

Intrusion Detection

Hauptseminar
„Malicious Code
(Viren, Würmer und Co.)“

Manfred Schreiber

Technische Universität München
Fakultät für Informatik
Lehrstuhl für theoretische Informatik und Grundlagen der künstlichen Intelligenz
Professor Prof. Dr. Dr. h.c. mult. W. Brauer

Inhaltsverzeichnis

Intrusion Detection	1
<i>Manfred Schreiber</i>	
1 Einführung	3
1.1 Begriffe	3
2 Grundlegende Architektur	3
2.1 Erfassungseinheit (Data Gathering Component)	4
2.2 Verarbeitungseinheit	5
Misused Based ID	5
Anomaly Based ID	6
2.3 Speichereinheit	6
2.4 Reaktionseinheit	6
3 Vernetzung von Intrusion Detection Systemen	7
3.1 Common Intrusion Detection Framework(CIDF)	7
3.2 Sicherheitsaspekte	8
4 Reale Beispiele für host-basierte Intrusion Detection Systeme	9
4.1 RealSecure	9
4.2 Linux Intrusion Detection System (LIDS)	9
5 Abschluß	10

1 Einführung

Das Internet gewinnt immer mehr an Bedeutung und ist aus unserer heutigen Zeit nicht mehr wegzudenken. Über dieses Netz lassen sich viele Firmennetzwerke verbinden. Wegen der Größe des Internets kennt man nicht mehr alle Teilnehmer und damit sind diese nicht unbedingt vertrauenswürdig. Schnell hat man erkannt, dass es nicht nur ehrliche Nutzer im Internet gibt, sondern auch welche eine Gefahr darstellen. Sicherheit in der Netzwerkumgebung ist ein Thema geworden, das immer wichtiger wurde. Um ein sicheres System zu bauen, reicht es nicht nur, dieses via einer Firewall zu schützen sondern man muss diesen Schutz auch überwachen. Ähnlich einer Burg, welche hohe Mauern hat, können diese mit einer Leiter überwunden werden. Erst die Wachen auf den Mauern machen diese Burg sicher. Ebenso verhält es sich mit sicheren Systemen. Diese müssen überwacht werden. Falls ein Angriff erfolgt, kann man Aktionen gegen jenen einleiten. Dieses soll ein Intrusion Detection System (IDS) leisten.

1.1 Begriffe

Mit Intrusion Detection bezeichnet man das Erkennen von Einbruchversuchen in ein Computersystem bzw. Computernetzwerk. Diese Einbruchversuche sind dabei Verstöße gegen die Sicherheitsregeln (Security Policy). Solche Regeln sind unabdingbar für ein IDS. Nur wenn es Sicherheitsregeln gibt, können diese vom IDS benutzt werden, um Aktionen entgegen der Regeln zu prüfen und evtl. Alarm zu melden oder Reaktionen auszuführen. Ein IDS muss folgende Eigenschaften besitzen, damit dieses brauchbar ist:

- **Korrektheit** Ein solches System muss sicher verdächtige Aktionen identifizieren. Nur wenn das System keine normalen Vorgänge als Angriffe oder Angriffe als normale Vorgänge deklariert, ist dieses verlässlich.
- **Performance** Die Verarbeitungsgeschwindigkeit der gesammelten Daten muss schnell genug sein um einen Angriff in Echtzeit zu erkennen. Das heißt, der Angriff muss frühzeitig erkannt werden, bevor Schaden entsteht. Hier wird eine Reaktionszeit unter einer Minute empfohlen.
- **Komplettheit** Ein IDS sollte alle anormalen Vorgänge erkennen und melden. Dies ist ein schwer zu erreichendes Ziel, denn man hat kein vollständiges Wissen aller Angriffsvarianten.
- **Fehler Toleranz** Um zuverlässig zu arbeiten ist auch die Fehlertoleranz eines solchen Systems wichtig. Ein Angriff auf das IDS darf es nicht außer Gefecht setzen.

