Intrusion Detection Systeme

- **am Beispiel von**

- Deniz Aksu

Inhaltsverzeichnis

Definition: 'intrusion' und 'intrusion detection'?

»Eine Intrusion ist ein unerlaubter Zugriff oder Aktivität in einem Informationssystem.« Immer größere Bedeutung erlangen aktive Analysemechanismen wie Intrusion Detection, die den Netzwerkverkehr mittlerweile in Echtzeit analysieren und anhand bestimmter Regeln etwaige Angriffe erkennen und bei Bedarf Aktionen ausführen, um den Angriff abzuwenden, ihn zu protokollieren oder den Administrator zu benachrichtigen.

Ziel des Intrusion Detection

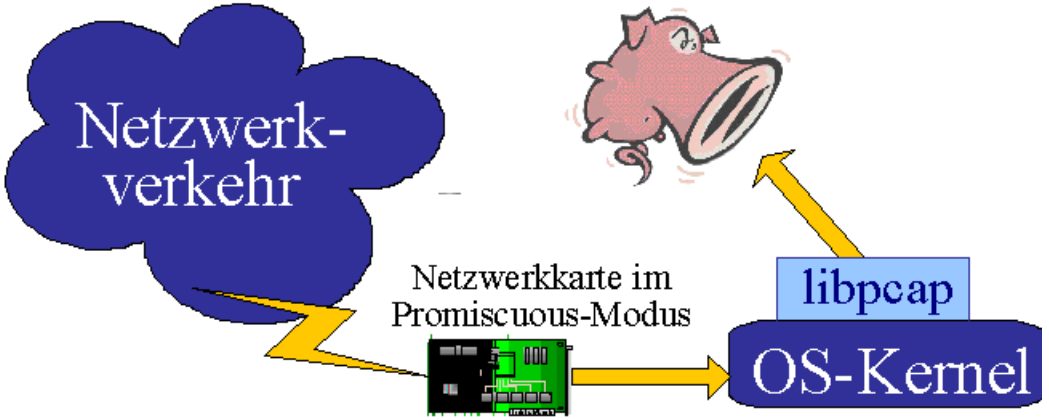
Überwachung von Rechnersystemen und Netzinfrastrukturen zur Erkennung von Einbrüchen und Systemmissbrauch.

Was kann man mit Intrusion Detection erreichen ?

- Überwachung der Einhaltung von Sicherheitspolitiken
  - >Erkennen von Eindringlingen
  - >Erkennen von System-Missbrauch
- Schadenserkennung und – begrenzung „*Response*-Mechanismen“
- Erfahrungsgewinn Verbesserung der „Erkennungs- und Abwehrstrategien“
- Abschreckungseffekt

Aufbau und Funktionalität eines IDS

- Network Engine
- Agent
- Management



Funktionalität eines IDS

Network Engine:

Überprüft die Datenpakete in Echtzeit, sendet Alarmmeldung und trifft automatisch Gegenmaßnahmen (Response). Sie besteht aus fünf Haupteinheiten.

1. Netzwerkadapter:

Die Netzwerkadapter werden im Promiscuous Mode betrieben. Zusätzlich besteht die Möglichkeit, die Network Engine auch im so genannten Transparent Mode zu fahren

2. Packet Capture Modul:

Das Paket Capture Modul organisiert den Transport von Datenpaketen vom Netzwerk Zum Auswertemodul der Network Engine.

3. Filterung:

Je nach Einstellung der Sicherheitspolicies werden Quell-IP- Adressen/Ports und Ziel IP-Adressen/Ports gefiltert.

4. Angriffsdetektor:

Untersucht den Datenstrom nach bekannten Angriffsmustern und werden die Daten-Pakete mit vorhandenen Angriffsmustern in der Datenbank verglichen. Zusätzlich können bestimmte Systemzustände als Angriff definiert.

5. Response Modul:

Falls ein Angriffsmuster erkannt wird, können verschiedene aktive Gegenmaßnahmen (Responses) wie „*Verbindung blocken*“ oder „*Alarm senden*“ eingeleitet werden:

- Überwachen aktiver Sessions
- Speichern von Sessions
- Führen von Log- Dateien.
- Blockieren von Verbindungen (TCP-Reset).

