TECHNISCHE UNIVERSITÄT ILMENAU

Klassifikation und Einsatzmöglichkeiten von Intrusion Detection Systems mit einer prototypischen Umsetzung

Projektarbeit

an der Fakultät für Informatik und Automatisierung
Institut für Praktische Informatik und Medieninformatik
Fachgebiet Telematik
der Technische Universität Ilmenau

vorgelegt von

Heiko Steigerwald

Matrikelnr.:	25978
Matrikel:	1997
Studienrichtung:	Wirtschaftsinformatik

Hochschullehrer:	Prof. Dr. Ing. habil. D. Reschke
Betreuer:	Thorsten Strufe
Datum der Ausgabe des Themas:	15. Juli 2002
Datum der Abgabe der Projektarbeit:	15. Januar 2003

Inhaltsverzeichnis

Dieses Dokument wurde mit TEX unter Verwendung des Makropaketes LATEX 2_ε und weiterer Pakete gesetzt.

Abkürzungsverzeichnis

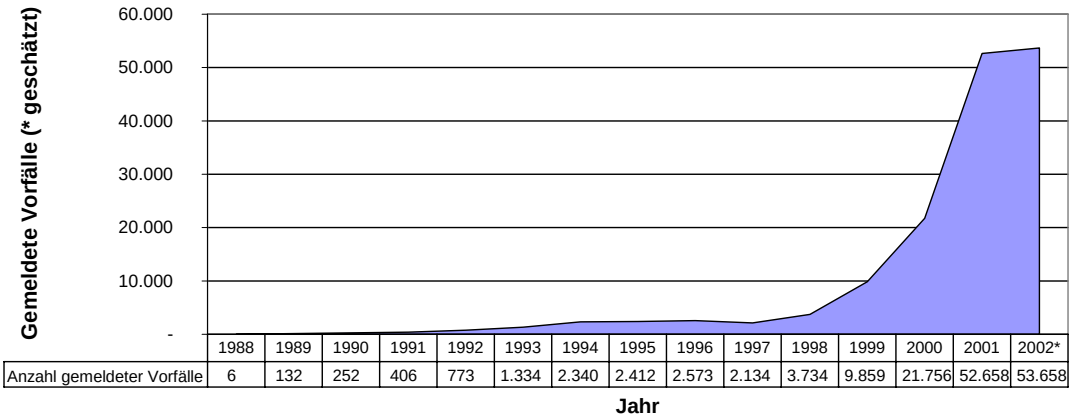
BSM	Basic Security Module
CERT/CC	Computer Emergency Response Team / Coordination Center
CIDF	Common Intrusion Detection Framework
CISL	Common Intrusion Specification Language
DMZ	Demilitarisierte Zone
DNS	Domain Name System
DoS	Denial of Service
EMERALD	Event Monitoring Enabling Responses to Anomalous Live Disturbances
FTP	File Transfer Protocol
GCC	GNU Compiler Collection
GNU	GNU's Not Unix!
HTTP	Hypertext Transfer Protocol
IDES	Intrusion Detection Expert System
ID	Intrusion Detection
IDS	Intrusion Detection System
IDWG	Intrusion Detection Working Group
IETF	Internet Engineering Task Force
IMAP	Internet Message Access Protocol
IP	Internet Protocol
IPSec	IP Security Protocol
ISO/OSI	International Organization for Standardization / Open Systems Interconnection
ISS	Internet Security Systems
LDAP	Lightweight Directory Access Protocol
NIDES	Next-Generation Intrusion Detection Expert System
NMS	Netzwerk-Management Systemen
OPSEC	Open Platform for Security
PDA	Personal Digital Assistant
RAID	Recent Advances in Intrusion Detection
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SNA	Systems Network Architecture
TAP	Test Access Port
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UID	User ID
VPN	Virtual Private Network

Abbildungsverzeichnis

1. Einleitung

1.1. Motivation

Nach Statistiken aus CERT/CC (2002c) hat sich die Anzahl der an das CERT Coordination Center (CERT/CC) gemeldeten sicherheitsrelevanten Vorfälle von 1998 bis 2001 jedes Jahr mehr als verdoppelt. Die Entwicklung ab dem Jahre 1988 ist in Abb. 1-1 dargestellt. Häufige und schwerwiegende Vorfälle im Jahr 2001 waren nach CERT/CC (2002b) unter anderem mehrere Schwachstellen im Berkeley Internet Name Domain (BIND) Server, der sadmind/IIS-Wurm, der "Code Red"-Wurm, W32/Sircam Malicious Code oder der W32/Nimda-Wurm.



	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002*
Anzahl gemeldeter Vorfälle	6	132	252	406	773	1.334	2.340	2.412	2.573	2.134	3.734	9.859	21.756	52.658	53.658

Abb. 1-1: Anzahl sicherheitsrelevanter Meldungen an das CERT/CC nach CERT/CC (2002c)

Neben der Zunahme von sicherheitsrelevanten Vorfällen sind auch die Auswirkungen der Angriffe schwerwiegender geworden. Für beide Tatsachen lassen sich verschiedene Gründe anführen.
Das Wachstum und die zunehmende kommerzielle Nutzung des Internet führen zu einer immer engeren Vernetzung von Computersystemen. Um E-Business erfolgreich betreiben zu können sind Zugriffsmöglichkeiten auf anfangs nicht-öffentliche Datenbestände und Netze zu gewähren. Damit bieten sich potentiellen Angreifern, die heute zunehmend politisch, militärisch oder finanziell motiviert sind, lohnenswerte Ziele. Gleichzeitig hat die Mächtigkeit von Angriffs-Werkzeugen enorm zugenommen und setzt oft nur noch sehr geringes technisches Wissen des Angreifers voraus (Allen u. a. 2000, S. 3 ff.).

Um eine IT-Infrastruktur gegen die vielfältigen Bedrohungen zu schützen, werden Schutzmaßnahmen in einer Sicherheitspolitik festgelegt und dann konsequent umgesetzt. Leider läßt sich trotzdem in der Regel kein vollständiger Schutz vor Angriffen erreichen. Durch die Komplexität heutiger IT-Systeme können Fehler in der Konzeption, der Implementierung und der Benutzung auftreten. Diese Schwachstellen können dann für einen Angriff ausgenutzt werden und zu einem unbefugten Eindringen in IT-Systeme führen.

Es existiert kein vollständiger Schutz vor Angriffen und diese sind letztlich nicht zu verhindern. Angriffe sollen jedoch schnellstmöglich und zuverlässig erkannt werden, um adäquate Maßnahmen treffen zu können. Diese Teilfunktion eines Sicherheitskonzeptes wird durch ein Intrusion Detection System (IDS) realisiert.

1.2. Zielsetzung und Vorgehensweise

Die Ziele dieser Arbeit leiten sich aus vier gestellten Aufgaben ab. Erstens soll ausführlich eine systematische Klassifikation von IDS vorgestellt werden. Zweitens sollen Gründe für den Einsatz von IDS im Rahmen eines Sicherheitskonzeptes dargestellt werden. Drittens sollen dabei erkannte Möglichkeiten und Grenzen aufgezeigt werden. Viertens sollen die theoretischen Erkenntnisse abschließend praktisch angewendet werden.

