Intrusion Detection Systeme

am Beispiel von 

Thematische Ausarbeitung zur Vorlesung
"Ausgewählte Kapitel der IT-Security" im WS03/04
bei Prof. Dittmann
an der Otto-von-Guericke-Universität Magdeburg

Von
Christian Graf &
Andreas Mähnz

Februar 2004

Inhaltsverzeichnis

Einführung

Motivation

Das moderne Privat- und Geschäftsleben ist heute ohne Computer und deren weltweite Vernetzung nicht mehr denkbar. Sie sind zu einem immer präsenten, unersetzlichen Werkzeug und Medium, aber damit auch zu einem potentiellen Angriffsziel für z.B. konkurrierende Mitbewerbern geworden. Dagegen gilt es sie zu schützen.

Grundlagen

Definition: Was ist 'intrusion' und was 'intrusion detection'?

Diese könnten versuchen, über eine "Menge von Handlungen, deren Ziel es ist, die Integrität, die Verfügbarkeit oder die Vertraulichkeit eines Betriebsmittels zu kompromittieren" [MHL94] (diese Handlungen werden im Englischen unter dem Begriff 'intrusion' zusammengefaßt) in das Zielsystem einzudringen. Einfacher gesagt: 'Intrusion' ist die Umgehung der Sicherheitsmaßnahmen eines (Computer-)Systems.

Systemadministratoren und die von ihnen ausgearbeitetenden Sicherheitslösungen sollen hier entgegenwirken und höchstmögliche Sicherheit vor Angriffen gewährleisten. Sie müssen letztendlich bewerten, welche Aktivitäten auf dem System als verdächtig, gefährlich und eindeutig kompromittierend zu sehen sind und über etwaige Gegenmaßnahmen entscheiden. Der automatisierte, rechnergestützte und in Echtzeit ablaufende Prozeß des Analysierens des Netzwerkverkehrs anhand bestimmter Regeln zum Aufspüren etwaiger Angriffe wird im Englischen als 'intrusion detection' (deutsch: Angriffs-/Einbruchserkennung) bezeichnet.

Abgrenzung: Was ist ein 'Intrusion Detection System' und was ist es nicht?

Angriffe auf Rechnersysteme zu erkennen, ist ein sehr zeitaufwendiger und kaum von Menschenhand zu bewältigender Prozeß. Das Datenaufkommen in den weltweiten Netzwerken und ist dafür einfach zu groß, und es kann nicht Aufgabe der Administratoren sein, die tagtäglich anfallende Menge an Protokolldaten des durchlaufenden Verkehrs per Hand zu analysieren, um etwaige Angriffe zu erkennen. Offene Strukturen, wie sie etwa dem Internet zu Grunde liegen (d.h. alles kann prinzipiell überall entlang geleitet werden) kommen erschwerend hinzu: Der Administrator kann sich nicht darauf verlassen, daß schon vorgeschaltete Systeme die Daten auf gewisse Sicherheitsrichtlinien geprüft haben und muss daher den kompletten Datenstrom als potentiell unsicher einstufen.

Die Überprüfung der spezifischen Merkmale einer Unmenge möglicher Angriff zur Übertragungszeit überstiege bei weitem jede von einem Menschen aufzubringende kognitive Leistung. Die alternative, zeitversetzte Analyse der vom Betriebssystem zur Verfügung gestellten Protokolldateien stellt auch keine Erleichterung dar und trägt den zusätzlichen Mängel in sich, daß so keine unmittelbare Gefahrenerkennung möglich ist. Konsequenterweise wird die Überprüfung des Datenstroms dem Computer überlassen, der unmittelbar und mit hoher Bandbreite die klar strukturierten Informationen (die Syntax und Semantik der Übertragungsprotokolle sind bekannt) fortwährend schnell verarbeiten kann. Dies wird mit sogenannten 'Intrusion Detection' Systemen (IDS), die eine effektive Analyse von teilweise sehr großem Datenaufkommen im Netzwerk gegen bestimmte Regeln zulassen, realisiert. Sie fungieren als eine Art Alarmanlage für einzelne Rechner oder ganze Rechnernetzwerke.

Bei einem etwaigen Angriff auf das System würde das IDS das Ergebnis des Analysevorgangs protokollieren, eventuell Alarm auslösen, den Systemadminstrator benachrichtigen oder Mechanismen zur Abwehr des Angriffs starten. Letztgenannte Funktionalität zählt nicht zu den Kernaufgaben eines klassischen 'Intrusion Detection' Systems, sondern gehört eher in die Kategorie der 'Intrusion Response' Systeme (IRS), die Gegenmaßnahmen einleiten, um den Angreifer abzuwehren (z.B. indem es dessen Verbindung kappt) und das System zu wieder abzusichern. An dieser Stelle soll nicht weiter auf IRS eingegangen, sondern nur festgehalten werden, daß moderne IDS gewisse Möglichkeiten bieten können, einen Angriff abzuwehren.

Einem ID System nicht unähnlich, aber doch mit einer anderen Aufgabe bedacht, sind sog. 'Firewalls'. Um eine Verwechslung zu vermeiden, soll hier ein Abgrenzung zwischen beiden erfolgen. Eine Firewall ist ein spezieller elektronischer 'Torwächter' auf Hard- oder Softwarebasis, der bestimmt, welcher Verkehr in oder aus einem Netz gelangen darf. Dies stellt eine andere qualitative Stufe dar, denn im Gegensatz dazu ist ein ID System per se nur Beobachter und greift nicht in den Datenverkehr ein. Sollte es einem Angreifer tatsächlich gelingen, in das System einzudringen und dort Veränderungen vorzunehmen, gelänge es mittels IDS bestenfalls nur, die ankommenden Pakete, die die entsprechenden Kommandos enthalten, festzustellen bzw. aufzuzeichnen. Integritätssicherung des Dateisystems oder einzelner Dateien sind nicht Bestandteil der Funktionalität (siehe Tripwire[1] als dafür geeignete Lösung).

