darin liegt, erforscht, attackiert und kompromittiert zu werden. Durch diese Definition ist offen gelassen, welchen Nutzen der Anwender daraus zieht. Mit dieser allgemein gehaltenen Definitionen können nun Ziele eines Honeypots im nächsten Kapitel beschrieben werden.

### 2.2 Ziele

Die Ziele eines Honeypots lassen sich in zwei Kategorien einteilen. Das sind zum einen die „Research Honeypots“ und zum anderen die „Production Honeypots“.

Die unterschiedlichen Ziele dieser Honeypots werden in den folgenden Kapiteln beschrieben.

#### 2.2.1 Research

Ein Research Honeypot wird für die Untersuchung und Forschung der Angriffe eingesetzt. Der Honeypot wird meist getrennt von allen Produktivsystemen aufgebaut. In Abb. 2.1 wird der Honeypot vor die Firewall des Produktivnetzes installiert. Dadurch wird versucht mögliche

Angreifer weit weg von dem Produktiv-Systemen zu halten.

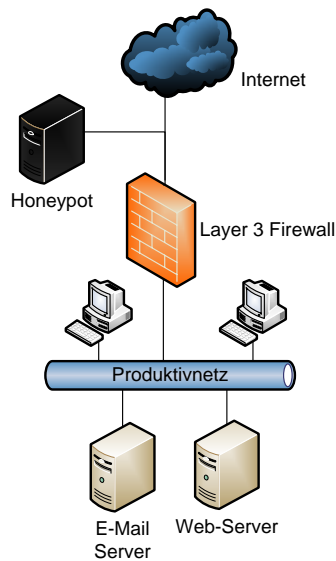


Abbildung 2.1: Research Honeypot - Variante I

Eine weitere Möglichkeit eines solchen Aufbaus ist in Abb. 2.2 dargestellt. Hier wird der Honeypot durch die Layer 3 Firewall von dem Produktivnetz getrennt. Bei diesem Aufbau muss genau auf die Firewall-Einstellungen geachtet werden, um den Angreifern nicht ungewollt einen Sprungserver zu den produktiven Systemen zur Verfügung zu stellen.

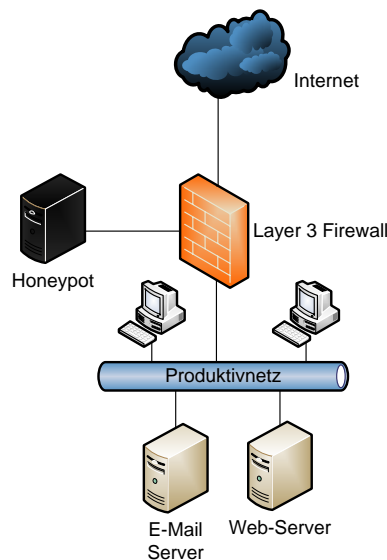


Abbildung 2.2: Research Honeypot - Variante II

Es geht hier primär um die Erkennung von neuen Würmern, Trojanern und weiterer Schadsoftware. Neben der Schadsoftware werden auch Angriffe von Hackern sowie der nicht so qualifizierten „Skript-Kiddies“ untersucht.

Bei jedem Angriff werden Daten und Informationen über den Angreifer gesammelt. Dabei lässt sich häufig ermitteln wer, womit und evtl. auch warum er angegriffen hat. Setzt man meh-

rere Honeypots an verschiedenen Standorten ein, lassen sich mit den gesammelten Daten neue Trends der Angreifer erkennen. Durch diese Informationen versuchen Unternehmen Angriffe auch vorhersagen zu können und die Vorgehensweise der Angreifer besser zu verstehen. Mit den gesammelten Daten können auch neue Tools, mit denen vorallem Skript-Kiddies arbeiten, aufgedeckt werden.

Um die Daten besser auswerten zu können teilen verschiedene Organisationen ihre gesammelten Daten und Erkenntnisse mit anderen Unternehmen. Die große Anzahl der Daten ermöglicht eine bessere Analyse sowie eine genauere Trend-Erkennung.

Research Honeypots dienen somit nicht direkt zur Risiko-Minimierung. Vielmehr helfen die Erkenntnisse, die Angriffs-Prävention und Erkennung zu verbessern.

### 2.2.2 Production

Im Gegensatz zu einem Research Honeypot wird ein Production Honeypot, wie der Name schon sagt, in einem produktiven Umfeld eingesetzt. Eine mögliche Implementierung ist in Abb. 2.3 zu sehen. Ziel eines Production Honeypots ist es, das Netzwerk sicherer zu machen. Er soll dafür sorgen die Infrastruktur vor Angriffen zu schützen und die Aufmerksamkeit der Angreifer auf sich zu lenken.

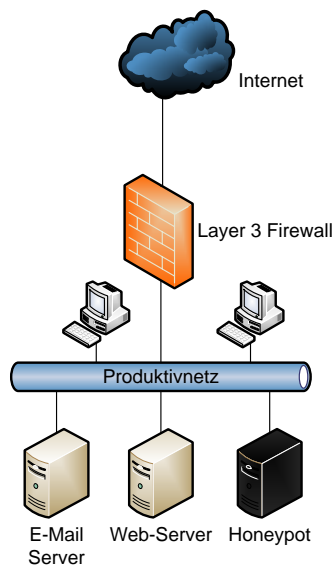


Abbildung 2.3: Production Honeypot

Um die Wirkung des Honeypots genauer zu beschreiben, werden im folgenden die drei Kategorien von „Sicherheit“, welche von —Verweiß— definiert wurden, aus Sicht des Honeypots erläutert.

- **Verhütung (Prevention)**

Um ein Netzwerk vor einem Angriff zu schützen werden häufig Firewalls und andere Sicherheits-Systeme eingesetzt. Ein Honeypot wird jedoch selten eingesetzt um sich vorbeugend vor Angriffen zu schützen. Sobald ein Honeypot aktiv wird, ist der Angreifer

bereits in das Netzwerk bzw. System eingedrungen. Wenn der Honeypot falsch konfiguriert ist, wird er sogar zu einer Gefahr für das eigene Netzwerk und zu einer Eingangstür für Angreifer. Trotz allem gibt es auch Punkte, die für einen Honeypot als vorbeugende Sicherheitsmaßnahme sprechen.

Zum einen werden die Angreifer verunsichert, da sie über die Existenz der Honeypots wissen. Bricht ein Angreifer in ein System ein, muss er sicher immer bewusst sein, dass er möglicherweise gerade in eine Falle gelockt wurde. Diese Tatsache könnte Angreifer verunsichern und von größeren Netzwerken fernhalten.

Des Weiteren kann argumentiert werden, dass ein Angriff auf einen Honeypot den Schaden von produktiven Systemen vorbeugend abwehrt. Der Honeypot beschäftigt den Angreifer und führt ihn meist für eine längere Zeit in die Irre. In dieser Zeit kann der Angreifer an keine wichtigen Daten gelangen und der eigentliche Angriff auf die produktiven Systeme wird dadurch verhindert. Dieser Aspekt zur Sicherung produktiver Daten kann ebenfalls als ein vorbeugender Schutz gesehen werden.

- **Erkennung (Detection)**

Im Gegensatz zu der Angriffs-Verhütung ist der Honeypot für die Erkennung von Angriffen besser geeignet. In einer Demilitarisierten Zone (DMZ) eines Netzwerks sind Systeme, welche meist mit dem Internet verbunden sind. Durch diese Verbindung ist das Risiko eines Angriffs an dieser Stelle erhöht. Wird hier ein Honeypot, wie es in Abb. 2.3 gezeigt, implementiert, kann dieser als eine Art Alarmanlage dienen. Da ein Honeypot nicht aktiv im Netzwerk arbeitet und somit auch keine Systeme eine Verbindung mit dem Honeypot benötigen, sollte es keine Verbindungsversuche geben. Bei Verbindungsversuchen auf Ports welche typischerweise für Internetdienste verwendet wird, handelt es sich meist um einen Port-Scan eines potenziellen Angreifers. Dieser versucht damit herauszufinden, um welches System es sich handelt und welche Dienste es zur Verfügung stellt. Oftmals werden dabei die Ports 25 für E-Mail Dienst und auch Port 80 für HTTP überprüft.

Alarmierend sind Zugriffe von Systemen, welche sich in der selben DMZ befinden wie der Honeypot. Ein solcher Zugriff bedeutet meist, dass ein produktiv System von einem Angreifer kompromittiert wurde. In einem solchen Fall sollten zwingend alle Systeme überprüft werden, da der Angreifer möglicherweise weitere Server der DMZ kompromittiert hat.

- **Reaktion (Response)**

Wenn ein System erfolgreich angegriffen wurde, ist es wichtig herauszufinden, wie der Angreifer vorgegangen ist. Hierfür muss das kompromittierte System untersucht werden. Wichtig sind dabei die MAC (Modify, Access, Change) Zeiten der Dateien. Mit den Zeitstempeln lässt sich nachvollziehen, welche Dateien der Angreifer manipuliert hat. Auf einem produktiven Webserver wäre eine solche Untersuchung sehr schwierig, da hier meist ein großes Datenvolumen anfällt und dementsprechend auch viele Schreib- und Lese-Zugriffe auf Dateien erfolgen. Der Honeypot hat den klaren Vorteil, dass Veränderungen einzelner Dateien auf den Angreifer zurückzuführen sind. Das macht eine Untersuchung

des Angriffs wesentlich leichter und schneller. Bei einer genauen Untersuchung kann festgestellt werden, wie der Hacker das System kompromittiert hat und womöglich auch welche Ziele er damit verfolgt hat. Mit diesen Informationen hat der System-Administrator die Möglichkeit, weitere Systeme in diesem Netz vor solchen Angriffen zu schützen. Zudem können mit diesen Informationen die übrigen Systeme auf eine Kompromittierung untersucht werden.

