Thomas Biege <tom@electric-sheep.org>

Veröffentlicht am: 20. April 2001

Inhalt

1. Was ist ein Intrusion Detection System?

Einfach betrachtet sind Intrusion Detection Systeme (ID Systeme) virtuelle Einbrecheralarmanlagen. ID Systeme dienen zur automatischen Erkennung von Angriffen gegen die Sicherheit von Netzwerken und Computersystemen, genauer gesagt überwachen sie die Einhaltung der Sicherheitspolitik (*Security Policy*).
Ein IDS sammelt Zustandsdaten von einem oder mehreren Computersystemen und/oder Computernetzen, analysiert diese und veranlasst vordefinierte Aktionen aufgrund des Analyseergebnisses.
Das Ergebnis der Analyse kann, abhängig von dem verwendeten Analyseverfahren, folgende Aussagen machen:

- kein Verstoß gegen die Sicherheitspolitik
- Verstoß gegen die Sicherheitspolitik
- nach einem bestimmten Grad auffälliges Verhalten, was zu einem Verstoß gegen die Sicherheitspolitik führen kann (s. *Zustandsautomaten*)

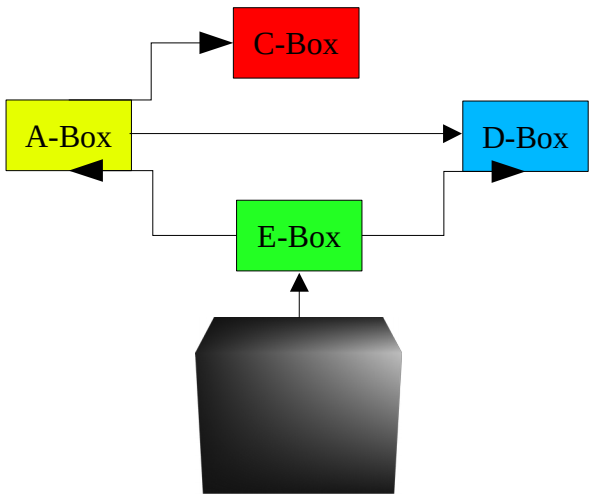
Zusätzlich kann immer noch der Grund für den Alarm angegeben werden.

Intrusion Detection Systeme können i.d.R. der passiven sowie aber auch der aktiven Abwehr von Angriffen dienen. Zusätzlich sollen ID Systeme die rechtliche und technische Verfolgung von Angreifern erleichtern sowie genug Informationen liefern um den verursachten Schaden einzuschätzen, zu beheben und das ausgenutzte Sicherheitsloch zu identifizieren.

2. Aufbau eines Intrusion Detection Systems

Für unsere Betrachtung soll die IDS-Spezifikation des *Common Intrusion Detection Frameworks* (CIDF) [8.3.1] verwendet werden. Nach dem CIDF besteht ein IDS aus vier Komponenten, sog. Boxen.

- Event (E-) Box
- Analysis (A-) Box
- Countermeasure (C-) Box
- Storage/Data (D-) Box



Die Event-Box ist dafür verantwortlich, dass Auditdaten, wie rohe Netzwerkpakete, User-Level und/oder Kernel-Level Logdaten, nahtlos gesammelt werden. Anschließend werden die Daten in einem einheitlichen Format [8.2.7] [8.2.8] [8.2.9] gebracht und an die Analysis-Box, und je nach Konfiguration für etwaige spätere, weitergehende Analysen auch an die Data Box, übertragen. Zuvor sollten jedoch redundante Informationen entfernt werden um Speicherplatz zu sparen und die Informationsverarbeitung zu beschleunigen.
In der Analysis Box findet die Hauptarbeit des IDS statt. Hier werden die Daten von der Event Box auf Angriffsmuster, Anomaltitäten und nicht eingehaltene Sicherheitsvorgaben überprüft. Existiert für das gewonnene Analyseresultat eine vom Benutzer definierte Aktion, wird diese durch die Countermeasure Box, oder auch Intrusion Response Einheit genannt, ausgeführt. Desweiteren wir das Analyseerbegnis an die Data Box weitergeleitet.
Folgende Analyseverfahren werden i. d. R. benutzt:

Misuse Detection:

- Experten System
- State Transition
- Beschreibungssprachen

Anomaly Detection:

- quantitative Analyse
- statistische Messungen
- regelbasierte Systeme
- Neuronale Netzwerke

Die von der Countermeasure Box durchgeführten Reaktionen können zwar in den verschiedensten Formen auftreten unterliegen aber grundsätzlich zwei Kategorien:

Passiv:

- Email verschicken
- Nachricht an einen Pager oder Handy senden
- Audiodaten abspielen
- Dialogfenster öffnen
- Menge der aufzuzeichnenden Auditdaten erhöhen
- kritische Systeme und Dienste abschalten
- Paketfilterregeln von Firewalls anpassen
- Email an den Administrator des attackierenden Systems schicken
- ...

Aktiv:

- Informationen über das System sammeln von dem die Angriffe ausgehen, z.B. eingeloggte Benutzer (*fingerd*, *rusersd* oder *identd*), angebotene Dienste (Portscanning) und das benutzte Betriebssystem (OS Fingerprinting) - die letzten beiden Informationen können für weitere Angriffe benutzt werden
- das attackierende Computersystem mit *Denial-of-Service* (DoS) Attacken lahmlegen
- ...

Die aktiven Reaktionen werden i.d.R. nicht angewandt, da der Angegriffene, das Opfer, selbst zum Täter von Straftaten werden kann. Erschwerend kommt noch hinzu, dass sich der Gegner garnicht auf dem System befinden muss, von dem die Attacken ausgehen. Der Angreifer kann mit Hilfe von *IP Address Spoofing* - hierbei wir die Absenderadresse eines IP-Paketes

gefälscht - seinen wahren Standort verschleiern. Das System, das gerade benutzt wird, kann auch nur ein Opfer des Angreifers sein.

Leider können die passiven Handlungen der C-Box ebenfalls von dem Angreifer gegen das zu schützende Netz gewandt werden. Wenn der Angreifer seine Attacken so aussehen lässt als würden sie von den IP-Adressen eines Geschäftspartners kommen und die C-Box daraufhin die Firewall-Konfiguration verändert um die Pakete aus dem Partnernetz zu blocken, so kann eine Zusammenarbeit verhindert werden. Desweiteren können durch zuviel vom IDS verschickte Alarmmeldungen über Email oder Pager die Mail- oder Modemserver überlastet werden (DoS Attacke), der *Security Officer* (SO) dazu gebracht werden die wirklich wichtigen Meldungen zu übersehen oder den Pager/Handy gleich ganz abzuschalten bzw. die Emails ungesehen zu löschen.

