10 Intrusion Detection und Intrusion Response

Hinter den Begriffen **Intrusion Detection** (**ID**) [Amor 99] und **Intrusion Response** (**IR**) [RRW 01] verbergen sich zwei Ansätze, bei denen mit mehr oder weniger intelligenten Programmen Angriffe automatisch erkannt und abgewehrt werden sollen. Ziel eines ID-Systems muß sein, einen durchschnittlichen Administrator soweit zu unterstützen, daß er ohne tiefgreifende Kenntnisse im Bereich Internetsicherheit in der Lage ist, aus einer großen Menge von Protokolldaten einen Angriff zu erkennen. IR-Systeme sollen dagegen dazu dienen, automatische Gegenmaßnahmen einzuleiten, sobald ein Angriff erkannt wurde.
Im Idealfall verfügen diese Programme über ebensoviel Kenntnisse wie ein guter Administrator und sind daher in der Lage, in beliebigen Protokolldaten nicht nur einen Angriff zu erkennen, sondern auch noch Aussagen über die Stärke der Bedrohung und die notwendigen Gegenmaßnahmen zu machen. Zur Zeit ist das allerdings noch ein Gebiet, das intensiv erforscht wird, so daß wesentliche Verbesserungen an den vorhandenen Programmen jederzeit möglich sind.
Genauere Ausführungen zu den folgenden Abschnitten finden Sie unter [Nort 99] und [EDZ 00].

10.1 Zeitliche Abfolge

Betrachtet man Intrusion Detection, so muß man sich Gedanken machen, zu welchen Zeitpunkten eine Auswertung der gesammelten Daten erfolgt. Dazu gibt es zwei Möglichkeiten.

10.1.1 Batch oder Intervall orientierte Auswertung

Bei der Batch orientierten, auch Intervall orientiert genannten, Variante schreiben Auditing Mechanismen des Betriebssystems oder andere Host basierende Agenten Informationen zu erkannten Ereignissen in Dateien, die dann von einen ID System periodisch nach Anzeichen von Eindringen oder Mißbrauch durchsucht werden. Dieser Vorgang ist somit nicht ständig in Aktion, sondern wird zu fest definierten Zeiten gestartet.

Vorteile:

- Batch orientierte Analyseverfahren erzeugen wenig Prozessorlast, besonders wenn die Intervalle kurz sind und deshalb das Datenvolumen gering bleibt.
- Sie eignen sich gut in Umgebungen, in denen der Bedrohungsgrad niedrig, das Verlustpotential durch Einzelangriffe aber sehr hoch ist. In diesen Umgebungen wird oft großer Wert darauf gelegt, verdächtige Vorfälle zu dokumentieren, um genügend Beweismateriel für Anzeigen zu sammeln. Der Fokus liegt daher weniger auf der sofortigen Behandlung und Abwehr von Vorfällen.

- Die Batch orientierte Sammlung und Analyse von Daten ist speziell geeignet für Organisationen, in denen System- und Personalresourcen limitiert sind. Organisationen, die kein Vollzeitsicherheitspersonal unterhalten, werden äußerst selten Echtzeit generierte Alarme bei ID Systemen handhaben können.

- Angriffe auf Computersysteme bestehen oft aus wiederholten Angriffen auf ein und dasselbe Zielsystem. Zum Beispiel, ein Angreifer kommt mittels eines Passwort-Grabbing-Angriffes auf ein System, installiert dort einen Trojaner als Hintertür, um später zurück zu kehren und den Angriff fortsetzen zu können.

- Viele aktuelle, juristische Praktiken in Verbindung mit Computern wurden eingeführt mit dem Gedanken an Batch orientiertes Sammeln und manuelles Analysieren. Somit ist es in Hinblick darauf einfacher, zur Beweisführung Systemlogdateien zu sammeln und Batch orientiert zu verarbeiten.

Nachteile:

- Bei der Überwachung werden selten Vorfälle erkannt, bevor sie abgeschlossen sind.

- Es ist keine aktive Einleitung von Gegenmaßnahmen zum Zeitpunkt des Angriffs möglich.

- Da Batch orientierte Analysen ein vorheriges Sammeln von Daten voraussetzen, kann das zu einer hohen Speicherplatzbelastung auf dem Analysesystem kommen.

10.1.2 Real-Time Analyse

Ein Real-Time System betreibet die Informationssammlung, Analyse der Daten und das Reporting (Berichterstattung) fortlaufend. Der Begriff Real-Time wird hier analog zu einem Prozesskontrollsystem verwendet, d.h. die Entdeckung passiert schnell genug, um den Angriff zu erkennen und durch geeignete Gegenmaßnahmen eventuell zu vereiteln. Diese Definition umfaßt sowohl Systeme, die im Millisekundenbereich Analysen durchführen, aber auch diese, die etwas langsamer arbeiten. Real-Time Systeme stellen eine Reihe von Alarmmechanismen zur Verfügung: z.B. E-Mail, Pager, Telefonnachrichten, etc., aber auch automatische Gegenmaßnahmen können eingeleitet werden. Typische Gegegenmaßnahmen reichen von der einfachen Benachrichtigung, Trennung der Netzwerkverbindung, Änderungen in der Konfiguration des angegriffenen Systems bis hin zu einem Gegenangriff.

Vorteile:

- Abhängig von der Geschwindigkeit der Analyse können Angriffe eventuell schnell genug entdeckt werden, um durch den Systemadministrator unterbrochen zu werden.

- Abhängig von der Geschwindigkeit und Sensitivität der Analyse können Systemadministratoren Vorfälle schneller behandeln, was zu einer schnelleren Wiederherstellung der Systemabläufe führt.
- Falls eine juristische Verfolgung der Vorfälle geplant ist, kann der Systemadministrator online Informationen sammeln, die eine Identifikation und Strafverfolgung der Eindringlinge effektiver machen.

Nachteile:

- Diese zeitliche Abfolge der Analyse bedarf mehr Speicherplatz und Prozessorleistung als Post-Facto-Systeme.
- Juristische betrachtet ist es fragwürdig, ob automatisch erzeugte Reaktionen auf einen Angriff, die das angreifende System schädigen könnten, legal sind.
- Die Konfiguration von Real-Time Systemen ist kritisch. Eine schlecht formatierte Signatur (siehe Abschnitt 10.2.1) kann so viele falsche Alarme erzeugen, daß ein echter Angriff übersehen werden kann.

10.2 Arten der Analyse

Nicht nur in der zeitlichen Abfolge gibt es unterschiedliche Ansätze, sondern auch in der Art, wie Angriffe zu erkennen sind.

10.2.1 Signaturanalyse

Bei der Signaturanalyse werden für jeden bekannten Angriff bestimmte Merkmale extrahiert, die diesen Angriff unverwechselbar kennzeichnen. Diese Signaturen werden in einer Datenbank gespeichert und mit den aktuellen Protokolldaten verglichen. Signaturen sind somit Muster von bekannten Angriffen oder Mißbräuchen von Systemen. Signaturen können in ihrer Komplexität sehr verschieden sein. Dies kann von sehr einfachen Mustern, wie z.B. Zeichenübereinstimmungen bei Sätzen oder Befehlen, bis hin zu komplexen Sicherheitsstati in Form von mathematischen Ausdrücken gehen. Generell kann eine Signatur Bezug auf einen Prozess (spezielle Ausführung eines Befehls) oder auf ein Ereignis (z.B. bei der Anforderung einer Root-Shell) nehmen.
Signaturanalyse ist das Vergleichen von Systemeinstellungen und Benutzeraktivitäten mit einer Datenbank, die bekannte Angriffsmuster enthält. Die meisten kommerziellen Intrusion Detection Systeme führen eine Signaturanalyse gegen eine vom Hersteller gelieferte Datenbank mit bekannten Mustern durch. Zusätzliche Signaturen, die durch den Kunden spezifiziert werden, können während der Konfigurationsphase eingefügt werden. Die meisten Hersteller bieten als Teil ihres Softwarewartungsabkommens einen periodischen Update der Signaturdatenbank an.

Ein Vorteil der Signaturanalyse ist, daß sie Sensoren ermöglicht, ein zielgerichtetes Sammeln von Systemdaten durchzuführen und somit den Overhead zu verringern.
Solange die Signaturdatenbank nicht zu groß ist, ist die Signaturanalyse wegen der fehlerhaften Flieskommarechnung wesentlich effizienter als die statistische Analyse, die wir im nächten Abschnitt behandeln werden.
Ein Beispiel ist die Erkennung eines Portscanningangriffes. Der Angreifer testet in einer schnellen Abfolge alle Portnummern einer Maschine, um festzustellen, welche Dienste auf dem Rechner verfügbar sind. Eine einfache Angriffssignatur sähe also so aus:

```
TCP-Verbindungsaufbau Pakete (ACK nicht gesetzt) im Anstand von max. 1s:
10.10.10.10 an 10.11.12.13 Port 1
10.10.10.10 an 10.11.12.13 Port 2
10.10.10.10 an 10.11.12.13 Port 3
10.10.10.10 an 10.11.12.13 Port 4
....
10.10.10.10 an 10.11.12.13 Port 65636
```

Hier zeigt sich sofort eine Schwierigkeit: Was passiert, wenn die Pakete nicht in festen, zeitlichen Abständen eintreffen, sondern in variablen Abständen auf zufällig gewählte Ports? Ist eine Abfogle von 100 Paketen zu verschiedenen Ports auch als Angriff zu bewerten? Das Hauptproblem bei der Signaturanalyse ist allerdings, daß es niemals möglich ist, alle denkbaren Angriffe zu erkennen und durch entsprechende Signaturen zu beschreiben. Die Systeme würden dabei dann auch viel zu langsam werden. Ein IDS, das mit Signaturanalyse arbeitet, kann also immer nur so gut sein wie die verwendete Datenbank mit Angriffssignaturen. Sie sind sehr stark vergleichbar mit den herkömmlichen Virenschutzprogrammen.
Typische Beispiele für Netzwerk basierende Echtzeit Intrusion Detection Systeme mit Signaturanalyse sind Real Secure von ISS [rea 02] und die freie Software Snort [sno 02].