Thematisch beschränkt sich die Arbeit auf IDS im engeren Sinne. Darunter werden Systeme verstanden, die Aktionen in einer definierten Systemumgebung dynamisch beobachten und entscheiden, ob diese Aktionen symptomatisch für einen Angriff sind oder eine legitime Nutzung der Systemumgebung darstellen (Debar u. a. 1999, S. 806).
Weitere Komponenten zur Umsetzung eines Sicherheitskonzeptes, wie etwa Authentifizierungs-, Autorisierungsdienste oder Paketfilter, werden nur im Kontext gestreift.
Desweiteren wird auch nicht detailiert auf spezifische IDS eingegangen. Eine Übersicht über IDS ist in Sobirey (2000) zu finden. Zur Thematik "Evaluierung von IDS" können etwa die Studien von Allen u. a. (2000) oder von Helden u. a. (1998) als Einstiegspunkte dienen.

Die Arbeit ist im Weiteren nach der folgenden Vorgehensweise aufgebaut. In Kapitel 2 werden grundlegende Begriffe der IT-Sicherheit geklärt und abgegrenzt. Außerdem wird auf Schwachstellen, Angriffe und deren mögliche Schäden eingegangen.
Das Kapitel 3 beginnt mit einem kurzen Abriß der Entwicklungsgeschichte von IDS. Weiter werden grundlegende Funktionsprinzipien und Komponenten von IDS dargestellt. Nach der Herleitung einer Taxonomie für IDS werden die einzelnen Klassifikationsmerkmale und deren Ausprägungen ausführlich erläutert.
Im Kapitel 4 werden Einsatzgründe und Möglichkeiten von IDS im Rahmen eines Sicherheitskonzeptes behandelt. Nachdem aktuelle Probleme und Herausforderungen für IDS gezeigt wurden, sollen zum Abschluß des theoretischen Teiles Ansatzpunkte zur Verbesserung und Entwicklungsrichtungen von IDS aufgezeigt werden.
Das Kapitel 5 beschreibt abschließend knapp die Grundgedanken eines Praktikumsversuches zum Thema IDS. Anhang A enthält die ausgearbeitete Aufgabenstellung des Praktikums und Anhang B Hinweise für den Praktikumsbetreuer.

Ein Hinweis zur Nutzung von englischen Ausdrücken in dieser Arbeit: Wo es notwendig und sinnvoll erschien werden englische (Fach-)ausdrücke aus dem Gebiet der Informatik und Computersicherheit verwendet. Sind diese nicht als Anglizismen im Deutschen verbreitet, so werden sie nach englischen Regeln klein geschrieben und zur Kennzeichnung *kursiv* hervorgehoben.

2. IT-Sicherheit

2.1. Grundlagen der IT-Sicherheit

Die Informationssicherheit "beschäftigt [sich] mit allen Fragen der sicheren Verarbeitung von Information." (Kersten 1995, S. 71) Unter Sicherheit wird im Kontext von IV-Systemen auch oft Ordnungsmäßigkeit als "Ziel oder Eigenschaft, Daten so zu verarbeiten, wie es der Vorstellung des Betreibers eines IV-Systems entspricht" (Kersten 1995, S. 72) verstanden.
Ein IV-System setzt sich aus organisatorischen und technischen Komponenten zusammen. Technische Systeme, wie Computer und Netzwerke, bestehen aus Hard- und Software und werden als IT-Systeme bezeichnet. (Kersten 1995, S. 71) In dieser Arbeit wird im weiteren primär auf technische Komponenten von IV-Systemen eingegangen.
Die "IT-Sicherheit kann als Resistenz informationstechnischer Systeme gegenüber bedeutsam eingestuften Gefahren und Bedrohungen interpretiert werden." (Sobirey 1999, S. 7) IT-Sicherheit betrachtet schutzwürdige Objekte und umfaßt sowohl "Sicherheit gegenüber vorsätzlich verursachten Ereignissen (Security) als auch Sicherheit gegenüber zufällig eintretenden bzw. unabsichtlich verursachten Ereignissen (Safety)." (Sobirey 1999, S. 8) Im Weiteren wird nur die Sicherheit gegenüber vorsätzlich verursachten Ereignissen (Security) betrachtet.

Nach (Kersten 1995, S. 75 f.) umfassen die klassischen Ziele der IT-Sicherheit die Verfügbarkeit (*availability*), die Vertraulichkeit (*confidentiality*), die Unversehrtheit (*integrity*) von Daten und oft auch die Zurechenbarkeit (*accountability*) von Aktionen. Grundlage dafür ist die Verbindlichkeit und Authentizität von handelnden Subjekten und bearbeiteten Objekten.

Das oberste Ziel der IT-Sicherheit besteht darin, Bedrohungen für IT-Systeme durch Sicherheitsfunktionen soweit zu reduzieren, daß nur noch ein vertretbares Restrisiko vorhanden ist. Wichtige Sicherheitsfunktionen sind beispielsweise Identifikation, Authentifizierung, Zugriffskontrolle oder Auditing.
Aber auch durch strikte Anwendung solcher Sicherheitsfunktionen ist absolute IT-Sicherheit nicht realisierbar! (Sobirey 1999, S. 10) Wichtige Gründe dafür sind konzeptionelle Schwachstellen in Systemspezifikationen und die Nichtanwendbarkeit formaler Verifikation bei komplexen Systemen. Weiterhin stellen Implementations- und Konfigurationsfehler oder undokumentierte Funktionalität eine Bedrohung der IT-Sicherheit dar. Durch Mißbrauch von Zugriffsrechten durch reguläre Nutzer oder mangelnde Sensibilität von Nutzern für IT-Sicherheit kann diese ebenfalls bedroht sein.

Um einen möglichst hohen Grad an IT-Sicherheit zu erreichen, wird ein umfassendes Sicherheitskonzept entwickelt. (Kersten 1995, S. 79 f.) Systematisch werden aus der Analyse der Bedrohungen (*threats*) die Risiken (*risks*) und ihnen zugrundeliegende Schwachstellen (*flaws*, *vulnerabilities*) sowie mögliche Schäden ermittelt. In Sicherheitspolitiken bzw. Sicherheitsrichtlinien (*security policies*) werden dann die schutzwürdigen Objekte, die gegen sie gerichtete Bedrohung und ein

angestrebtes Sicherheitsniveau beschrieben. Die Entwicklung und Umsetzung eines konkreten Sicherheitskonzeptes wird beispielsweise von Wolf (2002) durchgeführt. Mittels unterschiedlicher Komponenten, die Sicherheitsfunktionen realisieren, werden die Sicherheitsrichtlinien in die Praxis umgesetzt. Beispiele für solche Komponenten sind Zugriffskontrollmechanismen, Paketfilter und auch IDS.