Um diesen Gefahren entgegenzuwirken kann das IDS bspw. so konfiguriert werden, dass bestimmte IP-Adressen nicht gesperrt werden, sondern nur der SO alarmiert oder Rückfrage gehalten wird. Alarmmeldungen des selben Types sollten nur einmal unter Angabe der Menge an den SO gemeldet werden.

3. Analysetechniken

Bei den Analyseverfahren haben sich in den letzten Jahren drei Richtungen herauskristallisiert. Die älteste Methode um Einbrüche festzustellen ist *Misuse Detection*. Bei diesem Verfahren wird ein Mustervergleich (*Pattern Matching*) vorgenommen. Die Daten der *Event-Box* werden mit Angriffsmuster (*Attack Signatures*) aus einer Datenbank verglichen. Ist dieser Vergleich positiv, dann wurde eine Verletzung der *Security Policy* erkannt und es wird entsprechend reagiert. Im kommerziellen sowie im nichtkommerziellen Bereich ist *Misuse Detection* immer noch das am häufigsten benutzte Verfahren. Es ist einfach zu realisieren, anzuwenden und nicht sehr anfällig für falsche Alarme (*False Positives*). Der große Nachteil dieses Verfahrens ist, dass es nur bekannte Angriffe erkennt, was zur Folge hat, dass neue Angriffe, die sich noch nicht in der Signature-Datenbank befinden, keinen Alarm auslösen (*False Negatives*) und somit unbemerkt bleiben.

Um dieses Defizit auszugleichen wurde eine neuer Weg eingeschlagen und das *Anomaly Detection* entwickelt. *Anomaly Detection* geht davon aus, dass alles was nicht zur Menge des ``normalen'' Verhaltens gehört, also demnach ``anormal'' (anomaly) ist, ein Angriff sein muss. Diese Methode hat gegenüber *Misuse Detection* den Vorteil, dass sie es ermöglicht neue Angriffe zu erkennen, da sie ein abnormales Verhalten darstellen. Zudem muss keine Datenbank mit Angriffsmuster aktualisiert und gepflegt werden. Aber auch hier tauchen wieder Schwierigkeiten auf, die die Verbreitung von *Anomaly Detection* im kommerziellen Bereich stark behindern. Verfahren zur Anomalieerkennung müssen zuerst das ``normale'' Verhalten eines Netzes oder Computersystems erlernen indem sie Profile für die Benutzer und das System anlegen. Diese Phase stellt an sich schon eine Hürde da und könnte auch von einem Gegner ausgenutzt werden um dem IDS Angriffe als normales Verhalten beizubringen. Das ID System könnte somit zukünftige Angriffe dieser Art nicht mehr erkennen. Ein weiteres Manko besteht in der hohen Rate an *False Positives*, die durch Störungen in der normalen Systemaktivität ausgelöst werden, aber keine Angriffe darstellen. Zusätzlich ist die Implementierung von *Anomaly Detection*

gegenüber der *Misuse Detection* schwieriger, da die angewandten Verfahren komplexer sind.

Ein noch sehr junges Verfahren, das als *Buglar Alarm*, *Passive Traps*, oder *Strict Anomaly Detection* bezeichnet wird, benutzt einen relativ simplen aber dafür effektiven Ansatz. Es besagt, dass alles was nicht ``gut'' ist ``schlecht'' sein muss. Diese Aussage erinnert an *Anomaly Detection* es wird aber Mustererkennung wie bei *Misuse Detection* für die Erkennung benutzt, d.h. das bekannte, normale Verhalten des Systems wird als Signature in eine Datenbank abgelegt und jede Systemaktivität, die nicht einem der Muster aus der Datenbank entspricht, ist anormales Verhalten und signalisiert einen Angriff.
Es müssen nur relativ wenige Muster in der Datenbank abgespeichert werden. Diese Muster müssen nicht wie bei *Misuse Detection* für jeden neuen Angriff aktualisiert werden, sondern nur bei einer Veränderungen des IT-System. Dadurch ergibt sich ein geringer Verwaltungsaufwand im laufenden Betrieb, als bei normalen *Misuse Detection* Systemen.
Damit das ganze etwas klarer wird ein kleines Beispiel:
Der EMail-Proxy in einem DMZ-Netzsegment darf nur TCP-Pakete auf Port 25 empfangen und absenden. Desweiteren dürfen diese Pakete nur an das Internet oder an das gesicherte/interne Netz geschickt werden. Das ist das normale Verhalten, welches als ein Muster in einer Datenbank abgelegt wird.
Das IDS überprüft nun den Paketfluss des EMail-Proxies mit dem gespeicherten Muster. Sollte der EMail-Proxy diesem Verhalten einmal nicht folgen, so besteht die Wahrscheinlichkeit eines Angriffs. Die Muster müssen sich nicht auf abstrakte Netzverbindungen beziehen, sondern können auch tiefere Ebenen wie Header-Informationen der einzelnen Netzwerkschichten oder Systemaufrufe eines Betriebssystem wiederspiegeln.

Damit die Fehlerquote bei der Erkennung von Angriffen vergleichbar gering bleibt müssen Anomalien, wie sie immer mal wieder in Netzwerken auftreten, als Ausnahme explizit angegeben werden oder in einem Toleranzbereich fallen. Somit können *False Positives* zwar auftreten, sind aber weitaus geringer als bei *Anomaly Detection*, *False Negatives* sind nahezu ausgeschlossen. Wenn Angriffe unter der Toleranzgrenze bleiben, dann werden sie natürlich detektiert.

Nachfolgend sollen die verschiedenen Analyseverfahren für *Misuse* und *Anomaly Detection* kurz erläutert werden. Auf *Strict Anomaly Detection* und andere alternativen Analyseverfahren (Immunsystem, Genetische Algorithmen) soll hier nicht näher eingegangen werden, da sie noch zu neu sind und bisher noch keine Verwendung in der Praxis fanden.

3.1 Misuse Detection

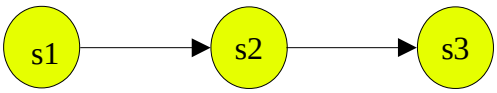
- Experten Systeme

  Für die Analyse der Daten wurden schon früh Fähigkeiten von Experten Systemen entdeckt. Folgenden ID Systemen benutzten Experten Systeme:
  MIDAS, *IDES*, *Next Generation IDES* (*NIDES*) [8.4.1], *DIDS*, *AID* (benutzt *RTworks*) [8.4.2], *Emerald* (benutzt eine Abwandlung von *P-BEST*) [8.5.5] und *CMDS* (benutzt das freie System *CLIPS* [8.3.2]).
  Experten Systeme können mit *if-then-else*-Ausdrücken von dem Anwender leicht programmiert werden. Sie haben aber das Problem, dass sie große Mengen dieser Regeln nicht schnell genug abarbeiten

können, da sie nur Interpreter - und damit bekanntlich langsam - sind. Hinzu kommt noch, dass ein Experten System natürlich immer nur so gut sein kann, wie die Person, die es programmiert hat. Eine Veränderung des Regelwerk gestaltet sich schwierig, da es Auswirkungen auf die restlichen Regeln hat. Durch die reinen *if-then-else*-Bedingungen sind Experten Systeme nicht in der Lage Unsicherheiten bzgl. der Angriffsanalyse zu handhaben, sondern können nur *Ja- oder Nein*-Aussagen machen.