10.2.2 Statistische Analyse, Anomalie-Analyse

Die statistische Analyse ermittelt Abweichungen von normalen Verhaltensmustern. Dieses Verfahren ist üblich in Forschungsumgebungen und findet sich nur in einigen kommerziellen IDS-Produkten wieder.
Statistische Profile werden für Systemobjekte über einen längeren Zeitraum hin erstellt, z.B. Benutzer, Dateien, Verzeichnisse, Geräte, etc., indem die verschiedenen Attribute des normalen Gebrauchs ermittelt werden, wie etwa Anzahl der Zugriffe, Anzahl der Fälle, in denen eine Operation scheitert, Tageszeit, usw.. Daraus errechnet sich die normale Anzahl und die Möglichkeit der Abweichung bei regulärer Benutzung. Mögliches Eindringen wird signalisiert, wenn entdeckt wird, daß diese Werte sich außerhalb des normalen Rahmens bewegen.
Eine statistische Analyse könnte z.B. die Anmeldung eines Benutzers um 03:00 Uhr als ungewöhnlichen Vorfall melden, wenn sich dieser Benutzer im Normalfall nur zwischen

08:00 Uhr und 18:00 Uhr angemeldet hat.

Vorteile von statistischen Verfahren:

- Es werden auch unbekannte Angriffe entdeckt.
- Es können Unregelmäßigkeiten im Userverhalten erkannt und somit unberechtigte Zugriffe entdeckt werden.
- Statistische Methoden erlauben es, komplexe Angriffe zu erkennen.

Nachteile der statistischen Analyse, die zur Zeit relevant sind:

- Es ist relativ einfach für einen Angreifer, sein Verhalten über einen längeren Zeitraum hinweg so zu verändern, daß ein Angriff als normales Benutzerverhalten interpretiert wird.
- Statistische Verfahren kommen nicht gut mit Änderungen im Benutzerverhalten zurecht. Das kann zu einer Vielzahl von Alarmen oder sogar zu einem übersehenen Angriff führen.

Ein typisches Beispiel für Netzwerk basierendes, Batch orientiertes ID Systeme mit statisitscher Analyse ist die freie Software Shadow [sha 01].

10.2.3 Integritätsanalyse

Bei einem Dateienintegritätscheck werden die Dateien auf einem Computer daraufhin untersucht, ob sie seit der letzten Überprüfung geändert wurden. Der Integritätschecker pflegt eine Datenbank mit Hashwerten für jede Datei. Jedesmal, wenn der Checker läuft, berechnet er die Hashwerte und vergleicht sie mit den gespeicherten Werten. Wenn sich die Hashwerte unterscheiden, wurde die Datei verändert. Ansonsten liegt keine Veränderung vor.

Stärken:

- Rechnerisch ist es unmöglich, die Mathematik eines Integritätscheckers zu besiegen. Das macht ihn zu einem sehr starken Werkzeug, um Veränderungen an Dateien zu entdecken.
- Die Konfiguration eines Integritätscheckers ist sehr individuell anpassbar. Die Überwachug kann sowohl das ganze System betreffen oder auch nur für explizit definierte Dateien oder Verzeichnisse aktiviert werden.

- Nach einem geglückten Einbruch werden in der Regel von dem Angreifer Binaries und Logdateien verändert, um die Spuren zu verwischen und ein späteres Zurückkehren in das System zu ermöglichen. Diese Veränderungen werden durch einen Integritätschecker erkannt. Möchte ein Angreifer diese Meldungen verhindern, so muß er den Checker deaktivieren, was dann aber wieder zur Folge hat, daß alle Bericht des Checkers ausbleiben, was dann wiederum das Mißtrauen des Administrators wecken sollte.

Schwächen:

- Alle Daten, die von einem Integritätschecker verwendet werden, liegen auf dem zu überwachenden System und können ebenfalls verändert werden. Diesem Problem kann entgegengetreten werden, indem die Vergleichsdaten auf einem Read-Only Medium, z.B. CD, gespeichert werden. Treten nun gewollte Konfigurationsänderungen auf, so muß der Datenbestand erneut mittels Hashwerten in die Datenbank aufgenommen werden, was natürlich bei einer CD nicht ganz so einfach ist. Die elegantere Lösung ist die Verschlüsselung der Vergleichsdatenbank. Somit werden unbefugte Veränderungen erschwert und vom Administrator gewollte Änderungen können online sofort wieder in den Datenbestand aufgenommen werden.

- Ein Integritätschecker muß für jedes System individuell konfiguriert werden.

- Integritätschecker benötigen gewaltige Systemresourcen wie Prozessorzeit, Speicher und Plattenplatz.

Ein typisches Beispiel für einen Host basierenden, Batch orientierten Integritätschecker ist die freie Software Tripwire [tri 02a].

10.3 Host basierende IDS-Systeme

Ein Host basierendes IDS achtet auf Anzeichen eines Eindringens auf dem lokalen System. Es benutzt die Auditing und Logging Mechanismen des Hostsystems als Informationsquelle für die Analyse. Diese IDS Architektur benutzt generell regelbasierte Engines.

Stärken:

- Ein Host basierndes IDS kann ein extrem mächtiges Werkzeug sein, um mögliche Angriffe zu analysieren. Z.B. kann dieses IDS exakt rekonstruieren, was ein Angreifer getan hat, welche Befehle er benutzt hat, welche Dateien er geöffnet hat und welche Systemaufrufe von ihm gestartet wurden. Eine Voraussetzung hierfür ist natürlich, daß die Logfiles und Auditingmechanismen nicht verändert wurden.

- Durch ein Host basierendes IDS werden detailiertere und oft auch relevantere Daten geliefert als das bei einem Netzwerk basierenden IDS (vgl. Abschnitt 10.5) der Fall sein kann.
- Host basierende Systeme tendieren dazu, weniger falsche Alarme zu generieren als Netzwerk basierende, da z.B. bei der Ausführung eines Befehls klar ist, was damit erreicht wird.
- Host basierende Systeme können komplett in sich abgeschlossen sein.
- Bei Host basierenden Systemen können bei erkannten Angriffen schnell automatisch aktive Gegenmaßnahmen eingeleitet werden, z.B. Ausloggen und Sperren eines Benutzers oder das Beenden eines Dienstes. Das hierbei eingegangene Risiko ist abschätzbar.
- Die Wahrscheinlichkeit, ein Host basierendes System zu hintergehen, ist äußers gering.
- Auch Angriffe über verschlüsselte Sessions können hier erkannt werden, da die Verschlüsselung auf dem Zielsystem endet.

Schwächen:

- Ein Host basierendes IDS muß auf jedem zu überwachenden System installiert und konfiguriert werden.
- Host basierende IDS verlassen sich auf servereigene Logging- und Überwachungsmechanismen.
- Bei Produktivsystemen kann der Einsatz eines Host basierenden IDS zu einem erhöhten Managementaufwand führen.
- Die Lizenzkosten für kommerzielle Host basierende IDS sind sehr hoch.
- Alle Ereignisse, die sich außerhalb des Hosts abspielen, werden nicht betrachtet.

Eine spezielle Unterart von Host basierenden IDS sind File Integrety Checker. Diese überprüfen in bestimmten Abständen, ob sich die zu überwachenden Dateien eines Systems verändert haben.