Der Begriff "*intrusion*" als Wortbestandteil in IDS bedeutet im Englischen einmischen, belästigen oder eindringen. Sobirey versteht darunter einen "Oberbegriff für sämtliche Aktionen, die einer Sicherheitspolitik zuwiderlaufen." (Sobirey 1999, S. 21) Definieren läßt sich ein IDS wie folgt:

> "An intrusion-detection system dynamically monitors the actions taken in a given environment, and decides whether these actions are symptomatic of an attack or constitute a legitimate use of the environment." (Debar u. a. 1999, S. 806)

Ähnliche Definitionen liefern auch Mukherjee u. a. (1994), von Helden u. a. (1998) und Allen u. a. (2000). Debar u. a. (1999) betonen den dynamischen Aspekt von IDS im engeren Sinne um sie von weiteren Werkzeugen zur Abwehr von *intrusions* abzugrenzen. Diese Werkzeuge sind beispielsweise *vulnerability scanner*, *file integrity checker* oder *honey pots* und können ergänzend zu IDS eingesetzt werden (Bace und Mell 2001, S. 23 ff.). Die vorliegende Arbeit betrachtet IDS im engeren Sinne und geht in Kap. 4.1.1 nur am Rande auf ergänzende Werkzeuge zur Erkennung von *intrusions* ein.

Die Notwendigkeit für den Einsatz von IDS ist ursächlich durch die bereits erwähnte nicht erreichbare absolute IT-Sicherheit gegeben. Im Kapitel 4.1 wird detailierter auf Gründe für den Einsatz eines IDS als zusätzliche Komponente im Rahmen eines Sicherheitskonzeptes eingegangen.

2.2. Schwachstellen, Angriffstypen und mögliche Schäden

Nach den Grundlagen der IT-Sicherheit sollen in diesem Kapitel weitere Begriffe aus dem Umfeld "*intrusion detection*" geklärt werden. Zudem sollen mögliche Schwachstellen von Systemen, Angriffstypen und mögliche Schäden durch erfolgreiche Angriffe vorgestellt werden.

Mit Verwundbarkeit (*vulnerability*) oder Schwachstelle (*flaw*) werden Eigenschaften eines Systems oder Objektes bezeichnet, die es einem Gegner ermöglichen, das System in einen von den Sicherheitsrichtlinien nicht erlaubten Zustand zu versetzen. Diese Eigenschaften verhindern also die erfolgreiche Anwendung von Sicherheitsrichtlinien für das System (Allen u. a. 2000, S. 120). Das Ausnutzen einer solchen Schwachstelle zur Erreichung eines bestimmten Zieles nennt man *exploit*. Die Ausführung eins *exploits* stellt einen Angriff (*attack*) auf die Systemsicherheit dar (Allen u. a. 2000, S. 116). Ein Angriff bezeichnet allgemein Aktionen, die ein Angreifer gegen ein Zielsystem (*target*) durchführt um dessen Vertraulichkeit, Unversehrtheit oder Verfügbarkeit zu verletzen (Bace und Mell 2001, S. 40). Damit verletzen Angriffe aus Sicht der Systemverantwortlichen die Sicherheitsrichtlinien bzw. Ziele der IT-Sicherheit. Aus dem Blickwinkel eines neutralen

Beobachters kann ein Angriff entweder erfolgreich (*intrusion*) oder nicht erfolgreich sein (*failed intrusion*) (Allen u. a. 2000, S. 115).
Wird von einem IDS ein erfolgreicher Angriff nicht als *intrusion* erkannt, spricht man von einem "*false positive*" Ereignis. Demgegenüber steht ein "*false negative*" Ereignis, wenn ein IDS irrtümlich einen nicht stattfindenden Angriff meldet (Allen u. a. 2000, S. 116).

Bereits Anderson (1980) hat verschiedene Typen von Angreifern identifiziert. Der *external penetrator* ist ein Nutzer, der sich unberechtigt Zugang zu einem Computersystem verschafft hat. Ein *masquerader* kann ein *external penetrator* oder ein legitimierter Nutzer des Systems sein. Er versucht die Identität eines anderen Nutzers zu erlangen. Unter einem *misfeasor* wird ein berechtigter Nutzer verstanden, der nichtautorisierte Zugriffe auf Daten durchführt. Der *clandestine user* arbeitet auf einer technischen Ebene unterhalb der normalen Erkennungsmöglichkeiten (z.B. als Superuser) und ist aus diesem Grund sehr schwierig zu erkennen.

Angriffe nutzen überwiegend Systemschwachstellen aus. Deshalb wurden verschiedene Klassifikationskriterien für Schwachstellen entwickelt (Alessandri u. a. 2001, S. 13 f.). Aus den Arbeiten von Howard (1997) und von Helden u. a. (1998) lassen sich die folgenden Fehlertypen als Ursache von Schwachstellen ableiten.
Designfehler können im Soft- und Hardwaredesign auftreten und betreffen häufig Kommunikationsprotokolle oder Dienstespezifikationen. Implementierungsfehler öffnen Schwachstellen durch eine fehlerhafte Umsetzung des Designs. Konfigurationsfehler entstehen durch falsche Konfiguration von Hard- und Software. Fehlbedienungen durch Nutzer und *social engineering* öffnen weitere Schwachstellen. Häufige Fehler aus den Kategorien Design- und Implementierungsfehler sind beispielsweise Fehler bei der Eingabeprüfung (z.B. *buffer overflow*), Fehler bei der Zugriffsprüfung, Fehler bei der Behandlung von Ausnahmebedingungen oder Fehler im Zusammenspiel mit der Systemumwelt (vgl. (Bace und Mell 2001, S 45 f.)).

Solche Schwachstellen können von Angreifern ausgenutzt werden, um verschiedene Ziele zu erreichen. In (Alessandri u. a. 2001, S. 12 f.) werden verschiedene Möglichkeiten der Klassifikation von Angriffen vorgestellt.
In (Lippmann u. a. 2000, S. 585ff.) werden die folgenden Kategorien nach dem Ergebnis von Angriffen gebildet: *probe*, *denial of service* (DoS) sowie *penetration attacks* mit der Unterteilung in *remote to local*, *user to root* und *data*. *probes* sind Angriffe mit dem Ziel detailierte Informationen über Netzwerke und angeschlossene Systeme zu erhalten und bereiten oftmals den eigentlichen Angriff vor. DoS-Angriffe versuchen ein Netzwerk, Dienst oder Server unbenutzbar zu machen. Im Fall *remote to local* hat ein Angreifer keine lokale Zugangsberechtigung auf dem Zielsystem und erlangt Zugang zu dem System, kann Daten vom System kopieren oder Daten im Transit manipulieren. Bei einem *user to root* Angriff versucht ein Angreifer mit einer lokalen Zugangsberechtigung Administrations- bzw. Root-Rechte zu erlangen. Zielt ein solcher Angriff auf die Erlangung von schützenswerten Informationen wird auch von einem *data*-Angriff gesprochen.