Anwendungsgebiete

Wie in der Einführung schon dargelegt, folgert die Notwendigkeit von ID Systemen aus der Tatsache der großflächigen Vernetzung von Rechnern und der Unsicherheit, wer in diesem Netz mit welcher Intention aktiv ist. Eine weitergehende Frage ist dann, ob jedes System ein IDS beinhalten sollte oder ob bestimmte Kategorien von Systemen existierten, in denen ein IDS eher hinderlich sein würde. Sicher ist der Einsatz eines IDS auf einem 'zentralen' (sofern sich dieser Begriff mit dem Ansatz der heutigen Netzwerke vereinbaren läßt) Router oder Switch wünschenswert, weil dort die meisten Daten entlangfließen, die man analysieren könnte. Doch nach was sollte man hier suchen? Solange es keine definierte Sicherheitspolitik in einem Netzwerk gibt, macht der Einsatz von ID Systemen keinen Sinn. Und solche Sicherheitspolitik baut auf einem bestimmten Nutzungsmodell eines Netzwerkes auf. So kann definiert werden, was erwünschter Datenverkehr ist und was nicht. Setzt man jedoch an einer zu zentralen Stelle an, sind auch die Daten zu generell und lassen sich nicht mehr in 'harmlos' und 'gefährlich' trennen. Als zweites Problem sind die Daten an sich zu sehen, oder besser: das Datenaufkommen. An manchen zentralen Knoten laufen Tera- und Petabytes an Daten durch, die selbst bei minimalster Filterung noch Unmengen an Protokolldaten, Warnungen etc. erzeugen würden, die kein Mensch jemals auswerten könnte. So könnten Angriffe oder Angriffsversuche auf bestimmte Rechner in der Flut an Protokolldaten praktisch untergehen. Daher ist ein ID System nur für Subnetze mit klarer Sicherheitspolitik und Zuständigkeiten angeraten und praktisch auch nur dort administrierbar.

Einbruchsversuche können nicht nur auf der Ebene von Firmennetzwerken oder auf kritische Rechner in globalen Verbünden geschehen, sondern z.B. auch auf jeden Rechner, der über ein Modem mit dem Internet verbunden ist. Diese kurzzeitigen Verbindungen mit ihren bei Verbindungsaufbau ständig neu vergebenen Adressen sind nicht sonderlich interessant für Angreifer. Doch Systeme, die eine länger andauernde Verbindung mit anderen Rechnern aufrecht halten, sind für sie immer wieder zu finden und könnten so

[1] Tripwire Open Source Projekt: *Homepage*, http://www.tripwire.org

auch systematisch ausspioniert werden. Zu diesen potentiellen Opfern gehören auch jene Privatleute, die über DSL mit einer fest vergebenen IP-Adresse mit ihrem Provider verbunden sind. Selbst wenn sie glauben, keine sensiblen Daten zu schützen zu haben, könnten sie Gefahr laufen, mit Trojanern verseucht zu werden, die Passworte, Kontoinformationen etc. ausspähen und diese anschließend vom heimischen System nach außen schmuggeln. Über den gleichen Weg könnten zum Teil fatale 'Distributed Denial of Service' (DdoS) Attacken auf andere Systeme initiiert werden - auch ohne Wissen des Computernutzers, der dann zum 'Mittäter' wird. ID Systeme sind somit auch für Einzelplatzrechner sinnvoll, obwohl diese meist nur als letztes Glied in einem Netzwerk hängt.

ID System "SNORT"

Das hier betrachtete frei erhältliche ID System 'SNORT'[2] wird von seinen Entwicklern konsequent für kleinere und mittlere Netzwerke empfohlen. Auf größeren Systeme ist Performanz (hier äquivalent zu Datendurchsatz) und einfachem Management komplexer Sicherheitsrichtlinien von höchster Bedeutung, zwei Anforderungen, bei denen SNORT professionellen Systemen nicht das Wasser reichen kann[3]. Sicher könnte man die GNU-lizenzierte Software auch in diesem Bereich einsetzen und sie würde nicht versagen, doch dies würde dem Einsatz einer MySQL-Datenbank in der Liga der ORACLE und INFORMIX Datenbanken entsprechen: prinzipiell leisten sie dasselbe, doch die einen haben spezielle, optimierte Verfahren für die Handhabung großer Datenmengen und sind mit ihren Werkzeugen darauf ausgelegt, die anderen nicht. Niemand würde ernsthaft argumentieren, ein Privatanwender solle sich mit einer völlig überproportionierten ID Lösung für seinen Heimcomputer ausstatten. Jedes der Systeme hat seine Existenzberechtigung und seinen optimalen Einsatzbereich.

Funktionsweise eines ID Systems

Datensammlung

Als grundlegende Voraussetzung für einen erfolgreichen Einsatz eines IDS muß die Netzwerkkarte den sogenannten „promiscuous mode“ unterstützen, in dem es möglich ist, nicht nur die für diesen Computer bestimmten Datenpakete zu lesen, sondern auch alle anderen. Weiterhin gelten insbesondere jene Überlegungen, die im Abschnitt ‚Wo das Schwein parken‘ erörtert wurden, damit sinnvolles Datensammeln überhaupt möglich ist. Diese erste Grundfunktionalität ist auch Voraussetzung für weitere Verarbeitungsschritte (siehe Abschnitte "Datenanalyse" und "Ergebnisvisualisierung").