10.4 Intregritätschecks mit TRIPWIRE

Ursprünglich wurde die Tripwire Software an der Purdue University durch die IT-Sicherheitsexperten Professor Dr. Eugene Spafford und Gene Kim entwickelt. Tripwire gilt als das meist geschätzte Tool zum Erkennen von unerlaubten Änderungen an Betriebssystemen und kritischer Anwendungssoftware. Zur Zeit ist Tripwire entweder als frei

erhältliche Software von Purdue oder als kommerzielle Lösung mit Herstellersupport von Tripwire, Inc. [tri 02b] erhältlich.
Tripwire 2.3 erzeugt eine Datenbank aller derzeit gültigen Stati mit kryptographischen Prüfsummen aller angegebenen Dateien und Verzeichnisse des Betriebssystems und der Anwendungssoftware. Der so abgelegte Status wird in periodischen Abständen mit dem aktuellen Status verglichen. Zwei Dinge müssen hier erfüllt sein, damit keine unerlaubte Veränderung vorliegt: zum einen muß der Initialstatus als sicher einstufbar sein, zum anderen müssen die Hashwerte in der Datenbank und die neu erzeugten Hashfunktionen bei jedem gefahrenen Test unverändert sein.
Tripwire stellt ein extrem nützliches Tool zur Überwachung aller Konfigurationsänderungen auf einem System dar. Somit kann Tripwire alle möglichen Arten von bekannten und auch unbekannten Angriffen entdecken, die Änderungen am System vornehmen. Diese Tatsache macht Tripwire bei Intrusion Response Verfahren so wertvoll.
Tritt ein Angriff mit wiederkehrenden Dateiänderungen auf, so kann Tripwire alarmieren und Anhaltspunkte geben, um herauszufinden, was gerade passiert. Die frei verfügbare Variante von Tripwire ist in beinahe jeder Linux Distribution vertreten. Somit ist es in diesem Fall besonders einfach, den sicheren Anfangszustand eines Systems festzustellen.
Bei SuSE 8.0 wird Tripwire 1.2 mitgeliefert. Hier ist noch keine Verschlüsselung der Datenbank integriert. Es kann aber das RPM-Paket für Redhat 7.x von `http://www.tripwire.org/` [tri 02a] heruntergeladen und mit `rpm --install` installiert werden. Das Programm `siggen` muß auf der Maschine vorhanden sein. Das kann mit `which siggen` überprüft werden.

```
sectest:/home/oberhait/tripwire # rpm --install tripwire-2.3-47.i386.rpm

----------------------------------------------
Generating Tripwire configuration file...

----------------------------------------------
Customizing default policy file...

A clear-text version of the Tripwire policy file
/etc/tripwire/twpol.txt
has been created for your inspection.  This implements
a minimal policy, intended only to test essential
Tripwire functionality.  You should edit the policy file
to describe your system, and then use twadmin to generate
a signed copy of the Tripwire policy.

IMPORTANT:  To complete the Tripwire 2.3 installation,
you must run the configuration script:
/etc/tripwire/twinstall.sh
```

```
This script walks you through the processes of
setting passphrases and encrypting the policy
and configuration files.  If you wish to change the
contents of your policy file, /etc/tripwire/twpol.txt
you may want to do so before running this script.

----------------------------------------------
The installation succeeded.

Please refer to /usr/doc/tripwire/README
for release information and to the printed user documentation for
further instructions on using Open Source Tripwire 2.3 for Linux.

sectest:/home/oberhait/tripwire #
```

Das Hauptkonfigurationsverzeichnis wird unter `/etc/tripwire` angelegt. Kurzanleitungen zu Tripwire sind unter `/usr/doc/tripwire/` zu finden, Dokumentationen liegen auf [tri 02a] als PDF-Files zum Download bereit. Um die Installation abzuschließen, muß `/etc/tripwire/twinstall.sh` ausgeführt werden. Hierbei werden zwei Schlüsselpaare erzeugt, mit denen die Verschlüsselung der Informationen erfolgt.

```
sectest:/home/oberhait/tripwire # /etc/tripwire/twinstall.sh

----------------------------------------------
The Tripwire site and local passphrases are used to
sign a variety of files, such as the configuration,
policy, and database files.

Passphrases should be at least 8 characters in length
and contain both letters and numbers.

See the Tripwire manual for more information.

----------------------------------------------
Creating key files...

(When selecting a passphrase, keep in mind that good passphrases typically
have upper and lower case letters, digits and punctuation marks, and are
at least 8 characters in length.)

Enter the site keyfile passphrase: server123
Verify the site keyfile passphrase:
Generating key (this may take several minutes)...Key generation complete.
```

```
(When selecting a passphrase, keep in mind that good passphrases typically
have upper and lower case letters, digits and punctuation marks, and are
at least 8 characters in length.)

Enter the local keyfile passphrase: praktikum
Verify the local keyfile passphrase:
Generating key (this may take several minutes)...
----------------------------------------------
Signing configuration file...
Please enter your site passphrase:
Wrote configuration file: /etc/tripwire/tw.cfg

A clear-text version of the Tripwire configuration file
/etc/tripwire/twcfg.txt
has been preserved for your inspection.  It is recommended
that you delete this file manually after you have examined it.


----------------------------------------------
Signing policy file...
Please enter your site passphrase:
Wrote policy file: /etc/tripwire/tw.pol

A clear-text version of the Tripwire policy file
/etc/tripwire/twpol.txt
has been preserved for your inspection.  This implements
a minimal policy, intended only to test essential
Tripwire functionality.  You should edit the policy file
to describe your system, and then use twadmin to generate
a new signed copy of the Tripwire policy.

sectest:/home/oberhait/tripwire #
```

Folgende Files sind nun in `/etc/tripwire` erzeugt worden:

```
sectest:~ # ls -l /etc/tripwire/
total 160
drwxr-xr-x    2 root     root         4096 Jun 29 15:59 .
drwxr-xr-x   40 root     root         4096 Jun 29 14:57 ..
-rw-r-----    1 root     root          931 Jun 29 13:53 sectest-local.key
-rw-r-----    1 root     root          931 Jun 29 13:52 site.key
-rw-r-----    1 root     root         4586 Jun 29 13:54 tw.cfg
```

```
-rw-r-----    1 root     root         4159 Jun 29 15:59 tw.pol
-rw-r--r--    1 root     root          480 Jun 29 13:50 twcfg.txt
-rwxr-x---    1 root     root        10112 Jun 29 13:50 twinstall.sh
-rwxr-xr-x    1 root     root        10109 Aug 15  2000 twinstall.sh.bak
-rw-r--r--    1 root     root        41256 Jun 29 13:50 twpol.txt
```

- `sectest-local.key`: Hier ist der **local key** abgelegt, nach dessen Passphrase bei der Ausführung von `twinstall.sh` gefragt wurde.
- `site.key`: Hier ist der **site key** abgelegt, nach dessen Passphrase bei der Ausführung von `twinstall.sh` gefragt wurde.
- `tw.cfg`: ist die von Tripwire benötigte Konfiguration, die aus dem Textfile `twcfg.txt` bei der Ausführung von `twinstall.sh` erzeugt wurde.
- `tw.pol`: ist die von Tripwire verwendete Policy. Ist nur die maschinenlesbare Policy vorhanden, so kann mit

  ```
  twadmin --print-cfgfile > twpol.txt-erzeugen
  ```

  ein ASCII File `twpol.txt-erzeugen` generiert werden, in dem die Policyinformationen menschenlesbar dargestellt sind. Hierbei wird die Passphrase für den **site key** abgefragt. `tw.pol` wird in der Regel aus dem ASCII File `twpol.txt` mit

  ```
  twadmin --create-polfile twpol.txt
  ```

  neu erzeugt. Dazu ist die Eingabe der Passphrase für den **local key** nötig.
- `twcfg.txt`: Dieses File beinhaltet die Hostkonfiguration für Tripwire, aus der dann das binäre Konfigurationsfile erzeugt wird. Die hier verwendeten Einstellungen werden bei der Ausführung von `twinstall.sh` aus dem System ausgelesen:

  ```
  ROOT          =/usr/sbin
  POLFILE       =/etc/tripwire/tw.pol
  DBFILE        =/var/lib/tripwire/$(HOSTNAME).twd
  REPORTFILE    =/var/lib/tripwire/report/$(HOSTNAME)-$(DATE).twr
  SITEKEYFILE   =/etc/tripwire/site.key
  LOCALKEYFILE  =/etc/tripwire/sectest-local.key
  EDITOR        =/bin/vi
  LATEPROMPTING =false
  LOOSEDIRECTORYCHECKING =false
  MAILNOVIOLATIONS =true
  EMAILREPORTLEVEL =3
  REPORTLEVEL   =3
  ```

```
MAILMETHOD    =SENDMAIL
SYSLOGREPORTING =false
MAILPROGRAM   =/usr/lib/sendmail -oi -t
```

- `twpol.txt`: ist das auf Red Hat ausgelegte Policyfile, das von Hand editiert werden kann. Darin ist festgelegt, welche Verzeichnisse und Dateien mit welcher Priorität zu behandeln sind. Für SuSE 8.0 muß das File angepaßt werden, da oft die Verzeichnispfade nicht übereinstimmen.
- `twinstall.sh`: Ist das auszuführende Installationsskript, um die Installation abzuschließen. `twinstall.sh.bak` ist eine Vorgängerversion und kann außer Acht gelassen werden.

