



Intrusion-Detection-System Snort im Einsatz

Trüffelschwein

Markus Manzke

Das Network Intrusion Detection System (NIDS) Snort findet zwar keine teuren Delikatessen, aber dafür die besonders interessanten Datenpakete: jene, die einen Angriff, Einbruch oder andere verdächtige Aktivitäten anzeigen.

Snort von Marty Roesch ist das wohl populärste Open-Source-NIDS. Administratoren können es flexibel an die eigene Netzwerk-Infrastruktur und an Betriebssystem- und Serverversionen der zu überwachenden Systeme anpassen. Derzeit fallen darunter viele Unix-Varianten, Linux, *BSD, (Open)Solaris, SunOS, Mac OS X und Windows. Serverseitig bietet Snort Optionen für Apache, IIS sowie diverse Einstellungen für FTP, SSH und andere Protokolle.

Bei Bedarf erledigt eine Datenbank das Loggen von Alarmen. Logs autonom arbeitende Sensoren kann ein Tool wie BASE (siehe Kasten „Onlinequellen“, [a]) zentral auswerten. Mehrere Snort-Instanzen lassen sich mit unterschiedlicher Konfiguration parallel betreiben; die Konfiguration für eine Linux-Webserverfarm sähe anders aus als für Solaris-Datenbankserver oder Windows-Server im Office-Backend.

Optional läuft Snort mithilfe von Add-ons wie SnortSam, FWSnort oder snort_inline wie ein Intrusion Prevention System (IPS), sperrt also IP-Adressen angreifender Hosts dynamisch oder modifiziert gar den Traffic auf Paketebene. Das Folgende gibt einen Überblick über die Möglichkeiten von Snort und einige Anregungen für den Betrieb in einer produktiven Umgebung. Snort ist ausführlich im Quellcode-Archiv dokumentiert, Installationsanleitungen für verschiedene Setups finden sich online [b]. Mehrere Bücher beschäftigen sich speziell mit Snort oder dem Aufbau und der Funktion eines IDS im Allgemeinen [1, 2].

Snort, Sourcefire und die Community

Marty Roesch entwickelte Snort 1998 ursprünglich als Sniffer und Werkzeug für die Analyse des Datenverkehrs. 2001 gründete er die Firma Sourcefire, die seither für die Weiterentwicklung des Quellcodes verantwortlich zeichnet. Snort steht unter der GPL-Version 2. Sourcefire nutzt Snort als Basis in eigenen kommerziellen IDS. Zukünftig geplant sind Verbesserungen bei der Konfiguration und die Möglichkeit, Regelsätze zur Laufzeit neu zu laden. Ab Version 3.0 soll Snort außer als Stand-alone-IDS als Plug-in (SnortSP) [c] zum Dekodieren, Analysieren und Alarmieren für verschiedene Security-Engines verfügbar sein.

Ohne Regelsätze zum Analysieren des Netzverkehrs wäre Snort nicht viel wert. Snort.org hat über 240 000 registrierte Anwender, die VRT-Regelsätze mit mehr als 15 000 Regeln durchschnittlich 50 000-mal im Monat herunterladen. Marty Roesch betrachtet Open Source nicht ohne Grund geradezu als Leitgedanken für alle Sourcefire-Angestellten [d].

Eine weitere Community hat sich um das Projekt „Emerging Threats“ (ET, früher Bleeding Edge Snort, [e]) von Matt Jonkman gebildet. Die Mailingliste emerging-sigs [f] mit über 2000 aktiven Nutzern dient zur Diskussion und zur Verbesserung der Signaturen. Die kompletten Sätze mit mehr als 10 000 Regeln lassen sich direkt oder via Oinkmaster [r] herunterladen. ET-Signaturen sind aufgrund von Lizenzbestimmungen meist aktueller als die freien Snort-VRT-Regeln; ET verzeichnet 120 000 Downloads täglich. Matt Jonkman ist ein Verfechter der Open-