[2] SNORT im WWW: http://www.snort.org

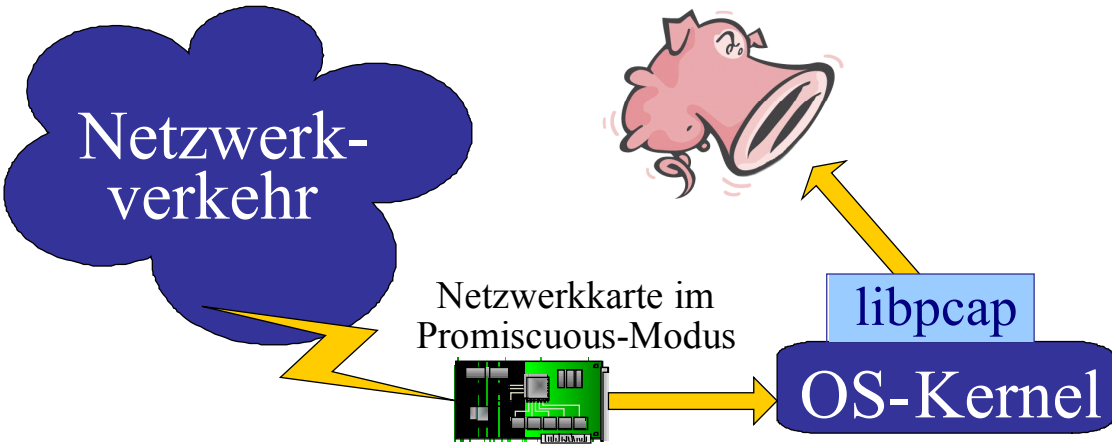

Abb.1: Datenfluß bei Analyse mittels SNORT

Sniffer

Im einfachsten Fall arbeitet das IDS als sog. ‚Sniffer', ein Programm, das einen kontinuierlichen Strom an Daten vom Netzwerk liest. Die einlaufenden Pakete müssen nicht für das System bestimmt sein, auf dem das IDS läuft, sondern können sich auch im Durchlauf zu anderen Systemen befinden. Sie werden mit eingebauten Methoden selbst sofort decodiert (SNORT als das hier benutzte Beispiel IDS kann TCP, IP, UDP und ICMP Pakete verarbeiten) und nicht gespeichert – die Arbeitsweise eines Sniffers ist transient. Dies erlaubt sofortige Erkenntnisse aus dem aktuellen Datenstrom, mit denen Angriffe erkannt werden können, während sie stattfinden. Dazu ist aber eine externe Logik zur Bewertung des Datenstroms nötig, der Sniffer beinhaltet diese nicht.

Logger

Aufbauend auf der einfachen Sniffer-Funktionanlität können die meisten Programme (und so auch SNORT) jedes durchlaufende Paket sammeln und permanent speichern. Dabei werden Metainformationen wie Datum und Zeit ebenfalls aufgezeichnet. Diese auf Persistenz ausgelegte Arbeitsweise ist der Hauptunterschied zum einfachen Sniffer. Ein kontunierliches Aufzeichnen allen Datenverkehrs ist nur in bestimmten Fällen zweckmäßig, denn es erfordert zum einen enormen Speicherbedarf (besonders an einem breitbandigen Anschluß), und es stellt sich die Frage, ob jemand wirklich die Zeit aufbringen kann, die aufgezeichneten Protokolldaten (die noch in keiner Weise inhaltlich in einen Zusammenhang gebracht worden sind!), irgendwann einmal analysieren können wird. Im Gegensatz zum **‚reaktionären' ID** mittels Sniffer (dort kann sofort reagiert werden), können durch das Verarbeiten von Logdateien im Nachhinein nur Maßnahmen für einen etwaigen nächsten Angriff auf die Schwachstelle getroffen werden – sogenanntes **‚präventives' ID**.

Neben den beiden hier genannten Ansätzen, die beide den Netzwerkverkehr überwachen, können zusätzlich Daten der Applikationsebene zur Auswertung hinzugezogen werden, etwa die Vergabe von Betriebsmitteln an laufende Prozesse (belegter Speicher, Auslastung der CPU, Netzwerkverbindungen usw.).

Datenanalyse

Wie schon festgestellt wurde, reicht das einfach Aufzeichnen des Datenverkehrs nicht aus, denn es erlaubt durch fehlende Verarbeitung erstens keine sofortige Erkenntnisse zum Qualität der einfließenden Daten und zweitens erfordert es Unmengen an Arbeitsmitteln

(Speicherplatz und menschliche Arbeitszeit bei der Auswertung). Eine Technik zur automatischen Erkennung ist gefragt. Dabei sind zwei Möglichkeiten zu unterscheiden:

Mißbrauchserkennung

Anhand vordefinierter Muster versucht das IDS Einbrüche zu erkennen. Mittels vordefinierter Signaturen und Pattern-Matching-Algorithmen werden einzelne Netzwerkpakete untersucht. Nach diesem Paradigma arbeitet auch SNORT. Um einen Angriff erfolgreich identifizieren zu können, muß also eine entsprechende Signatur vorliegen. Pakete mit Angriffen, für die keine Signatur vorliegt, werden demnach von der Mißbrauchserkennung als sog. ‚False Negative' unbehelligt durchgelassen. Das Internet bietet jedoch eine genügend große Anzahl Angriffssignaturen, so daß der Systemadministrator bei fortwährender Aktualisierung des IDSs davon ausgehen kann, daß sein System zumindest gegen bekannte Angriffe geschützt ist.