Unter `/var/lib/tripwire` findet man die Referenzdatenbank `sectest.twd` und unter dem Unterverzeichnis `report` die durch Vergleichsläufe erzeugten Berichte.
Eine ganz einfache und sicherlich auch noch sehr verbesserungswürdige Policykonfiguration könnte so aussehen:

```
#######################################
# Tripwire Policy TCS/PS, Barbara Oberhaitzinger, 020629
# SuSE
#######################################

#######################################
# Definitionen der Verzeichnisse von Tripwire
#######################################
@@section GLOBAL
TWDOCS="/usr/doc/tripwire";
TWBIN="/usr/sbin";
TWPOL="/etc/tripwire";
TWDB="/var/lib/tripwire";
TWSKEY="/etc/tripwire";
TWLKEY="/etc/tripwire";
TWREPORT="/var/lib/tripwire/report";
HOSTNAME=sectest;
######################################

######################################
# Definitionen der Wertigkeiten
######################################
@@section FS
SEC_CRIT      = $(IgnoreNone)-SHa ;  # Critical files that cannot change
SEC_SUID      = $(IgnoreNone)-SHa ;  # Binaries with the SUID or SGID flags set
```

```
SEC_BIN       = $(ReadOnly) ;        # Binaries that should not change
SEC_CONFIG    = $(Dynamic) ;         # Config files that are changed
                                     # infrequently but accessed often
SEC_LOG       = $(Growing) ;         # Files that grow, but that should never
                                     # change ownership
SEC_INVARIANT = +tpug ;              # Directories that should never change
                                     # permission or ownership
SIG_LOW       = 33 ;                 # Non-critical files that are of minimal
                                     # security impact
SIG_MED       = 66 ;                 # Non-critical files that are of
                                     # significant security impact
SIG_HI        = 100 ;                # Critical files that are significant
                                     # points of vulnerability
######################################

######################################
# Schutz der Tripwire Binaries
######################################
(
  rulename = "Tripwire Binaries",
  severity = $(SIG_HI)
)
{
  $(TWBIN)/siggen                      -> $(SEC_BIN) ;
  $(TWBIN)/tripwire                    -> $(SEC_BIN) ;
  $(TWBIN)/twadmin                     -> $(SEC_BIN) ;
  $(TWBIN)/twprint                     -> $(SEC_BIN) ;
}
######################################

######################################
# Tripwire Data Files
######################################
(
  rulename = "Tripwire Data Files",
  severity = $(SIG_HI)
)
{
  $(TWDB)                              -> $(SEC_CONFIG) -i ;
  $(TWPOL)/tw.pol                      -> $(SEC_BIN) -i ;
  $(TWPOL)/tw.cfg                      -> $(SEC_BIN) -i ;
  $(TWLKEY)/$(HOSTNAME)-local.key      -> $(SEC_BIN) ;
  $(TWSKEY)/site.key                   -> $(SEC_BIN) ;
```

```
  $(TWREPORT)                        -> $(SEC_CONFIG) (recurse=0) ;
}
######################################

######################################
# Verschiedene Mountpoints, alles eigene Patitionen
######################################
(
  rulename = "Low Security Mountpoints",
  severity = $(SIG_LOW)
)
{
  !/dev ;
  !/proc ;
  !/tmp ;
  !/var/lib ;
  !/var/log ;
  /home                              -> $(SEC_LOG) ;
  /var                               -> $(SEC_LOG) ;
}
(
  rulename = "Medium Security Mountpoints",
  severity = $(SIG_MED)
)
{
  /                                  -> $(SEC_BIN) ;
  /boot                              -> $(SEC_BIN) ;
  /lib                               -> $(SEC_BIN) ;
  /lost+found                        -> $(SEC_BIN) ;
  /root                              -> $(SEC_BIN) ;
}
(
  rulename = "High Security Mountpoints",
  severity = $(SIG_HI)
)
{
  /bin                               -> $(SEC_BIN) ;
  /etc                               -> $(SEC_CONFIG) ;
  /sbin                              -> $(SEC_BIN) ;
  /usr                               -> $(SEC_BIN) ;
}
```

Dabei ist zu bedenken, daß für jedes dieser Files und Verzeichnisse eine Hashfunktion erzeugt werden muß, was einiges an Zeit in Anspruch nimmt. Soll Tripwire produktiv eingesetzt werden, so muß man sich zuerst Gedanken machen, welche Anwendungen auf der Maschien laufen und was zu betrachten ist. Im Normalfall ist wichtig, daß sich die Konfigurationsfiles und die Binaries nicht ändern. D.h. es sollte `/etc`, `/usr`, `/sbin` und `/bin` überwacht werden. Hat man nun seine Policy festgelegt und das Policyfile erstellt, so muß das von Tripwire eingelesene Policyfile aus dem ASCII File erzeugt werden:

```
twadmin --create-polfile twpol-test-bo.txt
```

wobei `twpol-test-bo.txt` das gerade verfaßte ASCII Policyfile ist. Dazu ist die Passphrase für den **site key** nötig. Nun muß die Tripwiredatenbank initialisiert werden. Hierzu wird mit `tripwire --init` die Referenzdatenbank `/var/lib/tripwire/sectest.twd` erzeugt. Das kann einige Zeit in Anspruch nehmen.
Verändert man nun z.B. die `/etc/passwd`, so würde das nach einem Checklauf so aussehen:

```
sectest:/etc/tripwire # tripwire --check
Parsing policy file: /etc/tripwire/tw.pol
*** Processing Unix File System ***
Performing integrity check...
Wrote report file: /var/lib/tripwire/report/sectest-20020629-160623.twr


Tripwire(R) 2.3.0 Integrity Check Report

Report generated by:          root
Report created on:            Sat Jun 29 16:06:23 2002
Database last updated on:     Never

===============================================================================
Report Summary:
===============================================================================

Host name:                    sectest
Host IP address:              10.50.187.55
Host ID:                      None
Policy file used:             /etc/tripwire/tw.pol
Configuration file used:      /etc/tripwire/tw.cfg
Database file used:           /var/lib/tripwire/sectest.twd
Command line used:            tripwire --check

===============================================================================
Rule Summary:
```

```
===============================================================================

-------------------------------------------------------------------------------
  Section: Unix File System
-------------------------------------------------------------------------------

  Rule Name                       Severity Level    Added    Removed  Modified
  ---------                       --------------    -----    -------  --------
* Tripwire Data Files             100               1        0        0
  Tripwire Binaries               100               0        0        0
* High Security Mountpoints       100               0        0        1
  (/etc)

Total objects scanned:  3298
Total violations found:  2

===============================================================================
Object Summary:
===============================================================================

-------------------------------------------------------------------------------
# Section: Unix File System
-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
Rule Name: Tripwire Data Files (/var/lib/tripwire)
Severity Level: 100
-------------------------------------------------------------------------------

Added:
"/var/lib/tripwire/sectest.twd"

-------------------------------------------------------------------------------
Rule Name: High Security Mountpoints (/etc)
Severity Level: 100
-------------------------------------------------------------------------------

Modified:
"/etc/passwd"

===============================================================================
Error Report:
===============================================================================
```

```
No Errors

-------------------------------------------------------------------------------
*** End of report ***

Tripwire 2.3 Portions copyright 2000 Tripwire, Inc. Tripwire is a registered
trademark of Tripwire, Inc. This software comes with ABSOLUTELY NO WARRANTY;
for details use --version. This is free software which may be redistributed
or modified only under certain conditions; see COPYING for details.
All rights reserved.
Integrity check complete.
```

Es wurde erkannt, daß die `/etc/passwd` verändert und die `/var/lib/tripwire/sectest.twd` hinzugefügt wurde. Gehen wir nun davon aus, daß diese Änderungen gewollt waren. D.h. der nun sichtbare Status der Dateien muß als akzeptiert in die Refernzdatenbank aufgenommen werden. Es wäre aber nicht sinnvoll, die komplette Datenbank neu zu initialisieren. Ein solcher Datenbankupdate kann auf zwei Arten durchgeführt werden:

- Update anhand eines bestehenden Berichtes:

  ```
  tripwire --update --twrfile /var/lib/report/$(HOSTNAME)-DATUM.twr
  ```

  Dazu ist die Eingabe der Passphrase des **local keys** nötig.

- Interaktiver Vergleichslauf mit Angabe, welche Änderungen erwünscht sind:

  ```
  tripwire --check --interactive
  ```

  Alle Inkonsistenzen werden in dem Editor, der durch das `twcfg.txt` File festgelegt wurde (hier vi-Editor), aufgelistet. Dort kann man durch Entfernen der vorangestellten Kreuze den Update für bestimmte Files verhindern. Nach dem Abspeichern wird der Update der Datenbank angestoßen. Die Eingabe der Passphrase für den **local key** ist nötig.

Sollen nun Änderungen in dem Policyfile vorgenommen werden, so muß das von Tripwire verwendete `tw.pol` nicht komplett neu initialisiert werden, sondern mit

```
tripwire --update-policy twpol.txt
```

kann ein Update erfolgen. Dazu ist sowohl die Passphrase für den **local key** wie auch die für den **site key** nötig. Im Anschluß daran muß natürlich wieder die Datenbank erneuert

werden.
Alle diese ASCII Files sollten auf dem zu überwachenden System nicht mehr vorhanden sein, damit einem möglichen Einbrecher die Informationen, was überwacht wird, nicht frei zugänglich sind.
Aus Sicherheitsgründen sollten auch die Passphrases für die Keys von Zeit zu Zeit gewechselt werden:

```
twadmin --generate-keys --verbose --site-keyfile /etc/tripwire/site.key
twadmin --generate-keys --verbose --local-keyfile /etc/tripwire/sectest-local.key
```

Für weiter Informationen empfiehlt sich die FAQ-Seite auf [tri 02a] und die mit dem Sourcecode mitgeleiferte Dokumentation, die man auch unter [sou 02] finden kann.
Eine Möglichkeit, um diese Checks zu automatisieren, ist die Einrichtung eines Cronjobs

```
less /etc/cron.d/tripwire
0 6 * * * root /usr/sbin/tripwire --check 2>&1
         | /usr/bin/mail -s Tripwireoutput admin@[10.10.10.10]
```

wobei in diesem Fall der durch den Vergleichslauf erzeugte Output per Mail an die Kennung `admin` auf der Maschine `10.10.10.10` geschickt wird. In der Dokumentation zu Tripwire findet man auch noch andere Möglichkeiten, diese Vorgänge zu automatisieren.