Im Bereich der Reaktion auf einen Angriff, kann ein Honeypot sehr nützlich sein. Mit Hilfe eines Honeypots können andere Systeme gegen Angriffe geschützt werden und bereits kompromittierte Systeme aufgedeckt werden. Dabei ist jedoch immer zu beachten, sollte ein Angreifer den Honeypot auslassen und nur produktive Systeme attackieren, ist der Honeypot keine Unterstützung. Der Administrator ist darauf angewiesen, dass der Honeypot auch in den Angriff mit einbezogen wird.

## 2.3 Rechtliche Grundlage

Der Betreiber eines Honeypots stellt bewusst ein Angriffsziel für blackhats zur Verfügung. Nutzt ein blackhat den Zugriff auf einen Honeypot, um von dort aus weitere Straftaten zu begehen, kommen dabei auch Rechtliche Fragen auf. Kann dem Betreiber des Honeypots fahrlässiges handeln oder gar eine Beihilfe vorgeworfen werden?

Hinzu kommt die Frage, mit der sich auch schon [2] beschäftigt hat, dürfen die Daten der blackhats verwendet werden, ohne das Wissen gerade an einem Experiment teil zu nehmen?

In den folgenden Kapiteln werden diese Fragestellungen kurz beleuchtet.

### 2.3.1 Zivil- und Strafrecht

Sollte der Honeypot, von einem blackhat, für einen Übergriff auf einen unbeteiligten Dritten genutzt werden, könnte ihm Strafrechtlich eine Beihilfe vorgeworfen werden. Um eine Beihilfe einer Straftat zu leisten, erklärt [2], muss eine aktive Hilfeleistung seitens des Honeypots-Betreibers nachgewiesen werden.

Im Fall der Honeypots ist dieser Nachweis nicht möglich, da es gezielt Maßnahmen zum Schutz vor Übergriffen durch den Honeypot auf fremde Systeme gibt. Der Sicherheitsstandard eines Honeypots ist oft nicht geringer als der anderer Systeme. Zudem kann auch nicht behauptet werden, dass sich ein Honeypot einem blackhat explizit aufdrängt oder ihn zu einer Straftat verleitet.

Zivilrechtlich stellt sich nun die Frage, ob der Betreiber für einen aufgetretenen Schaden Haftbar gemacht werden kann. [2] beantwortet diese Frage ebenfalls mit nein. Um den Betreiber Haftbar zu machen, muss der Honeypot aktiv anderen Systemen schaden zufügen. Da ein solcher Sachverhalt nicht vorliegt, bleibt die Möglichkeit einer Unterlassung der Absicherung. Diese Anschuldigung stützt sich auf der sogenannten Verkehrssicherungspflicht. Hierbei geht es im groben darum, dass der Betreiber die Gefahren, die bewusst durch den Honeypot geschaffen werden, mit angemessenen Vorkehrungen gering hält. Somit soll der schaden Dritter vermieden werden.



Auch hier ist zu sagen, dass Honeypots ausreichend überwacht und gesichert werden, wodurch eine Unterlassung der Absicherung ausgeschlossen werden kann.

#### 2.3.2 Datenschutz

Ist ein Angreifer auf einem Honeypot zu Gange, wird er oftmals von Tools überwacht und aufgezeichnet. Diese Aufzeichnungen stehen dem Betreiber zur Verfügung und dass ohne das Wissen des Angreifers. Dornseif zweifelt dabei die Unwissenheit des Angreifers an und argumentiert damit, dass die blackhat-Community über die Existenz von Honeypots informiert sind. Somit nehmen sie bei ihren Angriffen eine Aufzeichnung und Überwachung in Kauf.

Hinzu kommt, dass bei den Angriffen von blackhats keine personenbezogenen Daten in die Hände des Betreibers fallen. Mit dieser Argumentation kann auch der Datenschutzrechtliche Hintergrund entkräftet werden.

# Kapitel 3

## Honeynets

Honeynets bestehen aus mehreren virtuellen oder physikalischen Honeypots, die ein komplettes Netzwerk darstellen. Jede Netzwerkkomponente und jeder Server kann dabei als Honeypot angesehen werden. So besteht die Möglichkeit ein komplettes Produktivnetz nachzustellen, welches dem Hacker den Eindruck vermittelt, in ein produktiv verwendetes Firmennetz eingedrungen zu sein. [?]

### 3.1 Anforderungen eines Honeynets

Die Anforderungen eines Honeynets kann grob in drei Kategorien unterteilt werden:

**Data Control:** Datenkontrolle bedeutet, dass der Betreiber eines Honeypots oder Honeynets die Kontrolle der ein- und ausgehenden Datenpakete behält. Gelingt es dem Hacker dem Honeynetbetreiber diese Kontrolle zu entreißen, besteht die Möglichkeit, dass der Hacker einen Angriff auf das Produktivnetz startet. Die Datenkontrolle muss für den Angreifer unsichtbar sein. Zu strenge Sicherheitsvorkehrungen können den Hacker jedoch verunsichern und ihn auf den Honeypot aufmerksam machen (z.B. ausgehende Verbindungen blockieren). Jedoch kann z.B. eine Limitierung der ausgehenden Verbindungen ein gutes Mittel gegen den Verlust der Datenkontrolle sein.

**Data Capture:** Die Informationen, die ein Hacker während eines Angriffs hinterlässt, müssen möglichst reichhaltig und unauffällig dokumentiert werden. Für die Datensammlung gibt es verschiedene Möglichkeiten die Aktionen eines Hackers aufzuzeichnen.

- Packet Sniffing: Zum aufzeichnen des kompletten Netzverkehrs (ein- und ausgehende Pakete).
- Keystroke Logging: Das aufzeichnen der vom Hacker ausgeführten Tastenanschläge.
- Snapshot Software: Vergleicht Betriebssystem vor- und nach der Kompromittierung und hält die Änderungen fest.
- Log-Dateien wie z.B. Log-Daten eines Netzwerkgerätes wie Switch und Router

Wichtig bei diesen Programmen ist es, dass sie für den Hacker nicht ersichtlich sein dürfen. Findet der Angreifer eines dieser Programme ist dieser alarmiert und wird die Flucht ergreifen.

**Data Collection:** Bei einem verteilten System wie z.B. bei einem Honeynet muss es eine zentrale Stelle geben in der die Informationen gesammelt und gespeichert werden. Daten werden nie direkt auf einem Honeypot protokolliert, sondern an ein zentrales System übertragen. Wichtig hierbei ist es, dass die Daten sicher und unverändert an das System übertragen werden. Der Angreifer soll nicht die Möglichkeit haben einen Angriff zu vertuschen, oder gar das System selbst anzugreifen.

**Data Analysis:** Die gewonnenen Informationen müssen dem Zweck entsprechend ausgewertet werden. Je nach Ziel des Honeypots müssen z.B. Gegenmaßnahmen getroffen, oder aus dem gelernten Wissen Schlüsse gezogen werden, um in Zukunft eine Kompromittierung eines Produktivsystems zu verhindern.

## 3.2 Architektur eines Honeynets

Es gibt zwei verschieden Arten von Architekturen die sich im Laufe der Zeit durchgesetzt haben. Diese werden in Generation I und Generation II Honeynets (Abk. GenI und GenII) unterteilt.

### 3.2.1 GenI Honeynet

Bei einem GenI Honeypot wird das gesamte Netz durch eine Firewall in drei Teile unterteilt. Der erste Teil ist das Produktivnetz indem sich das zentrale Management System befindet. Der zweite Teil ist das Internet, welcher das Zugangsmedium des Angreifers darstellt. Der dritte und letzte Teil ist das Honeynet.

Der Prozess der Datensammlung beginnt bereits mit passieren der Firewall. Dort können Informationen wie die verwendeten Protokolle, Zeitstempel, IP-Adressen und Ports gesammelt werden. Außerdem wird hier kontrolliert wie oft der Angreifer eine Verbindung eingehen kann (Data Control). Wie viele Versuche zugelassen werden hängt vom Verwendungszweck des Honeynets ab. Der Router zwischen Honeynet und Firewall unterstützt diese auf zwei verschiedene weisen. Zum Einem versteckt er die Firewall vor dem Hacker. Der Angreifer denkt, er greift auf einen produktiven Router zu. Zum Anderen unterstützt er die Firewall in Sachen Zugriffskontrolle. So kann ein Single-Point-of-Failure vermieden werden.

Ein IDS-System steht nun noch zwischen dem Angreifer und den Honeypots. Dieses ist meist über einen Switch (oder wie in Abb. 3.1 mit einem Router) mit dem gesamten Honeynet verbunden. Dort werden alle Netzwerkaktivitäten protokolliert und bei bestimmten Angriffsmustern gegebenenfalls ein Alarm ausgelöst.

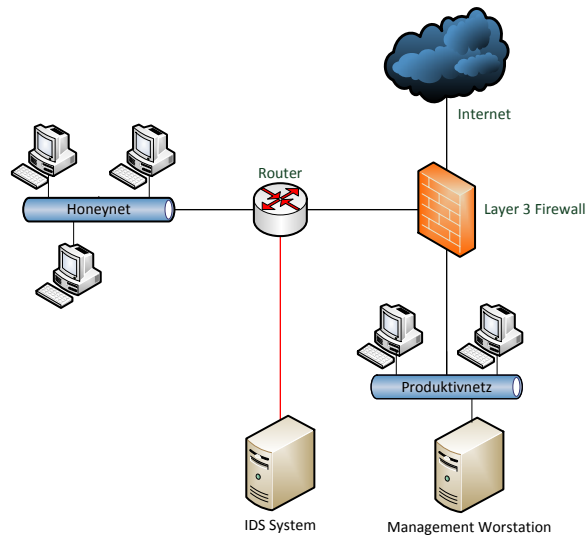


Abbildung 3.1: GenI HoneyNet

#### 3.2.2 GenII und GenIII Honeynets

Die zweite Generation von Honeynets verwendet ein Gateway, welches die Funktionen der in GenI verwendeten Komponenten enthält, und diese noch weiter ergänzt. Dieses Gateway (meist Honeywall genannt) besteht nicht wie in GenI aus einem Layer-3 Router sondern aus einer Layer-2 Bridge. Dies verhindert z.B. dass der Angreifer über die TTL-Zähler eines IP-Paketes den Router erkennen würde. Alle zuvor genannten Anforderungen werden in der Honeywall erfüllt. Für die Datenkontrolle werden hier wie in GenI die Verbindungen limitiert (oft mit dem Programm IPTables), um so DOS-Angriffe zu vermeiden und die Kontrolle über die Verbindungen zu erhalten. Zusätzlich bietet sich nun auch die Möglichkeit Zugriffe die über bestimmte Protokolle einzeln zu limitieren.

