

Intrusion Detection für Linux Server

©2003 Ralf Spenneberg*

Revision : 1.1

Zusammenfassung

Ist meine Firewall sicher? Konnte jemand in meinen Rechner oder in mein Netzwerk einbrechen? Werde ich angegriffen?

Häufig merkt der Anwender nicht oder viel zu spät, dass er Opfer eines Angriffes wurde. Dabei ist es essentiell schnell reagieren zu können, um den Schaden möglichst gering zu halten.

Dies ist die Aufgabe der Intrusion Detection. Die Intrusion Detection unterscheidet grundsätzlich rechnerbasierte (host-based) und netzwerkbaasierte (network-based) Intrusion Detection.

Zusätzlich existiert die Intrusion Prevention. Diese versucht mit geeigneten Mitteln einen Einbruch zu vereiteln oder den Einbrecher unschädlich zu machen.

1 Einführung: warum, was, wieso?

Einer der schlimmsten Alpträume jedes Systemadministrators stellt ein Einbruch in sein Netzwerk oder einem seiner Rechner durch dritte Personen dar. Viele Administratoren entgegennen dieser Gefahr mit Sicherheitslösungen wie zum Beispiel einer Firewall und wiegen sich dann in Sicherheit. Warum benötigen wir dennoch in der heutigen EDV-Umgebung Intrusion Detection Systeme?

Intrusion detection is needed in today's computing environment because it is impossible to keep pace with the current and potential threats and vulnerabilities in our computing systems. (Sans Institute ID FAQ: http://www.sans.org/newlook/resources/IDFAQ/ID_required.htm)

Wahrscheinlich ist es unmöglich, einen im Netzwerk kommunizierenden Rechner zu 100 % sicher zu konfigurieren. Daher besteht immer die Gefahr eines möglichen Einbruches durch Dritte von Außen auf diesem Rechner.

Häufig wird jedoch auch die Gefahr eines Einbruches von Innen unterschätzt. Eine Firewall ist meist nur in der Lage Einbrüche von Außen zu erkennen. Einbrüche durch Insider werden von der Firewall nicht gemeldet. Hier kann nur ein Intrusion Detection System, welches das interne Netz und die internen Rechner überwacht, Abhilfe schaffen. Intrusion Detection Systeme, die auf der Firewall selbst installiert sind, können die meisten internen Angriffe nicht erkennen.

Bei den Intrusion Detection Systemen unterscheidet man grundsätzlich zwei verschiedene Systeme: Hostbasierte IDS (HIDS) und netzwerkbaasierte IDS (NIDS). Ein HIDS überwacht einen einzelnen Rechner, dessen Dateien, Prozesse und Benutzer. Ein NIDS überwacht die Tätigkeiten in einem Netzwerk und sucht nach fehlerhaften Paketen und den Fingerabdrücken von bekannten Angriffen.

Eine Weiterentwicklung der Idee der Intrusion Detection Systeme stellen die Intrusion Prevention Systeme dar. Diese versuchen einen Angriff oder Einbruch zu vereiteln oder in seinem Ausmaß zu beschränken. Auch hier gibt es grundsätzlich zwei verschiedene Ansätze. Einige Systeme können die Rechte des Einbrechers nach dem Einbruch so beschränken, dass er keinerlei Schaden anrichten kann. Andere Systeme erkennen einen Angriff in der Netzwerkverbindung und brechen diese direkt ab.

2 IDS selbstgebaut

Einfache und wirkungsvolle Intrusion Detection Systeme können bereits mit Linux Bordmitteln ge-

*ralf@spenneberg.net

baut werden.

Eine große Gefahr unter UNIX geht von Befehlen aus, bei denen das SetUID oder das SetGID Bit gesetzt sind. Bei der Ausführung dieser Befehle erbt der Ausführende die Rechte des Besitzers beziehungsweise der Gruppe. Ist ein Angreifer in der Lage eine Kopie des vi-Editors so mit SetUID Rechten auszustatten, dass er bei Ausführung die Rechte von root erbt, so hat er sich eine Hintertür geschaffen, über die er immer wieder root-Rechte erhalten kann. Es ist daher sinnvoll regelmäßig sämtliche Dateien des Rechners auf diese Rechte zu überprüfen. Eine derartige Liste kann sehr einfach mit dem folgenden Befehl erzeugt werden:

```
$ find / -type f -perm +6000 > \  
/etc/setuid-setgid 2>/dev/null
```

Wird diese Datei als Baseline verwendet, so kann nun täglich der aktuelle Stand mit dieser Liste verglichen werden:

```
$ find / -type f -perm +6000 2> \  
/dev/null | diff /etc/setuid-setgid -
```

Wird dieser Befehl nun über den Scheduler Cron aufgerufen, so wird automatisch eine E-Mail versandt, wenn der diff Befehl einen Unterschied erkennt.