Eine weitere Klassifikationsmöglichkeit liefert Howard (1997), der den gesamten Angriffsprozeß und das Ergebnis in seine Klassifikation mit einbezieht (vgl. Abb 2-1). In ähnlicher Weise klassifizieren von Helden u. a. (1998) Angriffe nach möglichen Schäden.

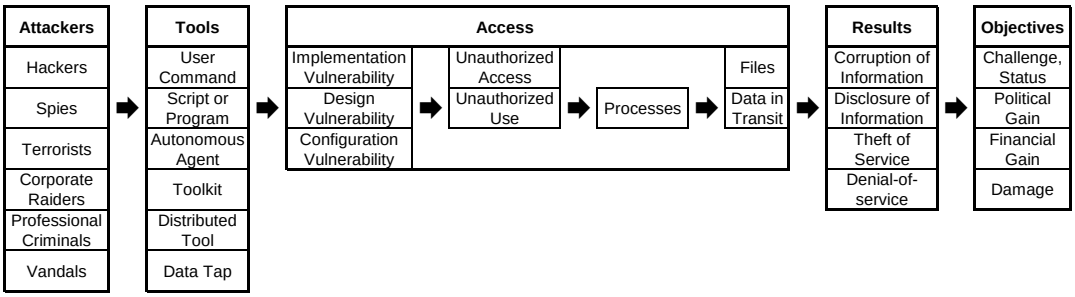


Abb. 2-1: "Complete Computer and Network Attack Taxonomy" aus Howard (1997)

Ein Ergebnis eines erfolgreichen Angriffes kann eine Integritätsverletzung sein (*corruption of information*). Die Datenechtheit (Authentizität) geht verloren, wenn die Quelle der Daten verfälscht wird. Gelingt es einem Angreifer Daten im Transit zu manipulieren, liegt ein Datenintegritätsverlust während des Transports vor. Zu einem Datenintegritätsverlust auf Rechnern kommt es durch Manipulation von auf Rechnern gespeicherten Daten.
Ein weiteres Ergebnis eines erfolgreichen Angriffes kann eine Vertraulichkeitsverletzung sein (*disclosure of information*). Dabei kann ein Angreifer Informationen über das anzugreifende Ziel erhalten. Gelingt es einem Angreifer Daten im Netzwerk bzw. auf Rechnern abzuhören bzw. zu lesen liegt ein Datenvertraulichkeitsverlust während des Transports bzw. auf dem Rechner vor.
Besteht das Ziel eines Angriffs im *theft of service*, so versteht man darunter die unautorisierte Nutzung von Ressourcen ohne andere Nutzer zu beeinträchtigen.
Besteht das Ergebnis eines Angriffes hingegen darin, Ressourcen in einen Zustand der Nichtverfügbarkeit zu bringen, spricht man von *denial of service*.

3. Klassifikation von IDS

3.1. Historie

Im folgenden Kapitel wird auf die historische Entwicklung von IDS und ihren grundlegenden Funktionsprinzipien und Komponenten eingegangen. Anschließend wird eine Taxonomie zur Klassifikation ausgewählt sowie wesentliche Dimensionen und Ausprägungen ausführlich diskutiert.

Für den Beginn der Forschungsarbeiten an IDS läßt sich nach (Frincke und Huang 2000, S. 541) kein genauer Zeitpunkt festlegen. Erste Ansätze zur Erkennung von *intrusions* wurden bereits Mitte der 50er Jahre des 20. Jahrhunderts in Form von Audits auf Hosts[1] entwickelt.
Im Jahre 1980 leitete James P. Anderson mit seiner Studie über "Computer Security Threat Monitoring and Surveillance" (vgl. Anderson (1980)) die systematische Erforschung und Entwicklung von IDS ein. In seiner Arbeit formuliert Anderson erstmals den Gedanken der automatisierten *intrusion detection* und schlägt vor, Audit-Daten zur Erkennung von unautorisierten Zugriffen auszuwerten. Damit wurde der Weg für IDS auf Mainframes geebnet.
Eine weitere wegweisende Arbeit ist die zwischen 1984 und 1986 von Dorothy E. Denning verfaßte Arbeit "An Intrusion Detection Model" (vgl. Denning (1987)). Darin entwickelt sie das Modell eines "real-time intrusion-detection expert system" um IT-Sicherheitsverletzungen zu erkennen. Dem Modell liegt die Hypothese zugrunde, daß IT-Sicherheitsverletzungen anhand von abnormalen Mustern in System-Audits erkannt werden können. Neben weiteren methodischen Grundlagen wurde von Dorothy E. Denning auch der Prototyp eines "Intrusion Detection Expert System (IDES)" entwickelt.
Dieser und die meisten anderen Ansätze basierten auf der Erkennung von Anomalien in System-Audits auf Hosts. Die Systeme waren allerdings wenig ausgereift und daher wenig effektiv. Die Forschung wurde lange Zeit überwiegend von der US-Regierung finanziert. Aus den frühen Forschungsansätzen wurden dann im Laufe der Zeit kommerziell verfügbare IDS entwickelt.

Diese Situation änderte sich in den 90er Jahren des 20. Jahrhunderts und führte zum vermehrten Einsatz von Standard-IDS. Für den umfassenderen und erfolgreicheren Einsatz von IDS gibt es nach (Frincke und Huang 2000, S. 541) drei wesentliche Gründe:
Erstens wurden zunehmend hybride IDS entwickelt und eingesetzt. Diese kombinieren verschiedene Techniken und Datenquellen um eine *intrusion* zu erkennen. Damit arbeiten sie wesentlich effektiver und die Zahl der falschen Alarme läßt sich deutlich senken.
Zweitens hat sich die Nutzung des Internet ab diesem Zeitpunkt sprunghaft entwickelt. Die zunehmende Nutzung des Internet für wichtige und geschäftskritische Anwendungen führt zu einer immer stärkeren Abhängigkeit von funktionierenden IT-Systemen. Im Zuge dieser Entwicklung hat sich auch das Sicherheitsbewußtsein entsprechend entwickelt. IDS stellen nun ein probates Mittel dar, um kritische Teile der Infrastruktur zu überwachen und zu schützen.

[1] Der Begriff Host wird in dieser Arbeit nicht zur Bezeichnung einer Bestimmten Rechnerkategorie (wie etwa anfänglich Mainframes) verwendet. Vielmehr wird damit allgemein der jeweilige zu überwachende Computer gemeint.

Drittens setzte vor einigen Jahren verstärkt ein Prozeß der Konsolidierung von Firmen und von zugehörigen IT-Systemen und Netzen ein. Dies führt zu einer immer engeren Verzahnung von host- und netzwerk-basierten IDS. Außerdem wurde der Bedarf nach einem unternehmensweiten Management von IDS in großen und heterogenen IT-Systemen deutlich.