- State Transition (Zustandsautomaten)

  Für Misuse Detection bieten optimierte Zustandsautomaten zur Mustererkennung eine flexible und kraftvolle Möglichkeit Eindringlinge zu erkennen.
  Das erste ID System, das diese Technik benutzte war *STAT* und später *USTAT*, *NetSTAT* und *WinSTAT*, die alle an der Universität von Kalifornien, Santa Barbara entwickelt wurden.
  Auf dem ACM Workshop im Jahr 2000 wurde eine flexible und kraftvolle Beschreibungssprache namens *STATL* für state-transitions-basierte ID Systeme von drei Leuten, die ebenfalls von der Universität von Kalifornien, Santa Barbara kommen, vorgestellt.



  Das obige Bild zeigt den Weg von einem Startzustand (Kreis s1) über Transitionen (Pfeile) zu einem möglichen Endzustand (Kreis s3), der einen Einbruchsversuch signalisiert. Ein Eindringen in das System besteht i.d.R. aus mehreren Aktionen. Jede Aktion wird durch eine Transition wiedergegeben. Die States spiegeln den Systemzustand wieder, der bspw. aus User-Privilegien, vorhandene Dateien oder dem derzeitigen Zustand eines Netz- oder Systemdienstes bestehen kann.
  Das ID System verwaltet für jedes Angriffsszenario ein State-Transition-Diagramme. Wenn eine Systemaktivität stattfindet, überprüft die Analyseeinheit diese mit jedem State-Transition-Diagramm und überführt jeden Zustandsautomaten in den neuen State, wenn die Aktivität mit der Transition übereinstimmt. Dieses Verfahren macht es möglich, die Wahrscheinlichkeit und den Verlauf von Angriffen zu ermitteln und die Schritte des Angreifers vorherzusagen, indem die verschiedenen Zustandsautomaten bzgl. ihres aktuellen States untersucht werden. Wurde der Endzustand erreicht, d.h. ein Eindringen liegt vor, so wird ein entsprechendes Ereignis von der Intrusion Response Einheit (*C-Box*) ausgeführt. Die Zustandsdiagramme bieten ein sehr abstraktes Abbild von den Angriffsmuster und sind somit leicht verständlich und unabhängig vom Format der Auditdaten und technischen Details.
  Zustandsautomaten stellen das Minimum an Aktivität für einen Angriff dar. Aus diesem Grund können Varianten einer Attacke schon mit wenigen, kleinen Zustandsautomaten abgedeckt werden. Auf State-Transition basierte ID Systeme ermöglichen es sogar koordinierte und stark verlangsamte Attacken zu erkennen. Ein verbesserte Variante der *State-Transition-Analyse* wurde mit dem sog. *Colored Pertri* (*CP*)-Netz im *IDIOT* IDS implementiert.

- Beschreibungs-/Interpretersprachen
  Damit sich die Benutzung von Analyseeinheiten in ID Systemen noch etwas einfacher gestaltet wurden Sprachen zur Beschreibung von Angriffsszenarien entwickelt.
  Bekannte Beispiele sind:
    - RUSSEL
    - STALKER
    - N-Code vom Network Flight Recorder (NFR)
    - Snort's Beschreibungssprache
    - STATL
    - und natürlich auch reguläre Ausdrücke, wie in SWatch oder LogSurfer u.ä.

3.2 Anomaly Detection

- Quantitative Analyse
  Es gibt verschiedene Arten der *quantitativen Analyse*, die geläufigste ist die Schwellwerterkennung (*Threshold Detection* oder auch *Threshold and Triggers* genannt). Bei der Schwellwerterkennung werden System-und Benutzeraktivitäten mit Hilfe von Zählern dargestellt. Der Wert der Zähler ist bis zu einem gewissen Schwellwert als normales Verhalten definiert, bei der Überschreitung dieses Schwellenwertes wird ein möglicher Angriff gemeldet. Als Beispiel kann man sich vorstellen, dass sich ein Benutzer in den letzten zwei Stunden drei Mal erfolglos versucht hat an einem System anzumelden. Der Wert 3 ist in diesem Fall der Schwellwert und das IDS reagiert, indem es z.B. den Benutzer-Account sperrt.
  Als Erweiterung der Schwellwertanalyse wurde die *Heuristische Schwellwert Analyse* entwickelt. Bei der *Heuristischen Schwellwert Analyse* sind die Grenzwerte nicht starr sondern werden dynamisch angepasst. In unserem Beispielfall würde der Grenzwert nicht stumpf auf 3 gesetzt werden, sondern wäre anhand des Login-Verhaltens des Users in der Vergangenheit berechnet worden.
  Der Grenzwert könnte bspw. aus der alten Anzahl von fehlerhaften Logins zuzüglich einer Standardabweichung errechnet werden.
  Als weiteres *quantitatives Analyseverfahren* kann man noch die Integritätsprüfung (*Target-Based Integrity Check*) nennen. Bei dieser Methode wird die Veränderung von Systemobjekten (Dateien, Programme, Hardware etc), die sich eigentlich nicht verändern dürfen, überwacht. Das bekannteste Beispiel ist *Tripwire*. *Tripwire* überwacht Dateien und Programme, indem es kryptographische Hash-Werte (*MD5* Algorithmus) von ihnen generiert, sie in einer Datenbank ablegt und diese *MD5*-Hash-Werte in regelmäßigen Zeitabständen mit den Hash-Werten der z.Zt. im System befindlichen Dateien bzw. Programmen vergleicht. Stimmen die Hash-Werte nicht überein, so wurde die entsprechende Datei/Programm verändert, z.B. indem sie von dem Angreifer durch ein *Trojanisches Pferd* ersetzt oder der Inhalt der Passwort-Datenbank verfälscht wurde. Die quantitative Methoden werden aber nicht nur zur Erkennung von Attacken benutzt, sondern auch zur Datenreduzierung (*Data Reduction*).
  Die Eigenschaft Systemattribute und User-Verhalten in Zahlen auszudrücken macht sich das *NADIR* IDS zunutze um die redundanten und nutzlosen Informationen in den Auditdaten zu reduzieren. Die *NADIR*-Entwickler stellten Benutzeraktivitäten mit

Hilfe von Vektoren dar. Diese Profile-Vektoren können für die Experten Systeme und für die Statistische Analyse benutzt werden.