2 Grundlegende Architektur

Es gibt verschieden Ansätze ein IDS mit diesen Eigenschaften zu realisieren. All diese Ansätze haben trotzdem einen ähnlichen Aufbau. Sie sammeln Daten, speichern und verarbeiten jene. Deshalb kann man ein solches System in folgende Komponenten unterteilen:

- Erfassungseinheit (Data Gathering Component)
- Verarbeitungseinheit (Data Processing Component)
- Speichereinheit (Data Storage Component)
- Reaktionseinheit (Response Component)

2.1 Erfassungseinheit (Data Gathering Component)

Die Erfassungseinheit ist für das Sammeln von Informationen verantwortlich. Hier gibt es eine Vielzahl von Parametern, die überwacht werden können. Je nachdem wo diese Werte gesammelt werden, unterscheidet man zwischen host-basierten IDS und netzwerk-basierten IDS. Um die obig definierten Ziele Korrektheit, Komplettheit und Fehlertoleranz zu erreichen, müssen die Daten vor Manipulationen geschützt werden. Hier ist nicht nur darauf zu achten, dass die Informationen nicht geändert werden können, auch das komplette Unterschlagen bzw. Hinzufügen von Daten muss verhindert werden.

Bei host-basierten IDS werden die Zustände des Systems protokolliert. Hier können z.B. die Aktionen der Benutzer oder System Calls interessant sein. Es gibt mehrere Verfahren um an diese Informationen zu gelangen. Ein Verfahren ist das Auswerten von sogenannten Operating System Audit Trails. Ein Audit Subsystem erstellt diese Trails und fügt bei bestimmten Ereignissen einen Eintrag hinzu. Solch ein Eintrag enthält Daten zu der Aktion, sowie dem Ausführer(Subjekt), dem veränderten Objekt und dem Zeitpunkt. So werden die Systemaktivitäten chronologisch mitprotokolliert. Dadurch dass dieses Subsystem ein Teil des Betriebssystems bzw. Kerns ist, ist es nicht all zu leicht es zu manipulieren. Ein anderer ähnlicher Ansatz ist, die Daten mittels Syslogd zu ermitteln. Hier wird ein eigenständiges Programm benutzt, welches die Logausgaben sammelt und in entsprechende Files ablegt. Auch hier werden Daten über die Zeit, Aktion, Subjekt und Objekt abgelegt. Diese Eigenschaften sind ein Grund dafür, dass Informationen von Syslogd leichter manipuliert werden können. Einer der Vorteile von Syslogd ist dessen Flexibilität. Die Daten können z.B. einfach über die serielle Schnittstelle auf einem anderen Rechner gespeichert werden, was eine Manipulation erschwert. Auch kann Syslogd Loginformationen von anderen Programmen erhalten, was eine umfassendere Auswertung erlaubt. Diese dynamische Erfassung kann durchaus sehr umfangreich sein und somit können sehr viele Daten anfallen. Diese Informationen müssen auch noch analysiert werden, weshalb ein solches IDS sehr ressourcen-intensiv sein kann. Dies kann wiederum die Performance der Maschine beeinträchtigen. Ein weiterer Ansatz ist das target-based Monitoring [2]. Dieser host-basierte Ansatz ist statisch im Gegensatz zu den vorherig vorgestellten dynamischen Ansätzen. Hier werden vorher zu überwachende Objekte festgelegt und auf Veränderungen überprüft. Oftmals werden kryptographische Hashwerte über Systemdateien wie z.B. ls oder netstat gebildet und sicher abgespeichert. Diese Hashwerte werden dann bei jeder Überprüfung neu gebildet und gegen die gespeicherten Werte verglichen. Bei Änderungen an den Systemdateien unterscheiden sich diese Werte.

Ein netzwerk-basiertes IDS überwacht den Verkehr im Netzwerk. Die Idee dabei ist, verdächtige Verbindungen zu identifizieren. Durch die große Menge an

anfallenden Daten haben solche Systeme einen großen Ressourcenbedarf und oft Performance-Probleme. Vor allem bei Fast-Ethernet und Gigabit-Ethernet wird dies zu einem riesigen Problem. Diese Systeme müssen so konzipiert sein, dass sie auch bei hoher Netzwerklast alle Verbindungen überprüfen. Ein großer Vorteil netzwerkbasierter IDS ist, dass keine Veränderungen an den eigentlichen Hosts vorgenommen werden müssen. Das IDS „lauscht“ einfach den Netzwerkverkehr ab und fällt deshalb nicht auf.