Diese Tendenzen begünstigten die Erforschung und Entwicklung von zahlreichen Produkten, die beispielsweise in (Allen u. a. 2000, S. 18 ff.), Bruneau (2001) oder auch von Helden u. a. (1998) beschrieben werden. Dort sind unter anderem Produkte wie EMERALD, NIDES (vgl. beide SRI (2002)), die Entwicklung des Netranger der Wheelergroup bis zu seinem Aufgehen in Cisco IDS Produkten (vgl. Cisco (2000)) oder die Entwicklung der RealSecure Produkte von ISS (vgl. ISS (2002)) dargestellt. Eine sehr umfangreiche und aktuelle Zusammenstellung von IDS ist auf der Web-Seite von Sobirey (2000) zu finden.

Der aktuelle Stand der Entwicklung wird von (Frincke und Huang 2000, S. 542) charakterisiert.
Die in den 80er Jahren entwickelten Technologien kommen in aktuell verfügbaren Systemen zum Einsatz. In den Bereichen Data Mining, Benutzerschnittstelle, Ausfallsicherheit und *software engineering* für IDS wurden verbesserte Ansätze und Technologien erprobt und implementiert. Vor allem der Grad der Nutzbarkeit (*usability*) wird für kommerzielle Systeme eine wesentliche Anforderung. Zunehmend wird das umfassendere Management von Mißbrauch bzw. Sicherheitsverletzungen in IT-Systemen gegenüber dem Ansatz der Abschottung vor Eindringlingen und ihrer Identifizierung forciert.
Alle diese neuen Entwicklungen ändern aber wenig an den grundlegenden Funktionsprinzipien und Komponenten von IDS.

3.2. Grundlegende Funktionsprinzipien und Komponenten

In diesem Kapitel sollen nach einer Darstellung der ersten grundlegenden Modelle die funktionalen Hauptkomponenten eines IDS herausgearbeitet werden. Anschließend wird kurz auf Frameworks und mögliche Architekturen von IDS eingegangen.

Bereits in (Anderson 1980, S. 28 ff.) schlägt James P. Anderson die Struktur eines Host- und Audit-basierten Überwachungssystems mit den funktionalen Komponenten "*selection program*", "*sort*", "*session builder*" und "*surveillance program*" vor. Das "*selection program*" erhebt Audit-Daten und leitet diese an eine "*sort*-Funktion weiter. Die sortierten Audit-Daten werden dann durch den "*session builder*" zu einer Nutzer-Sitzung zusammengeführt und durch das "surveillance program" auf mögliche Sicherheitsverletzungen untersucht.
Von Dorothy E. Denning wird in Denning (1987) das Funktionsmodell eines IDS mit sechs Hauptkomponenten dargestellt. Dabei führen *subjects* (Nutzer bzw. Prozesse) Aktionen an *objects* (Ressourcen) aus, wobei die Aktionen in *audit records* aufgezeichnet werden. In *profiles* wird das normale Verhalten der Subjekte in Form von statistischen Metriken hinterlegt. Durch *activity rules* werden Abweichungen vom normalen Verhalten erkannt und entsprechende *anomaly records* generiert.

Ein generelles Modell für ein IDS zeichnet Axelsson (1998) in seiner Arbeit. Dort werden wesentliche funktionale Komponenten eines IDS dargestellt. Ähnliche funktionale Hauptkomponenten unterscheiden auch Alessandri u. a. (2001), von Helden u. a. (1998), Allen u. a. (2000) und Bace und Mell (2001).
In der Abb. 3-1 sind die funktionalen Elemente eines IDS dargestellt. Die Aufgabe des *sensors* besteht in der Sammlung von Daten aus den unterschiedlichsten Quellen. Der *analyzer* (auch *detector* oder *intrusion detection engine*) analysiert die von den Sensoren bereitgestellten Daten um Sicherheitsverletzungen feststellen zu können. Die Komponenten zur Ergebnisdarstellung (*management console*) können ebenfalls als Teil des IDS betrachtet werden.

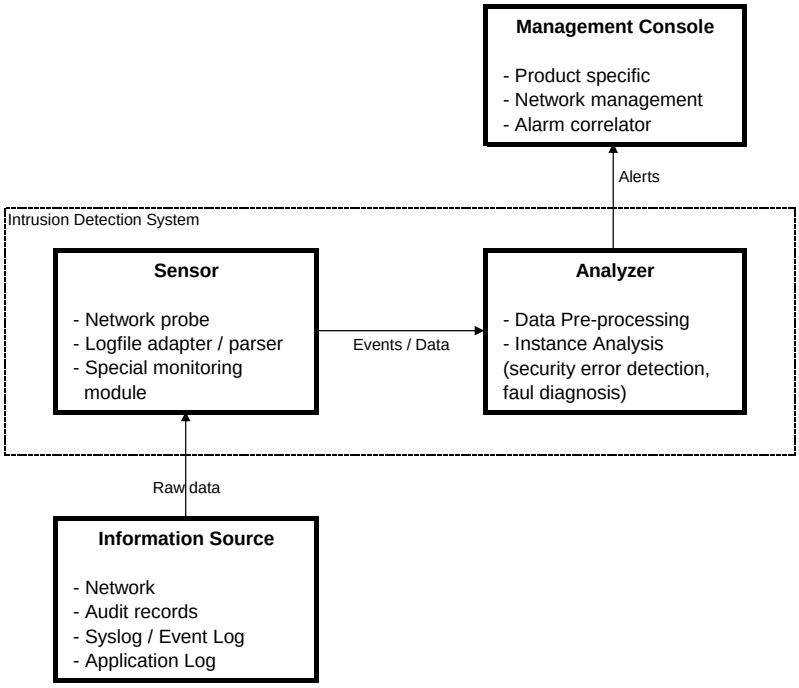


Abb. 3-1: "Intrusion Detection System Model" aus Alessandri u. a. (2001)

Da die Forschung und Entwicklung von IDS ein junges Gebiet ist, existieren zur Zeit noch keine verabschiedeten und allgemein akzeptierten Standards oder Rahmenwerke für die Architektur von IDS. Deshalb sollen hier verschiedene Entwicklungen nur kurz erwähnt werden.
Das Common Intrusion Detection Framework (CIDF) der gleichnamigen Arbeitsgruppe unterscheidet vier Hauptkomponenten (vgl. Porras u. a. (1999)). *Event generators* entsprechen Sensoren und *event-analyzer* erfüllen die Funktion des *analyzers*. In einer *event-database* werden Ereignisse zur späteren Weiterverarbeitung abgelegt. Die Komponente *response unit* ist für alle Arten von passiver und aktiver Reaktion des IDS zuständig. Die Komponenten des CIDF tauschen Nachrichten im Format der Common Intrusion Specification Language (CISL) (vgl. Feiertag u. a. (1999)) aus.

Die Intrusion Detection Working Group (IDWG) innerhalb der Internet Engineering Task Force (IETF) entwickelt ebenfalls Standards, um eine Interaktion der verschiedenen Komponenten von Sicherheitsarchitekturen zu ermöglichen (vgl. IDWG (2002)). Diese Arbeiten befinden sich allerdings noch in einem sehr frühen Stadium.
Weitere Bemühungen zur Erreichung von Interoperabilität von Komponenten einer Sicherheitsarchitektur sind in (Allen u. a. 2000, S. 215) z.B. mit der Open Platform for Security (OPSEC) genannt.