- Statistische Messungen
  IDES und *NIDES* benutzen statistische Verfahren und waren die ersten funktionierenden *Anomaly Detection Systeme* (eigentlich sind es hybrid ID Systeme, da sie sowohl state-transition-basiertes *Misuse Detection* als auch *Anomaly Detection* benutzen).
  Die Systeme verwalten eine Menge von statistischen Profilen, die das Normalverhalten der Benutzer und der Systemkomponenten beinhalten. Die Profile werden in periodischen Zeitabständen aktualisiert. Ein Sicherheitsverstoß wird gemeldet sobald eine starke Abweichung vom Normal-Profil auftritt.
  Ein weiteres IDS, das auf statistische Messungen basiertes, ist *Haystack*. *Haystack* benutzt zwei Mengen von statistischen Profilen. Eine Menge dient zum Vergleich des User-Verhaltens gegenüber Angriffsverhalten und die zweite Menge wird benutzt um eine Abweichung vom Normalverhalten festzustellen. Mit statistischen Verfahren können Angreifer erkannt werden, die sich als andere Benutzer maskieren, indem sie deren Account benutzen. Da sich der Angreifer anders verhält als der Benutzer, dessen Account missbraucht wird, wird das IDS darauf aufmerksam und alarmiert den SO. Zudem können auch völlig neue Angriffstaktiken und -methoden erkannt werden, die ein System mit quantitativer Analyse nicht erkennen kann. Der große Nachteil vieler statistischen Verfahren ist die Unfähigkeit im Realtime-Betrieb zu arbeiten, was eine sofortige Reaktion auf einen Angriff unmöglich macht. Zur Entwicklungszeit dieser Analysemethode war der Batchmode-Betrieb völlig ausreichend, da sie für die Überwachung von Mainframes konzipiert waren und es noch keine so ausgeprägte Vernetzung wie heute gab. Der Versuch statistische Analysen für Realtime-ID-Systeme zu verwenden scheiterte an der hohen Performanz, die dafür nötig war. Hinzu kommt, dass statistische Messungen nur eine Aktivität und nicht eine Abfolge von Aktivitäten analysieren können. Diese Eigenschaft schränkt die Verwendung solcher Systeme im Hinblick auf die Menge der zu detektierenden Attacken stark ein, da die meisten Angriffe aus einer Abfolge von Handlungen bestehen.

Die o.g. Verfahren basieren alle auf Annahmen bzgl. des normalen und nicht-normalen Verhaltens eines Systems. Der Grenzwert, den das IDS benutzt, darf nicht zu niedrig sein um falsche Alarmierungen (*False Positives*) zu vermeiden, was nur dazu führt, dass das IDS nicht mehr beachtet wird. Andererseits darf der Grenzwert nicht zu hoch liegen, sodass Angriffe nicht erkannt werden (*False Negatives*).

Die folgenden Ansätze benötigen keine Parameter für den Grenzwert und versuchen die Nachteile der parameterbehafteten Analyseverfahren auszugleichen.

- regelbasierte Systeme
  Diese Systeme funktionieren im Grunde genau wie die statistischen Systeme, nur mit dem Unterschied, dass Regeln anstatt Statistiken benutzt werden. *Wisdom and Sense* (*W&S*) ist bspw. ein regel-basiertes IDS. Die Regeln können *W&S* entweder durch den Benutzer

eingegeben werden (um die *Security Policy* darzustellen) oder es kann die Regel von alten Auditdaten generieren.
Das *Time-Based Inductive Machine* (*TIM*) IDS benutzt auch Regeln, geht aber einen ganz anderen Weg als andere *Anomaly Detection* Systeme. Es fandet nicht nach individuellen Ereignissen sondern überwacht deren zeitlich Abfolge. Eine Anomalie liegt vor, wenn die zeitliche Abfolge von Systemereignissen nicht eingehalten wurde.

- Neuronale Netzwerke
  Neuronale Netzwerke können anormales Verhalten mit Hilfe von adaptiven Lerntechniken erkennen und benötigen deshalb keine Parameter, die durch den Benutzer vorgegeben werden müssen. Das Neuronale Netzwerk muss erst mit sauberen, d.h. nicht mit Angriffsaktivitäten verunreinigten, Daten trainiert werden. Aufgrund dieser Lerneigenschaft stellen Neuronale Netzwerke einen großen Wert für die Anomalieerkennung dar. Leider lassen sich durch Benutzung von Neuronalen Netzwerken nicht die Ursachen einer Anomalie herausfinden. Dem SO kann nur das Vorhandensein eines Sicherheitsverstoßes aber nicht dessen Grund gemeldet werden.
  Um dieses Problem zu umgehen werden Neuronale Netzwerke entwickelt, die sich nur auf einen Angriffstyp beschränken. Das bedeutet, wenn das Neuronale Netzwerk für *SYN-Flooding* feuert, dann weiß man, um welchen Angriff es sich handelt.

4. Architekturen

Neben der Vielfalt der Analyseverfahren gibt es noch verschiedene Architekturen für ID Systeme. Hier sollen nur die verbreitetsten Varianten angesprochen werden. Als erstes müssen einige Unklarheiten bei den Begriffen hostbasiertes und netzbasiertes IDS aus dem Weg geräumt werden.
Der hostbasiertes IDS mag suggerieren, dass ein IDS zur Überwachung der Sicherheit von Computersystemen verwendet wird; das ist aber nicht ganz korrekt. Als hostbasierte ID Systeme werden Produkte bezeichnet, die ihre Auditdaten von einem Host bekommen, also dessen Log-Dateien oder ausgeführte Systemaufrufe (Syscalls). Hostbasierte ID Systeme sind in der Lage 1-n Rechner zu überwachen; sie sind also nicht auf einen Rechner beschränkt.

Netzbasierte ID Systeme analysieren Netzwerkdaten, es egal wie sie diese erhalten und wo sie sie sammeln.
Dabei überwachen sie die Sicherheit von Computersystem in einem Netzwerk. Häufig wird der Fehler gemacht nur solche ID Systeme als netzbasierend zu bezeichnen, die ihre Daten von einem Netzwerkinterface erhalten, was im Promiscuous-Mode arbeitet um alle vorbeikommenden Netzpakete mitzulesen. Solche Systeme sind die klassischen sensorbasierten ID Systeme, modernere Ansätze verwenden Agenten, die auf den zu überwachenden Computersystem installiert sind um die Netzpakete aus den unterschiedlichen Schichten des TCP/IP Stacks abzufangen und an eine zentrale Analyseeinheit zu schicken.

4.1 Hostbased IDS (H-IDS)

Die hostbasierten ID Systeme sind die ältesten. Sie wurden vom amerikanischen Militär eingesetzt um Datenbank-Hosts o.ä. zu überwachen.