Anomalieerkennung

Wenn also ein System mittels SNORT abgesichert ist, bleiben weiterhin die Angriffe problematisch, deren Angriffslogik bisher unbekannt war. Die Techniken zur Identifizierung möglicher Attacken werden unter dem Begriff Anomalieerkennung zusammengefaßt. Eine Anomalie ist jede Abweichung vom ‚normalen' Netzverkehr, wobei nicht jede Anomalie sofort eine Gefahr darstellen muß. Und nicht jeder Angriff wird sich als Anomalie darstellen, man denke an ausgehorchte Login-Passwort-Kombination, die zu schädlichen Zwecken eingesetzt werden. Das IDS könnte dies beim Login niemals erkennen, wenn der ordnungsgemäß erfolgt (siehe Abschnitt "Verwundbarkeit - Klassische Angriffe").

Um eine Anomalieerkennung überhaupt möglich zu machen, muß das IDS erst einmal vermittelt bekommen, was normales Systemverhalten überhaupt ist. Dazu bedarf es einer längeren Einlaufzeit des System mit möglichst durchschnittlichen Kenngrößen, denn diese bestimmen letztendlich die Schwellenwerte zum ‚unnormalen'. Verläßt ein zu überwachender Parameter im Einsatz des IDS die definierten Akzeptanzschwellen, wird Alarm ausgelöst. Diese auf Reverenzdaten basierende Anomalieerkennung verfolgt einen **statistischen Ansatz**. Im Gegensatz dazu wird beim **logischen Ansatz** die zeitliche Abfolge von Ereignissen untersucht. Das IDS nimmt bestimmte Ereignisfolgen als typisch an. Wenn es dann den Beginn einer solchen erkennt, der weitere Verlauf aber nicht den Regeln entspricht, schlägt das System Alarm. So kann das Potential genutzt werden, nicht bekannte Angriffsarten aufzuzeichnen und daraufhin Abwehrstrategien zu entwickeln. Der Präprocessors ‚Spade'[4] etwa sorgt dafür, daß SNORT fähig ist, Anomalien zu erkennen (zur Funktionsweise von SNORT siehe entsprechenden Abschnitt).

IDS richtig plazieren

Trotzdem ist ein IDS nicht etwa eine minderwertiger Ersatz für eine Firewall, beide haben ihre Existenzberechtigung und arbeiten komplementär: die Firewall als Bastion gegen Angriffe von außen nach innen und umgekehrt; und die IDS zur Validierung der ordnungsgemäßen Funktion der Firewall, falls diese doch einmal schädlichen Verkehr durchlassen sollte - das IDS als zweite Sicherheitsstufe. Darüber hinaus kann sie im Gegensatz zur Firewall auch den internen Verkehr im LAN auf gefährliche Aktionen hin überprüfen, denn heute kommt die Bedrohung für das eigene Netz und seine Daten nicht mehr unbedingt von außen, sondern von den eigenen Mitarbeitern. Ein IDS außerhalb des Firewalls kann den eintreffenden Verkehr auf Angriffe prüfen und durch einen Abgleich

[4] Spade = Statistical Packet Anomaly Detection Engine

mit der inneren IDS feststellen, wie gut die Firewall funktioniert, denn die sollte eben jene Pakete herausfiltern. Dafür müssen natürlich alle drei 'Wächter' aufeinander abgestimmt sein - eine vor der Installation definierte Sicherheitspolitik ist hierzu unablässig.

Nun stellt sich natürlich die Frage, wo genau das IDS in die lokale Netzwerkarchitektur eingebunden werden soll. Prinzipiell gibt es hierbei 3 Möglichkeiten. Zunächst kann das IDS vor der Firewall plaziert werden - z.B. kann es hier den hereinkommenden Verkehr auf etwaige Angriffe überprüfen. Jedoch gibt es dabei keine Möglichkeit zu sagen, ob der Angriff auch erfolgreich war. Daher bietet sich ein zweites IDS direkt hinter der Firewall an. Damit kann festgestellt werden welche Angriffe die Firewall nicht geblockt hat. Drittens gibt es die Möglichkeit eines IDS im internen Netz. Es ermöglicht die Erkennung von Angriffen aus dem Inneren des Netzwerks und verringert somit die Gefahr des unberechtigten Zugriffs durch Mitarbeiter, Fremde oder Würmer und Trojaner. Letztendlich kann der dreifache Einsatz eines IDS sehr sinnvoll sein (siehe Abbildung 2).

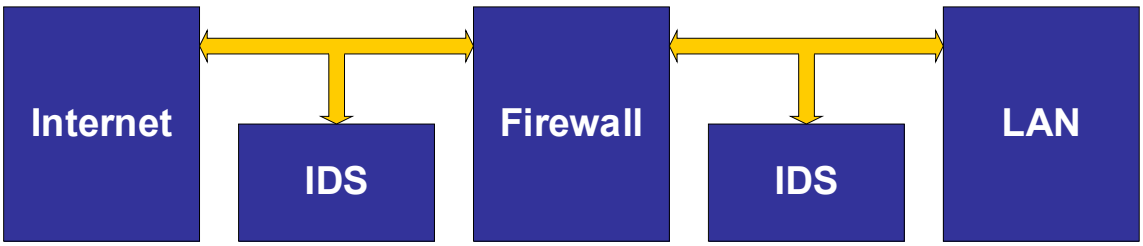

Abb.2: Empfohlene Plazierung eines ID Systems im Netzwerk

Zur Arbeitsweise von "SNORT"

Grundfunktionalität

Um die Arbeitsweise von SNORT zu verstehen, muß man sich darüber bewußt werden, daß SNORT kein monolithisches System ist, sondern durch Plug-Ins erweitert werden kann. Die Grundfunktionalität besteht in dem Decodieren der Netzwerkpakete (siehe Abschnitt „Sniffer“) und das Prüfen derselben gegen bestimmte Regeln. Vor dem eigentlichen Prüfen kann der Systemoperator jedoch noch verschiedenste sogenannte Präprozessoren einsetzen, die den Datenstrom zusätzlich analysieren. Nach dem Prüfen sorgen sogenannte Module dafür, daß die etwaigen Informationen über Angriffsversuche entsprechend verarbeitet werden (siehe Abbildung 3).