2.2 Verarbeitungseinheit

Die Verarbeitungseinheit bildet den Kern des IDS. Sie muss alle gesammelten Daten verarbeiten und abgleichen. Um einen Angriff zu entdecken kann man verschiedene Wege beschreiten. Je nachdem ob man feste Regeln oder das Benutzerverhalten zugrunde legt, unterscheidet man zwischen Misused Based ID oder Anomaly Based ID.

Misused Based ID Bei einem Misused Based ID System geht man davon aus, dass Aktionen und Sequenzen die einen Angriff bilden, vorher bekannt sind. Diese Muster aus Aktionen werden Signaturen genannt. Durch Speichern dieser Muster in einer Datenbank können Aktionen, welche von der Erfassungseinheit aufgezeichnet werden, mit den Signaturen verglichen werden. Es gibt mehrere Arten dies zu machen:

- **Experten System** In einem Experten System wird das Wissen als ein Regelwerk formuliert. Dabei werden die Aktionen jeweils gegen Bedingungen getestet. Falls diese Bedingung erfüllt wird, wird eine Aktion ausgeführt.
- **Model Based Reasoning System** Bei diesem Ansatz werden Angriffszenarien als Sequenzen von Aktivitäten gespeichert. Das System nimmt an, dass zu jeder Zeit ein gewisser Angriff statt findet und sucht nach Anzeichen dafür in den Audit Trails/ Logfiles. Werden Hinweise für den gerade untersuchten Angriff gefunden, wird der Wahrscheinlichkeitswert, dass dieser Angriff tatsächlich stattfindet, erhöht. Steigt dieser Wert über einen bestimmten Grenzwert, wird dieser Angriff gemeldet.
- **Zustandsübergang Analyse** Das System wird als ein endlicher Automat betrachtet. Dabei bilden Teile des Systems und deren Status die Zustände. Diese Systemteile können z.B. Systemdateien oder der Netzwerkstack sein. Die Zustandsübergänge bilden die Aktionen, welche die Systemteile verändern. Geht bei einem solchen Zustandsübergang das System in einen Zustand über, welches ein Sicherheitsproblem darstellt, so wird Alarm gegeben.
- **Tastatureingabeüberwachung** Bei diesem Ansatz wird die Tastatureingabe überwacht. Es wird einfach angenommen, dass gewisse Eingaben nicht erlaubt sind und einen Angriff bilden. Wird eine solche Zeichenkette gefunden, wird sie als Angriff wahrgenommen. Diese Methode ist leicht zu realisieren, hat aber den Nachteil, dass sie einfach umgangen werden kann. Hierzu kann man z.B. Aliase benutzen.
- **Signatur Erkennung** Hier wird nach bestimmten Zeichensequenzen gesucht. Wird dabei ein bestimmter String gefunden, wird dies gemeldet.

Anomaly Based ID Misused Based ID Systeme haben den Nachteil, dass die Angriffsszenarien vorher bekannt sein müssen. Neue oder variierte Angriffe werden nicht erkannt. Um ein System zu entwerfen, welches nicht diesen Nachteil hat, schlug Dorothy Denning [4, Seite 6] einen anderen Ansatz vor. In diesem Ansatz wird davon ausgegangen, dass sich das Verhalten eines Benutzers nicht sehr stark ändert und regelmäßig genug ist um in einem Profil abgespeichert zu werden. Das Profil wird aus einer Vielzahl von Parametern mittels Heuristik und statistischen Mechanismen erstellt und in Variablen mit Werten gefasst. Auch künstliche Intelligenz und neuronale Netze sind für das Erstellen von Profilen denkbar. Ändert sich das Verhalten des Benutzers, so wird davon ausgegangen, dass dies nur stufenweise und nicht abrupt geschieht. Diese Änderungen werden im Profil angepasst. Es wird weiterhin angenommen, dass ein Angriff eine plötzliche Veränderung des Verhaltens des Benutzers beinhaltet. Sind diese Änderungen zu massiv, so wird nicht das Profil angepasst, sondern ein Alarm gegeben.