Im Normalfall werden Userlevel-Logdaten (syslog bei Unix Systemen) und/oder Kernellevel-Logdaten (z.B. BSM Auditing Subsystem von Solaris, AIX Audit Subsystem, NT Event Log o.ä.) auf Einbruchsmuster untersucht. Andere H-ID Systeme, wie z.B. PortSentry [8.3.4], lauschen auch an TCP-/UDP-Ports und alarmieren den SO, wenn eine Verbindung zu den entsprechenden Ports aufgebaut wird.
H-ID Systeme haben den Vorteil, dass sie weniger *False Positives* auslösen als netzbasierte ID Systeme, da sie genau wissen wie das Computersystem reagiert. Zudem können sie auch Angriffe sehen, für die ein N-IDS blind ist, z.B. Angriffe über die Konsole, Modems, verschlüsselte Verbindungen oder bösartige Programme (wie *Trojanische Pferde* o.ä.). Nachteilig macht sich bemerkbar, dass ein H-IDS auf jeden zu überwachenden Rechner installiert und gewartet werden muss.
Denial-of-Service Angriffe, die Fehler im TCP/IP Stack eines Systems ausnutzen, können von einem H-IDS nicht ausreichend oder sogar garnicht erkannt werden, weil das System vorher abstürzt und ein hostbasiertes ID System i.d.R. auf einer höheren Ebene operiert als z.B. ein netzbasiertes IDS.

4.2 Netbased IDS (N-IDS)

Sensorbasierte N-ID Systeme erfreuen sich seit einigen Jahren großer Beliebtheit, da die Anzahl der Netzwerkverbindungen und damit auch das Sicherheitsbedürfnis immer stärker wuchsen. Zudem bot die Technik genug Leistung um große Datenmengen in Echtzeit - wie immer man das definieren möchte - analysieren zu können.
Sensorbasierte N-ID Systeme lassen sich gut zentral verwalten und einsetzen. Somit kann ein IDS ganze Computernetzwerke einfach mit Hilfe eines Rechners überwachen und es muss nicht an jeden Hosts einzeln Hand angelegt werden.
Diese Eigenschaft macht sensorbasierte N-ID Systeme sehr attraktiv, da sie weniger administrativen und finanziellen Aufwand erfordern.
Die Event-Box eines sensorbasierten N-IDS ließt alle Daten, die über einen Netzwerksegment transportiert werden, und reicht sie anschließend an die Analyse-Box weiter.
Anhand der Informationen im Header (Flags, Attribute, etc) können DoS-Attacken oder Scans (Abtasten des Netzwerks auf Rechner oder Abtasten von Rechnern auf angebotene Dienste) erkannt werden. Die Payload enthält die Daten, die von dem TCP/IP Stack an die Applikationsebene weitergereicht werden. Diese Daten werden von der A-Box auf Angriffsmuster geprüft.
Bei der Verwendung von N-ID Systemen treten massive Probleme auf, die teilweise in der Zukunft noch größere Auswirkungen haben werden.

Das offensichtlichste Problem, dem sensorbasierte N-ID Systeme gegenüberstehen, ist die Verschlüsselung der Daten. Nicht nur VPN-Lösungen wie IPsec o.ä. sondern auch chiffrierte Login-Verbindungen, wie bei SSH, macht ein klassisches N-IDS blind. Die Daten können nicht dechiffriert und somit auch nicht analysiert werden. In solchen Fällen ist ein hostbasiertes System klar im Vorteil. Der Versuch dieses Problem mit einer *Public Key Infrastruktur* oder ähnlichen Keymanagement-Verfahren begegnen zu wollen ist nicht sinnvoll, da dadurch die Komplexität stark erhöht würde und somit die Robustheit des IDS sinkt. Zudem würde sich die Analyse extrem verlangsamen, da kryptographische Algorithmen sehr

rechenintensiv sind.

Die wachsende Netzwerkbandbreite macht es einem sensorbasiertes N-IDS schwer alle Daten nahtlos zu lesen ohne das interne (Ring-) Puffer überlaufen und Daten verloren gehen. Der Network Flight Recorder schafft es ein saturiertes 100MBit Netzwerk zu überwachen.

Ein weitere Hürde die es zu bewältigen gilt stellt die geswitchte Netzwerkinfrastruktur dar. Im herkömmlichen Ethernet teilen sich alle Rechner eines logischen Segments das Übertragungsmedium (physikalisches Segment). Das bedeutet, wenn zwei Rechner gleichzeitig Pakete übermitteln wollen, dann kollidieren die Pakete u. U. und müssen erneut versandt werden. Die Rechner befinden sich also in einer gemeinsamen Collision-Domain. Ein Switch vermeidet Kollisionen, indem er jeden Rechner einen eigenen physikalischen Netzwerkstrang über Ports am Switch zuweist und die Pakete genau zu dem Rechner schickt an den sie adressiert sind - ein Switch erzeugt also kleinere physikalische Segmente in denen sich nur ein Rechner befindet. Wenn die Netzpakete nur an den dafür vorgesehenen Rechner gelangen, dann kann die E-Box des N-IDS die Pakte nicht mehr mitlesen das IDS ist somit wiedereinmal blind.
Einige Switches haben einen sog. Spy oder Monitoring Port, der es dem N-IDS erlaubt die Pakete zu lesen. Für ein 100MBit Netzwerk hat jeder Port, incl. Spy Port, des Switches eine Bandbreite von 100MBit. Wenn der Switch 10 Ports, ohne Spy Port, hat, müsste der Spy Port 10 x 100MBit Bandbreite haben um alle Daten von allen Ports transportieren zu können. Er hat nur 1/10 dieser Bandbreite und kann somit in einem voll ausgelasteten Netzwerk garnicht alle Daten lesen. Der Spy Port ist also eher eine Notlösung für den Intrusion Detection Bereich.

Um die Daten analysieren zu können muss das sensorbasierte N-IDS alle gängigen Netzprotokolle wie TCP/IP, IPX, Appletalk etc verstehen. Wenn in dem zu schützenden Netz ein sehr exotischen Protokoll gesprochen wird, welches der IDS Hersteller nicht implementiert hat, dann ist das IDS wertlos.

Die Spezifikationen für den TCP/IP Stack sind zwar in den RFC Dokumenten vorgegeben, werden aber von jedem Betriebssystemhersteller anders implementiert. Das hat zur Folge, dass das sensorbasierte N-IDS nie genau weiß, in welchem Zustand sich der TCP/IP Stack eines Hosts befindet. Die Diskrepanz in den verschiedenen Stack-Implementierungen kann sich ein Angreifer zunutze machen um seine Pakete gegenüber dem IDS zu „verstecken", oder er kann Pakete im Datenstrom des IDS einfügen, die das Zielsystem nicht anerkennt und ignoriert. [8.2.6]
Dieser Sachverhalt soll an einem einfachen Beispiel erklärt werden:
Ein Angreifer schickt ein TCP-Paket, mit den Angriffsdaten, an einen unsicheren Netzdienst des Zielsystems.
Die Checksumme im TCP Header ist absichtlich falsch berechnet. Das IDS sieht das Paket, ignoriert es aber, weil die Checksumme falsch ist. Das Zielsystem empfängt das Paket überprüft die Checksumme aber nicht, weil es nicht implementiert wurde und reicht die Nutzdaten an den Server weiter. Resultat: Das IDS hat das Paket verworfen, aber das Zielsystem akzeptierte es und der Angreifer konnte das Sicherheitsloch unbemerkt ausnutzen.