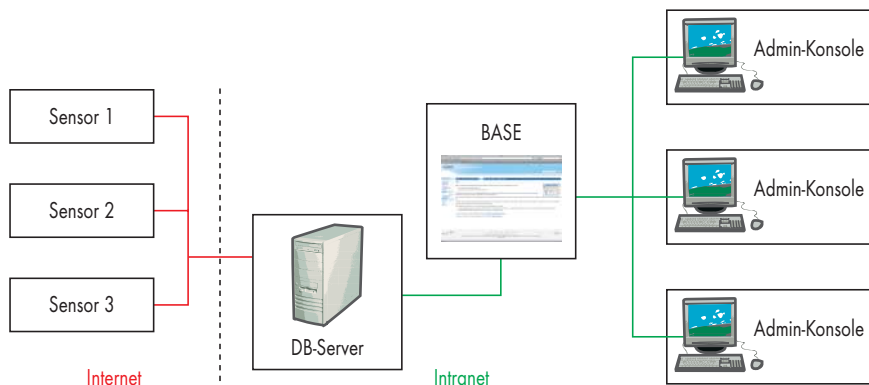
Source-Security, wie er in einem online verfügbaren Interview anlässlich dieses Artikels verdeutlichte [g]. Sie bedeutet nicht nur die Offenlegung des Quellcodes von Kernel und Anwendungen, sondern auch Veröffentlichung von Schwachstellen, sicherheitsrelevanten Informationen, Exploits sowie Signaturen und Gegenmaßnahmen.

Der produktive Betrieb eines NIDS setzt ein aktives Sicherheitsmanagement voraus. Das Verfolgen distributionsspezifischer und sicherheitsorientierter Mailinglisten ist genauso unerlässlich wie Patch-Management und definierte Notfallmaßnahmen. Des Weiteren sind Fragen wie Standort des Sensors (vor oder hinter einer Firewall), zu schützende Netze, Server und Dienste, einzubindende Regelsätze und Hardware/Betriebssystem vor dem Installieren zu klären. Für den Snort-Host gelten die gleichen Sicherheitskriterien wie für Firewalls: gehärtetes Betriebssystem, gesicherter Zugang, keine unnötigen Dienste.

Snort liest wie ein Sniffer den Netzwerkverkehr mit, leitet die Pakete an interne Präprozessoren weiter, dekodiert sie und löst bei Bedarf Aktionen aus, etwa einen Port-Scan-Alarm. Die dekodierten Pakete gelangen zur regelbasierten Detection-Engine, die sie anhand der geladenen Regelsätze analysiert und bei Treffern die angegebene Aktion (Alarm, Log-Eintrag) über ein Ausgabe-Plug-in ausführt. Präprozessoren zum Dekodieren der Pakete, Regelsätze und Output-Plug-ins lassen sich flexibel konfigurieren [1].

Funktionsweise und Installation

Einen guten Überblick über die Funktionen gibt das Snort-Users-Manual [h]. Snort sollte immer in der aktuellen Version (zurzeit 2.8.x) zum Einsatz kommen, damit keine Schwierigkeiten beim Einbinden aktueller Regeln auftreten. Ausführliche Installationsanleitungen sind auf den Snort-Seiten [b]



Beispielhaftes Sensornetzwerk mit mehreren Sensoren und zentraler Datenbank mit BASE als Auswertungswerkzeug (Abb. 1)

und in *doc/INSTALL* des Quellcode-Archivs zu finden.

Globale Einstellungen für den Sensor enthält die Datei *snort.conf*, dort sind die Variablen entsprechend der eigenen Infrastruktur zu verändern: *var HOME_NET* kennzeichnet eigene Netze oder IP-Adressen, *var EXTERNAL_NET* wird häufig mit *!\$HOME_NET* oder dem Platzhalter *any* angegeben, *var RULE_PATH* / enthält einen absoluten Pfad und verweist auf das Verzeichnis mit den installierten Regelsätzen.

Der nächste Schritt aktiviert das Logging in der Datei *snort.conf*. Snort bietet verschiedene Plug-ins zum Schreiben von Log-Dateien oder -Datenbank-Einträgen. Derzeit kommen Postgres, MySQL, Oracle und MSSQL als Datenbank-Backends infrage. Mehrere Sensoren lassen sich für dieselbe Datenbank konfigurieren und mehrere Output-Plug-ins können parallel arbeiten. Ab einer gewissen Datenmenge bildet der Datenbankzugriff zum Loggen der Alarme ein Nadelöhr. Hierfür gibt es ein weiteres Add-on namens Barnyard, das Snort von den Datenbankzugriffen entkoppelt und damit beschleunigt.