Dieses System weist jedoch auch Schwächen auf. Erkennt der Einbrecher, dass dieses System aktiv ist, so kann er die Datei /etc/setuid-setgid modifizieren oder den Cronjob deaktivieren, so dass keine Meldung mehr erzeugt wird. Die Modifikation der Datei kann nicht erkannt werden.

3 Hostbased IDS

Moderne hostbasierte IDS bieten daher zusätzlich einen Schutz gegen die Modifikation durch den Angreifer. Hierzu werden meist digitale Signaturen oder Verschlüsselungen eingesetzt. Der Modulare Syslog, ein alternativer zentraler Protokolldienst für UNIX, bietet zum Beispiel die digitale Signatur jeder geschriebenen Protokollmeldung. So kann eine spätere Modifikation der Protokolldatei eindeutig erkannt werden.

Der System Integrity Verifier (SIV) Tripwire erzeugt eine Datenbank, in der alle überwachten Dateien mit ihren Eigenschaften und Prüfsummen ihres Inhaltes eingetragen werden. So können Ände-

rungen von Systemdateien zuverlässig erkannt werden. Um eine Modifikation der Datenbank zu verhindern, wird diese Datenbank mit einem privaten Schlüssel verschlüsselt. Eine Modifikation der Datenbank ist nur bei Kenntnis des privaten Schlüssels möglich.

Tripwire ist in der Version 2.x für Linux als Open Source veröffentlicht worden. Daher kann Tripwire ohne zusätzliche Lizenzkosten sowohl für private als auch kommerzielle Zwecke eingesetzt werden.

Die Konfiguration von Tripwire erfolgt in zwei Dateien: `twpol.txt` und `twcfg.txt` in dem Verzeichnis `/etc/tripwire/`. Die Datei `twcfg.txt` erfordert meist keine Änderung. In der Datei `twpol.txt` müssen die zu überwachenden Verzeichnisse und Dateien angegeben werden. Im folgenden ist eine kleine einfache Beispielkonfiguration dargestellt:

```
# Tripwire Richtliniendatei  
!/etc/mtab ;  
/etc -> +ugisMS (mailto="tw@spenneberg.de") ;
```

Diese Datei konfiguriert die Überwachung des Verzeichnisses `/etc` einschließlich aller Dateien und Unterverzeichnisse. Ausgenommen wird lediglich die Datei `/etc/mtab`. Modifikationen der Dateien sollen per E-Mail an `tripwire@spenneberg.de` gemeldet werden. Hierbei sollen der Besitzer (`u-user`), die Gruppe (`g-group`), der Inode (`i-inode`), die Größe (`s-size`), die MD5 (`M`) und die SHA-1 (`S`) Prüfsumme überwacht werden.

Nach der Erzeugung der Konfiguration ist die Erzeugung der Schlüssel für die Verschlüsselung der Konfigurationsdateien und der Datenbank erforderlich. Dann kann die Datenbank erzeugt werden. Hierzu werden die folgenden Befehle benötigt:

```
# /etc/tripwire/twinstall.sh  
# tripwire --init
```

Nun kann in regelmäßigen Abständen (täglich oder auch stündlich) der Stand des Systems mit der Datenbank verglichen werden.

```
# tripwire --check
```

Werden Änderungen an den überwachten Systemdateien vorgenommen, so ist eine Aktualisierung der Datenbank erforderlich, da Tripwire ansonsten diese Änderungen bei jedem Aufruf meldet.

```
# tripwire --update
```

Tripwire wird nun den lokalen Rechner auf Modifikationen untersuchen. Jede Modifikation wird nun per E-Mail an tripwire@spenneberg.de gemeldet. Der Administrator muss dann im Zweifelsfall ermitteln, ob die Modifikation von ihm oder durchgeführt wurde, oder auf einen Einbrecher zurückzuführen ist.

4 Networkbased IDS

Netzwerkbasierter Intrusion Detection Systeme überwachen den Netzwerkverkehr und melden ungewöhnliche Vorkommnisse. Hierbei kann es sich um Pakete handeln, die in dieser Form nicht erlaubt sind oder die von der Signatur typischen bekannten Angriffen entsprechen.