Ein IDS oder IPS verhindert weiterhin dass der Hacker vom Honeynet aus einen Angriff auf das Produktivsystem oder in das Internet starten kann. Die Zugriffskontrolle und das IDS sammeln wie beim GenI Honeynet die Daten. In Abb. 3.2 befindet sich ein Beispiel einer GenII Honeynet Architektur.

Ende 2004 wurde die vorerst letzte Generation von Honeynets vorgestellt. Ein GenIII Honeynet besitzt die selbe Netzwerkarchitektur wie dessen Vorgänger, behebt jedoch einige dessen Schwachstellen. Bei dem Versuch einen Honeynet Standard und eine Möglichkeit zu finden, ein Honeynet leichter zu Erstellen, veröffentlichte das Honeynet-Project ([www.honeynet-project.org](http://www.honeynet-project.org)) eine CD, die alle Anforderungen eines Honeynets beinhaltet. Diese CD (*Roo* genannt) wird als dritte Generation angesehen. Die aktuelle Version (1.4, Stand: 2014) bietet neben der verbesserten Datensammlung, eine grafische Web-Oberfläche zur Datenanalyse, und unterstützt weitere Tools wie Sebek und Hflow2 (Siehe Kapitel Tools).

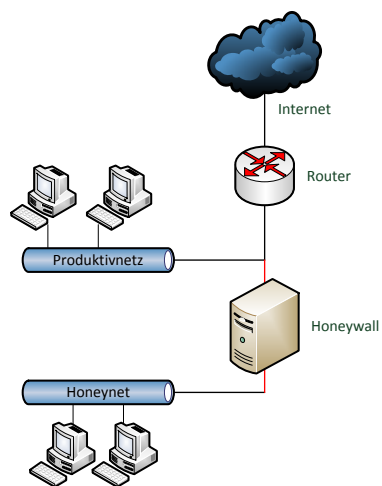


Abbildung 3.2: GenII Honeynet

# Kapitel 4

## Implementierungsarten

Wie viele Informationen ein Honeypot sammeln kann, hängt von seiner Implementierungsart ab. Je nach Zweck und Aufwand kann zwischen den in diesem Kapitel genannten Arten unterschieden werden.

### 4.1 Typen von Honeypots

Die Funktionsweise von Honeypots ist von der Implementierungsart abhängig. Dabei spielt der gewünschte Grad der Interaktion, sowie der Zweck des Honeypots eine Rolle (Research oder Production). Je nach Anwendungsgebiet wird die dafür nützlichste Implementierung verwendet. Grundsätzlich wird dabei zwischen folgenden Implementierungsmöglichkeiten unterschieden:

#### 4.1.1 High-Interactive

Bei High-Interactive Honeypots werden alle Dienste und Software "real" verwendet. Alle Anwendungen, Ports und alle Schichten des OSI-Modells werden dem Angreifer zur Verfügung gestellt. Diese Variante wird häufig dazu verwendet, um manuellen Hackern das Gefühl zu geben, in einen echten Server oder Computer eingedrungen zu sein. Die Überwachungs-routinen befinden sich dabei meist im Betriebssystem Kernel, um ein Entdecken des Honeypots zu erschweren. Der Vorteil dieser Methode ist, dass der Hacker alle Möglichkeiten besitzt, sein Ziel zu erreichen. Somit kann das meiste forensische Material gesammelt werden. Jedoch hat diese vollkommene Freiheit des Hackers den Nachteil, dass es für ihn leichter ist, in das Produk-tivsystem einzudringen.

Diese Art der Implementierung findet sich meist bei der Anwendung von Honeynets (vgl. Kapitel 2). Der Implementierungsaufwand bei High-Interaction Honeypots ist wesentlich höher als bei anderen Implementierungsarten, da nach einer erfolgreichen Kompromittierung das komplette System neu aufgesetzt werden muss.

#### 4.1.2 Low-Interactive

Ein Low-Interaction Honeypot bietet im Vergleich zu einem High-Interactive Honeypot keine "reale" Umgebung. Meist emuliert ein Programm (z.B. honeyd) einen oder mehrere Dienste die dem Hacker einen eher eingeschränkten Zugang auf ein System gewähren. So wird meist

auch nur ein Zugriff auf die Netzwerkschicht des OSI-Modells und auf gewünschte Ports zugelassen. Da es sich bei dieser Implementierungsart um eine Emulation bestimmter Angriffsschnittstellen handelt muss nach einer versuchten Kompromittierung auch das Betriebssystem nicht vollständig neu aufgesetzt werden. Dadurch wird der Wartungsaufwand im Vergleich zu High-Interactive Honeypots erheblich vermindert.

Bei einem versuchten Angriff durch einen Wurm oder einem Hacker wird z.B. nur eine Verbindungsanfrage zugelassen, jedoch nicht akzeptiert. Dadurch kann z.B. über einem Portscanner die Herkunft der Verbindungsanfrage ermittelt werden. Das forensische Material das bei Low-Interactive Honeypots gesammelt werden kann ist deswegen im Vergleich zu High-Interactive Honeypots eingeschränkt.

#### 4.1.3 Virtualisierung

Honeypots können wie normale Betriebssysteme über eine Virtualisierungssoftware auch virtualisiert werden. Dabei werden alle Dienste, Ports und OSI-Schichten emuliert. Zur Realisierung dieser Implementierungsart gibt es verschieden Möglichkeiten.

##### **Virtual Machine Honeypots**

Hierbei werden reale Betriebssysteme über eine Virtualisierungssoftware (z.B. VMWare oder VirtualBox) auf einem Rechner installiert. So ist es Möglich, auf einem physikalischen Rechner oder Server mehrere Honeypots in Betrieb zu halten. Der Vorteil dieser Methode ist, dass dem Hacker ein vollständiges OS mit allen Diensten, Netzwerkschichten und Anwendungen zur Verfügung gestellt wird, und trotzdem sich der Implementierungsaufwand in Grenzen hält. Wird ein System kompromittiert, so kann es ohne viel Aufwand wiederhergestellt werden. Durch die Möglichkeit mehrerer Virtuelle Honeypots auf einem physikalischen Server zu betreiben ist es Möglich, ein komplettes virtuelles Honeynet auf einem Server zu erstellen.

##### **Emulated Honeypots**

Um einen Honeypot zu emulieren wird meist eine Software verwendet, die die grundlegenden Funktionen eines Betriebssystems oder Dienstes emuliert. Je nach Konfiguration kann ein emulierter Honeypot auch alle Dienste, OSI-Netzwerkschichten und Applikationen zur Verfügung stellen. Die Wiederherstellung des Systems ist dabei ähnlich einfach wie die der Virtuellen Honeypots. Durch die eingeschränkten Möglichkeiten der Emulation von verschiedenen Diensten oder Systemen werden emulierte Honeypots als Low-Interactive eingestuft. Emulated Honeypots eignen sich besonders für Honeypot Einsteiger, da sie meist einfach zu Konfigurieren sind und so schnell erste, zumeist automatisierter, Kompromittierungsversuche entdecken.

Viele Tools wie z.B. honeyd arbeiten mit diesem Prinzip.

### 4.2 Risiken

Da ein Honeypot oder Honeynet von echten Angreifern attackiert wird, muss damit gerechnet werden das der Angreifer eventuell versucht über den Honeypot in das Produktive Netz zu gelangen. Es muss dringend vermieden werden das eine solche Möglichkeit besteht, indem das/die Hostsystem/e genügend abgesichert wird. Außerdem muss vermieden werden, dass der Angreifer weiter Angriffe über das Internet vom Kompromittierte Netz aus startet (z.B. durch limitierte Verbindungsversuche aus dem Netz hinaus).

Neben den Risiken die eine Honeypot für das eigene Netz bringt, können auch unbewusst einige rechtliche Gesetzte verletzt werden (laut Strafgesetzbuch):

*§ 26. Anstiftung. Als Anstifter wird gleich einem Täter bestraft, wer vorsätzlich einen anderen zu dessen vorsätzlich begangener rechtswidriger Tat bestimmt hat.*

*§27. Beihilfe. (1) Als Gehilfe wird bestraft, wer vorsätzlich einem anderen zu dessen vorsätzlich begangener rechtswidriger Tat Hilfe geleistet hat. (2) Die Strafe für den Gehilfen richtet sich nach der Strafdrohung für den Täter. Sie ist nach § 49 Abs. 1 zu mildern.*

Über einen Emulierten Honeypot wäre eine Straftat schwer zu verwirklichen, da jedoch in Deutschland schon der versuch einer Straftat als Verbrechen gilt könnten sich dies in diesem Fall als problematisch erweisen. Des weiteren könnte ein absichtlich schlecht gewartetes Betriebssystem (z.B. aus Research-Gründen), das von einem Angreifer kompromittiert wurde, als Plattform für weitere Angriffe genutzt werden. Da das Betriebssystem vorsätzlich ungesichert ist, würde dies als Beihilfe für eine Straftat gelten.



# Kapitel 5

## Hacking

Das eigentliche Ziel jedes Honeypots ist das Anlocken von Hackern. Die dabei gesammelten Daten sind jedoch nichts wert, wenn sie nicht korrekt erkannt und analysiert werden. In diesem Kapitel wird über die Grundlagen, Vorgehensweisen und die Absichten von Hacking-Angriffen genauer eingegangen.