10.5 Netzwerk basierende IDS-Systeme

Bei einem Netzwerk basierenden Intrusion Detektion System handelt es sich um einen Netzwerksniffer mit Analysetool und eine Managementstation. Der Sniffer nimmt die Daten auf und das Analysetool verarbeitet sie weiter, um sie auf eventuelle Angriffe hin zu analysieren.
Der Netzwerksensor meldet nun die erkannten Unregelmäßigkeiten mittels Alarme an die Managementstation, die diese für den Operator darstellt. Netzwerksensor und Mangementstation sind die logischen Komponenten, können sich aber auf dem gleichen System befinden. Netzwerksensoren besitzen in der Regel zwei oder mehr Netzwerkkarten. Eine dieser Karten ist zur Administration und dem Kontakt zur Managementstation gedacht. Die restlichen Karten sind an Netzwerksegmente angeschlossen, aus denen sie den gesamten Traffic, nicht nur die an sie gerichteten Pakete, aufnehmen und verarbeiten sollen.

Stärken:

- Bei einer Netzwerk basierenden IDS Lösung ist keine Modifikation an den Produktivservern oder Hosts nötig. Somit werden diese Systeme nicht mit zusätzlichen Aufgaben belastet.

Schwächen:

- Netzwerk basierende IDS können nur Angriffe entdecken, die sich über das Netzwerksegment bewegen, an das sie angeschlossen sind.
- Um die Leistungsanforderungen zu erfüllen, tendieren Netzwerk basierende IDS zu Signaturanalysen. Dadurch können zwar bekannt Angriffsmuster erkannt werden, komplexere oder neuartige Angriffe werden aber nicht erkannt.
- Da für jedes analysierte Paket eine große Menge an Analysedaten anfällt und somit die Leitungen zur Managementstation belastet werden, werden viele Entscheidungsprozesse auf den Sensor verlagert. Oft wird die zentrale Managementstation nur noch als Display oder Kommunikationszentrum benutzt anstatt für die Analyse.
- Netzwerk basierende IDS können keine Angriffe innerhalb verschlüsselter Verbindungen erkennen.

10.5.1 Platzierung des Sensors

Eines der größten Geheimnisse bei der Konzeptionierung von Intrusion Detection Systemumgebungen stellt die Platzierung des Sensors dar. Ein Netzwerksensor muß ein Interface in dem Segment besitzen, über das die zu überwachenden Daten fliesen. Handelt es sich dabei um ein geswitchtes Netzwerk, kann keiner der Standardswitchports verwendet werden, da dort nicht der gesamte, in diesem Netzbereich fließende Datenverkehr anfällt. Hierzu sind sogenannte Mirroring Ports nötig, an die der Traffic aller anderen Ports gespiegelt wird. Noch schwieriger wird es bei der Überwachung von Punkt-zu-Punkt-Verbindungen, wie z.B. Standleitungen oder Funkverbindungen.
Ein versierter Hacker wird bei einem Angriff natürlich versuchen, das IDS außer Kraft zu setzen. Um das zu erschweren, gibt es spezielle Netzwerktreiber, die die zum Aufgreifen von Daten bestimmten Netzwerkkarten in den sogenannten **promiscous mode** setzen. Dabei können zwar alle Daten aufgenommen werden, das Interface besitzt in diesem Segment aber keine IP-Adresse.
Sind nun alle Server in dem zu überwachenden Netzwerk mit 100 Mbit angebunden, kann es bei einer 100 Mbit Karte in einem IDS zu Problem kommen, da die Datenmengen, die auf dem Netzwerk fließen, das Fassungsvermögen des Interfaces übersteigen können. Somit ist es ratsam, abhängig von dem zu erwartenden Durchsatz eine schnellere Netzwerkkarte zu verwenden, wenn das durch den verwendeten Netzwerkzugang (Switch, Hub, etc.) unterstützt wird.
Weiterhin ist zu bedenken, daß bei der Analyse von großen Datenvolumen Problem auftreten können. Da ein IDS die Daten auf verschiedene Angriffsformen hin analysiert, sind IDS oftmals die leistungsfähigsten Systeme auf dem Markt. Auf Grund dieser Problematik ist die große Kunst bei der Konzeptionierung eines Netzwerk basierenden IDS neben der Platzierung des Sensors die Optimierung des Systems an die Anforderungen der Umgebung. Aus diesem Grund sollten alle Dienste, die nicht benötigt werden, auch nicht überprüft bzw. protokolliert werden.

10.6 Der Netzwerkanalyst SNORT

Snort ist weder ein reinrassiger Protokollanalysator noch ein vollständiges Intrusion Detection System. Er ist von beidem etwas und kann bei Incidence Response wertvolle Dienste leisten. Snort ist Freeware unter der GNU General Public License und läuft mittlerweile auf eigentlich allen Unix Arten und auch auf Windowssystemen. Die neuesten Signaturen und Updates bekommt man natürlich über die Webseite [sno 02]. Dort ist auch eine ausführliche Dokumentation zu finden. Derzeitig ist die Version 1.9.0 aktuell. Bei SuSE 8.0 ist zur Zeit die 1.8.4 mitgeliefert, was für unsere Testzwecke ausreichend ist. Auf der Webseite [sno 02] ist auch eine Vielzahl von Zusatzsoftware für Snort zu finden. Möchte man Snort als Produktivnetzwerksniffer einsetzen, so muß man sich über die Darstellbarkeit der gewonnenen Daten Gedanken machen. Die für solche Zwecke vorgesehenen GUIs können ebenfalls unter Addons von der Seite geladen werden.
Snort ist ein typischer Vertreter der Gattung Netzwerk basierendes IDS mit Signaturanalyse, das in Echtzeit arbeitet. Das Konfigurationsverzeichnis mit dem Konfigurationsfile und den Signaturen liegt unter `/etc/snort`:

```
-rw-r--r--    1 root     root         1419 Mar 23 20:55 attack-responses.rules
-rw-r--r--    1 root     root        21765 Mar 23 20:55 backdoor.rules
-rw-r--r--    1 root     root         1687 Mar 23 20:55 bad-traffic.rules
-rw-r--r--    1 root     root         3339 Mar 23 20:55 classification.config
-rw-r--r--    1 root     root         5884 Mar 23 20:55 ddos.rules
-rw-r--r--    1 root     root         3369 Mar 23 20:55 dns.rules
-rw-r--r--    1 root     root         3398 Mar 23 20:55 dos.rules
-rw-r--r--    1 root     root        10560 Mar 23 20:55 exploit.rules
-rw-r--r--    1 root     root         2986 Mar 23 20:55 finger.rules
-rw-r--r--    1 root     root         7244 Mar 23 20:55 ftp.rules
-rw-r--r--    1 root     root        15934 Mar 23 20:55 icmp-info.rules
-rw-r--r--    1 root     root         4269 Mar 23 20:55 icmp.rules
-rw-r--r--    1 root     root         1298 Mar 23 20:55 info.rules
-rw-r--r--    1 root     root          209 Mar 23 20:55 local.rules
-rw-r--r--    1 root     root         4323 Mar 23 20:55 misc.rules
-rw-r--r--    1 root     root         3130 Mar 23 20:55 netbios.rules
-rw-r--r--    1 root     root         5830 Mar 23 20:55 policy.rules
-rw-r--r--    1 root     root         1862 Mar 23 20:55 porn.rules
-rw-r--r--    1 root     root        12908 Mar 23 20:55 rpc.rules
-rw-r--r--    1 root     root         2225 Mar 23 20:55 rservices.rules
-rw-r--r--    1 root     root         4587 Mar 23 20:55 scan.rules
-rw-r--r--    1 root     root         4087 Mar 23 20:55 shellcode.rules
-rw-r--r--    1 root     root         4073 Mar 23 20:55 smtp.rules
-rw-r--r--    1 root     root        18251 Mar 23 20:55 snort.conf
-rw-r--r--    1 root     root        10069 Mar 23 20:55 sql.rules
-rw-r--r--    1 root     root         2809 Mar 23 20:55 telnet.rules
```

```
-rw-r--r--    1 root     root         1125 Mar 23 20:55 tftp.rules
-rw-r--r--    1 root     root        14853 Mar 23 20:55 virus.rules
-rw-r--r--    1 root     root         9107 Mar 23 20:55 web-attacks.rules
-rw-r--r--    1 root     root        21428 Mar 23 20:55 web-cgi.rules
-rw-r--r--    1 root     root         7635 Mar 23 20:55 web-coldfusion.rules
-rw-r--r--    1 root     root         8056 Mar 23 20:55 web-frontpage.rules
-rw-r--r--    1 root     root        18773 Mar 23 20:55 web-iis.rules
-rw-r--r--    1 root     root        43529 Mar 23 20:55 web-misc.rules
-rw-r--r--    1 root     root          676 Mar 23 20:55 x11.rules
```