Dieser Ansatz eignet sich gut um unbekannte Angriffe zu entdecken, doch kann auch dieser Ansatz nicht alle Angriffe erkennen. Behutsam vorgenommene Angriffe können unentdeckt bleiben. Durch langsames Ausführen der Angriffsschritte wird das Profil langsam geändert und die feindlichen Schritte können als legal vom System angenommen werden. Da nicht klar zwischen Angriff und legalen Handeln unterschieden werden kann, hat dieser Ansatz Probleme mit einer erhöhten false positiv Rate. Es kann passieren dass ein Benutzer eine einmalige, spezielle Aktion tätigen muss, die als illegal angesehen wird, ob wohl diese keinen Angriff bildet. Dazu kommt das Problem, dass der Angriff nicht spezifiziert wird und deshalb nur ein Angriff gemeldet wird, aber nicht die Art des Angriffes.

2.3 Speichereinheit

Die von der Erfassungseinheit gesammelten Daten werden vor Ort gespeichert und der Datenverarbeitungseinheit zugeführt. Dies erlaubt eine Pufferung zwischen Datenerfassung und Verarbeitung. Für längerfristige Speicherungen eignet sich dieses Vorgehen nicht. Durch den kontinuierlichen Strom an Daten können diese einen erheblichen Umfang erreichen. Hier ist es zweckmässig eine Hierarchie über die Daten einzuführen. Diese Hierarchie soll dafür sorgen, dass die Daten kompakt und eindeutig abgespeichert werden. Bei der Reduzierung des Datenvolumens muss darauf geachtet werden, dass die Semantik nicht verloren geht. Es sollte weiterhin möglich sein Vorgänge aus diesen reduzierten Datensätzen zu erkennen. Außerdem wird bei den Daten nach Relevanz unterschieden. Je nach Wichtigkeit werden sie für Tage, Wochen oder Monate aufgehoben. Durch das Speichern in einer Datenbank kann der Zugriff beschleunigt werden und für das Entdecken von Langzeitvorgängen herangezogen werden.

2.4 Reaktionseinheit

Die Reaktionseinheit ist für Aktionen verantwortlich, die ausgeführt werden sobald ein Angriff entdeckt wird. Dabei kann eine Person benachrichtigt werden

oder die Einheit kann weitere automatische Aktionen ausführen. Solche vom System aktiv ausgeführten Reaktionen können direkt gegen Angreifer ausgeführt werden, indem z.B. die Netzwerkverbindung getrennt wird. Das System kann auch weitere Informationen sammeln wie z.B. den Ursprung des Angreifers. Aktionen, die eine Person benötigen werden als passiv angesehen, denn diese werden nicht vom System selbst ausgeführt.

3 Vernetzung von Intrusion Detection Systemen

Als die ersten ID Systeme entwickelt wurden, waren diese eigenständige Systeme die jeweils auf den einzelnen Hosts installiert wurden und nur diese auf Angriffe überwachten. So wurden die Daten lokal gesammelt, ausgewertet und auf Angriffe untersucht. Eine Verbindung zu anderen Systemen gab es nicht. Schnell stellte man fest, dass ein Angriff oder verbotene Aktion nicht nur lokal beschränkt blieb sondern auch andere Systeme betraf. Auch konnte ein Angriff von mehreren Seiten ausgeführt werden. Jeder Ursprung führt hierbei eine an sich unauffällige Aktion aus, doch die Gesamtheit ergibt dann sehr wohl einen Angriff. Um solche Angriffe zu entdecken, begann man IDS miteinander zu vernetzen. In diesen Netzwerken gibt es sowohl host-basierte als auch netzwerk-basierte IDS. Diese sammeln Daten und bereiten sie auf. Dann werden die Daten zu einem zentralen Server geschickt um sie auszuwerten. Diese Client - Serverarchitektur ermöglicht zwar eine klar strukturierte Aufgabenverteilung, doch bringt das auch Probleme mit sich. So ist eine zentrale Verarbeitung auch ein „Single Point of Failure“. Sollte diese Systemkomponente ausfallen, ist das Netzwerk ohne Schutz. Auch ist die Skalierbarkeit eines solchen Systems nicht sehr gut. Eine zentrale Einheit kann nur ein bestimmtes Kontingent an Daten verarbeiten und deshalb ist auch die Zahl der Clients beschränkt.

3.1 Common Intrusion Detection Framework(CIDF)

Computernetzwerke bestehen oft aus verschiedenen Computersystemen, e.g. auf einem läuft ein Unixsystem, auf dem anderen ein Windowssystem. Auf beiden Systemen können dazu noch IDS verschiedener Hersteller laufen, welche auf unterschiedliche Arten von Angriffen spezialisiert sind. Dieser Umstand führte zu dem Wunsch, jene miteinander zu vernetzen und beiderlei Datenmengen zu verarbeiten. Um diese herstellerübergreifende Kommunikation zu erlauben, wurde das Common Intrusion Detection Framework entworfen.