Für die Alarmierung stehen verschiedene Dienste zur Verfügung:

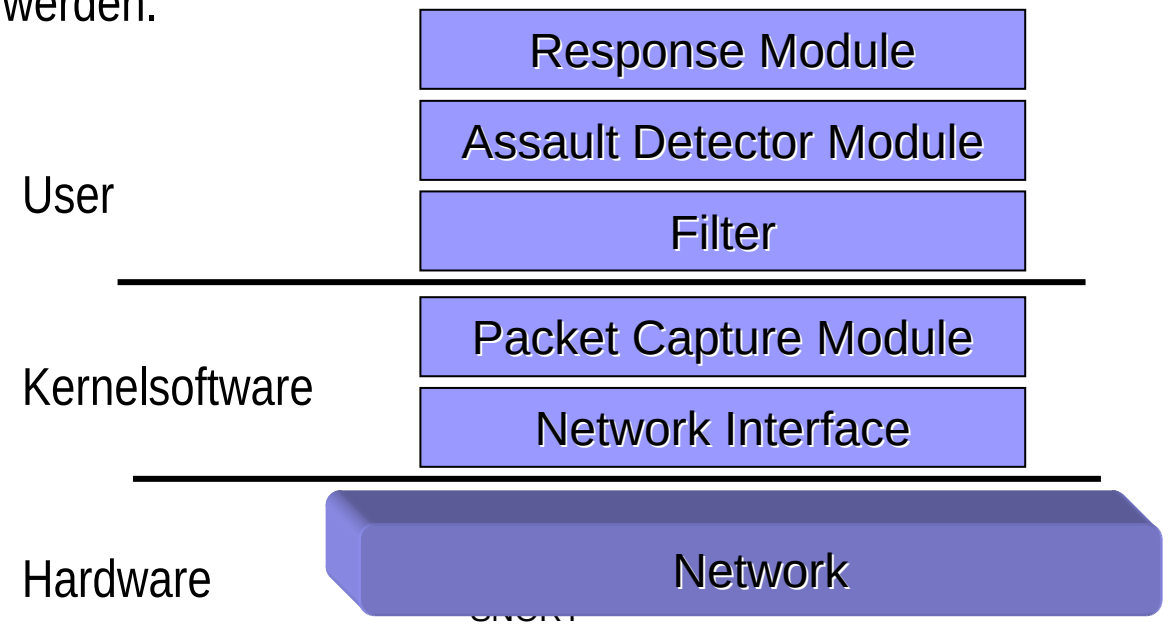
- Alarmmeldung zur Management Console
- E-mail
- SMS, Pager oder Ausführen von nutzerdefinierten Programmen
- SnMP- Trap, z.B. an ein Tivoli Überwachungssystem

Agent:

Überwacht alle Systemaktivitäten eines IT-Systems im Netz. Alle vom Betriebsystem oder einzelnen Applikationen erzeugten Auditdaten werden auf mögliche Angriffe und nach evidenten anormalen Ereignissen untersucht. Dadurch kann das System auch vor unbekannten Angriffen geschützt werden

Management:

Administriert und konfiguriert die Network Engines und den Agent. Auch die Updates Erfolgen über das Management. Es bildet eine Sammelstelle für alle Network Engine und Agent gesendeten Events. Die Angriffe können protokolliert werden, so können Informationen über Bedrohungen für Informations- und Risikomanagement zur Verfügung gestellt werden.



Platzierung der ID-Sensoren

Vor der Firewall

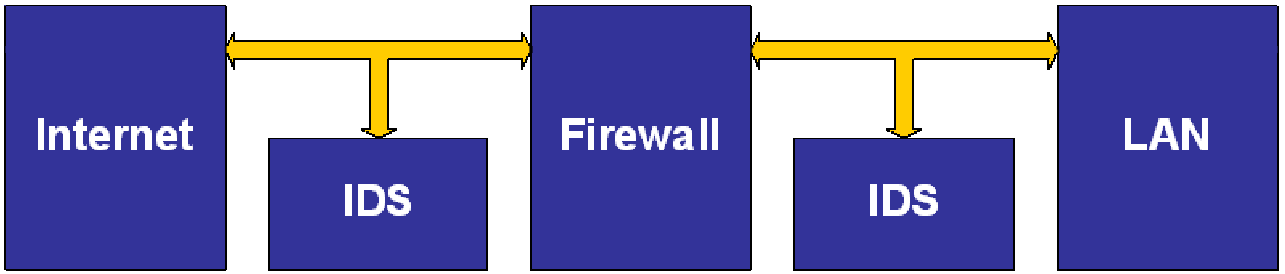
- Kann auch angriffsvorbereitende Erkundungen erkennen
- Viele False Positives