Die implementierte Architektur eines IDS kann mit der funktionalen Struktur identisch sein. Größere Netze, eine höhere Menge von Eingangsdaten und die performante Analyse der Daten erfordern aber zunehmend die Verteilung funktionaler Komponenten. In (Bace und Mell 2001, S. 9 ff.) wird nur allgemein hinsichtlich der Plazierung des IDS im Bezug zum überwachten System nach *host-target co-location* und *host-target seperation* unterschieden. In (Sobirey 1999, S. 28ff) wird der Verteilungsgrad sogar als Klassifikationskriterium für IDS herangezogen. Da sich gleiche Funktionen eines IDS oft unterschiedlich realisieren und auf Komponenten verteilen lassen, ist der Verteilungsgrad kein geeignetes Klassifikationskriterium für IDS.

3.3. Taxonomie

Nachdem grundlegende Funktionsprinzipien und Komponenten von IDS vorgestellt wurden, soll nun eine Taxonomie für IDS abgeleitet werden und anschließend eine detailierte Erläuterung der einzelnen Klassifikationskriterien erfolgen.
Unter Klassifikation versteht Britannica (1997–1999) das systematische Einordnen von Begriffen in Gruppen oder Kategorien nach etablierten Kriterien. Als Taxonomie wird hingegen die Wissenschaft bezeichnet, die sich mit den generellen Prinzipien der wissenschaftlichen Klassifikation beschäftigt. Die vermutlich erste ausgearbeitete Taxonomie, die eine sinnvolle Klassifikation von IDS erlaubt, wurde im Jahre 1999 von Debar u. a. (1999) vorgestellt. Es wurden aber bereits vor dieser Arbeit ähnliche funktionale Klassifikationskriterien in Untersuchungen über IDS herangezogen (vgl. z.B. (Alessandri u. a. 2001, S. 10)).

Die ursprüngliche Taxonomie von Debar u. a. (1999) besteht aus vier Dimensionen mit jeweils zwei Ausprägungen (vgl. Abb. 3-2). Zu den funktionalen Charakteristiken zählen die Dimensionen *detection method*, *behavior on detection* und *audit source location*. In der Dimension *detection method* wird die Methodik zur Erkennung von *intrusions* in *behavior-based* und *konwledege-based* unterschieden. Die Reaktion auf eine erkannte Sicherheitsverletzung wird mit den Ausprägungen *passive* und *active* in der Dimension *behavior on detection* beschrieben. Die Dimension *audit source location* gibt mit ihren Ausprägungen *host log files* und *network packets* die Quelle der zu analysierenden Daten an.
Als nicht-funktionale Charakteristik für ein IDS wird die Dimension *usage frequency* mit den beiden Ausprägungen *continuous monitoring* und *periodic analysis* angesehen.

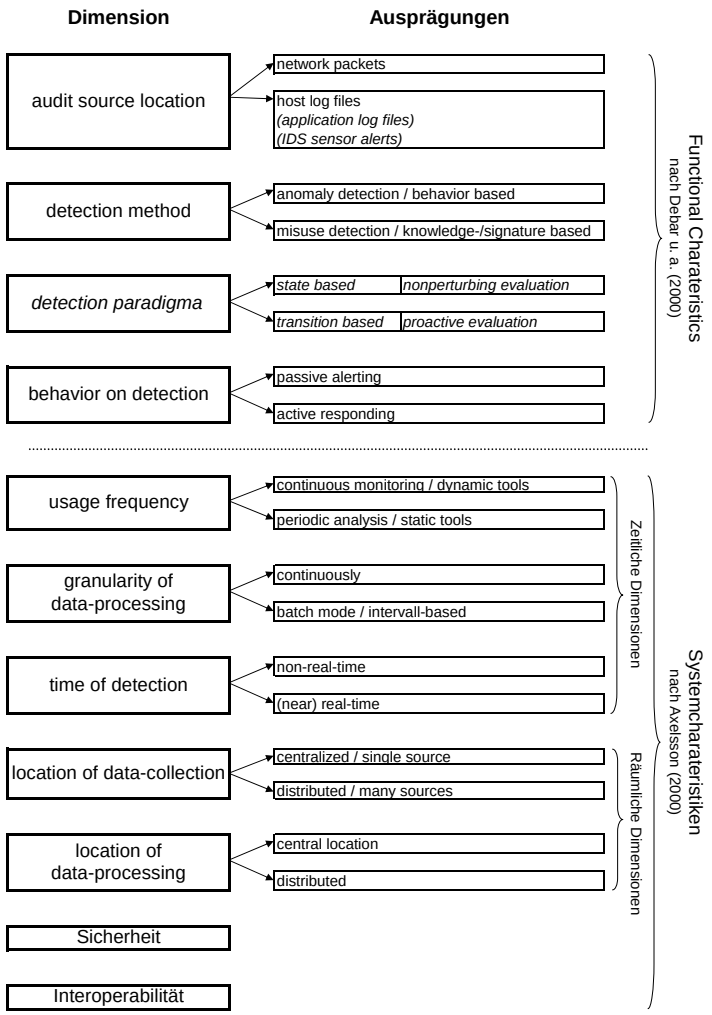


Abb. 3-2: IDS Taxonomie

In einer weiteren Arbeit von Debar u. a. (2000) wurde das ursprüngliche Klassifikationsschema um die Dimension *detection paradigma* mit den Ausprägungen *state-based* und *transition-based* sowie *nonperturbing evaluation* und *proactive evaluation* erweitert. Zudem wurden in der Dimension *audit source location* zusätzlich die Ausprägungen *application log files* und *IDS sensor alerts* eingeführt.

Durch Axelsson (2000) wurde die ursprüngliche Taxonomie von Debar u. a. (1999) ebenfalls aufgegriffen und modifiziert. Axelsson (2000) nimmt eine sehr tiefgehende Untergliederung der Dimension *detection method* vor. Demgegenüber unterscheidet er eine Gruppe von Dimensionen, die er als System-Charakteristiken bezeichnet (vgl. Abb. 3-2). Dazu gehören die Dimensionen *time of detection*, *granularity of data-processing*, *source of audit data*, *response to detected intrusions*, *location of data-processing*, *location of data-collection*, *security* und *degree of interoperability*. Bei einer anschließenden Klassifikation von verschiedenen IDS nach seiner aufgestellten Taxono-

mie stellt Axelsson (2000) fest, daß manche Systeme sich nicht unzweifelhaft klassifizieren lassen und in mehrere Kategorien eingeordnet werden können. Dies ergibt sich aus der Tatsache, daß in einem IDS oft mehrere Ausprägungen einer Dimension realisiert werden.
Ebenfalls ist festzustellen, daß Taxonomien für IDS oft unterschiedlich differenziert angewendet werden. Beispiele für weniger detailierte Klassifikationsschemata sind in Bace und Mell (2001), von Helden u. a. (1998) oder Allen u. a. (2000) zu finden.