Aufgrund dieser gravierenden Probleme ist der Einsatz von reinen sensorbasierten N-IDS in Zukunft fraglich.

Knotenbasierte (nodebased) N-ID Systeme vermeiden viele Probleme

der klassischen N-ID Systeme, sind aber aufgrund ihres dezentralen Charakters nicht so einfach einzusetzen wie ihre älteren Kollegen.
Auf den einzelnen Computersystemen werden kleine Monitore eingesetzt, die in den TCP/IP Stack eingebunden werden um die Netzwerkpakete auf einer höheren Ebene zu lesen um sie dann zentral (oder auch dezentral) zu analysieren. Inkonsistenzen bei der Interpretation von Netzverbindungen, überfüllte Puffer und Verschlüsselungen auf niedrieger Ebene (IPsec etc.) sind somit kein Problem mehr. Verschlüsselung auf höherer Level z. B. durch Secure Shell, PGP, SSL usw. können aber immer noch nicht im Klartext betrachtet werden, außer natürlich mit kompliziertem Key Management.

4.3 Alternativen

Neben den host- und netzbasierten Ansätzen gab es noch andere Entwicklungen in der Architektur von Intrusion Detection Systemen, die zum großen Teil aber keinen Einsatz in der Praxis fanden (z. B. *mobil autonomous Agents*). Die bekanntesten Formen sind wohl das agentenbasierte IDS, dass nicht mehr den Weg der monolitischen Architektur von klassischen ID-Systemen geht, und der *Honeypot*, der den Angreifer ein verwundbares System vorgaukelt.

4.3.1 Honeypot Systeme

Honeypots sind Systeme, die vorgeben Sicherheitslücken zu haben. Wenn ein Angreifer dieses System entdeckt, dann wird er es attackieren und keine Probleme haben die Sicherheitsmechanismen zu überwinden. Der Angreifer denkt, dass er sich auf einem normalen Computersystem befindet, wurde in Wahrheit aber in eine abgeschottete Umgebung umgeleitet. Der SO wird darüber benachrichtigt und hat nun die Möglichkeit das Verhalten des Gegners zu beobachten und zu analysieren. Somit ist es vielleicht möglich das Bestreben des Gegners zu erkennen oder sogar neue Angriffsformen zu entdecken.

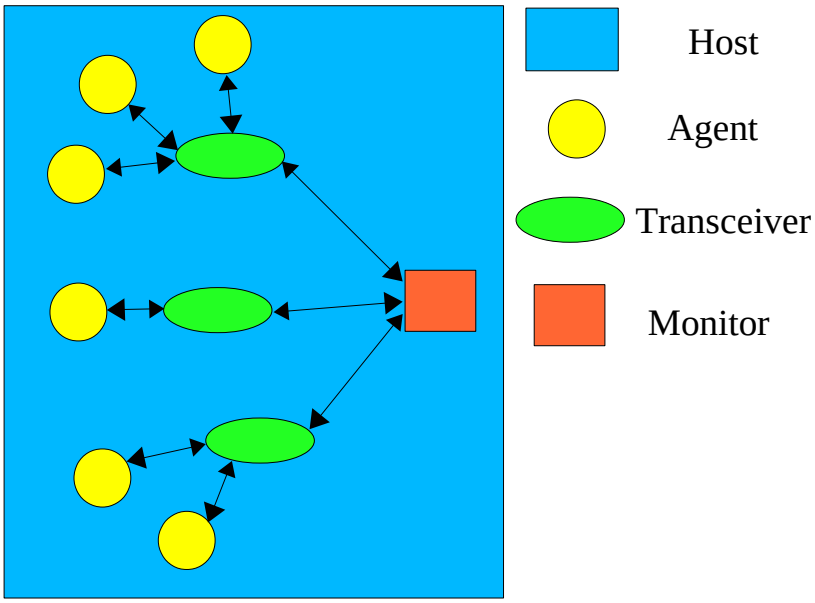
Das *Deception Toolkit* (DTK) und *ManTrap* sind gute Beispiele für ein hostbasiertes Honeypot System.
Theoretisch sind auch netzbasierte Honeypots (Honeynet) möglich, die dem Angreifer ein ganzes Netz vorgaukeln, indem sie auf einem Rechner mehrere *Virtual Machines* (VM) mit virtuellen Netzverbindungen implementieren.
Secure Networks, Inc (SNI) hatte mal ein Honeynet angekündigt, es aber anscheinend nie bis zur Auslieferungsreife gebracht.

Honeypots sind sehr zuverlässig beim Erkennen von Attacken, können dem SO aber unnötig Arbeit machen, da jeder noch so unbegabte Angreifer in das System einbrechen kann. Diese Angreifer wollen häufig keine speziellen Informationen stehlen, sondern nur ihre Langeweile vertreiben oder IRC Bots aufsetzen. Solche Angriffe sind zeitraubend und liefern dem SO keine neuen Erkenntnisse.
Da ein Honeypot nur eine Falle darstellt kann es natürlich passieren, dass der Angreifer nicht in sie hineintappt und er somit unentdeckt bleibt. Honeypots eignen sich besonders in einer DMZ oder vor einer Firewall, da der Angreifer dort nur wenig Angriffsfläche findet und so die Wahrscheinlichkeit höher ist, dass der Honeypot Ziel der Attacken

wird.

4.3.2 agentenbasierte Intrusion Detetcion Systeme (AA-IDS)

AA-ID Systeme sind ein gutes Beispiel für *Distributed Computing*. Auf den zu observierenden System laufen eigenständige, kleine Softwareprogramme (Agenten), die einfach nur Kommandos überwachen oder sogar nach Angriffen Ausschau halten. Die Informationen, die von den Agenten gesammelt wurden, werden an andere Einheiten des ID Systems weitergeleitet um dort verarbeitet zu werden.
Zwei sehr bekannte Beispiele für AA-ID Systeme sind Autonomous Agents for Intrusion Detection (AAFID) [8.2.5] und EMERALD. Anhand von AAFID soll dieser IDS-Type näher erläutert werden.