Bei CIDF wird das System in vier Komponenten aufgeteilt, die im Kapitel 2 erläutert wurden. Diese sind Erfassungseinheit (E-Box), Verarbeitungseinheit (A-Box), Speichereinheit (D-Box) und Reaktionseinheit (R-Box). Die Komponenten kommunizieren über Intrusion Detection Objects (idos) welche jeweils Informationen über einzelne Ereignisse mit zugehörigen Zeitdaten enthalten. Die Informationen selbst werden mit einer Sprache (Common Intrusion Detection Specification Language - CISL) dargestellt, was einen semantisch korrekten Datenaustausch erlaubt[1]. Ein weiterer Punkt ist eine sichere Kommunikation. Das

CIDF definiert hierfür Nachrichten- und Verzeichnisdienste, mit deren Hilfe sich die Komponenten finden, authentifizieren und sicher kommunizieren können.

Beispiel für CISL

```
(Delete
  (When
    (Time '12:24 15 Mar 1999 UTC')
  )
  (Initiator
    (UserName 'joe')
    (UserID 1234)
    (HostName 'foo.example.com')
  )
  (FileSource
    (FullPathName '/etc/passwd')
    (HostName 'foo.example.com')
  )
)
```

3.2 Sicherheitsaspekte

Ein IDS soll das Netzwerk vor Angriffen schützen, indem es Angriffe aufdeckt. Für einen Hacker ist deshalb das IDS an sich ein interessantes Ziel. Wird dieses außer Gefecht gesetzt bleibt der Angriff unbemerkt. Es sind folgende Arten von Angriffen denkbar:

- **Denial of Service Attack** Ein Denial of Service Angriff (DOS) zielt darauf ab, ein System durch Überlastung unbrauchbar zu machen. Vor allem netzwerkbasierende IDS haben eine große Menge an Daten zu erfassen und zu verarbeiten. Um die Datenmenge zu reduzieren, wird schon in der Erfassungseinheit vorgefiltert. Schickt ein Angreifer nun viele für das IDS potentiell interessante Netzwerkpakete, so können diese den Vorfilter passieren und werden zwischengespeichert. Irgendwann wird der Puffer überlaufen und das IDS muss Daten verwerfen, weil es mit der Verarbeitung nicht mehr nachkommt. In diesem Zustand ist das IDS blind für weitere Angriffe und kann diese nicht mehr entdecken. Um einen solchen Angriff abzuwehren muss der Netzwerkverkehr beschränkt werden, was auch eine Verschlechterung des eigentlichen Service bedeutet.
- **Communication Channel Attack** Ein Angriff auf die Kommunikation hat ebenfalls fatale Folgen. Wird z.B. der Mailserver kompromittiert, können keine Benachrichtigungen per Mail an den Administrator geschickt werden. Auch die Kommunikation in einem verteilten IDS kann gestört werden. Schafft es der Angreifer die Kommunikation zwischen der zentralen Verarbeitungseinheit und den Erfassungseinheiten zu unterbrechen, ist das System blind und kann seinen Dienst nicht mehr verrichten.

- **Direct Attack** Wie jede Software, haben auch IDS Schwachstellen welche ausgenutzt werden können. Ein direkter Angriff zielt darauf ab, das System zum Absturz zu bringen. Hier können z.B. Buffer Overflows ausgenutzt werden.
- **Subterfuge Attack** Je nachdem welches IDS eingesetzt wird, kann man deren Eigenheiten ausnutzen um unentdeckt zu bleiben. So muss z.B. ein Hacker bei einem anomaly based IDS nur langsam genug vorgehen. Durch die geringe Angriffsgeschwindigkeit wird davon ausgegangen, dass sich nur das Verhalten des Benutzers geringfügig ändert. Damit kann man das Benutzerprofil ändern und einen Angriff tarnen.

4 Reale Beispiele für host-basierte Intrusion Detection Systeme

Nachdem die prinzipielle Funktion von IDS vorgestellt wurde, möchte ich zwei reale host-basierte IDS vorstellen.