### 5.1 Definition von Hacking

Unter Hacking wird oft das gesetzwidrige aushebeln von Programmfunktionen um einer illegalen Aktivität nachzugehen, verstanden. Laut (Jon Erickson) handelt es sich jedoch um "das Auffinden von unbeabsichtigten oder übersehenen Anwendungsmöglichkeiten von Regeln, die auf eine neue und originelle Weise angewendet werden, um ein Problem zu lösen, wie immer dieses auch sein mag. Ob ein Hack nun bösartig ist oder nicht, hängt vom ausführenden Hacker und dessen Absichten ab. Deswegen unterscheidet man im allgemeinen unter verschiedenen Typen von Hackern [3]:

#### **White-Hat**

Als White-Hat Hacker werden jene bezeichnet, die sich mit ihren Aktionen im gesetzlichen Rahmen befinden. Sie verwenden ihr Wissen, um Sicherheitslücken zu erkennen, und andere daraufhin aufmerksam zu machen. Sie werden oft von größeren Firmen eingesetzt, um deren Systeme mit professionellen Penetrationstests auf Schwachstellen zu überprüfen.

#### **Black-Hat**

Black-Hat Hacker wollen durch ihre Hacking Angriffe Schaden verursachen. Dabei kann es sich z.B. um Datendiebstahl, Systembeschädigungen oder das Blockieren von Diensten handeln. Dies ist die Art von Hackern, die man letztendlich mit einem Honeypot anlocken möchte.

#### **Grey-Hat**

Diese Form von Hackern ist eine Mischform der zuvor genannten. Sie halten sich womöglich nicht in einen gesetzlichen Rahmen, verfolgen jedoch mit ihren Aktionen meist ein höheres Ziel, wie z.B. Firmen zu Veröffentlichung von Informationen zu bekannten Sicherheitslücken zwingen.

Neben diesen Kategorien gibt es noch die s.g. Scriptkiddies, welche meist aus Jugendlichen besteht, die mit wenigen Grundkenntnissen vorgefertigte Automatismen verwenden, um in Systeme einzudringen oder Schaden anzurichten. Script-basierte Angriffe werden meistens von Honeypots registriert, da sie oft versuchen auf Dienste zuzugreifen, die das System nicht unterstützt.

## 5.2 Grundlagen von Netzwerkhacks

Um ein gewisses Grundwissen über die Vorgehensweise von Hackern zu erhalten werden in diesem Kapitel die Grundlagen von Angriffen, die über ein Netzwerk stattfinden, genauer erläutert. Mit diesem Wissen können später die vom Honeypot aufgezeichneten Log-Dateien ausgewertet werden. Zunächst werden kurz die Grundlagen der netzbasierten Kommunikation, danach verschiedene Angriffsszenarien genauer betrachtet.

### 5.2.1 Netzkommunikation

Die Kommunikation in einem Netzwerk findet über s.g. Netzprotokolle statt. So wird gewährleistet, dass der Sender und der Empfänger die selbe SSprache sprechen. Wird z.B. eine Web Seite in einem Browser aufgerufen, wird meist das HTTP-Protokoll verwendet. Dieses wird über weitere Protokolle in ein Paket gepackt, und an den gewünschten Web Server gesendet. Dieser entpackt das Paket in umgekehrter Reihenfolge wie der Sender, und gelangt so letztendlich zur ursprünglichen Nachricht [1]. Für die Reihenfolge der anzuwendenden Protokolle gibt es zwei Grundlegende Referenzmodelle:

#### OSI-Referenzmodell

Das OSI-Schichtenmodell beschreibt die Struktur der gesamten Kommunikation zweier Kommunikationsteilnehmer. Es besteht aus sieben Schichten, die aus jeweils mehreren Protokollen bestehen, die für unterschiedliche Aufgabenstellungen zuständig sind [1]:

- Anwendungsschicht: Diese Schicht sorgt für eine Umsetzung der Anforderungen der Anwendung
- Darstellungsschicht: Diese Schicht sorgt dafür dass die Nachricht in eine für die Anwendung verwertbare Sprache zur Verfügung gestellt wird
- Kommunikationsschicht: Diese Schicht kümmert sich um den Aufbau und Überwachung von Verbindungen
- Transportschicht: Diese Schicht sorgt für eine transparente Ende-zu-Ende Verbindung mit einem Kommunikationspartner
- Vermittlungsschicht: Die Vermittlungsschicht kümmert sich um die Vermittlung der oberen Schichten. Zu ihren Aufgaben gehören unter anderem das Routing und die Adressierung.

- Sicherungsschicht: Diese Schicht bietet Funktionen wie Fehlerkorrekturen und Flusssteuerung an
- Bitübertragungsschicht: Die unterste Schicht behandelt die physikalische Verbindung zweier Punkte und ist für die Bitstream-Übertragung zuständig

Die Kommunikation über diese Schichten findet über Paketen statt, die z.B. in der Anwendungsschicht erstellt, und danach in die darauf folgenden sechs Schichten gekapselt verpackt werden. Beim Empfänger werden die Schichten zur Anwenderschicht wieder entfernt bis die eigentliche Nachricht interpretierbar wird. In Abb. 5.1 wird ein Beispielkommunikation über das Internet dargestellt.

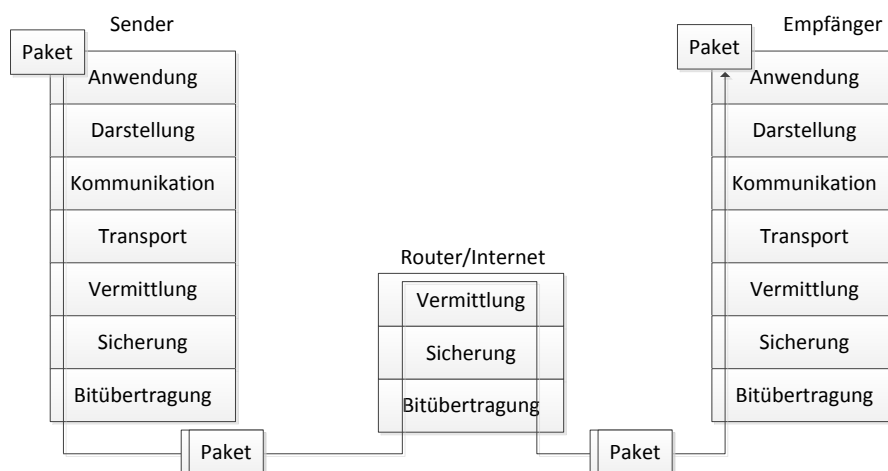


Abbildung 5.1: Kommunikation über das OSI-Schichtenmodell

Das OSI-Schichtenmodell wird häufig als Grundlage für den Entwurf neuer Netzwerkprotokolle verwendet. In der Realität finden Kommunikation meist über weniger Schichten statt, wie z.B. das TCP/IP-Referenzmodell.

### TCP/IP-Referenzmodell

Das TCP/IP-Referenzmodell basiert auf dem OSI-Modell, fasst jedoch die Aufgaben einiger Schichten zusammen. So wird statt der Bitübertragungsschicht und Sicherungsschicht eine allgemeine Netzzugangsschicht, und aus Kommunikations-, Darstellungsschicht und Anwendungsschicht eine allgemeine Anwendungsschicht. Die Vermittlungsschicht und Transportschicht werden beibehalten.

### 5.2.2 Netzwerkprotokolle

Netzwerkprotokolle besitzen je nach Zweck und Netzwerkschicht verschiedene Merkmale und Funktionen. Viele Netzwerkprotokolle hängen von der obersten Schicht bis zur Untersten einen eigenen Protokoll Header an. Dieser besitzt wichtige Merkmale zum Transport der darauf

folgenden Daten. Einige der wichtigsten Protokolle und deren Header Funktionen werden in diesem Kapitel genauer betrachtet:

- **IP:** Das Internet Protocol ist ein sehr weit verbreitetes Netzwerkprotokolle welches die Grundlagen des Internet darstellt. Es befindet sich in der Vermittlungsschicht des OSI-Schichtenmodells und ist somit für die Vermittlung über IP-Adressen zuständig.
- **TCP:** Das Transmission Control Protocol ist ein Netzwerkprotokoll der Transportschicht. Dieses garantiert eine zuverlässige, verbindungsorientierte Transport von Paketen in Computernetzwerken. Es zählt wie das IP-Protokoll zu den Grundlagen des Internets. In diesem Protokoll-Header werden der Quell und Ziel Port genannt, um dem meist zuvor erstellten IP-Paket einen bestimmten Dienst zuzuweisen. Eine TCP-Verbindung erfolgt immer über eine vorgegebene Startsequenz. Diese wird 3-Way-Handshake genannt und wird in Abb. 5.2 dargestellt. Dabei werden die Control-Flag-Bits im Header jeweils geändert.

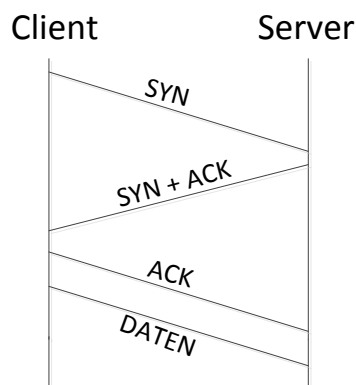


Abbildung 5.2: Drei-Wege-Handshake mit TCP

Nach dem Aufbau der Verbindung können Daten gesendet werden.

- **UDP:** Beim User Datagram Protocol handelt es sich um ein verbindungsloses Transportprotokoll. Wie bei TCP befindet sich in seinem Header Ziel- und Quell-Port, jedoch erfolgt kein Verbindungsaufbau. Außerdem wird nicht überprüft, ob alle Pakete tatsächlich beim Benutzer ankommen.
- **HTTP/HTTPS:** Das Hyper Text Transfer Protocol wird von Web-Servern verwendet, um deren Webseiten an den Benutzer zu senden. Diese Protokoll befindet sich bereits in der Anwenderschicht, und baut auf einer TCP/IP Verbindung auf.
- **SSH:** Das Secure Shell Protokoll ist ein Netzwerkprotokoll, welches sich ebenfalls auf der Anwenderschicht befindet. Es öffnet eine verschlüsselte Verbindung auf einen entfernten Server, um die Kommandozeile dessen lokal zur Verfügung zu stellen.