AAFID ist hierarchisch strukturiert. An unterster Stelle stehen die Agenten. Es können mehrere Agenten, die jeweils unterschiedliche Aufgaben haben, pro Host eingesetzt werden. Alle Agenten senden ihre Informationen an einen Transceiver, der sich auf demselben Host befindet. Der Transceiver überwacht und steuert die Agenten, er kann sie starten, anhalten und rekonfigurieren. Zusätzlich ist der Transceiver noch für die Datenreduktion verantwortlich und übergibt seine Ergebnisse an einen oder mehrere Monitore, die die nächst höhere Instanz der Hierarchie darstellen. Die Monitore selbst können wieder hierarchisch angeordnet werden. Sie kontrollieren die Daten von den Transceivern und fügen sie zusammen. Die Fähigkeit der Monitore die Daten von verschiedenen Hosts zu vereinigen macht es möglich, dass Attacken erkannt werden können, die von mehreren Hosts ausgehen.
Der Aufbau von AAFID erlaubt eine leichte Skalierung. Wenn dem Netz ein neuer Server hinzugefügt wird, müssen einfach nur die AAFID-Komponenten installiert werden, die dann an einen übergeordneten Monitor ihren Bericht schicken.

5. Anforderungen

Folgende Punkte sollten von einem Intrusion Detection Hersteller berücksichtigt werden. [8.2.2]

1. Das System muss einfach zu installieren sein.

2. Die Wartung des IDS muss einfach und kostengünstig sein.

3. Die Verwaltung des ID Systems sollte mit Hilfe einer GUI erleichtert werden.

4. Das IDS soll eine ausführliche Online-Hilfe zur Verfügung stellen.

5. Das IDS muss Benutzern verschiedener Erfahrungsstufen die nötigen Informationen liefern können, d.h. das IDS muss in der Lage sein einem unerfahrenden Benutzer einen Angriff sehr abstrakt darzustellen. Benutzer mit mehr technischem Wissen können von dem IDS nach Bedarf mit detailierteren Darstellungen versorgt werden.

6. Der Grund für ein Alarm muss vom IDS angezeigt werden können.

7. Große Datenmengen müssen so darstellbar sein, dass eine manuelle Verarbeitung möglich ist.

8. Technischer Support sollte vom Hersteller geleistet werden.

9. Angriffssignaturen sollten von dem Hersteller schnell bereitgestellt werden.

10. Eigene Angriffsmuster müssen vom Benutzer manuell eingegeben werden können.

11. Die Herstellerfirma sollte Trainingskurse für das Personal des Kunden anbieten.

12. Das System, auf dem die IDS-Software läuft, muss gut abgesichert sein.

13. Das IDS selbst muss gegen Angriffe gehärtet worden sein.

14. Das Produkt muss leicht skalierbar sein.

15. Um effizienter zu arbeiten sollte das IDS externe, sicherheitsrelevante Informationen aufnehmen können. Bsp.: Urlaubs-, Feier- und Krankheitstage

16. Das ID System sollte hybrid sein, d.h. es sollte Misuse- und Anomaly Detection unterstützen.

17. Die Daten sollten in "Echtzeit" analysiert werden.

18. Daten aus mehreren Quellen (Netzwerk, Logfiles, etc) sollten verarbeitet werden können?

19. Die Authentizität der gesammelten Auditdaten muss gewährleistet werden können, z. B. durch kryptographischen Signaturen. Andernfalls sind die Daten

rechtlich nicht zulässig.

20. Aus rechtlichen Gründen müssen Benutzerdaten pseudonymisiert werden. Mit Hilfe der Pseudonymisierung ist es z.B. nicht möglich die Leistungsfähigkeit von Arbeitnehmern zu überwachen. Dies ist häufig Voraussetzung für den Einsatz von ID Systemen in Firmen. Achtung: Pseudonymisierung kann das Analysieren der Auditdaten stark verlangsamen, da hierfür komplizierte Kryptoalgorithmen benutzt werden.

21. Die unter den einzelnen IDS-Komponenten ausgetauschten Daten müssen verschlüsselt werden. Dem Angreifer werden somit keine nützlichen Informationen gegeben und ihm wird die Möglichkeit der Manipulation verwehrt.

22. Die ID Systemkomponenten sollten ihre Daten über mehrere Kanäle transportieren können damit beim Ausfall eines Kanals keine Informationen verloren gehen. Als Transportmedium kann bspw. das Computernetzwerk, die Telefonanlage, serielle Kabel, Funkstrecken etc dienen.

23. Die Integrität der gesammelten Auditdaten muss erhalten bleiben. Dies kann durch Speicherung auf einmal beschreibbare Medien oder durch Kryptographie erzielt werden.

24. Es muss möglich sein die Granularität der Auditdaten zu verändern. Bspw. kann während eines Angriffes die Granularität erhöht werden.

25. Mehrere Angriffe müssen vom IDS erkannt und priorisiert werden.

26. Beim Ausfall von Komponenten des ID Systems muss sofort Alarm gegeben werden.

27. et cetera

6. Standortabhängige Fragen

Damit ein IDS sinnvoll eingesetzt werden kann muss man sich zuvor einige Fragen beantworten und nicht dem Markt-Hype hinterherrennen. Nur wenn man das richtige IDS für seine Ansprüche findet und sich über dessen Stärken und Schwächen bewusst ist, kann ein IDS eine effiziente Ergänzung der IT-Sicherheits-ifrastruktur bilden.

1. Kann das IDS meine Sicherheitspolitik umsetzen?
   Das IDS und dessen Konfiguration muss flexibel genug sein um sich der Security Policy anzupassen. Natürlich kann ein N-IDS nicht alle Aufgaben eines H-IDS, wie z.B. keine Logins an der Konsole nach der Arbeitszeit, erfüllen und vice versa.

2. Was will ich schützen?
   Will man die Netzwerkinfrastruktur gegen Ausfälle, z.B. durch DoS Attacken gegen Router etc, schützen, dann muss man ein N-IDS einsetzen, da ein H-IDS, weil es auf einer höheren Ebene agiert, solche Attacken nicht oder nicht ausreichend erkennen kann. Zudem ist es unwahrscheinlich, dass hostbasierte ID Systeme für die verschiedenen Betriebssysteme für Router, Server, Workstations

etc. vorhanden sind.
Workstations und/oder Server können einfach und effektiv durch ein N-IDS überwacht werden. Voraussetzung ist, dass die Systeme in einem separaten Netzwerk untergebracht sind - grundsätzlich sollten sich Server, Workstations, Netzwerk-Drucker/Druckerserver und Modem- und Faxserver immer in eigenen Netzsegmenten und Räumen befinden, weil es sich um eigene Administrations- und Sicherheitsdomänen handelt. Um bei der Aufklärung von Angriffen aus dem eigenen Netz helfen zu können, sollte man grundsätzlich Angriffe auf sowie von seinen Systemen überwachen.

3. Wo soll geschützt werden?
   Soll ein Server in der DMZ mit einer 3 GBit Internet-Anbindung überwacht werden? Oder das Subnetz der Forschungsabteilung? Der NIS Server im langsamen 10 MBit Netzwerk, das nur von Besuchern, Praktikanten und Studenten genutzt wird? Dieser Punkt beeinflusst stark die Art und die Ausstattung des ID Systems.