Das bekannteste NIDS im Open Source Bereich ist sicherlich Snort. Snort wurde von Marty Roesch geschrieben. Es bietet Funktionen, wie sie normalerweise nur von sehr teuren kommerziellen NIDS angeboten werden. Dabei liefert Snort eine erstaunliche Leistung und muss in Punkto Geschwindigkeit und Genauigkeit keinen Vergleich mit kommerziellen Lösungen scheuen. Es ist der defacto Standard in vielen Umgebungen und auch wissenschaftlichen Veröffentlichungen geworden.

Im einzelnen bietet Snort unter anderem die folgenden Funktionen:

- IP Defragmentierung
- TCP Streamreassemblierung
- UNICODE Dekodierung
- Dynamische Regeln
- Stateful IDS
- Output Plugins: XML, Datenbank, etc.
- Flexible Antworten
- Anomalie Detektor
- Back Orifice Detektor
- ARP Spoof Detektor

Im folgenden soll eine Regel zur Erkennung eines Bufferoverflows gezeigt werden:

```
alert tcp any any -> $HOME_NET any \
(msg:"Possible Bufferoverflow";flags: A+; \
content:"|9090 9090 9090 9090 9090 9090|";\
content "/bin/sh"; )
```

Ein Bufferoverflow nutzt einen Fehler des Programmierers aus, um in dem Benutzerkontext des fehlerhaften Programmes ein weiteres Programm (meist eine Shell) zu starten. Anschließend kann er auch über das Netzwerk auf diese Shell zugreifen. Wenn das fehlerhafte Programm mit `root`-Rechten gestartet wurde, so wird auch die Shell `root`-Rechte erhalten. Aus technischen Gründen enthalten Bufferoverflows meist große Bereiche mit dem Byte `0x90`. Diese kodiert den Intel Assembler Befehl `NO Operation` oder kurz `NOP`. Diese Bereiche werden daher auch als `NOP-Schlitten` oder `NOP-Sled` bezeichnet. Diese Snort Regel erkennt einen derartigen `NOP Sled` und die Zeichenkette `/bin/sh` die zum Aufruf einer Shell genutzt wird.

Ein großes Problem bei der Anwendung von Snort sind falsch positive Meldungen. Dabei handelt es sich um ein grundsätzliches Problem von Intrusion Detection Systeme. Diese IDS müssen immer in einer Anpassungsphase für die lokalen Gegebenheiten konfiguriert werden, so dass sie eine möglichst geringe Anzahl von falsch positiven Meldungen liefern. Dies ist jedoch bei kommerziellen Systemen in dem selben Maße erforderlich.

Bei einem Intrusion Detection System ist aber auch die Auswertung und Verwaltung der Meldungen ein sehr wichtiger Aspekt. Hier sollte der Administrator möglichst unterstützt werden. Für Snort existiert hier eine Reihe von Werkzeugen, die auch eine grafische Auswertung erlauben:

- Analysis Console for Intrusion Detection (ACID)
- SnortCenter
- SnortSnarf
- IDS Policy Manager

Hier soll nun ein Screenshot des ACID Systems vorgestellt werden. Dieser läßt die Möglichkeiten des Systems erahnen.

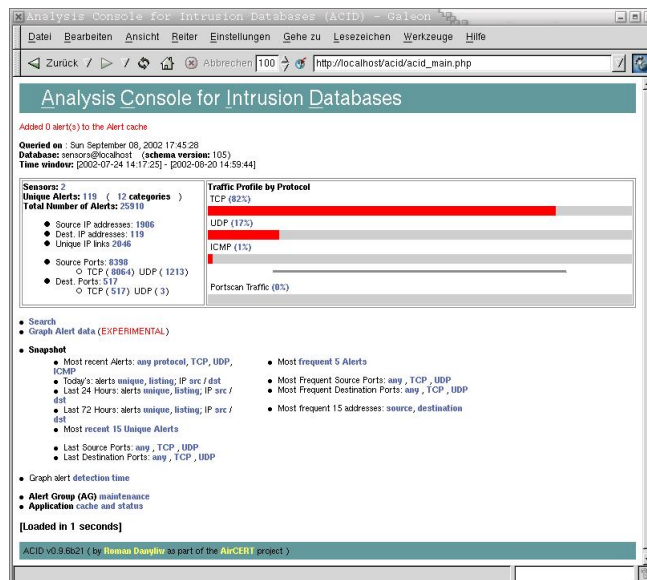


Abbildung 1: Ein web-basiertes Frontend für Snort: ACID

5 Intrusion Prevention Systems

Ein sehr bekanntes und auch schon ausreichend getestetes Intrusion Prevention System ist LIDS, das Linux Intrusion Detection System. Trotz seines Namens ist es eher den Intrusion Prevention Systemen zuzurechnen.