- FTP: Das File Transfer Protocol wird zur Übertragung von Daten allgemein verwendet. Das Anwendungsprotokoll wird verwendet um vom Server Daten herunter- oder hochzuladen.

Die Protokolle die in den jeweiligen Schichten angewendet werden, sind oft Angriffsziel vieler Hacker. Diese können über bewusste Manipulation so abgeändert werden, dass das System oft unerwartet reagiert. Einige mögliche Angriffsszenarien werden im folgenden Kapitel betrachtet.

## 5.3 Angriffsszenarien

Durch Manipulation einiger Funktionen von Netzwerkprotokollen können Hacker Zugriff auf das System erhalten, oder dort Schaden anrichten. Einige dieser Angriffsarten werden in diesem Kapitel genauer betrachtet, speziell dieses, die auch auf dem Honeypot vermehrt festgestellt wurden.

## 5.4 Datenanalyse

Nach einer erfolgreichen Kompromittierung eines Angreifers muss unabhängig des verwendeten Honeypotsystems eine Datenanalyse stattfinden. Abhängig des eigentlichen Nutzen des Honeypots für das Netzwerk kann eine Netzwerkanalyse mehr oder weniger aufwendig werden. Wird ein Honeypot als Early-Warning System verwendet wird eine aufwendige Datenanalyse nicht nötig sein. Wird er jedoch als Research Honeypot verwendet, oder ein und die selbe Sicherheitslücke wird auf längere Zeit mehrmalig ausgenutzt, so wird eine ordentlich durchgeführte Datenanalyse notwendig. Folgende Merkmale bestimmen ob eine ausführliche Datenanalyse nach einer Kompromittierung durchgeführt werden soll [5]:

- Was ist der eigentliche Zweck des Honeypots? (Production oder Research)
- Was wird versucht zu schützen?
- Ist jeder Angriff interessant oder nur die die erfolgreich sind?
- Ist die Angriffsmethode interessant?
- Soll herausgefunden werden wer der Hacker ist? Welcher Tools, Techniken oder Mechanismen dieser verwendet?
- Ist nur die Methode des Zugriffs auf den Honeypot interessant oder auch das was der Hacker auf diesen vorhatte?

Allgemein sollte bei einer ausführlichen Datenanalyse (wenn z.B. wie im Falle des Projektes ein Research Honeypot verwendet wird) folgende drei Fragen geklärt werden:

- War der Angriff manuell oder automatisch?
- Wie fand die initiale Kompromittierung statt?

- Was macht der Hacker nachdem er sich Zugriff verschafft hat?

Automatisierte Angriffe kommen häufiger vor als manuelle. Manuelle Kompromittierungsversuche sind jedoch um einiges Gefährlicher als automatisierte, da der Angreifer unerwartet seine Strategie ändern, und unberechenbar auf seinem Zielsystem handeln kann. Automatisierte Angriffe nutzen meist bekannte Sicherheitslücken aus, und hoffen dabei bei einer neue Anwendung eine Zero-Day-Attack auszuführen (nutzt Sicherheitslücken aus die die Programmierer bei der erstmaligen Erscheinung eines Programmes noch nicht behoben haben). Automatisierte Angriffe sind relativ einfach an folgenden Eigenschaften erkennbar:

- Schnelle Zugriffsversuche auf verschieden Art und Weise(zeitlich sehr nah beieinander)
- Ports oder Exploits, die nicht für das eigene System verfügbar sind werden ausprobiert
- Dieselbe Zugriffsmethode wird in kurzer Zeit mehrmals ausgeführt, ohne das eine Änderung von Parametern stattfindet
- Das Tippen ist so schnell das kein Mensch dazu fähig wäre

Im Gengenzug dazu sind manuelle Hacker durch folgende Merkmale erkennbar:

- Der verwendetet Exploit Code passt zu dem Zielsystem
- Wesentlich häufiger vorkommende Tippfehler und wiederholende Eingaben
- Zeitabstände zwischen Eingaben sind ungleichmäßig
- Versucht häufig vorher Informationen zu sammeln (Ping, Portscan)

#### **Initialer Zugriff auf das System**

Die meisten Angriffe bestehen aus zwei verschiedenen Phasen. Das Ziel der ersten Phase besteht darin, Zugriff auf das System zu erhalten, das Ziel der zweite um die eigentlichen Absichten zu verfolgen.

#### **Nach dem Zugriff**

Der meist wichtigere Teil der Datenanalyse ist der Teil der nach der initialen Kompromittierung stattfindet. Das Ziel ist es herauszufinden, was die Absichten des Hackers waren, welche Daten er gesucht hat (z.B. Kreditkartendaten, Werbeinformationen)oder welche Daten er hinterlassen hat. Meist versucht der Hacker den kompromittierten Rechner als Speicherplatz (für z.B. illegale Daten wie DVDs oder Spiele), oder als Teil eines Botnetzes zu verwenden.

Die Möglichkeiten eines Hackers sind vielfältig, deswegen gibt es(je nach Genauigkeit der Analyse) mehr oder weniger aufwendige Analysearten. Diese sind durch die Implementierungsart des Honeypots meist mit eingeschränkt:

### **Low-Interactive Honeypots**

Bei Low-Interactive Honeypots hält sich die Datenauswertung in Grenzen. Meist bietet sich nur die Möglichkeit, den Netzwerkverkehr, IDS Log-Files und die Honeypot Log-Files zu analysieren. MEHR!

### **High-Interactive Honeypots**

High-Interactive Honeypots bieten einen hohen Informationsgehalt bei der Datenanalyse. Diese ist jedoch im Vergleich zu den anderen Honeypot Implementierungsarten sehr aufwendig. Für eine ausführliche Analyse sind folgende Schritte notwendig [5]:

1. Den Honeypot aus dem Netz entfernen
2. RAM Speicherinhalte falls möglich sichern
3. Eine Kopie der Festplatte erstellen
4. Netzwerkverkehr, Dateisystem, Schadcode (falls vorhanden), Betriebssystem, Log-Daten analysieren
5. Schlüsse daraus ziehen (Sicherheitslücken erkennen)
6. Sicherheitslücken schließen/Honeypot modifizieren (falls gewünscht)
7. Honeypot neu aufsetzen

### **Den Honeypot offline nehmen**

Um eine forensische Analyse an einem Honeypot vorzunehmen, muss zunächst sicher gestellt werden, dass sich dieser nicht mehr im Netz befindet. So wird sichergestellt, dass niemand mehr weitere Modifikationen am Honeypot vornehmen kann. Außerdem kann der Angreifer ab dem Zeitpunkt, an dem die Verbindung gekappt wurde, nicht mehr seine Spuren verwischen, oder gar entdecken, dass er sich auf einem Honeypot befindet. In diesem Fall kann er seine Taktik ändern und eventuell versuchen, das System zu beschädigen oder sogar die Festplatte zu formatieren.

#### **5.4.1 Dateien sichern**

Zu Beginn einer Datenanalyse sollte eine Momentaufnahme (Snapshot) des aktuellen Systems gespeichert werden. Dazu gehört ein Abbild des RAM- sowie des Festplattenspeichers.

### **RAM Speicher sichern**

Bei einer ausführlicheren Systemanalyse sollte auch der RAM Speicher untersucht werden. Für Linux/Unix Systeme gibt es Programme, die ein Speicherabbild des RAM Speichers erstellen können (z.B. Memfetch). Für Windows gibt es keine solche Anwendungen. Jedoch kann über Umwege auf das Pagefile zugegriffen werden. Das Pagefile enthält die Speicherseiten, die bei Swap-Operationen (vom RAM-Speicher werden Speicherseiten auf die Festplatte ausgelagert,

um den Arbeitsspeicher zu entlasten) auf der Festplatte gespeichert werden. Um dies zu erreichen, muss das System über den Power-Button (nicht die konventionelle Methode über das Menü) ausschalten. Dadurch wird verhindert dass das PAGEfile gelöscht wird. Um nun Zugriff auf dieses zu erhalten darf von der Festplatte nicht mehr gebootet werden, da dabei das PAGEfile überschrieben werden würde. Um nun Zugriff auf die Datei zu erhalten, muss auf die Festplatte von einem anderen System zugegriffen werden (z.B. als Slave-Festplatte). Danach kann über verschiedene Tools auf die Datei zugegriffen werden. Ein anderer Weg, um Zugriff auf den RAM Speicher zu erhalten ist, einen STOP-Error herbeizuführen. Zuvor kann in den Erweiterten Systemeinstellungen (hängt vom verwendeten Betriebssystem ab) eingestellt werden, dass ein komplettes Speicherabbild erstellt werden soll (Abb. 5.3).