4. Wie viele Systeme muss ich schützen?
   Ist es nur ein Server, oder sind es drei Duzend? Liegen die zu schützenden Systeme in einem Netz oder sind sie verteilt? Liegen die Systeme vielleicht sogar in geographisch weit voneinander entfernten Standorten/Netzen meiner Firma?

5. Wie sieht mein Netzwerk aus?
   Mit Hilfe von Network Management Konsolen sollte man sich einen Überblick von seinem Netz verschaffen. Die Administratoren können wichtige Fragen bzgl. der Geschwindigkeit, Verwendung von Switches, Virtual Private Network (VPN) etc. beantworten. Diese Faktoren nehmen Einfluss auf die Art des IDS und die zu verwendende Hardware.

6. Arbeitet das IDS mit anderen Komponenten meines Netzwerkes zusammen?
   Kann das IDS z.B. *SNMP* Nachrichten an Network Management Stations wie *OpenView* oder *TIVOLI* schicken? Dadurch könnten Verletzungen der Sicherheit einfach und wirkungsvoll angezeigt werden (ein attackierter Host kann bspw. in der grafischen Abbildung des Netzes auf der Network Management Station rot blinken). Die Countermeasure-Box sollte Firewall-Konfigurationen anpassen können oder mit anderen ID Systemen anderer Hersteller kommunizieren können. Um das zu erreichen sind offene Protokolle nötig, die die Interoperabilität wahren. Folgende Standards sind z.Zt. weit verbreitet
   - Common Intrusion Detection Framework (CIDF)
   - Open Plattform for Security (OPENSEC)
   - Adaptive Network Security Alliance (ANSA)
   - Intrusion Detection Message Exchange Format (IDMEF) [8.2.7]

7. Was bringt die Zukunft?

Im Grunde ist es dumm solch eine Frage beantworten zu wollen, da es reine Spekulation ist; aber ich werde trotzdem versuchen einen vorsichtigen Blick in die Zukunft zu geben.

Es halten immer mehr Computer bzw. intelligente Elektronik im alltäglichen Leben Einzug. Sie werden untereinander mit immer höheren Bandbreiten und

exotischeren Protokollen (s. Bluetooth, UMTS etc.) kommunizieren. Die Menge der sensiblen Daten, wie Ergebnisse von medizinischen Untersuchungen, Kreditkartennummern, Liquidität, Einkaufsverhalten etc., die in digitalem Zustand gespeichert werden nimmt, ebenso, wie das Verlangen diese Systeme an öffentliche Netze anzuschließen, stätig zu. Diese Entwicklung muss ein IDS Produkt standhalten. Zudem werden immer mehr Verbindungen verschlüsselt, was den Einsatz von sensorbasierten N-ID Systemen nahezu sinnlos macht.
Neben der steigenden Performanz ist die Interoperabilität zwischen den einzelnen Sicherheits- und Managementprodukten eine Herausforderung für die IDS-Entwicklung. Bisher haben die verschiedenen Hersteller von Network Management Software, Firewalls, Intrusion Detection Systeme, I&A-Produkte etc. ihre eigenen kleinen Insellösungen entwickelt. Durch offene Schnittstellen und die damit verbundene Fähigkeit Daten auszutauschen um zusammenzuarbeiten kann die Sicherheit und Verwaltung von Netzwerken enorm erhöhen.

Heutige ID Systeme beschränken ihre Observation noch zu stark auf den digitalen Bereich. Zukünftige ID Systeme sollte auch Informationen von Alarmanlagen, physikalischen Zugangskontrollen (Magnetkartenschlösser, etc) und Telefonanlagen/Modems berücksichtigen. Somit ist ein ID System in der Lage auch Angriffe auf nicht virtueller Ebene zu erkennen, bzw. die dadurch gewonnen Erkenntnisse bei der Analyse von virtuellen Angriffen zu Rate zu ziehen. Zum Beispiel kann das IDS Alarm geben, wenn sich ein Benutzer A an der Konsole eines Servers anmeldet, aber laut der Magnetkartenschlösser nur Benutzer E im Raum ist.

Wie auch bei Firewalls wird die Entwicklung von freien und kraftvollen ID Systemen zunehmen. Snort und PakeMon sind gute Beispiele dafür.

8. Referenz

8.1 Bücher

8.1.1. Rebecca Gurley Bace, Intrusion Detection, Macmillan Technology Series

8.1.2. Paul E. Proctor; The Practical Intrusion Detection Handbook; Prentice Hall

8.2 Papers

8.2.1. Aurobindo Sundaram, ``An Introduction to Intrusion Detection''

8.2.2. Dr. Josef von Helden, ``Grundlagen, Forderungen und Marktübersicht für Intrusion Detection Systeme (IDS) und Intrusion Response Systeme (IRS)''

8.2.3. Sandeep Kumar und Eugene H. Spafford, ``An Application of Pattern Matching in Intrusion Detection''

8.2.4. J. P. Anderson, ``Computer Security Threat Monitoring and Surveillance''

8.2.5. Mark Crosbie und Gene Spafford, ``Active Defense of a Computer System using Autonomous Agents''

8.2.6. Thomas H. Ptacek und Timothy N. Newsham, Inseration, ``Evasion and
Denial of Service: Eluding Newtork Intrusion Detection''

8.2.7. David A. Curry, Herve Debar, Ming-Yuh Huang; Intrusion Detetction

Message Exchange Format; http://www.silicondefense.com/idwg/draft-ietf-idwg-idmef-xml-02.txt

8.2.8. Matt Bishop; Standrad Audit Trail Format; 1995 National Information Systems Security Conference, pages 136-145

8.2.9. Common Intrusion Specification Language (CISL); http://www.gidos.org/drafts/language.txt

8.2.10. Vern Paxons; Bro: A System for Detecting Network Intruders in RealTime

8.3 Web

8.3.1. Common Intrusion Detection Framework (CIDF); http://www.gidos.org

8.3.2. CLIPS; http://www.unet.univie.ac.at/~a9715573/res/clips.html

8.3.3. IDWG; http://www.silicondefense.com

8.3.4. Psionic Software; http://www.psionic.com/

8.3.5.

8.4 kommerzielle ID Systeme

8.4.1. Next Gerneration IDS (NIDS); http://www.sdl.sri.com/nides/index.html

8.4.2. AID; http://www-rnks.informatik.tu-cottbus.de/de/security/aid.html

8.4.3. Specter; http://www.specter.ch/

8.4.4.

8.5 nichtkommerzielle ID Systeme

8.5.1. Snort; www.snort.org

8.5.2. Packet Monster; http://www.inas.mag.keio.ac.jp/ids/pakemon/index.html

8.5.3. Deception ToolKit; http://www.all.net/dtk/

8.5.4. Autonomous Agents for Intrusion Detection (AAFID); http://www.cerias.purdue.edu/coast/projects/aafid.html

8.5.5. Emaerlad; http://www.sdl.sri.com/emerald/

8.5.6. The Honeynets Project; http://project.honeynet.org