4.1 RealSecure

RealSecure ist ein Produkt der Firma Internet Security Systems Inc. Es gehört zu den weitverbreitetsten Systemen und wird auf Windows, Linux, HP-UX und Solaris angeboten[3]. Es handelt sich um ein zentrales, verteiltes System, welches sowohl host-basierte als auch netzwerk-basierte Sensoren benützt. Da es sich um ein misused based IDS handelt, werden nur bekannte Angriffsmuster erkannt. Es wird zwischen Sensoren und Managers unterschieden. Sensoren erfassen Daten, scannen Logfiles und nutzen Audit Subsysteme des Kerns. Weiterhin verarbeiten die Sensoren die Informationen und können Verletzungen der Richtlinien feststellen. Der netzwerk-basierte Teil läuft auf den Hosts und überwacht den Netzwerkverkehr. Dieser Teil ist für die Erkennung von Angriffen zuständig, während der host-basierte Teil zeigen kann, ob der Angriff erfolgreich war und das System kompromittiert wurde. Managers sind Module welche zuständig für die Verwaltung der Sensoren sind und bilden den zentralen Teil des Systems. Wie eine Masterconsole führen sie die gesammelten Daten zusammen und zeigen die Ergebnisse an. Außerdem können über die Manager die Signaturen der Sensoren aktualisiert werden, ohne diese neu starten zu müssen.

4.2 Linux Intrusion Detection System (LIDS)

LIDS ist ein unter der GPL-Lizenz veröffentlichtes Projekt und nur unter Linux lauffähig. Es ist zwar ein host-basiertes IDS, verfolgt aber einen anderen Ansatz als ein misused based IDS oder anomaly based IDS. Bei LIDS handelt es sich um ein Kernelpatch, welches eine zusätzliche Rechteverwaltung realisiert. Bei einem normalen Linuxsystem darf ein User, welcher in Besitz von Root-Rechten ist, alles machen. Dies wird von LIDS geändert. So kann das Laden von zusätzlichen Kernelmodulen verboten werden. Auch das Ändern von bestimmten Dateien

kann untersagt werden. Damit kann ein Hacker, welcher in das System eingebrochen ist, keinen Rootkit installieren. Um das Belauschen von Netzwerkverbindungen zu verhindern, wird der Promiscuous Mode der Netzwerkkarte verboten. Firewallregeln können außerdem als nicht änderbar gekennzeichnet werden. Sollte man Wartungsarbeiten an dem System durchführen wollen, welche die kompletten Rechte benötigen, gibt es zwei Möglichkeiten das LIDS abzuschalten. Bei der ersten muss das System abgeschaltet und mit einem bestimmten Bootparameter gestartet werden. Die andere Möglichkeit erlaubt das Abschalten über ein Administrationstool. Hierfür wird ein Passwort benötigt und es funktioniert nur von der lokalen Konsole aus. Wird versucht einer der obig aufgezählten Ressourcen zu ändern, so kann LIDS einen Administrator benachrichtigen oder die entsprechende Konsole beenden.

5 Abschluß

IDS sind ein gutes Mittel um die Sicherheit in einem Netzwerk oder auf einem System zu steigern. So kann man mit einem solchen IDS frühzeitig Sicherheitsprobleme entdecken und rechtzeitig beheben, bevor Schaden entsteht. Doch auch das beste IDS kann keine absolute Sicherheit bieten. Nur die Kombination aus guter Security Policy, sicheren Rechnersystemen und Pflege der Systeme bietet einen guten Schutz. Sicherheit ist kein Produkt, das man installieren kann, sondern ein fortlaufender Prozess.

Literatur

1. *A CISL Tutorial*.
URL: <http://www.isi.edu/gost/cidf/tutorial.html>.
2. *Intrusion Detection Systeme*.
URL: http://www.sec.informatik.tu-darmstadt.de/de/lehre/WS03-04/seminar_fw/ausarbeitungen/Seminar_Gremm_Ausarbeitung.pdf.
3. *RealSecure Server*.
URL: http://www.iss.net/products_services/enterprise_protection/rsserver/protector_server.php.
4. *A Survey on Intrusion Detection Systems*.
URL: <http://www.infosys.tuwien.ac.at/reports/repository/TUV-1841-2000-11.ps>.