In der DMZ

- Kann helfen, die Effektivität der implementierten Sicherheitsmaßnahmen zu beurteilen

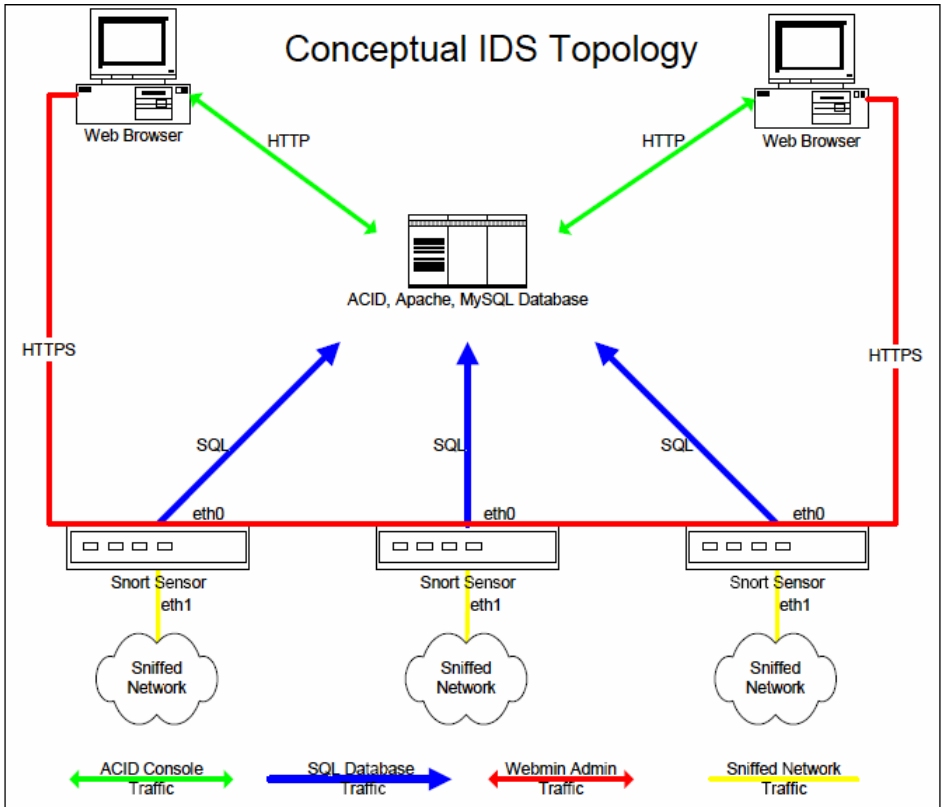
Im lokalen Netz

- Kann vom eigenen Netz ausgehende Angriffe erkennen



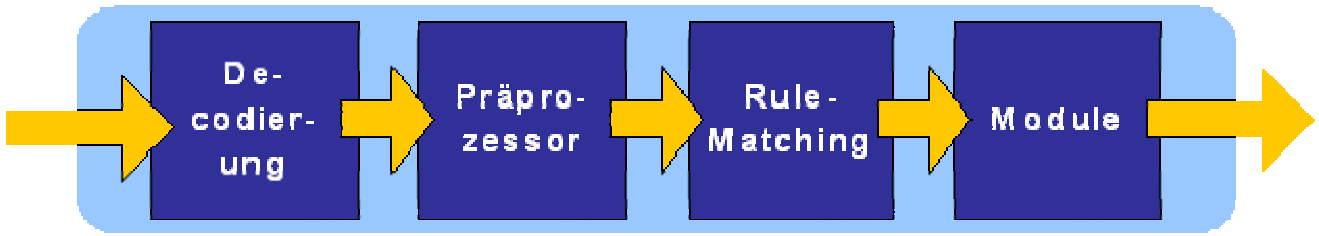
Was ist und Warum Snort?