LIDS bietet viele verschiedene Funktionen. Die beiden Kernfunktionen des Systems sind jedoch:

- Schutz des Linux Kernels vor einem Einbruch. Wenn ein Einbrecher in der Lage ist den UNIX Kernel zu kompromittieren, so hat der Administrator kaum noch eine Möglichkeit dies zu erkennen, da er keinerlei Information des Betriebssystems mehr vertrauen darf. Jede Information kann von dem Angreifer modifiziert worden sein.
- Schutz des Systems vor der Allmacht von root. Der Systemadministrator unter UNIX ist root. root hat alle Rechte und alle Macht. Es gibt unter einem klassischen UNIX keine Möglichkeit diese Rechte einzuschränken. root ist in der Lage sich sämtliche Rechte erneut zu vergeben. LIDS bietet die Möglichkeit root spezielle Rechte zu entziehen und diese an gewisse Programme wieder zu verteilen. So kann

schließlich ein Rechner so konfiguriert werden, dass root am Ende über keinerlei privilegierte Rechte verfügt. Ein Einbrecher, der root-Status erreicht, kann daher keinerlei Schaden auf dem System anrichten.

Ein Auszug einer typischen Konfigurationsdatei für LIDS ist im folgenden abgebildet:

```
# Erlaube keinen schreibenen Zugriff auf /etc
lidsconf -A -o /etc -j READONLY
# Ausnahme für /etc/motd
lidsconf -A -o /etc/motd -j WRITE
# Ausnahme für den mount-Befehl und /etc/mtab
lidsconf -A -s /bin/mount -o /etc/mtab -j WRITE
lidsconf -A -s /bin/umount -o /etc/mtab -j WRITE
# Erlaube keinen Zugriff auf /etc/shadow
lidsconf -A -o /etc/shadow -j DENY
# Ausnahme login, su, sshd
lidsconf -A -s /bin/login -o /etc/shadow -j READONLY
lidsconf -A -s /bin/su -o /etc/shadow -j READONLY
lidsconf -A -s /usr/sbin/sshd -o /etc/shadow -j READONLY
```

Die hier definierten Rechte gelten auch für root. Das bedeutet, dass selbst root nicht in der Lage ist, die Konfigurationsdateien des Linux Systems in dem Verzeichnis /etc zu ändern. Zu Administrationszwecken ist dann die Abschaltung von LIDS durch ein Kennwort oder ein Reboot erforderlich.

Um Snort als Intrusion Prevention System zu nutzen kann die oben definierte Regel erweitert werden:

```
alert tcp any any -> $HOME_NET any \  
  (msg:"Possible Bufferoverflow";flags: A+; \  
  content:"|9090 9090 9090 9090 9090 9090|";\  
  content "/bin/sh"; resp: rst_all;)
```

Wenn ein Paket diesen Inhalt besitzt, wird jetzt die Verbindung, zu der dieses Paket gehört, zurückgesetzt (`resp: rst_all`).

6 Fazit

Für Linux existieren eine große Anzahl von Intrusion Detection und Intrusion Prevention Systemen, die es ermöglichen, eine sehr sichere Infrastruktur aufzubauen. So ist es möglich Angriffe und Einbrüche, die von einer Firewall nicht erkannt werden, dennoch zu identifizieren. Dabei ist aber eine gute Kenntnis des Betriebssystems, seiner Schwächen und Probleme erforderlich. Dies sollte jedoch nicht als Abschreckung, sondern als Herausforderung angesehen werden, da nur ein entsprechend ausgebildeter Administrator in der Lage ist Angriffe zu erkennen, abzuwehren und anschließend ähnliche Angriffe zu verhindern. Dies kann keine noch so bunte kommerzielle Software erreichen.

7 Weiterführende Literatur

- Intrusion Detection für Linux Server, Ralf Spennberg, 2002, Markt+Technik, ISBN 3827264154
- Linux Sicherheit, Tobias Klein, 2001, Dpunkt Verlag, ISBN 3932588045