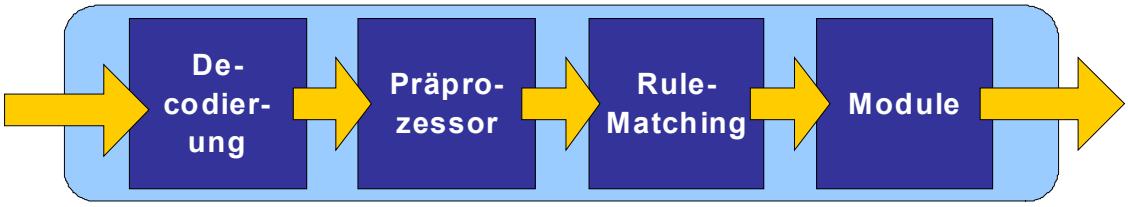

Abb.3: Verarbeitungsschritte in SNORT

Präprozessoren

Mittels Präprozessoren wird der Datenstrom zusätzlich annalysiert und Metainformationen über die ankommenden Pakete erzeugt, die in späteren Verarbeitungsschritten benutzt werden können. Einige dieser ergänzenden Funktionseinheiten seien im Folgenden exemplarisch genannt.

- **frag2**: setzt Paketfragmente zuammen, ideal für erfolgreichen Mustererkennung für zerstückelte Angriffsmuster
- **stream4**: setzt TCP-Stream wieder zusammen und erlaubt so die Analyse eines vorher fragmentierten Angriffs (siehe Abschnitte zu „Invasion“ und „Evasion“)
- **portscan**: entdeckt Ausspähversuche nach offenen Ports
- **spade**: Statistical Packet Anomaly Detection Engine - ist ein selbstlernendes Modul, das nach einer Lernphase anormalen Netzwerkverkehr feststellen kann
- **rpc_decode**: defragmentiert RPC-Pakete
- **telnet_decode**: trennt Telnet-Kontrollzeichen von den Sessiondaten
- **flow**: Tracking um eindeutigen IPv4-Stream sicherzustellen
- **Performance Monitor**: mißt die Echtzeit- und theoretische Maximumperformance
- **HTTP Inspect**: grundlegende Analyse der HTTP-Pakete (benötigt ein vorgeschaltetes Zusammensetzungs-Modul)

Das Prüfen (‚rule matching‘)

Eine Regel für SNORT setzt sich immer aus einem Kopfteil für die grobe Filterung sowie einem Teil für die Optionen der Feinjustierung zusammen. Im Kopf werden dabei Informationen über die Regelaktion, das Protokoll, die Quell- und Ziel-IP-Adresse, die Netzmaske, der Quell- und Ziel-Port sowie die Datenflußrichtung festgelegt. Die definierten Regelaktionen in SNORT sind:

- **alert:** generiert einen Alarm und loggt das Paket
- **pass:** ignoriert das Paket
- **activate:** generiert einen Alarm und wendet weitere Regeln an
- **dynamic:** loggt das Paket nur wenn durch 'activate' aufgerufen
- **log:** loggt das Paket

Im Optionsteil werden u.a. Informationen über die noch weiter zu untersuchenden Pakete sowie die Alarmmitteilungen abgelegt. Möglich sind dafür z.B.:

- **meta-data:** Metainformationen über die Regel - hat keinen Effekt während der Analyse (z.B. eine Nachricht ausgeben)
- **payload:** die Optionen beziehen sich auf das Paket selbst (z.B. den Inhalt)
- **non-payload:** die Optionen beziehen sich auf Metadaten des Pakets (z.B. die Größe)
- **post-detection:** regelspezifischer Auslöser (z.B. Verkehr aufzeichnen)

Hierbei sei noch zu sagen daß eine Vielzahl fertiger Regeln bereits im Internet verfügbar sind (http://www.snort.org/dl/rules/).

Die Ergebnisweiterverarbeitung

Die Weiterverarbeitung der Analyseergebnisse kann in SNORT auf verschiedene Weise ablaufen (z.B. Visualisierung, Alarmierung des Administrators etc.). Dieses wird durch Module realisiert, wobei verschiedene Module auch hintereinander angewendet werden können. Im nachfolgenden Absatz werden einige Module näher vorgestellt.