- Snort ist ein Open Source Intrusion Detection System. Das bedeutet, dass der Quellcode frei verfügbar ist und von dem Anwender auch verändert werden darf. Es lässt sich sehr gut an die Bedürfnisse des Anwenders anpassen und verschlingt dabei keine Unsummen an Lizenzgebühren.
- Es ist ein netzwerkbasiertes IDS, das den Datenverkehr Ihres Netzwerks nach frei definierbaren Regeln analysiert. Aufgrund der einfachen Regelsyntax können leicht eigene Filterregeln programmiert und integriert werden.



Arbeitsweise von "SNORT"

SNORT ist kein monolithisches System, sondern kann durch Plug-Ins Erweitert werden. Die Grundfunktionalität besteht in dem Decodieren der Netzwerkpakete 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



Präprozessoren

Mittels Präprozessoren wird der Datenstrom zusätzlich analysiert 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 zusammen, ideal für erfolgreichen Mustererkennung für zerstückelte Angriffsmuster

stream4: setzt TCP-Stream wieder zusammen und erlaubt so die Analyse eines vorher fragmentierten Angriffs.

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

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)

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

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

Mögliche Fehler eines IDS

- Falsch positiv (eine erlaubte Aktion wird als Angriff identifiziert)
- Falsch negativ (ein Angriff wird vom IDS nicht erkannt oder als unbedenklich klassifiziert)
- Subversionsfehler (sehr komplexe Fehler; der Angreifer kann das IDS unterminieren)

IDS als Schwachpunkt

Ein ID System ist nur dann effektiv und kann seinen Dienst zuverlässig leisten, wenn es nicht von einem Angreifer 'ausgehebelt' werden kann.

- Klassische Angriffe
- Insertion
- Evasion
- Denial-of-Service Attacken
- Gegenmaßnahmen

Klassische Angriffe

Bevor ein Angreifer überhaupt ein IDS attackieren kann, muß er erst einmal ihre Identität herausfinden und Das kann man über IRS Funktionalität machen. 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 einzig verbleibende Schwierigkeit bei dieser Angriffsart ist die Erlangung einer gültigen Login-Passwort-Kombination, um in das System einzudringen.

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.

Evasion

Das IDS nimmt diesmal ein Paket nicht mit in die Analyse auf, welches jedoch vom Zielhost angenommen wird. 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

Denial-of-Service Attacken

Bei diesen Angriffen geht es darum, eine Ressourcenerschöpfung im IDS oder dem dafür benutzten Rechner herbeizuführen. Wenn das System keine Ressourcen mehr anfordern kann, wird es sich selbst abschalten oder vom Betriebssystem gestoppt. In jedem Fall wäre es dann nicht mehr funktionstüchtig und könnte keine Angriffe mehr erkennen.
Der Angreifer versucht eine Bedingung zu schaffen, die das IDS dazu bringt, Systemressourcen komplett zu beanspruchen. z.B. CPU-Zeit, Arbeitsspeicher, Plattenplatz und Netzwerkkapazität. 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

Möglichst nicht zu erratende oder Standardpasswörter. Um die Sicherheit zu erhöhen, sollten nur wenige, im Extremfall nur ein Benutzer Zugriff auf das IDS haben.
Was man nach einem Angriffsversuch 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.

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.

Referenzen

- Alexis Hildebrandt, Mathias Meyer: *Intrusion Detection am Beispiel Snort*, http://www.pl-forum.de/work/snort/
- Biswanath Mukherjee, Todd Heberlein, Karl Levitt: *Network Intrusion Detection*, http://seclab.cs.ucdavis.edu/papers/mhl94.pdf
- Nobert Pohlman, Hartmut Blumberg, *Der IT- Sicherheitsleitfaden*
- Prof. Dr.-Ing. H. König,Technische Universität Cottbus, *Intrusion Detection*
- Edin Dizdarevic, System Developer, Internet & e-Security, *Netzwerk Intrusion Detection mit Snort in schnellen Netzen*