Eine weitere Möglichkeit ist die Klassifikation von IDS nach Merkmalen der Architektur, wie etwa dem Verteilungsgrad (vgl. (Sobirey 1999, S. 26 f.)). Dieser Ansatz ist aber nicht geeignet IDS sinnvoll zu klassifizieren, da nicht auf funktionale Kriterien eingegangen wird.

In dieser Arbeit sollen im weiteren die Dimensionen *audit source location*, *detection method* mit *detection paradigma* und *behavior on detection* als bestimmende funktionale Merkmale von IDS detailiert diskutiert werden. Als zusätzliche Klassifikationsmerkmale sollen die von Axelsson (2000) so genannten System-Charakteristiken (*usage frequency*, *time of detection*, *granularity of data-processing*, *location of data-collection*, *location of data-processing*, *security* und *interoperability*) betrachtet werden.
In den folgenden Kapiteln werden Ausprägungen der jeweiligen Dimension beschrieben, spezifische Vor- und Nachteile diskutiert sowie Einsatzmöglichkeiten und eine Bewertung dargestellt.

3.4. Klassifikationskriterien

3.4.1. Audit Source Location

Ein charakteristisches Merkmal für ein IDS ist die Quelle der zu analysierenden Daten. Mit der Datenquelle steht die Art der Daten und ihre Eignung zur Erkennung von Sicherheitsverletzungen in engem Zusammenhang. Die Erhebung der Daten wird in einem IDS durch den Sensor realisiert. Die Funktionalität der Sensoren und ihre Plazierung wirken sich ebenfalls erheblich auf die Quantität und Qualität der erhobenen Daten aus (vgl. Kap. 3.4.4).

Nach Debar u. a. (1999) gibt es für die Dimension *audit source location* zwei wesentliche Datenquellen als Ausprägungen. Zum einen *host log files* und zum anderen *network packets*.
In der Ausprägung *host log files* werden die Daten aus *audit trails* des überwachten Hosts gewonnen. Unter *audit trail* ist die elektronische Aufzeichnung von (sicherheitsrelevanten) Aktivitäten eines Systems zu verstehen. Als geeignete Quellen für Audit-Daten führen (Debar u. a. 1999, S. 812 f.) die Syslog-Mechanismen und *C2 security audits* an. Syslog ist ein einfacher Audit-Dienst, der auf Unix-Derivaten zur Verfügung steht und von vielen Applikationen genutzt wird. Der Nachteil des Syslog-Dienstes besteht in geringer Sicherheit, mangelndem Schutz der Audit-Daten vor nachträglichen Manipulationen und wenig detailierten Audit-Daten. Im *C2 security audit* hingegen werden sicherheitsrelevante Ereignisse differenzierter (detailierte Nutzer- und Gruppen-Identitäten, Parameter von Systemaufrufen, ...) aufgezeichnet. Das *auditing subsystem* ist in

einem vertrauenswürdigen Bereich des Betriebssystems untergebracht (vgl. Kap. "Intrusion Detection and the Classic Security Model" aus Escamilla (1998)). Deswegen bestehen deutlich weniger Möglichkeiten zur Umgehung oder nachträglichen Manipulation von *audit trails*.
Da die ersten IDS auf Hosts aufgesetzt wurden, lag die Verwendung von Host-Logfiles in Form von Audit-Daten als Datenquelle nahe (vgl. u.a. Anderson (1980), Denning (1987)). Mit der Verwendung von leicht zugänglichen Host-Logfiles versuchte man lokale Angriffe zu erkennen. Angriffe über Netze spielten kaum eine Rolle, da sich (lokale) Netze noch nicht etabliert hatten. Weil sich die Datenquelle auf dem Host befindet, spricht man auch von host-basierten IDS.

Die zweite Ausprägung der Dimension *audit source location* ist *network packets*.
Darunter werden nach (Debar u. a. 1999, S. 811 f.) Daten verstanden, die ursprünglich von einem Netzwerkinterface (*packet capturing*) erhoben werden. Dies kann sowohl Netzwerkverkehr (*traffic*) für den lokalen Host als auch eines ganzen Teilnetzes sein. Um *traffic* ganzer Teilnetze auszuwerten werden typischerweise auf Ebene 2 des ISO/OSI-Referenzmodells (vgl. Tanenbaum (1998)) Ethernet-Frames einer Broadcast-Domain erfaßt (*network sniffing*). Dies läßt sich im ursprünglich als *shared medium* spezifizierten Ethernet leicht realisieren. Da Ethernet heute als Industriestandard für lokale Netze angesehen werden kann, verarbeitet die überwiegende Anzahl von IDS außer Ethernet-Frames keine weiteren Protokolle der Ebene 2. Auf Angriffsmerkmale werden dann meist Protokolle der Vermittlungsschicht (IP), der Transportschicht (TCP, UDP) und der Anwendungsschicht (SNMP, DNS, SMTP, IMAP, Telnet, FTP, HTTP, ...) des TCP/IP-Referenzmodells (vgl. Tanenbaum (1998)) untersucht. Andere Protokolle, wie etwa die SPX/IPX-Protokoll-Familie von Novell oder SNA-Protokolle in IBM-Hostumgebungen werden hingegen kaum von IDS als Datenquelle herangezogen. Wegen der Erhebung der Daten von Netzwerkkomponenten werden solche IDS als netzwerk-basierte IDS bezeichnet.

In ihrer überarbeiteten Taxonomie nehmen Debar u. a. (2000) zwei weitere Ausprägungen der Dimension *audit source location* auf, um eine genauere Klassifikation zu ermöglichen. Die Ausprägung *application log files* umfaßt alle Logfiles von Anwendungen wie etwa Datenbanken, Web-Servern usw. Mit *IDS sensor alerts* werden Nachrichten von anderen IDS-Sensoren oder weiteren Komponenten einer Sicherheitsinfrastruktur als Datenquelle genutzt. Damit wird der Tendenz zum Aufbau hierarchischer IDS mit verteilten Sensoren Rechnung getragen.
Die Detailierung der Dimension *audit source location* wird in dieser Arbeit nicht weiter untersucht. *IDS sensor alerts* können als Spezialfall von *application log files* aufgefasst werden. Logfiles von Applikationen sind wiederum *audit trails* in den aufgezeichneten Daten sehr ähnlich.
Damit sind die beiden Ausprägungen *host log files* und *network packets* wegen ihrer einfachen und trennscharfen Abgrenzbarkeit gut zur Klassifikation von IDS geeignet.

Im weiteren werden nun die spezifischen Vor- und Nachteile der beiden Ausprägungen gegenübergestellt (vgl. (Bace und Mell 2001, S. 15 ff.)). Netzwerk-basierte IDS bieten folgende Vorteile:

- Mit wenigen, strategisch günstig, plazierten Sensoren läßt sich ein großes Netzwerk überwachen. Strategisch günstige Positionen sind zentrale Stellen eines Netzwerkes (z.B. *core*

router) oder Netzübergänge (z.B. *gateway router*). In vielen Fällen werden IDS auch in einer Demilitarisierten Zone (DMZ) eingesetzt, um *traffic* von und zu besonders gefährdeten Systemen zu überwachen.