Module

Das Regelsystem für Module ist ähnlich dem für Präprozessoren. Folgende Aktionen werden von Snort bei einem vermuteten Angriff bereitgestellt:

- **Alert_xxx:** ein Warnhinweis wird wahlweise in eine Logdatei oder zu einem Port in kurzer oder ausführlicher Form geschrieben
- **Log_tcpdump:** die Pakete werden im tcpdump-Format in eine Datei geschrieben

- **Log_null:** es erfolgt kein Paketmitschnitt, nur ein Warnhinweis wird ausgegeben
- **Database:** die Daten werden direkt in eine angeschlossene Datenbank geschrieben
- **CSV:** die Warndaten werden in ein einfach zu importierbares Datenbankformat geschrieben
- **Unified:** die Daten werden getrennt in eine Warndatei und eine Paketdatei geschrieben - dies stellt die schnellste Ausgabemöglichkeit dar

Die folgende Auflistung gibt einen kleinen Überlick über die große Anzahl der verfügbaren Module:

- **libnet:** ermöglicht Flexible Response, d.h. es ermöglicht Reaktionen auf Angriffe wie z.B. Beendigung der TCP-Verbindung über RST-Paket oder ICMP ('Net unreachable', 'Host unreachable', 'Port unreachable' oder alle drei auf einmal) [http://www.packetfactory.net/libnet/]
- **oinkmaster:** ist ein Perl-Programm welches die Snort-Regeln auf dem neuesten Stand hält [http://oinkmaster.sourceforge.net/]
- **guardian:** ermöglicht die (auch zeitweise) Sperrung der Angreifer-IP-Adresse [http://www.chaotic.org/guardian/]
- **acid:** PHP-Skripte zur grafischen Aufbereitung der Logfiles [http://acidlab.sourceforge.net/]
- **snortsnarf:** Perl-Skripte zur grafischen Aufbereitung der Logfiles [http://www.silicondefense.com/software/snortsnarf/]
- **Snort Webmin Interface:** ein Plugin für Webmin zur Konfiguration von Snort, d.h. es ist kein Snort-Modul [http://msbnetworks.net/snort/]

Visualisierung

Nach einem erfolgten Angriff ist es elementar, die gesammelten Daten zu sichten, um sie für etwaige Maßnahmen auszuwerten. Für diesen Zweck bieten sich Zusatzprogramme zur grafischen Aufbereitung wie z.B. ACID (basiert auf PHP, greift auf Daten zurück die Snort in eine Datenbank loggt) oder SnortSnarf (besteht aus Perl-Skripten die aus Logfiles oder Datenbankeinträgen HTML-Output erzeugen) an.

Praktischer Einsatz

Gefahr einer Fehlkonfiguration

Bei der Konfiguration ist es essentiell wichtig, daß die Integrität der Konfigurationsdateien gesichert ist und sie vor unerlaubter Zugriff geschützt sind. Selbst das Lesen dieser Dateien könnte einen Angreifer dazu befähigen, Pakete zu konstruieren, die keiner Regel entsprechen und sie so zum Zielrechner durchschleusen.
Soll das IDS eine Anomalieerkennung leisten, besteht eine Abwägungsfrage: Wird das System zu kurz trainiert, liegen keine zuverlässigen und ausgewogenen Daten vor, und es kann leicht zu 'False Positives' kommen: unbegründete Alarmmeldungen. Wird es zu lange trainiert und muß eine große Bandbreite an Paketen, die nur wenig gemeinsam haben, fertig werden, sind die Regeln am Ende zu schwach und erkennen Angriffe nicht mehr zuverlässig, sog. 'False Negatives' (u.a. auch durch eine bisher unbekannte Angriffsart möglich, der Administrator muß sich ständig über neue Angriffstechniken informieren). Der Systemadministrator muß hier also Fingerspitzengefühl aufweisen, um das IDS in seinem Sinne zu trainieren - auch darf er seltenen Netzwerkverkehr wie z.B. ein monatliches Backup bei der Einrichtung des IDS nicht vergessen. Dies kann zu einer nahezu optimalen Arbeitsweise führen, aber auch leicht zu Konfigurationslücken oder -fehlern. Das IDS muß genau wie eine Firewall ständig gewartet werden!

Testmöglichkeiten

Mit einem Paketgenerator wie z.B. IP-Packet kann der Regelsatz von Snort getestet werden. Möchte man anhand seiner Regelsätze Alarme erzeugen, so bietet sich snot an. Mittels stick kann man einen eventuellen Flaschenhals im IDS ermitteln. Als letzte Testmöglichkeit fungiert der Administrator. Er muß die gesammelten Daten auswerten und entsprechende Maßnahmen ergreifen. Dabei besteht die Gefahr daß umfangreiche Logfiles unter Umständen nicht mehr vollständig überblickt werden können.

IDS als Schwachpunkt

Ein ID System ist nur dann effektiv und kann seinen Dienst zuverlässig leisten, wenn es nicht von einem Angreifer 'ausgehebelt' oder umgangen werden kann. Zwei Angriffspunkte sind hier denkbar: der Computer, auf dem das IDS ausgeführt wird, und das IDS selbst als Opfer. Erste sind allgemein bekannte Unsicherheiten eines jeden Computersystems, zweite speziell auf IDS bezogen. Wir wollen hiernach beide Ansätze vorstellen.

Klassische Angriffe

Ein IDS muß zwecks Absicherung eines Systems auf mindestens einem Computer dieses Systems ausgeführt werden. Wenn dieser jedoch selbst nicht als sicher einzustufen ist oder seine Funktion nicht durchgängig gewährleistet werden kann, ist die zuverlässige Arbeit des ID Systems gefährdet.

Bevor ein Angreifer überhaupt ein IDS attackieren kann, muß er erst einmal herausfinden, daß ein solches überhaupt gibt. Die Existenz kann nur dann herausgefunden werden, wenn das IDS sich in irgendeiner Art 'verrät', etwa über IRS Funktionalität. Es ist fraglich, ob es empfehlenswert ist, die Aufmerksamkeit eines Angreifers auf das IDS zu ziehen. Denn selbst, wenn es nicht abwehrt, kann es immer noch den Angriff mitschneiden. Gelänge es einem Angreifer, das IDS zu lokalisieren und auszuschalten oder zu umgehen, hätte der Systembetreuer gar keine Daten über den Angriff vorliegen. Dies ist der schlechteste denkbare Fall, deshalb ist es angeraten, ein IDS stets ruhig im Hintergrund zu halten.