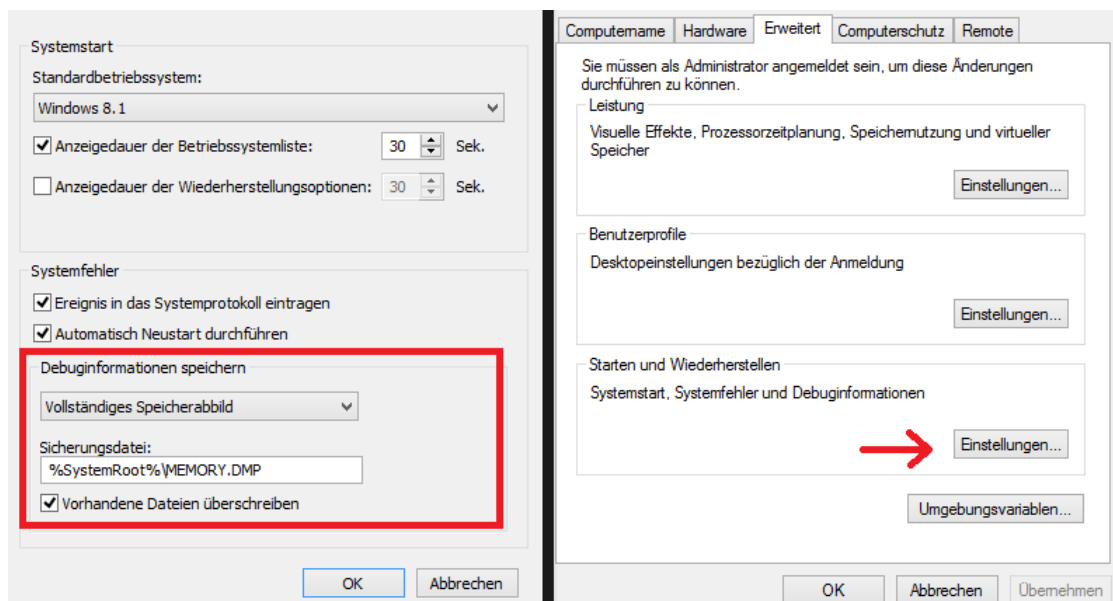


Abbildung 5.3: Einstellungen für RAM-Seicher Dmp-File

Ruft man nun einen STOP-Error herbei, so wird ein gesamtes Speicherabbild im angegeben Pfad gespeichert. Ein solchen Fehler absichtlich auszulösen kann über eine Tastenkombination, die zuvor in einem Registry Eintrag definiert wird, ausgelöst werden. Ein Beispiel dafür wäre:

```
HKEY_LM \System\CurrentControlSet\Services\i8042prt\Parameters Value
Name: CrashOnCtrlScrollData Type: REG_DWORD Data:(1 = enabled)
```

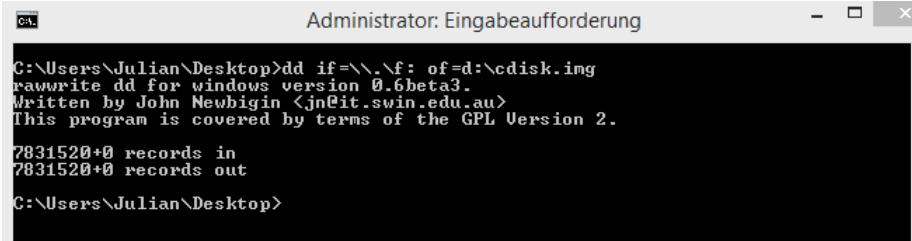
Nun kann durch das Halten der Strg-Taste und das zweimalige Tippen auf Rollen ein STOP-Error ausgelöst werden. Programme, mit denen man nun das Dmp-File bearbeiten kann werden von Windows selbst bereitgestellt.

### Speicherabbild des Festplattenspeichers

Neben ein RAM-Speicherabbild wird bei einer ausführlichen Analyse ein Speicherabbild der Festplatte benötigt. Es gibt spezielle Anwendungen die ein komplettes Speicherabbild der Festplatte erstellen können. Als kommerzielle Lösung stellt Symantec Norton Ghost zur Verfügung.



Um eine forensische Analyse durchzuführen werden meist jedoch speziellere Tools verwendet. Eine kostenlose Möglichkeit ein Festplattenabbild zu erstellen ist durch das dd Tool möglich. Dieses ist sowohl für Linux als auch für Windows verfügbar und kopiert Blockweise Speicher von einer Festplatte oder Partition auf eine andere. In Abb. ?? wird die komplette Partition f: auf d: als cdisk.img gespeichert.



```
C:\Users\Julian\Desktop>dd if=\\.\f: of=d:\cdisk.img
rawwrite dd for windows version 0.6beta3.
Written by John Newbigin <jn@it.swin.edu.au>
This program is covered by terms of the GPL Version 2.
7831520+0 records in
7831520+0 records out
C:\Users\Julian\Desktop>
```

Abbildung 5.4: Erstellung eines Speicherabbilds mit dd

Neben dieser vereinfachten Methode gibt es noch weitere Möglichkeiten über kommerzielle Software, die für den Zweck einer forensischen Analyse entwickelt wurden, ein professionelles Festplattenabbild zu erstellen. Zu diesen gehören:

- EnCase
- Winhex
- ProDiscover
- SafeBack

Nachdem alle relevanten Daten gesichert wurden kann der Netzverkehr überprüft werden.

### 5.4.2 Netzwerkanalyse

Mit einer Netzwerkanalyse kann am schnellsten erkannt werden, was auf einem Honeypot passiert ist. Die Netzwerkanalyse kann grundsätzlich auf zwei Wege genutzt werden. Zum einen kann erkannt werden welche Ports verwendet wurden, zum anderen kann über die Payload der Pakete nach Details geforscht werden. Wenn der übertragene Datenverkehr nicht verschlüsselt ist, kann ein Packet-Sniffer (z.B. WireShark) den kompletten Verkehr aufzeichnen. Fragen, die sich bei der Netzwerkanalyse stellen, sind:

- Wie viele Pakete wurden aufgezeichnet
- Welche IP Adressen waren beteiligt
- Wer kommunizierte mit wem
- Welche Ports wurden verwendet
- Welche zeitlichen Abstände befinden sich zwischen den abgesendeten Paketen
- Wie groß waren die Pakete

# Kapitel 6

## Tools und Anwendungen

In diesem Kapitel werde die von Honeypots und Honeynets verwendeten Tools zum Sammeln von Daten und der Datenaufbereitung genauer betrachtet.

### 6.1 Honeypot und Honeynet Tools

Snort Snort\_inline Hflow2(Gen3) Tcpdump Sebek

#### 6.1.1 Sebek

Sebek ist ein Kernel basiertes Tool, welches die Daten eines Angreifers auf dem Host-System mitliest und loggt.

Zu Beginn der Überwachung von Host-Systemen wurden häufig Netzwerk-Sniffer verwendet. Diese sind in der Lage alle Pakete zwischen Angreifer und Opfer-System mitzulesen. Damit sind Netzwerk-Sniffer in der Lage sowohl den Input als auch den Output zu rekonstruieren. In der heutigen Zeit wird es immer schwieriger an Daten wie beispielsweise Tastenanschläge zu kommen. Viele Hacker benutzen verschlüsselte Verbindungen zwischen ihren Opfer-Systemen und umgehen dadurch einem Netzwerk-Sniffer. Häufig wird für die Verschlüsselung SSH verwendet. Einige Angreifer verwenden eigene Tools, welche sie auf dem Opfer-System installieren um ihre Verbindung zu verschlüsseln. Um in solchen Fällen an die Daten des Angriffs zu kommen, mussten neue Methoden gefunden werden.

Aus dieser Not heraus entwickelte The Honeynet Project das Tool Sebek. Sebek konzentriert sich bei der Daten-Sammlung nicht auf die übertragenen Daten, sondern auf die Daten die im Kernel des Systems verarbeitet werden. Hintergrund dieses Vorgehens ist die Tatsache, dass verschlüsselte Daten nicht von dem System verarbeitet werden können. Somit kommen die Daten im Kernel unverschlüsselt zur Verarbeitung an. Durch dieses Vorgehen ist es für Sebek unerheblich welche Verschlüsselung von den Angreifern eingesetzt wird, da die Angreifer auch die Entschlüsselung für Sebek übernehmen. Sobald die Daten an dem System-Aufruf `read()` ankommen greift Sebek die Daten ab ohne dabei Aufsehen zu erregen.

Möglich macht dieses Vorgehen die Manipulation des System-Aufrufs. Durch Sebek wird der Pointer, welcher in der Syscall Tabelle gespeichert wird, auf die `read()` Methode geändert. Der

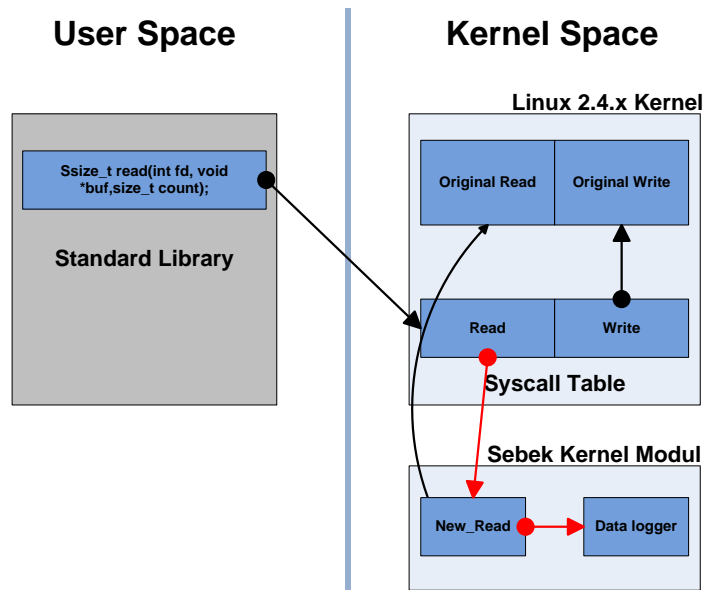


Abbildung 6.1: Abfangen der Daten durch Sebek [4]

kompromittierte Pointer verweist nun auf eine von Sebek initialisierte `read()` Methode. Diese schreibt alle Informationen in einen "Data logger" und ruft danach die Originale `read()` Methode auf. Somit bekommt der Angreifer nicht mit, dass die Daten über einen Umweg abgefangen werden konnten. Um den Vorgang zu verdeutlichen wird dieser in Abb. 6.1 bildlich dargestellt.