Wobei die Dateien mit der Endung `.rules` die vordefinierten Signaturen darstellen. Das Konfigurationsfile heißt `snort.conf`. Hier nun die voreingestellten Werten in der `/etc/snort/snort.conf` im Überblick:

```
##################
# Variablendefinition
##################
var HOME_NET any
var EXTERNAL_NET any
var SMTP $HOME_NET
var HTTP_SERVERS $HOME_NET
var SQL_SERVERS $HOME_NET
var DNS_SERVERS $HOME_NET
var RULE_PATH ./
##################
# Preprozessorkonfiguration
##################
preprocessor frag2
preprocessor stream4: detect_scans
preprocessor stream4_reassemble
preprocessor http_decode: 80 -unicode -cginull
preprocessor rpc_decode: 111 32771
preprocessor bo: -nobrute
preprocessor telnet_decode
preprocessor portscan: $HOME_NET 4 3 portscan.log
##################
# Einbinden des Klassifikationsfiles
##################
include classification.config
##################
# Einbinden der Signaturen
##################
include $RULE_PATH/bad-traffic.rules
```

```
include $RULE_PATH/exploit.rules
include $RULE_PATH/scan.rules
include $RULE_PATH/finger.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/telnet.rules
include $RULE_PATH/smtp.rules
include $RULE_PATH/rpc.rules
include $RULE_PATH/rservices.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/tftp.rules
include $RULE_PATH/web-cgi.rules
include $RULE_PATH/web-coldfusion.rules
include $RULE_PATH/web-iis.rules
include $RULE_PATH/web-frontpage.rules
include $RULE_PATH/web-misc.rules
include $RULE_PATH/web-attacks.rules
include $RULE_PATH/sql.rules
include $RULE_PATH/x11.rules
include $RULE_PATH/icmp.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/attack-responses.rules
# include $RULE_PATH/backdoor.rules
# include $RULE_PATH/shellcode.rules
# include $RULE_PATH/policy.rules
# include $RULE_PATH/porn.rules
# include $RULE_PATH/info.rules
# include $RULE_PATH/icmp-info.rules
# include $RULE_PATH/virus.rules
# include $RULE_PATH/experimental.rules
include $RULE_PATH/local.rules
```

Mit den im ersten Teil des Konfigurationsfiles zu definierenden Variablen werden die Daten für das zu überwachende Netz festgelegt. Durch die Variable `RULE_PATH` wird definiert, in welchem Verzeichnis die Signaturen mit der Endung `.rules` zu finden sind.
Die mit `preprocessor` definierten Verfahren werden zur Paketanalyse noch vor der Detection Engine gestartet, so daß das Paket zwar schon angenommen aber noch nicht anderweitig verarbeitet worden ist. Laut [sno 02] kann das Paket in einer Art **out of band** durch diesen Mechanismus modifiziert oder analysiert werden.

- `frag2` bewirkt IP Defragmentierung: so kann erkannt werden, wenn Fragmentierungsangriffe gegen Hosts gefahren werden.
- `stream4: detect_scans` dient zur Stateful Detection des TCP Flußes: zusammen mit `detect_scans` können Stealth Scans erkannt werden.
- `stream4_reassemble` ohne Argumente bewirkt, daß nur Traffic mit vorangegangenem Dreiwegehandshake für die `stream4` Option betrachtet wird.
- `http_decode` betrachtet HTTP URLs und konvertiert sie in ASCII-Strings. Als Argumente können die HTTP Ports getrennt durch Leerzeichen angegeben werden. Um Alarme bei unicode Traffic und null bytes bei CGI Anfragen zu unterbinden, werden die Optionen `-unicode` und `-cginull` verwendet.
- `rpc_decode: 111 32771` untersucht den RPC Verkehr, um Anomalien zum normalen RPC zu erkennen. Als Argumente werden die zu überwachenden Ports mitgeliefert.
- `bo: -nobrute` scannt die Daten nach Back Orifice Traffic. `-nobrute` deaktiviert das Brute Forcing Plugin.
- `telnet_decode` überwacht Telnet und FTP Verkehr und analysiert, wann eine Abweichung des normalen Telnet oder FTP Verhaltens vorliegt.
- `portscan: $HOME_NET 4 3 portscan.log` aktiviert den Portscandetektor und legt fest, welches Netz überwacht werden soll, ab welcher Anzahl von angesprochenen Ports ein Scan vorliegt und welcher Detectionzeitraum betrachtet werden soll. Außerdem wird noch die Datei festgelegt, in der die gesammelten Informationen gespeichert werden sollen.

Das eingebundene Klassifikationsfile `classification.config` legt fest, wie die erkannten Angriffsmuster zu bewerten sind. D.h. jede Übereinstimmung mit einer Signatur kann anders gewichtet werden. Standard ist dafür die Priorität 10. Werden im Klassifikationsfile andere Prioritäten angegeben, so zählen diese. Somit erfolgt eine Klassifizierung und Wertung der Alarme, wobei 1 die höchste Priorität darstellt:

```
#
# config classification:shortname,short description,priority
#
config classification: not-suspicious,Not Suspicious Traffic,3
config classification: unknown,Unknown Traffic,3
config classification: bad-unknown,Potentially Bad Traffic, 2
config classification: attempted-recon,Attempted Information Leak,2
config classification: successful-recon-limited,Information Leak,2
config classification: successful-recon-largescale,Large Scale Information Leak,
2
```

```
config classification: attempted-dos,Attempted Denial of Service,2
config classification: successful-dos,Denial of Service,2
config classification: attempted-user,Attempted User Privilege Gain,1
config classification: unsuccessful-user,Unsuccessful User Privilege Gain,1
config classification: successful-user,Successful User Privilege Gain,1
config classification: attempted-admin,Attempted Administrator Privilege Gain,1
config classification: successful-admin,Successful Administrator Privilege Gain,
1

# NEW CLASSIFICATIONS
config classification: rpc-portmap-decode,Decode of an RPC Query,2
config classification: shellcode-detect,Executable code was detected,1
config classification: string-detect,A suspicious string was detected,3
config classification: suspicious-filename-detect,A suspicious filename was dete
cted,2
config classification: suspicious-login,An attempted login using a suspicious us
ername was detected,2
config classification: system-call-detect,A system call was detected,2
config classification: tcp-connection,A TCP connection was detected,4
config classification: trojan-activity,A Network Trojan was detected, 1
config classification: unusual-client-port-connection,A client was using an unus
ual port,2
config classification: network-scan,Detection of a Network Scan,3
config classification: denial-of-service,Detection of a Denial of Service Attack
,2
config classification: non-standard-protocol,Detection of a non-standard protoco
l or event,2
config classification: protocol-command-decode,Generic Protocol Command Decode,3
config classification: web-application-activity,access to a potentially vulnerab
le web application,2
config classification: web-application-attack,Web Application Attack,1
config classification: misc-activity,Misc activity,3
config classification: misc-attack,Misc Attack,2
config classification: icmp-event,Generic ICMP event,3
config classification: kickass-porn,SCORE! Get the lotion!,1
```

Normalerweise verwendet man als Grundlage für die Ablage der Daten eine Datenbank, bei Snort meist MySQL. Da wir hier in unserer Testumgebung ohne Datenbank arbeiten wollen, werden die Logfiles des erkannten Netzwerktraffics unter `/var/log/snort` abgelegt. Das Startskript von Snort heißt `/etc/init.d/snort`. Auf der `10.50.187.55` wurde der Snort mit der oben angegebenen Defaultkonfiguration gestartet und von der `10.50.181.1` eine Portscan mit `nmap 10.50.187.55` gestartet. Hier das Ergebnis des Portscans:

```
nmap 10.50.187.55

Starting nmap V. 2.54BETA30 ( www.insecure.org/nmap/ )
Interesting ports on  (10.50.187.55):
(The 1542 ports scanned but not shown below are in state: closed)
Port       State       Service
21/tcp     open        ftp
22/tcp     open        ssh
23/tcp     open        telnet
111/tcp    open        sunrpc
6000/tcp   open        X11
8080/tcp   open        http-proxy
12345/tcp  open        NetBus
```