Sollte es einem Eindringling dennoch gelingen, Zugriff mit entsprechenden Rechten auf den ID-Rechner zu erlangen, könnte er dort ungehindert Konfigurationsdateien ändern oder das IDS ganz ausschalten. Dies hätte zur Folge, daß spätere Angriffe nicht mehr aufgezeichnet werden. Die Organisationseinheit, die sich unter Schutz des IDS wähnt, schöpfte besonders dann keinen Verdacht, wenn nur bestimmte Überwachungsfunktionen und nicht das komplette System außer Kraft gesetzt würden. Aus diesem Grund ist es eher wahrscheinlich, daß ein Angreifer nur für ihn 'gefährliche' Regeln der Angriffserkennung ausschaltet als das komplette System. Die einzig verbleibende Schwierigkeit bei dieser Angriffsart ist die Erlangung einer gültigen Login-Passwort-Kombination, um in das System einzudringen. Dort erst einmal angekommen, gibt es einige hier den Rahmen sprengende (unlautere) Mittel, etwaig fehlende Privilegien zu erlangen. Es sei nur kurz auf den oben erwähnten Schritt und seine Bedeutung in Bezug auf ein IDS eingegangen.

Methoden

Logins und Passwörter sind entweder über automatisierte Methoden wie Massentestmails (da fast jeder zum Login berechtigte Benutzer auch ein Mailaccount besitzt, kann hiermit auf Existenz des Loginnamens geprüft werden) und Wörterbücherattacken auf Paßwörter möglich. Oder aber 'manuell' über die Ausspähung eines zulässigen Benutzers und dessen Passworts mittels Trojanischen Pferden oder einfacher Beobachtung (welche eine

physikalische Anwesenheit voraussetzt, die meist nur unter Kollegen gegeben ist - doch diese können Angreifer 'von innen' sein, siehe oben). Eine andere, sehr trickreiche und besonders auf den Menschen abgezielte Methode ist das sog. 'social engineering', das Austricksen der Person mittels z.B. vorgetäuschter Identität, so daß sie freiwillig die Zugriffsdaten herausgibt (z.B. gibt sich der Angreifer als Servicepersonal aus, der ein "Problem" im Netzwerk ausfindig machen muß und dazu bestimmte Angaben braucht, etwa Username und Passwort). Meldet sich der Angreifer dann mit den gestohlenen Daten am System an, kann das IDS dies nicht erkennen, denn es ist ein regulärer Login-Prozeß, nichts verdächtiges. Erst nach dem Anmelden verrät sich der unerwünschte Besucher möglicherweise durch ungewöhnliche Aktionen, die dann wieder vom IDS bemerkt werden könnten, weil entsprechende Regelsätze vorliegen oder die Anomalieerkennung aktiviert ist.

Im nachfolgenden Abschnitt sollen abschließend die Probleme eines IDS thematisiert werden, die nicht mit der physikalischen & logischen Grundlage (dem Rechner und seinem Betrieb) verbunden, sondern die dem IDS selbst inhärent sind. Sie beruhen auf zwei Techniken, "die es ermöglichen, das IDS mit falsch erscheinenden Pakete zu verwirren oder Inkonsistenzen zwischen dem IDS und den Endsystemen im lokalen Netzwerk auszunutzen" [HM01]: 'Insertion' und 'Evasion'.

Insertion

Der Angreifer manipuliert Datenpakete so, daß sie nur vom IDS angenommen und analysiert werden, aber vom Zielrechner nicht, d.h. diese Daten sind ausschließlich für das IDS gedacht, um es zu verwirren und die Bedeutung anderer Pakete (nämlich die mit dem eigentlichen Angriff) zu verschleiern. Ein Beispiel: die Sendung einzelner Datenbrocken, die zusammengesetzt im IDS eine andere Bedeutung haben, als im Zielsystem, das nicht alle annimmt. Der Angreifer schickt die Einzelpakete 'a', 't', 't', 'a', 'X', 'c' und 'k', die auch alle und in dieser Reihenfolge vom IDS empfangen und zusammengesetzt werden. Da es aber nur nach der Zeichenfolge 'attack' sucht, passieren alle Pakete das IDS unauffällig - es gibt keinen Alarm. Das Paket 'X' ist aber so manipuliert, daß es nicht vom Zielcomputer angenommen wird. Dieser setzt den String 'attack' zusammen. Der Angreifer hat sein Ziel erreicht und seinen Angriff in das Netzwerk und den Computer eingeschleust.

Evasion

Diese Methode funktioniert analog zu Insertion, bloß mit 'vertauschten Rollen'. Das IDS nimmt diesmal ein Paket nicht mit in die Analyse auf, welches jedoch vom Zielhost angenommen wird. Der Angreifer sendet diesmal 'a', 't', 't', 'a', 'c' und 'k' und hat das Paket 'c' manipuliert. Das IDS erkennt es daher nicht und analysiert nur die Zeichenkette 'attak'. Diese kommt nicht in den Regeln vor - es gibt keinen Alarm! Das Zielsystem hat jedoch alle Einzelpakete angenommen, setzt 'attack' zusammen, und der Angreifer hat abermals erfolgreich einen Angriff eingeschleust.

Glücklicherweise stellen Insertion- und Evasion-Attacken keine einfache Möglichkeit dar, ID Systeme zu umgehen, denn die Möglichkeiten, Pakete gezielt in einen Datenstrom einzufügen, sind begrenzt. Prinzipiell sind aber vor allem regelbasierte IDS mit ihren starren Mustervergleichen anfällig für die beiden eben ausgeführten Methoden. Eine dritte Angriffsmethode setzt nicht am Regelsystem an, sondern am IDS als ganzes und versucht es als Ganzes in die Knie zu zwingen: 'Denial of Service' (DoS) Attacken.