Grundsätzlich sollte der Datenbankserver nicht auf derselben Maschine laufen wie Snort selbst, sondern sich in einer gesicherten, nicht öffentlich zugänglichen Umgebung befinden. Dort legt der Admin eine entsprechende Datenbank nebst User-Accounts an und spielt den zugehörigen SQL-Dump aus dem Quellcode-Verzeichnis *schema/create_<db_type>* ein. Sollten die Da-

tenbankzugriffe der Sensoren quer durchs Internet gehen oder anderweitig zu protokollieren sein, empfiehlt sich der Einsatz einer SSL-Verschlüsselung via *stunnel*.

Es bieten sich weitere Optionen an, den Sensor auf das zu überwachende Netz abzustimmen und die Performance zu verbessern. So lassen sich die einzelnen Präprozessoren an die Zielsysteme anpassen, etwa *http_inspect* an die Webserver Apache/IIS, oder die Defragmentierung der Pakete einstellen.

Das Herzstück von Snort bilden die Dateien mit den Regelsätzen. Snorts Detection-Engine analysiert jedes Paket anhand der beim Start geladenen Regel und führt bei Treffern eine definierte Aktion aus.

Eine Regel besteht aus Header und Optionen, wobei im Header die Aktion (*alert*, *log*, *pass*, *drop*, *reject*, *sdrop*), das zu verwendende Protokoll (*tcp*, *udp*, *icmp*, *ip*) und die zu überwachenden Quell- und Zieladressen sowie -ports stehen müssen. In den Optionen finden sich der Name der Regel (*msg*), Inhalt (*content/uricontent*) als Klartext, Bitmuster oder Teil der URI, Match-Kriterien wie TCP-Flags, Flussrichtung (*flow*) des Pakets, Referenzen (*ref*), eine eindeutige Snort-ID (*sid*) und die Klassifizierung nach der Schwere des vermuteten Angriffs (*classification*). Weiterhin erleichtern Perl-kompatible reguläre Ausdrücke (PCRE) das Suchen. Das offizielle Snort-Manual [h] enthält ein Kapitel „Writing Snort Rules“, das die Optionen erklärt. Eine Regel, die zum Beispiel beim versuchten Zugriff auf das Verzeichnis */noaccess* eines Webserver vom Internet aus einen Alarm auslöst, sähe so aus:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS
  $HTTP_PORTS (msg:[WEB_ACCESS unauthorized
  Internet-access to /noaccess]; flow:to_server,
  established; uricontent:[/noaccess];
  classification:web-application-activity;
  sid:10203040; rev:1;)
```

Wer den Einsatz von Snort in Produktivumgebungen plant, muss sich mit



- Mit dem Network Intrusion Detection System (NIDS) Snort steht Administratoren ein flexibles Open-Source-Werkzeug für die Netzüberwachung zur Verfügung.
- Zum Auswerten und Darstellen der umfangreichen Log-Daten gibt es webbasierte Auswertungstools, die das Monitoring großer Netze erleichtern.
- Nach dem Installieren gilt es Snort zunächst an die lokalen Gegebenheiten anzupassen, damit wichtige Alarme im Ernstfall nicht im Grundrauschen untergehen.

den Regeln beschäftigen und bei Bedarf eigene hinzufügen. Die aktuellen Snort-Community-Regeln (*snortrules-snapshot-CURRENT.iar.gz*) kann man nach dem Registrieren von snort.org herunterladen [i]. Sie sind auf jeden Fall zu installieren [j].

Die Dateien mit den Regelsätzen gehören in das mit der Variable `var RULE_PATH` gekennzeichnete Verzeichnis entpackt, üblicherweise `/etc/snort/rules`. Die einzelnen Regeln stehen in Dateien (Regelsätzen) mit sprechendem Namen, etwa *backdoor.rules* oder *web-attacks.rules*. Die ET-Dateien sind mit *emerging-[name].rules* gekennzeichnet.

Welche Regelsätze Snort einbinden soll, legt *snort.conf* nach dem Schema `include $RULE_PATH/[ruleset-name].rules` fest. Eigene Regeln stehen in einer separaten Datei in `/etc/snort/rules`, da Updates die Community-Regelsätze überschreiben.

Nach dem Auswählen und Einbinden der Regeln kann der Snort-Test beginnen:

```
/usr/sbin/snort -T -c /etc/snort/snort.conf
```

startet einen Test der Konfiguration, gibt die Schritte und gegebenenfalls Fehlermeldungen aus. Es sollte mit „Snort successfully loaded all rules and checked all rule chains!“ sowie dem Hinweis auf eine eventuell verwendete Datenbank schließen.

Snort-spezifische Begriffe

Alert/Alarm: von Snort anhand einer Regel generierter Alarm

ET: Emerging Threats, ein Community-Projekt mit aktuellen Regeln, früher bekannt als „Bleeding Edge Snort“

HOME_NET: die zu schützenden Server/Netzwerke

Regel/Rule: Signatur zum Erkennen von Angriffen und Auslösen von Alarmen, in der Kurzform einfach nur [sig]

Regelsatz/Ruleset: unter einem Oberbegriff zusammengefasste Regeln, die in einer Datei liegen, zum Beispiel *web-attacks.rules*

Sensor: ein Client, auf dem Snort läuft

var: globale Konfigurationsvariable, die in Regeln zum Einsatz kommt, zum Beispiel `HOME_NET`

VRT: Vulnerability Research Team von Sourcefire. Es bietet eigene Regelsätze („Community Rules“) zum Download an.

Damit ist die Basisinstallation von Snort abgeschlossen. Es startet wie folgt:

```
/usr/sbin/snort -D -c /etc/snort/snort.conf
```

Ein distributionsspezifisches RC-Skript zum automatischen Starten lässt sich in `/etc/init.d` platzieren; zu Konfigurationsbeispielen, Skripten, Installationsanleitungen und weiteren Ressourcen siehe den Kasten „Onlinequellen“ [k].

Laufende Überwachung mit BASE

Datenbank-Logs kann man entweder mit viel Bash-Voodoo und Script-Foo selbst auswerten – oder sich einer komfortableren Weboberfläche bedienen, die Alarme laufend aktualisiert darstellt – zum Beispiel BASE [l].

BASE kann die Alarme der Sensoren unter verschiedenen Gesichtspunkten auswerten und korrelieren: Quell- und Zieladresse oder -port, Untersuchung der IP-Adressen auf Anzahl, Art und Priorität der ausgelösten Alarme, Analyse der Alarme bis auf Paketebene, zeitbasierte Auswertungen und vieles mehr. BASE umfasst außerdem eine umfangreiche Datenbanksuche. In der aktuellen Version (1.4.3.1) unterstützt BASE Postgres, MySQL, MSSQL und Oracle. Eine große Hilfe ist die Verlinkung von Referenzen und Snort-Signaturen in der Weboberfläche. Advisories von Bugtraq, CVE und andere Quellen, die in den Signaturen referenziert sind, befinden sich im direkten Zugriff. Die ET-Regeln enthalten jeweils einen Link auf die zugehörige Signatur.

Als PHP-basierte Webapplikation erfordert BASE einen Zugangsschutz per `.htaccess` und Authentifizierung. Außer den Administratoren sollten keine anderen Benutzer Dienste auf dem Webserver nutzen können, damit die kritischen Snort-Daten nicht Gefährdungen durch Sicherheitslücken in Webapplikationen ausgesetzt sind [m]. Grundsätzlich sollten Zugriffe SSL-verschlüsselt sein. BASE lässt sich einfach installieren und ist gut dokumentiert [b].

Neben der webbasierten gibt es weitere Methoden, sich über Snort-Alarme zu informieren. Mit Swatch oder Tenshi [n] kann man Log-Dateien überwachen, Mails versenden oder weiterführende Logs zur ausführlichen Auswertung schreiben. Dies ergibt Sinn in Setups, in denen die Sensoren einen gemeinsamen Log-Server benutzen. Tools wie SnortSnarf oder SnortLog erstellen Statistiken, Sguil [o] ist ein

weiteres Analysewerkzeug, das auf einer Snort-Alarm-Datenbank basiert. Teilweise sind umfangreiche Installationsmaßnahmen erforderlich oder die Tools etwas betagt.

False Positives und etwas Tuning

Nach dem Installieren ist es angebracht, die Regelsätze an das eigene Netz und dessen Datenverkehr anzupassen. Wer etwa eine reine Web- und Applikationsserver-Umgebung überwacht, benötigt keine Chat-, SIP- oder P2P-Regeln, sollte aber auf jeden Fall Regelsätze wie *web-cgi.rules* und *web-attack.rules* einbinden.

Zunächst kommt es je nach Auswahl der Regelsätze und der Positionierung des Sensors zu Fehlalarmen. Mit BASE kann man dann zum Beispiel ergründen, ob Zugriffe auf den eigenen DNS-Server DNS-Poisoning-Alarme auslösen oder ob hinter der IP-Adresse, die regelmäßig zu einem TCP-PortswEEP-Alarm führt, womöglich der eigene Nagios-Server steckt, dessen `check_tcp-` und `check_http-`Service-Checks Snort als Port-Scans wertet. Nach einigen Tagen ergibt sich ein erster Überblick und die Regelsätze lassen sich so anpassen, dass möglichst keine False Positives mehr auftreten.

Zum Deaktivieren von Regeln gibt es mehrere Ansätze: Man kann die zugehörige Regel komplett in der entsprechenden Datei oder den kompletten Regelsatz durch Auskommentieren der Option `include $RULE_PATH/name-des-sets.rules` deaktivieren. Regeln können auch global aktiviert bleiben, und Alarme lassen sich für bestimmte Quell- oder Zieladressen unterdrücken. Dazu dienen die `suppress-`Regeln, meist in der Datei `threshold.conf`.

Das Hauptziel des Sensor-Tunings ist es, bei allen geladenen Regeln und maximalem Netzwerkverkehr alle ankommenden Pakete mit Snort zu untersuchen und keines zu verwerfen. Snort schreibt eine ausführliche Statistik ins Syslog, wenn es ein SIGUSR1-Signal erhält: `kill -s SIGUSR1 `pidof snort``.

Gerade die Bereiche Netzwerksicherheit, Angriffsvektoren, neue Sicherheitslücken und verfügbare Exploits unterliegen laufend starken Änderungen. Wer die einschlägigen Mailinglisten abonniert hat oder hin und wieder bei PacketStorm sowie Milw0rm vorbeischaud, weiß ein Lied davon zu singen. Glücklicherweise ist die Security-Com-

munity sehr aktiv, wenn neue Bedrohungen auftauchen. Nach Bekanntwerden aktiver Exploits für eine BIND-Lücke am 29. Juli 2009 [p] waren innerhalb von drei Stunden die ersten Signaturen auf Emerging Threats verfügbar [q].

Regelmäßige Updates mit Oinkmaster

Produktive Sensoren sind regelmäßig mit Updates der geladenen Regelsätze zu versorgen. Als Grundsatz gilt: Regel nie ungeprüft auf aktive Sensoren ausrollen, da eine Regel mit falscher Syntax dazu führen kann, dass der Sensor nicht mehr arbeitet.

Das Deployment kann skriptgesteuert [k] von einer Admin-Workstation aus erfolgen, die SSH-Zugriff auf die Sensoren hat, und auf der Snort mit der gleichen Regelkonfiguration wie auf den Sensoren installiert ist. Folgender Arbeitsablauf hat sich dabei bewährt:

1. Backup der funktionierenden Regelsätze,
2. Download der aktuellen Regelsätze (VRT+ET),
3. Ansehen der Änderungen, implementieren oder verwerfen,
4. Testen der aktualisierten Regelsätze,
5. Ausrollen der aktualisierten Regelsätze auf den/die Sensoren, wobei skriptgesteuert erst ein Test der neuen Regel erfolgen und ein Backup der Regelsätze angelegt werden kann,
6. Kontrolle, ob Snort mit den aktualisierten Regeln startet und
7. gegebenenfalls Kontrolle via BASE, ob Regeln ungewöhnliches Verhalten zeigen, etwa massives Auftreten von False Positives.

Download und Änderungsverfolgung (Punkte 2 und 3) lassen sich komfortabel mit Oinkmaster [r] bewerkstelligen, einem Perl-Skript, das die aktuellen Regelsätze von snort.org sowie emergingthreats.org herunterladen und die eigenen Regeln aktualisieren kann. Oinkmaster bietet weitreichende Konfigurationen an. So kann man einzelne Regeln deaktivieren oder das Überschreiben lokaler Änderungen an bestehenden Regeln verhindern.

Nach erfolgreichem Update und Test der neuen Regelsätze rollt der Administrator sie per *rsync* auf die Sensoren aus und startet Snort dort neu. Er sollte immer kontrollieren, ob Snort nach einem Neustart läuft. Fehler bei der Übertragung sind nie ganz ausgeschlossen. Ein lokales Backup der letzten funktionierenden Regelsätze auf

Onlinequellen

[a] Basic Analysis and Security Engine (BASE)	base.secureideas.net/
[b] Snort installieren und konfigurieren	www.snort.org/docs/setup-guides
[c] Snort-Download	www.snort.org/downloads/snort-3-0/
[d] Interview mit Marty Roesch	www.mare-system.de/interview-with-marty-roesch.html
[e] Emerging Threats (ET)	www.emergingthreats.org/
[f] ET-Mailingliste	lists.emergingthreats.net/mailman/listinfo/emerging-sigs
[g] Interview mit Matt Jonkman	www.mare-system.de/interview-with-matt-jonkman.html
[h] Snort-Dokumentation	www.snort.org/docs/
[i] Snort-Regelwerke	www.snort.org/snort-rules
[j] Regeln zu Emerging Threats	www.emergingthreats.net/rules/emerging.rules.tar.gz
[k] Snort-Ressourcen des Autors	www.mare-system.de/arbeiten-mit-snort.html
[l] BASE auf Sourceforge	sourceforge.net/project/showfiles.php?group_id=103348
[m] SecurityFocus-Meldung zu BASE	www.securityfocus.com/archive/1/504487
[n] Log-Daten überwachen mit Tenshi	dev.inversepath.com/trac/tenshi
[o] Analysewerkzeug Sguil	sguil.sourceforge.net/
[p] BIND 9 DoS attacks in the wild	isc.sans.org/diary.html?storyid=6886
[q] Signaturen auf Emerging Threats	www.emergingthreats.net/cgi-bin/cvsweb.cgi/sigs/CURRENT_EVENTS/CURRENT_Bind
[r] Oinkmaster	oinkmaster.sourceforge.net/

den Sensoren hilft bei einem Rollback. Es empfiehlt sich, Updates manuell einzuspielen – es sei denn, man implementiert umfangreiche Tests, die im Zweifelsfall das Update abbrechen. Die Frequenz der Updates hängt von der eigenen Paranoia, Manpower und Bedrohungslage ab. Tägliche Updates sind natürlich vorbildlich, aber häufig reicht ein wöchentlicher Rhythmus. Trotzdem muss man in der Lage sein, kritische Signatur-Updates sofort einzuspielen.

Fazit

Snort, ein leistungsfähiges Intrusion Detection System, lässt sich auf vielfältige Weise an zu schützende Server und Netze anpassen. Frei verfügbare Plug-ins erlauben es, Snort in Kombination mit Firewalls zu einem IPS umzubauen, das aktiv auf Angriffe reagiert, IP-Adressen blockt oder Traffic umleitet. Snort-Anwender können die Detection Engine um eigene Regeln erweitern und mit genau denjenigen laden, die zur Überwachung nötig sind. Die Snort-internen Präprozessor gestatten betriebssystem- und serverspezifische Einstellungen.

Eine aktive Community liefert ständig aktualisierte Regeln und reagiert sehr schnell auf kritische Sicherheitslücken oder Exploits. Auf Snort können ganze dezentrale Sensornetze basieren, die sich zentral verwalten lassen. Damit eignet sich Snort für jede Organisation, die Dienste in öffentli-

chen Netzen anbietet, seien es Hoster, ISPs oder ASPs. Add-ons vereinfachen das Administrieren. Einer Analyse oder einem Monitoring der Alarmmeldungen sind kaum Grenzen gesetzt.

Vor Exploits oder Sicherheitslücken schützt Snort aber nicht. Wer unsichere Software einsetzt und seine Systeme nicht laufend aktualisiert, wird mit Alarmen zu Exploits, Scannern und unerlaubten Zugriffen bombardiert. Konfiguration und laufende Wartung erfordern, je nach Komplexität der zu schützenden Netze und Server, einigen Aufwand, der sich aus dem Blickwinkel des Sicherheitsmanagements aber durchaus lohnen kann – nicht zuletzt, weil Snort dabei hilft, den Überblick über das eigene Netz zu bewahren. (un)

MARKUS MANZKE

ist selbstständiger Systemadministrator und beschäftigt sich mit den Themen Monitoring, Security und Servermanagement.

Literatur

- [1] Brian Caswell, Jay Beale, Andrew Baker; Snort IDS and IPS Toolkit; Syngress, 2007
- [2] Stephen Northcutt, Judy Novak; Network Intrusion Detection (3rd Edition); Newriders, 2003