Betrachtet man sich nun die Logfiles und das Logverzeichnis, bekommt man folgendes zu Gesicht:

```
drwx------    2 root     root         4096 Jun 30 19:51 10.50.181.1
drwx------    2 root     root         4096 Jun 30 19:51 10.50.187.55
-rw-------    1 root     root         1863 Jun 30 19:53 alert
-rw-------    1 root     root        81104 Jun 30 19:52 portscan.log


more portscan.log

Jun 30 19:51:52 10.50.181.1:1388 -> 10.50.187.55:80 SYN ******S*
Jun 30 19:51:52 10.50.181.1:1219 -> 10.50.187.55:5902 SYN ******S*
Jun 30 19:51:52 10.50.181.1:1220 -> 10.50.187.55:1383 SYN ******S*
Jun 30 19:51:52 10.50.181.1:1221 -> 10.50.187.55:1471 SYN ******S*
Jun 30 19:51:52 10.50.181.1:1222 -> 10.50.187.55:134 SYN ******S*
Jun 30 19:51:52 10.50.181.1:1223 -> 10.50.187.55:2005 SYN ******S*

....

Jun 30 19:51:53 10.50.181.1:2655 -> 10.50.187.55:89 SYN ******S*
Jun 30 19:51:53 10.50.181.1:2670 -> 10.50.187.55:2430 SYN ******S*
Jun 30 19:51:53 10.50.181.1:2671 -> 10.50.187.55:220 SYN ******S*
Jun 30 19:51:53 10.50.181.1:2722 -> 10.50.187.55:1543 SYN ******S*
Jun 30 19:51:53 10.50.181.1:2731 -> 10.50.187.55:2038 SYN ******S*
Jun 30 19:51:53 10.50.181.1:2748 -> 10.50.187.55:6110 SYN ******S*
Jun 30 19:51:53 10.50.181.1:2756 -> 10.50.187.55:618 SYN ******S*
```

```
more alert

[**] [100:1:1] spp_portscan: PORTSCAN DETECTED from 10.50.181.1 (THRESHOLD 4 co
nnections exceeded in 0 seconds) [**]
06/30-19:51:52.689978

[**] [1:618:1] INFO - Possible Squid Scan [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/30-19:51:52.691011 10.50.181.1:1229 -> 10.50.187.55:3128
TCP TTL:63 TOS:0x0 ID:17589 IpLen:20 DgmLen:60 DF
******S* Seq: 0x1CBBFE94  Ack: 0x0  Win: 0x3EBC  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 55651472 0 NOP WS: 0

[**] [1:620:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/30-19:51:52.695947 10.50.181.1:1255 -> 10.50.187.55:8080
TCP TTL:63 TOS:0x0 ID:17615 IpLen:20 DgmLen:60 DF
******S* Seq: 0x1C43B62D  Ack: 0x0  Win: 0x3EBC  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 55651473 0 NOP WS: 0

[**] [1:1227:1] X11 outgoing [**]
[Classification: Unknown Traffic] [Priority: 3]
06/30-19:51:53.103162 10.50.187.55:6000 -> 10.50.181.1:2132
TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:60 DF
***A**S* Seq: 0xBC08CDBD  Ack: 0x1BF7F8F2  Win: 0x16A0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 55589205 55651512 NOP
TCP Options => WS: 0
[Xref => http://www.whitehats.com/info/IDS126]

[**] [1:615:2] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
06/30-19:51:53.246758 10.50.181.1:2395 -> 10.50.187.55:1080
TCP TTL:63 TOS:0x0 ID:18765 IpLen:20 DgmLen:60 DF
******S* Seq: 0x1C5C5853  Ack: 0x0  Win: 0x3EBC  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 55651528 0 NOP WS: 0
[Xref => http://help.undernet.org/proxyscan/]

[**] [100:2:1] spp_portscan: portscan status from 10.50.181.1: 1170 connections
 across 1 hosts: TCP(1170), UDP(0) [**]
06/30-19:52:49.766811

[**] [100:3:1] spp_portscan: End of portscan from 10.50.181.1: TOTAL time(1s) h
osts(1) TCP(1170) UDP(0) [**]
```

```
06/30-19:53:49.682603


ls -l 10.50.181.1/

total 12
drwx------    2 root     root         4096 Jun 30 19:51 .
drwxr-xr-x    4 root     root         4096 Jun 30 19:51 ..
-rw-------    1 root     root          339 Jun 30 19:51 TCP:2395-1080


more 10.50.181.1/TCP\:2395-1080

[**] SCAN Proxy attempt [**]
06/30-19:51:53.246758 10.50.181.1:2395 -> 10.50.187.55:1080
TCP TTL:63 TOS:0x0 ID:18765 IpLen:20 DgmLen:60 DF
******S* Seq: 0x1C5C5853  Ack: 0x0  Win: 0x3EBC  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 55651528 0 NOP WS: 0
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+


ls -l 10.50.187.55/

total 20
drwx------    2 root     root         4096 Jun 30 19:51 .
drwxr-xr-x    4 root     root         4096 Jun 30 19:51 ..
-rw-------    1 root     root          347 Jun 30 19:51 TCP:3128-1229
-rw-------    1 root     root          359 Jun 30 19:51 TCP:6000-2132
-rw-------    1 root     root          339 Jun 30 19:51 TCP:8080-1255
```

10.7 Bewertung der Möglichkeiten von Intrusion Detection Systemen

IDS stellen eine Ergänzung der vorhandenen Sicherheitsstruktur dar. Sie fügen zusätzliche Schichten an Sicherheit hinzu und helfen, Angriffe auf die vorhandene Sicherheitsinfrastruktur zu erkennen und diese unter Umständen auch zu verhindern. Mittels der Protokolldaten eines Intrusion Detection Systems ist ein Administrator im nachhinein in der Lage, den Vorfall zu rekonstruieren und das Problem in der Sicherheit zu beheben.
Da in ein IDS Protokolldaten von verschiendenen Systemen zusammen geführt werden können, kann man sich einen Überblick verschaffen, was auf den einzelnen Systemen los ist. Verteilte Angriffsmuster sind so zu erkennen. Benutzeraktivitäten können komplett verfolgt werden. Hat sich ein Angreifer durch Erraten eines Passworts Zutritt zu einem

System verschafft, so können unerlaubte Aktivitäten auf dem System erkannt werden. Erfolgte der Zugriff von Intern oder über eine unbekannte Hintertür, so ist dieser Angriff am Firewall nicht erkennbar. Ein entsprechend platziertes IDS sollte die Unregelmäßigkeiten aber erkennen.
Es ist aber zu bedenken, daß ein IDS nicht das Allheilmittel für alle Sicherheitsprobleme darstellt. IDS sind nur als Unterstützung der Sicherheitspolitik gedacht. Für die Sicherung der Server, die Sensibilisierung der Mitarbeiter und den Einsatz von geeigneten Authentisierungsmechanismen ist nach wie vor zu sorgen. Ebenso ersetzt ein IDS niemals einen qualifizierten Administrator. Nur ein geeigneter Administrator kann das IDS nicht nur bedienen, sondern auch benötigte Änderungen und Anpassungen vornehmen. Ebenso kann keine vollständig automatische Bewertung von Vorfällen erfolgen. Auch Intrusion Detection Systeme haben Schwachstellen, die ausgenutzt werden können.

10.8 Verhalten bei einem erkannten Einbruch

Werden anhand von Protokolldaten Anzeichen für einen Angriff erkannt, so muß man auf diesen Fall vorbereitet sein. Obwohl die wenigsten Angriffe erfolgreich sind und auch bei geglückten Angriffen keine übereilten Reaktionen erfolgen sollten, muß eine Verhaltensstrategie festgelegt werden, mithilfe der eine vorgegeben Checkliste abgearbeitet werden kann. Somit wird der Gefahr von Panikaktionen vorgebeugt.
In [Fuhr 98] wird als Tip zur Erkennung von Angriffen folgendes angegeben:

> Um einen Angriff auf einen Rechner zu erkennen, können die folgenden Anzeichen verwendet werden. Wenn diese auf einem Rechner gefunden werden, sollten auf jeden Fall alle anderen auch kontrolliert werden:
>
> - Verbindungen von und zu ungewöhnlichen Rechnern.
> - Ungewöhnliche Login-Zeiten.
> - Programme mit unnötigen Rechten (z.B. `SUID`, `SGID`-Programme, etc.).
> - Intregritätsverletzungen.
> - Angriffsprogramme (z.B. Netzwerksniffer wie `tcpdump` oder Programme für Wörterbuchangriffe wie z.B. `Crack`).
> - Unnötige Dienste.
> - Veränderungen an der Passwortdatei oder den Zugriffsrechten.
> - Versteckte Dateien (unter Unix z.B. mit .. , sprich Punkt Punkt Leerzeichen, zu erzeugen).

Die folgenden Punkte sollten vorab geklärt und das daraus resultierende Verhalten geplant werden:

1. **Erkennung des Angriffs:** Anhand der Art und Weise, wie ein Angriff erkannt wird, ändert sich das Vorgehen, was passieren soll. Ist ein unerfahrener Administrator mit der Überwachung der Protokolldaten betraut, so sollte er nicht über mögliche erste Schritte alleine entscheiden, da es so zu Unverhältnismäßigkeiten in den Reaktionen kommen kann.