Denial-of-Service Attacken

Bei dieser Art des Angriffs geht es darum, eine Ressourcenerschöpfung im IDS oder dem dafür benutzten Rechner herbeizuführen. Die Logik dahinter ist, daß, wenn das System keine Ressourcen mehr anfordern kann, um seinen ordnungsgemäßen Betrieb fortzufahren, es sich selbst abschaltet oder vom Betriebssystem gestoppt wird. In jedem Fall wäre es dann nicht mehr funktionstüchtig und könnte keine Angriffe mehr erkennen. Um dies zu provozieren, versucht der Angreifer eine Bedingung zu schaffen, die das IDS dazu bringt, Systemressourcen komplett zu beanspruchen. Dazu gehören z.B. CPU-Zeit, Arbeitsspeicher, Plattenplatz und Netzwerkkapazität. Aber auch Protokollüberläufe, die eine automatische Deaktivierung des verursachenden Prozeßes nach sich ziehen könnten, wären denkbar.

Speziell auf IDS, die ein IR Komponente enthalten, sind DoS-Attacken ausgelegt, bei denen der Angreifer das System dazu bringt "überzureagieren". Er erzeugt dabei so viele Angriffspakete, daß das IRS kontinuierlich reagiert und anderer Netzverkehr nicht mehr analysiert wird oder nicht mehr passiert. Im ersten Fall könnte er nun erfolgreich ein Angriff starten, im zweiten Fall müßte er nur noch warten, bis ein (ob der Klagen anderer Nutzer) entnervter Systemoperator das IDS abschaltet.

Gegenmaßnahmen

Aus den eben geschilderten Angriffsmöglichkeiten ergeben sich einige einfache Folgerungen: Die allgemeinen Sicherheitsrichtlinien für Computer und -systeme müssen insbesondere für ein IDS eingehalten werden - also z.B. möglichst nicht zu erratende oder Standardpasswörter. Um die Sicherheit zu erhöhen, sollten nur wenige, im Extremfall nur ein Benutzer Zugriff auf das IDS haben. Maximale Sicherheit wäre gegeben, wenn ein einzelner, ausgezeichneter Rechner das IDS beherbergte, dieser nur lokale Logins von einem einzigen Benutzer zuließe und auch nur lesend das Netzwerk analysieren würde (kein Rückkanal, über spezielle Patchkabel zu realisieren). Natürlich sind diese restriktiven Forderungen nicht immer zu erfüllen, ein höherer Bedienkomfort würde aber auf Kosten der Sicherheit des IDS gehen. Dies ist wiederum eine Abwägungsfrage, die der Sicherheitsbeauftragte auf dem Hintergrund der Sicherheitsrichtlinien fällen muß. Wir haben jedoch im vorherigen Abschnitt festgestellt, daß es neben diesen in der Hand des Systemadministrators liegenden Problemen jedoch auch Schwachpunkte gibt, die an der Logik des ID Systems ansetzen und daher eher genereller Natur sind. Hierfür sind Lösungen nicht so einfach zu finden.

Abschließend stellt sich die Frage was man nach einem Anriffsversuch mit den gesammelten Daten macht. Nutzt man sie zum Gegenangriff, so zieht man die Aufmerksamkeit des Angreifers auf das IDS. Eine vorübergehende oder dauerhafte Sperrung der IP-Adresse des Angriffsursprungs ist nicht unbedingt immer sinnvoll da nicht garantiert werden kann daß der Angriff wirklich von der verwendeten IP-Adresse stammte - u.a. IP-Spoofing oder dynamisch vergebene IP-Adressen können die Identifizierung des Angreifers stark erschweren. Alternativ ist eine Sperrung des entsprechenden UDP/TCP-Ports oder eine Terminierung des betroffenen Programmes möglich. Weitere Möglichkeiten der Reaktion auf Angriffe fallen in das Gebiet der 'Intrusion Response Systeme' und werden hier nicht weiter betrachtet.

Möchte man rechtliche Schritte gegen den Angreifer einleiten, so sollte man daran denken daß die Auditdaten nachträglich modifiziert und manipuliert werden können und somit nach §416 der Zivilprozeßordnung kein rechtsverbindliches Beweismittel darstellen. Diese Daten unterliegen der freien Beweisführung und haben somit den gleichen

gerichtlichen Status wie eine Zeugenaussage; eine Beurteilung liegt im Ermessen des Richters. Um solchen Problemen aus dem Weg zu gehen empfiehlt es sich, daß das IDS die gesammelten Auditdaten mittels einer vertrauenswürdigen Signatur signiert.

Referenzen

[HM01] Alexis Hildebrandt, Mathias Meyer: *Intrusion Detection am Beispiel Snort*, http://www.pl-forum.de/work/snort/

[MHL94] Biswanath Mukherjee, Todd Heberlein, Karl Levitt: *Network Intrusion Detection*, http://seclab.cs.ucdavis.edu/papers/mhl94.pdf

Quellennachweise

http://www.atis.org/tg2k/_data_logging.html
http://www.atis.org/tg2k/_sniffer.html
http://www.atis.org/tg2k/_transient.html
http://www.hyperdictionary.com/dictionary/transient
http://www.netaxs.com/~vossenjp/infosec/snort.html
http://www.net-lexikon.de/Sniffer.html
http://www.packetattack.com/network_analysis_sniffers.html
http://www.silicondefense.com
http://www.snort.org
http://www.thefreedictionary.com/transient

Alle WWW-Quellen vom 28.01.2004