Ein weiteres Problem was nun entsteht, ist die Frage was mit den gesammelten Daten passiert. Um weiterhin im verborgenen zu bleiben, können die Daten nicht auf dem Host gespeichert werden. Hier hätte der Angreifer die Möglichkeit nach auffälligen Dateien zu suchen oder auch durch Zufall darauf zu stoßen. Dieses Problem umgeht Sebek indem die Daten kurz in dem Data logger gespeichert werden und von dort an einen Server geschickt werden. Diese Übertragung muss jedoch so vollzogen werden, dass der Angreifer keine Möglichkeit hat sie zu entdecken. Um unerkannt zu bleiben werden die Pakete nicht über den TCP/IP Stack in das

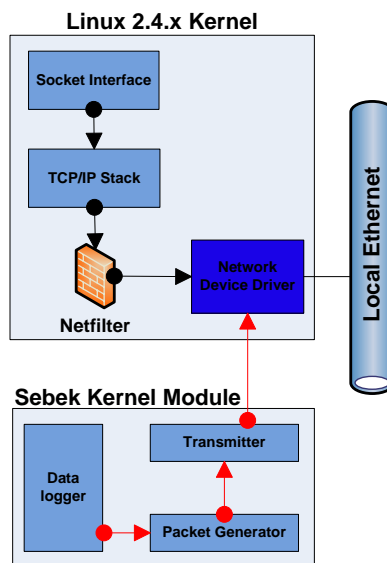


Abbildung 6.2: Zugriff auf Netzwerk Driver [4]

Netzwerk geleitet, sondern, wie ich Abb. 6.2 veranschaulicht, über den direkten Weg an den Netzwerk-Driver gesendet. Dadurch kann das System die Pakete weder durch einen Sniffer erkennen, noch über IPTABLES blockieren. Der Angreifer hat nun noch die Möglichkeit über das mitlesen des LAN-Traffics Sebek-Pakete zu entdecken. Sollten sich mehrere Honeypots im Netzwerk befinden, können auch Sebek-Pakete von anderen Systemen mitgelesen werden. Dies gilt es zu verhindern um keinen Verdacht zu erwecken. Um die Pakete nicht entdecken zu können, implementiert Sebek eine eigene Version des Raw Socket Interface. Das Interface erkennt anhand des Ziel UDP-Ports und einem "Magic-Value", welcher im Header der Pakete steht, dass es sich um ein Sebek-Paket handelt. Das Raw Socket Interface wird von Sebek so verändert, dass alle Sebek-Pakete verworfen werden und damit für jeden Sniffer auf einem Honeypot-Systeme unerkannt bleiben. Mit diesem Vorgehen schafft Sebek eine lückenlose Überwachung ohne dabei entdeckt zu werden. Ein beispielhafter Aufbau eines Angriffs-Szenario ist in Abb. 6.3 dargestellt. [4]

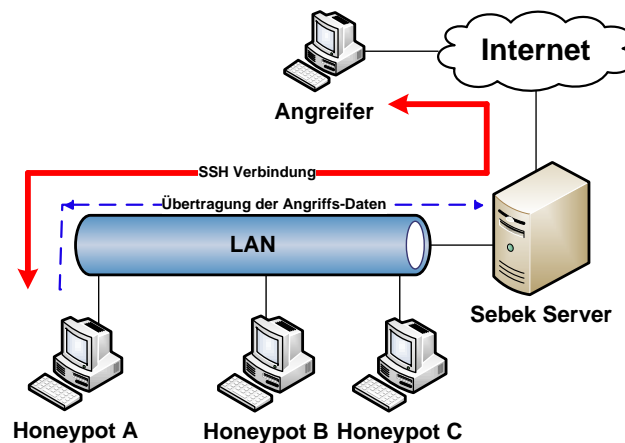


Abbildung 6.3: Ablauf einer Sebek-Überwachung [4]

### 6.1.2 Intrusion Detection System

In Netzwerken werden Intrusion Detection Systeme zur Überwachung und Erkennung von Unregelmäßigkeiten eingesetzt. Im Gegensatz zu einer Firewall, welche erkennbar vor einer DMZ eingesetzt wird, arbeiten IDS im Verborgenen ohne aktiv in das Geschehen einzuschreiten. Um den vielfältigen Aufgaben gerecht zu werden, besteht ein IDS aus folgenden vier Modulen:

- **Collection (Sammeln)**

Für das Sammeln aller relevanten Daten ist das Collection Modul zuständig. Hierbei können zwei verschiedene Vorgehensweisen unterschieden werden. Zum einen gibt es die Host-basierten IDS und zum anderen Netzwerk-basierte Systeme. Soll ein ganzes Netzwerk überwacht werden, eignet sich ein Host-basiertes IDS nicht, da hierfür auf jedem Host einzeln ein IDS installiert werden muss. Sinnvoller wäre hier eine Überwachung des Netzwerk-Traffics. Umgesetzt wird das durch sogenannte Sensoren die im Netz eingebunden werden und dort die Daten abgreifen. Möglich ist das an einem Switchport der als Monitoring-Port konfiguriert wird. Ein solches Vorgehen wird Netzwerk-basiertes Intrusion Detection System genannt.

- **Detection (Erkennen)**

Um einen Angriff erkennen zu können gibt es diverse Möglichkeiten. Ein Pattern basiertes System vergleicht die gesammelten Daten mit gespeicherten Pattern, welche aus bekannten Gefahren bestehen. Sobald es Übereinstimmungen gibt handelt es sich meist um einen Angriff. Nachteil hierbei ist, dass regelmäßig Updates eingespielt werden müssen, ähnlich wie es bei einem Anti-Viren Programm ist.

Eine Alternative ist ein Anomalie-Detection-System. Bei diesem Vorgehen wird in einer ersten Phase der Netzwerk-Verkehr mitgeschnitten und gespeichert. Aus den gesammelten Daten leitet das System einen „normalen“ Zustand ab. In der zweiten Phase vergleicht das System den aktuellen Traffic mit dem von ihm als „zulässigen“ gespeicherten Traffic und sucht nach Abweichungen. Vorteil hierbei ist, dass auch nicht bekannte Angriffe erkannt werden können.

- **Analyse (Analysieren)**

Das Analyse Modul bereitet die Ergebnisse des vorherigen Moduls grafisch auf und in gibt Statistiken über die ausgewerteten Daten aus.

- **Response (Reaktion)**

Häufig greift ein IDS nicht aktiv in den Netzwerk-Traffic ein um weiterhin unerkannt zu bleiben. Dennoch sollte ein Intrusion Detection System handeln wenn ein verdächtiges Verhalten entdeckt wird. Um das Vorgehen des möglichen Angriffs besser rekonstruieren zu können sollten die verdächtigen Aktivitäten mitgeloggt werden. Bei möglichen kritischen Problemen können unter Umständen auch Alarme, in Form von SMS oder E-Mail an den Administrator, erzeugt werden.

Bei genauer Betrachtung der Arbeitsweise zeigt sich, warum ein Honeypot oft auch als Teil eines IDS angesehen wird. Sowohl ein Host basiertes, als auch ein Netzwerk basiertes IDS bietet die Möglichkeit einen Honeypot zu überwachen und das Verhalten eines Angreifers zu dokumentieren. Um Handlungen auf dem Honeypot nachvollziehen zu können eignet sich jedoch ein Host basiertes System deutlich besser, da hier auch die Aktionen auf dem Host geloggt werden können. Die Überwachung des Honeypots gestaltet sich deutlich einfacher als die eines normalen Systems. Denn auch hier gilt jeder Zugriff auf den Honeypot als potentieller Angriff. Somit muss das IDS nicht entscheiden, nach welchen Mustern Angriffe gedeutet werden.

Abschließend kann festgehalten werden: Für diese Arbeit dient ein IDS aus Sicht eines Honeypots zur Aufzeichnung und Auswertung der Angriffe, sowie zur Alarmierung. Dies schließt ausdrücklich nicht die Ansicht aus, dass ein Honeypot Teil eines IDS sein kann. In diesem Fall kommt es nur auf den Blickwinkel an, aus dem das Thema betrachtet wird.

## 6.2 Honeypot Software

Für die Erstellung eines Honeypot Systems gibt es einige Softwarepakete die es ermöglichen ein solches System recht schnell zu realisieren.

### 6.2.1 Honeyd

Honeyd ist ein von Niels Provos entwickelter Honeypot Daemon, der verschiedene virtuelle Host Systeme wie Workstations oder Netzwerkgeräte emulieren kann. Somit ist es möglich, eine komplette Netzstruktur zu emulieren. Die Open Source Software wird unter GPL veröffentlicht, und ist somit ohne Lizenzkosten benutzbar. Die aktuelle Version 1.5c wurde am 27.05.2007 veröffentlicht. Über die Benutzung von Konfigurationsdateien kann eine Vielzahl von Betriebssystemen gewählt werden. Dabei werden s.g. Templates erstellt, für die zuerst eine Identität gewählt werden muss. Danach können verschiedene Parameter das simulierte Verhalten des virtuellen Clients festgelegt werden. Dazu gehören z.B. das eine emulierte Bandbreite des Übertragungskanal, Default-Aktionen verschiedener Ports und die zu aktivierenden Scripts beim Zugriff auf verschiedenen Ports. Zuletzt wird dem Template die gewünschte IP zugewiesen. In einer Honeyd Konfigurationsdatei können mehrere Clients eingetragen werden. Ein minimales Beispiel einer Honeyd-Konfigurationsdatei befindet sich in Abb. 6.4. Die Skripte, die bei einem versuchten Zugriff auf einem Port gestartet werden sollen können beliebig erstellt oder manipuliert werden.

```
create windows
set windows personality "Microsoft Windows 2000 SP2"
set windows default tcp action block
set windows default udp action block
set windows default icmp action block

add windows tcp port 23 "sh /usr/share/honeyd/scripts/win32/win2k/exchange-telnet.sh $ipsrc $sport $ipdst $dport"
add windows tcp port 25 "sh /usr/share/honeyd/scripts/win32/win2k/exchange-smtp.sh $ipsrc $sport $ipdst $dport"
add windows tcp port 110 "sh /usr/share/honeyd/scripts/win32/win2k/exchange-pop3.sh $ipsrc $sport $ipdst $dport"

bind 193.196.7.11 windows
```