2. **Erste Bewertung des Angriffs und Festlegung des Verhaltens:** Zuerst muß die Schwere des Angriffs und somit die Gefährdung des zu schützenden Netzes bewertet werden. Daraus resultieren dann die zu treffenden Maßnahmen, die aber vorab konsequent beschrieben und in all ihren Folgen abgeschätzt sein müssen. Ist dem Anschein nach nur ein System kompromittiert, so kann es ausreichen, nur dieses System vom Netz zu trennen. Stellt sich aber heraus, daß mehrere Systeme betroffen sind, oder sogar die Integrität des Firewallsystems gefährdet ist, so kann es Sinn machen, den kompletten Internetzugang abzuschalten. Dazu muß bereits in der Planungs- und Realisierungsphase eines Internetzugangs und aller anderer Systeme beachtet werden, daß auch von nicht so hoch qualifiziertem Personal im Notfall die Abschaltung vorgenommen werden kann. D.h. die Beschriftung von Rechnern und Kabeln muß eindeutig sein. Ebenso muß geplant werden, wie im Falle eines Angriffs kommuniziert wird, um dem Angreifer keinen Hinweis auf seine Entdeckung zu geben. Es müssen alle Anwender generell darüber informiert sein, daß eine Abschaltung des Internetzugangs im Notfall erfolgen kann. Dies kann anhand der in Anhang A.6 beschriebenen Benutzerrichtlinie geschehen, um den Unmut von Mitarbeitern möglichst gering zu halten. Soll ein Angreifer zurückverfolgt und ausgespäht werden, so empfiehlt sich der Einsatz eines sogenannten **Quarantänerechners**, auch **Honeypot** genannt. Auf diesem Rechner werden dem Anschein nach sehr wertvolle Daten und große Sicherheitslöcher zur Erleichterung des Angriffs bereitgestellt. Während nun der Angreifer versucht, dieses System zu erstürmen und sich die Daten zieht, kann durch den Administrator das Ausspähen des Vorgehens und eine Rückverfolgung des Angreifers erfolgen.

3. **Genauere Analyse und Bewertung der Fakten:** Bislang wurde die unmittelbare Gefahr verringert, so daß nun eine genauere Analyse vorgenommen werden kann. Diese Phase kann je nach Schwere des Angriffs einige Stunden bis hin zu Monaten dauern und sollte nicht alleine von dem zuständigen Systemadministrator vorgenommen werden. Das Hinzuziehen von externem Sachverstand ist ratsam: z.B. Revisor und andere Administratoren. Auch wird empfohlen, den Hersteller und evtl. auch Mitarbeiter eines CERTs [52] hinzuzuziehen. War der Angriff so gravierend, daß ei-

[52] **Computer Emergency Response Team**: Unter diesem Namen existieren weltweit ca. 70 Gruppen, deren Aufgabe es ist, Anwendern bei einem Sicherheitsvorfall zu helfen. Sie werden häufig auch als **Incidence Response Team** (**IRT**) bezeichnet. Eine weitere Aufgabe ist es, Fehler in Programmen und Betriebssystemen zu analysieren, zusammen mit den Herstellern Lösungen oder Patches zu erarbeiten und diese zusammen mit den entsprechenden Warnmeldungen zu verteilen. Viele **CERTs** sind im **Forum of Incidence Response and Security Team** (**FIRST**) (Informationen zu den Mitgliedern finden Sie unter [fir 02]) zusammengeschlossen.

ne Strafanzeige oder personelle Konsequenzen in Betracht gezogen werden, so muß von Anfang an auf eine genaue und lückenlose Dokumentation der durchgeführten Aktivitäten Wert gelegt werden. Für diese Aufklärungsphase müssen dem verantwortlichen Administrator genügend Zeit, Hilfsmittel und Personal zur Verfügung stehen, um falls nötig, alle Protokolldaten des betroffenen Rechners und eventuell auch von anderen Rechnern einsehen und bewerten zu können. Besteht die Wahrscheinlichkeit, daß der Angriff von einer anderen Organisation aus erfolgt ist, so sollte Kontakt zu den verantwortlichen Stellen dort aufgenommen werden, um gemeinsam den Vorfall zu rekonstruieren. Es ist fraglich, ob der zuständige Administrator des angreifenden Systems miteingeschaltet werden sollte, da dieser in die Vorfälle verstrickt sein könnte. Gemeinsam kann festgestellt werden, ob das angreifende System nur als Sprungbrett benutzt wurde. D.h. der Angreifer hat zuvor schon diese Systeme eingenommen, um von ihnen aus weitere Angriffe zu fahren und seine Identität zu verschleiern. Alle diese Maßnahmen setzen voraus, daß die rechtlichen Grundlagen geschaffen worden sind, um Protokolldaten auswerten zu dürfen (siehe auch Kapitel 2.5).

4. **Sicherstellung von Beweismaterial:** Wenn möglich sollte vor einer Veränderung eine komplette Kopie des Systemzustandes der angegriffenen Maschinen gemacht werden, damit diese später als Beweismittel verwendet werden kann. Mit diesen Backups können Analysen auch auf anderen Geräten durchgeführt werden, während das betroffene Gerät nach Beseitigung des Schadens wieder eingesetzt werden kann. Es ist empfehlenswert, die Daten auf ein Read-Only Medium zu speichern, damit kein Verdacht der Verfälschung aufkommen kann. Die so erzeugten Datenträger müssen mit Datum, Uhrzeit, Name des betroffenen Systems und Name der Person, die das Medium erzeugt hat, versehen werden. Ebenso sollten alle Telefongespräche mitprotokolliert werden, die zu diesem Thema geführt wurden. Auch sollte dokumentiert werden, wie das System wieder in seinen Betriebszustand überführt wurde.

5. **Wiederherstellung eines sicheren Zustandes:** Oft dienen kompromittierte Systeme nur dazu, einen Ausgangspunkt für weitere Angriffe zu schaffen. Somit werden meistens Programme auf den Systemen installiert, die ein späteres Zurückkommen des Angreifers durch einfache Hintertüren erlauben. Ist durch einen Integritätschecker feststellbar, welche Dateien und Programme verändert oder neu installiert wurden, kann sich die Wiederherstellung auf die betroffenen Dateien beschränken. Sind keine Integritätschecker eingesetzt worden, so stellt sich die partielle Wiederherstellung des Systems als äußers schwierig dar. Oftmals muß in solchen Fällen das System komplett neuinstalliert werden. Das trifft ebenfalls zu, wenn bei den gemachten Backups nicht klar ersichtlich ist, welches Backup noch sauber und welches schon durch den Angriff verseucht ist. Grundsätzlich sollten die Backups immer deutlich mit Datum und Maschinenname gekennzeichnet sein. Ebenso sollten die Datenträger auch leicht auffindbar sein. Bei der Wiederherstellung von Systemen sollte großer Wert darauf gelegt werden, daß die Lücken und Fehler, durch die der Angriff ermöglicht wurde, ausgemerzt werden, da sonst der nächste Angriff nicht lange auf sich warten läßt.

Sind die ausgenutzten Schwachstellen erkannt, so sollte eine eindeutige Signatur dieses Angriffs in die Datenbank der automatischen Protokollauswertung aufgenommen werden.

6. **Ausführliche Dokumentation und abschließende Bewertung des Vorfalles:** Zusätzlich zur ausführlichen Dokumentation der Tätigleiten und Fakten für die rechtliche Verfolgung ist eine Dokumentation der verwendeten Schwachstellen unerlässlich. Somit können die Hersteller die entdeckte Schwachstelle durch einen Patch entschärfen. Mit Hilfe der so erstellten Dokumentation der Vorgehensweise können im nachhinein Verbesserungen in der Handlungsweise vorgenommen werden.

10.9 Praktische Aufgaben

10.9.1 Tripwire

1. Laden Sie sich das RPM Paket für Red Hat 7.x von `http://www.tripwire.org/` herunter und installieren Sie es, nachdem Sie überprüft haben, das das Programm `siggen` auf Ihrem Rechner existiert.

2. Ändern Sie das mitgelieferte Policyfile so ab, daß

   - die Verzeichnisse von Tripwire definiert sind,
   - Sie auf Ihrem Rechner nur `/etc` überwachen
   - die Wertigkeiten der zu überwachenden Files und Verzeichnisse festgelegt sind,
   - die Tripwire-Binaries überwacht werden,
   - die Tripwire-Konfigurationsfiles überwacht werden

   und generieren Sie das von Tripwire einzulesende Policyfile.

3. Initialisieren Sie die Datenbank mit dem derzeitig gültigen Stand.

4. Ändern Sie eines der in `/etc/` beheimateten Konfigurationsfiles ab, fügen Sie ein Testfile innerhalb eines Unterverzeichnisses von `/etc` hinzu. Starten Sie nun einen Integritätscheck. Was sehen Sie?

5. Der so erzeugte Stand soll anhand des erzeugten Berichtes als Ist-Stand in die Integritätsdatenbank aufgenommen werden.

6. Machen Sie die vorher gemachten Änderungen rückgängig und ändern Sie die Datenbank anhand eines interaktiven Integritätschecks.

10.9.2 Snort

1. Installieren Sie Snort über YaST2.

2. Lassen Sie die Defaultkonfiguration unverändert und starten Sie Snort über das Startskript.

3. Lassen Sie von Ihrem Partnerrechner einen Nmap auf Ihre Maschine laufen. Was sehen Sie in den Logfiles?

4. Lassen Sie einen Nessusscan auf Ihre Maschine laufen. Was sehen Sie in den Logfiles?