Abbildung 6.4: Beispielkonfiguration einer Honeyd Config-Datei

Honeyd wird unter Unix Systemen in der Kommandozeile ausgeführt. Dabei stellt das Programm verschiedene Konfigurationsmöglichkeiten zur Verfügung. Über den Parameter honeyd -d wird das Programm nicht als Daemon gestartet. So kann man in der Konsole nach starten des Programmes die erfolgten Zugriffe beobachten. Diese möglicherweise versuchten Angriffe (je nach Netz werden auch alle Pakete die an die Broadcast Adresse gesendet werden empfangen) müssen danach analysiert werden. Um ein möglichst leicht auswertbares Ergebnis zu erhalten kann eine feste Angabe des abzuhörenden Netzbereichs sowie die Erstellung einer Log-Datei erfolgen. Der folgende Befehl kann so auf das zuvor genannte Beispiel angewandt werden:

```
honeyd -f honeyd.conf 193.196.7.11 -l honeyd.log
```

Hierbei werden nur die Pakete die für die Zieladresse 193.196.7.11 bestimmt sind für eine Auswertung in Betracht gezogen. Die erfolgten Zugriffe werden in der Datei honeyd.log gespeichert. Die Log-Dateien speichern folgende Datensätze:

*2014-02-22-15:50:59.5296 tcp(6) S 109.193.95.133 21473 193.196.7.11 23 [Windows 2000 RFC1323]*

Zunächst wird der Zeitpunkt des Angriffes festgestellt. Danach ob der Zugriff über TCP, UDP oder ICMP stattgefunden hat. Das S kennzeichnet den Start einer Verbindung. Bei UDP oder Paketen, die das Ende einer Verbindung kennzeichnen befindet sich an dieser Stelle ein E. Die nächsten beiden Werte sind die Quelle und Ziel IP-Adressen mit dem dazugehörigen Ports. Zuletzt befindet sich ein Versuch über Passive Fingerprinting das Betriebssystem des Angreifers zu erraten. Bei diesem Beispiel handelt es sich um einen Versuch über Telnet auf den Honeypot zuzugreifen. Da Honeyd seit 2007 nicht mehr weiterentwickelt wird verwendet es in diesem Beispiel eine veraltete Fingerprint Datenbank und gibt statt des verwendeten Windows 8.1 einen Windows 2000 Rechner an.

#### 6.2.2 Tiny oder Tarpit

#### 6.2.3 Roo

#### 6.2.4 High-Interactive?

### 6.3 Anwendungen zur Datenanalyse

#### 6.3.1 Honey View

#### 6.3.2 ...

# Kapitel 7

## Implementierung des Honeypots

### 7.1 Systemspezifikationen

Für die folgenden Implementierungen wurde ein Ubuntu 12.04.4 LTS Server verwendet. Bei dem Betriebssystem handelt es sich um eine reine Konsoleninstallation. Der Server besitzt folgende Hardwarespezifikationen:

Prozessor	Intel(R) Xeon(R) CPU X3220 @ 2.40GHz
Arbeitsspeicher	4GB
Plattenspeicher	250GB
Netzwerkadapter	82573L Gigabit Ethernet Controller

Der Server hatte die externe IP Adresse 193.196.7.11. Alle ankommenden Ports waren offen, abgehende Ports waren alle in Richtung des Internets offen, in das lokale Netz war alles geschlossen. So konnte der Honeypot über jeden Port über das Internet angesprochen werden, stellte jedoch keine Gefahr für das lokale Netz der Dualen Hochschule dar. Der Port 25 wurde von Seiten des Netzproviders gesperrt und kann dementsprechend nicht freigeschaltet werden.



## 7.2 Honeyd Implementierung

In diesem Kapitel wird genauer auf die konkrete Erstellung des Honeyd Honeyd mit allen Problemstellungen eingegangen.

### 7.2.1 Installation

Die zum Zeitpunkt der Studienarbeit aktuelle Version von Honeyd ist 1.5c. Diese Version wurde am 27.05.2007 veröffentlicht und ist demnach recht veraltet. Die Webseite von Niels Provos zu diesem Projekt wurde zuletzt am 15.07.2008 bearbeitet, demnach davon ausgegangen werden kann das er das Projekt nicht mehr weiter verfolgt. Honeyd ist jedoch mit Einschränkungen auf den meisten Unixartigen System lauffähig. Vor der Installation des eigentlichen Honeyd Deamons müssen einige Abhängigkeiten installiert werden:

#### **libdnet:**

libdnet ist eine Programmbibliothek die Zugriff auf low-level Netzwerkrountinen gewährt. Dazu gehören z.B. Netzwerkadressmanipulation, Netzwerkfirewallmanipulation, Netzwerinterfacezugriff, IP-Tunneling und Zugriff auf einzelne IP-Pakete.

#### **libevent:**

libevent ist eine Programmbibliothek, die eine asynchrone Benachrichtigungsfunktion besitzt, die bei bestimmten Events oder nach vordefinierten Timeouts ausgelöst werden kann.

#### **libpcap:**

libpcap ist eine freie Programmierschnittstelle zur Überwachung des Netzverkehrs. Viele Netzwerkanalysetools greifen auf die Funktion von pcap (packet capture) Bibliotheken zurück. libpcap ist eine Programmbibliothek für unixartige System, WinPcap besitzt für Windows Betriebssysteme die selbe Funktion. Wireshark, Snort und Nmap greifen z.B. auf die selbe Schnittstelle zurück.

Für erweiterte Funktionen wurde das Paket Honeyd-commons ebenfalls heruntergeladen. In diesem befinden sich zusätzliche Dokumentationen und Skripte für die Nachahmung verschiedener Services. Nach der Installation aller notwendiger Pakete müssen Konfigurationen für den Daemon erstellt werden.

### 7.2.2 Konfiguration und Inbetriebnahme

Um einen Honeyd mit Honeyd zu erstellen muss eine Konfiguration erstellt werden. Als ersten Versuch wurde versucht ein Suse 7.0 Rechner zu simulieren. Dieser blockiert zunächst alle Ports bis auf Port 21 (FTP) und Port 23 (Telnet). ICMP Anfragen werden ebenfalls beantwortet. Die restlichen Ports werden blockiert. Diese eher weniger reale Konfiguration wurde zunächst als Test verwendet. Da die meisten Angriffe auf Honeyd automatisch ablaufen, dauerte es nicht lange bis die ersten Angriffsversuche auf diesen aufgezeichnet wurden.

```
create suse70
set suse70 personality "Linux 2.2.12 - 2.2.19"
set suse70 default tcp action reset
set suse70 default udp action block
set suse70 default icmp action open
set suse70 uptime 92315
set suse70 droprate in 2
add suse70 tcp port 21 "sh /usr/share/honeyd/scripts/unix/linux/suse7.0/proftpd.sh"
add suse70 tcp port 23 "sh /usr/share/honeyd/scripts/unix/linux/suse7.0/telnetd.sh"

bind 193.196.7.8 suse70
```

Abbildung 7.1: Erst Testkonfiguration mit Honeyd

Durch ein Problem mit der veralteten Version von Honeyd hörte der Daemon meist nach einiger Zeit mit dem Aufzeichnen der Angriffe auf. Da es zu diesem Problem keinerlei Fehlermeldung gab und das Projekt nicht mehr weiter verfolgt wurde gelang es nicht diesen Fehler zu beheben. Durch mehrere Neustarts des NETzwerkinterfaces und der Deamons konnten jedoch einige Angriffe aufgezeichnet werden.

Die zweite Testkonfiguration ist ein Windows XP Konfiguration. Diese besitzt nur leicht andere Konfigurationseinstellungen als das zuvor verwendete Suse 7.0 Skript. Da das Problem mit dem Abbruch der Aufnahmen weiterhin bestand, machte es keinen Unterschied welche Konfiguration verwendet wurde, die Angriffe waren meist identisch.

```
create windows
set windows personality "Microsoft Windows XP Professional SP1"
set windows default icmp action open
set windows default tcp action reset
add windows tcp port 21 "sh /root/Configs/scripts/ftp.sh"
add windows tcp port 80 open
add windows tcp port 135 "sh /root/Configs/scripts/ftp.sh"
add windows tcp port 139 open
add windows tcp port 445 open

bind 193.196.7.8 windows
```

Abbildung 7.2: Zweite Testkonfiguration mit Honeyd

Trotz das nur wenige Daten aufgezeichnet werden konnten wurde der Honeypot jedes mal kurz nach Produktivstellung angegriffen. Was für Angriffe dies waren wird im folgendem Kapitel genauer betrachtet.

7.2.3 Datenanalyse

**7.3 Kommerzielle Implementierung**

7.3.1 Installation

7.3.2 Konfiguration

7.3.3 Datenanalyse

**7.4 Open Source Implementierung**

7.4.1 Installation

7.4.2 Konfiguration

7.4.3 Datenanalyse

## Kapitel 8

# Auswertung der gesammelten Daten

## Kapitel 9

## Fazit

## **Anhang**

# Literaturverzeichnis

- [1] ANDREW S. TANENBAUM, P. D. J. W.: *Computernetzwerke*. Pearson, 2012.
- [2] DORNSEIF, M.: *Ermittlung von Verwundbarkeiten mit elektronischen Koedern*. <http://www.ei.rub.de/media/emma/veroeffentlichungen/2012/08/24/Koeder-DIMVA04.pdf>, 2012.
- [3] ERICKSON, J.: *Hacking - Die Kunst des Exploits*. dpunkt, 2009.
- [4] PROJECT, T. H.: *Know Your Enemy: Sebek*. <http://old.honeynet.org/papers/sebek.pdf>, 2003.
- [5] ROGER A., G.: *Honeypots for Windwos*. Apress, 2003.
- [6] SPITZNER, L.: *Honyepots - Tracking Hackers*. Addison-Wesley, 2002.