- Sensoren von netzwerk-basierten IDS besitzen einen passiven Charakter. Sie können praktisch ohne störende Auswirkungen auf den laufenden Netzbetrieb implementiert werden.
- Aus dem passiven Charakter der Sensoren ergibt sich außerdem, daß sie praktisch nicht von Angreifern erkannt werden können. Somit steigt die Wahrscheinlichkeit, den unwissenden Angreifer zu beobachten, den Angriff zu dokumentieren und rechtzeitig reagieren zu können.
- Die Sensoren werden meist als Software auf dedizierten Rechnern oder als Hardware-Appliance implementiert. Damit ergeben sich keine negativen Auswirkungen auf die Performanz von bereits vorhandenen und genutzten Systemen durch den Einsatz der Sensoren.
- Durch die dedizierte Realisierung der Sensoren ergibt sich weiterhin der Vorteil, diese gut gegen Angriffe auf Sensoren selbst absichern zu können.
- Analysiert ein IDS Netzwerk-Datenpakete, können insbesondere netzwerk-spezifische Angriffe erkannt werden. Außerdem erlaubt es die Analyse von *network traffic* Angriffe oft in einem sehr frühen Stadium zu erkennen, weil der überwiegende Teil der Angriffe heute über Netzwerke durchgeführt wird. Netzwerk-basierte IDS können deshalb viele Angriffe zeitlich früher als host-basierte IDS detektieren.

Mit der Nutzung von Netzwerk-Paketen als Datenquelle ergeben sich aber auch einige Nachteile:

- In den ersten lokalen Netzwerken (10 MBit/s Ethernet) war es wenig problematisch alle Netzwerk-Daten annähernd mit Leitungsgeschwindigkeit (*wire speed*) zu verarbeiten. Um bei den heute üblichen Übertragungsraten (100 MBit/s oder 1 GBit/s Ethernet) alle Netzwerk-Daten verarbeiten zu können sind erheblich höhere Anforderungen an die Hard- und Software des IDS zu stellen. Können Netzwerk-Daten dennoch den Sensor wegen zu hoher Auslastung passieren, werden unter Umständen Angriffe nicht erkannt.
- Durch den heute üblichen hohen Datendurchsatz ist es auch möglich, den Sensor durch eine DoS-Attacke anzugreifen. Wird das Netzwerk mit Paketen geflutet, kann der Sensor an die Grenzen seiner Verarbeitungsgeschwindigkeit gelangen und analysiert nicht mehr alle Pakete. Dies ermöglicht wiederum die unerkannte Durchführung von Angriffen.
- Eine weitere Problematik für netzwerk-basierte IDS stellt die *switching*-Technologie dar. Dabei kommunizieren Ethernet-Stationen über Switches bidirektional miteinander; Die Eigenschaft des Ethernet als *shared medium* wird damit aufgehoben. Der Sensor, als an der Kommunikation unbeteiligte Station, kann den entsprechenden *traffic* also nicht aufzeichnen und analysieren. Hier bieten sich allerdings Network *"test access ports"* (TAP) als Lösung an (vgl. Einwechter (2002)).

- Mit der verschlüsselten Übertragung von Nutzdaten auf dem Netzwerk (*encrypted traffic*) wird ebenfalls eine Analyse verhindert. Es kann nur der nicht verschlüsselte Teil (z.B. *protocol header*) ausgewertet werden. Auf der einen Seite kann durch Nutzung von Verschlüsselungstechniken die Kommunikation abgesichert werden (z.B. verschlüsselte Emails, VPNs mit IPSec). Auf der anderen Seite wird eine Analyse des *traffic* durch ein IDS erschwert bis unmöglich gemacht.

- Von den meisten IDS werden Ethernet-Frames aufgezeichnet und Protokolle des TCP/IP-Protokoll-Stapels ausgewertet. Werden in einem Netzwerk andere Protokolle genutzt, so können diese nicht vom IDS ausgewertet werden. Als Beispiele können hier die SNA-Protokolle von IBM in Großrechner-Umgebungen oder die SPX/IPX-Protokolle in Novell-Netzwerken genannt werden. Dieser Nachteil relativiert sich, da die Mehrzahl der Firmennetze der relevanten Anwendungen heute auf Ethernet- bzw. TCP/IP-Protokollen basieren.

- Ein weiterer Schwachpunkt von netzwerk-basierten IDS ist ihre Unfähigkeit den Erfolg eines Angriffes zu ermitteln. Da die meisten Angriffe sich gegen einen Host richten und das Netzwerk nur als Transportmedium gebrauchen, kann nur auf dem Host eindeutig der Erfolg bzw. Schaden eines Angriffs festgestellt werden.

- Fälle von internem Mißbrauch von Zugriffsrechten (*trusted insider misuse*) werden meist von netzwerk-basierten IDS ebenfalls nicht erkannt. Zum einen werden solche Angriffe oft lokal (d.h. mit physischem Zugang) auf einem Host ausgeführt, wobei keine Netzwerk-Pakete generiert werden. Zum anderen werden interne Netze oft nicht durch Sensoren überwacht. Deshalb können interne Angriffe vom IDS unbemerkt durchgeführt werden.

Host-basierte IDS bieten demgegenüber folgende Vorteile:

- Der Ziel-Host eines Angriffes wird von host-basierten Systemen direkt überwacht. Hierdurch können auch lokal gegen den Host ausgeführte Angriffe erkannt werden. Damit kann dem Problem des *trusted insider misuse* durch interne Angreifer (z.B. vom Typ *masquerader*, *misfeasor*) begegnet werden. Solche Angriffe lassen sich z.B. durch abweichende Verhaltensmuster erkennen.

- Im Gegensatz zu netzwerk-basierten IDS schränken *encrypted traffic* oder *switched networks* host-basierte IDS meist nicht in ihrer Funktionalität ein. Da die Nutzdaten auf dem Host vor bzw. nach dem Transport über das Netzwerk in unverschlüsselter Form vorliegen, ist eine Analyse durch das IDS möglich. Die eingesetzte Netzwerk-Technologie bzw. Topologie spielt ebenfalls keine Rolle, da die Analysedaten direkt auf dem Host erfaßt werden.

- Ein weiterer Vorteil für host-basierte IDS sind die oft sehr umfangreichen *security audit trails*. Diese bieten sehr viel detailliertere Informationen als reine Logfiles (z.B. Syslog) und erfassen vor allem sicherheitsrelevante Aktionen von Subjekten an Objekten (vgl. Denning (1987)). Beispiele hierfür sind das Solaris SunSHIELD Basic Security Module (BSM)

(vgl. Sun (2002)), die Audit-Funktionalität von AIX (vgl. IBM (2002)) oder der Microsoft Windows NT/2000 Produkte. Diese Produkte können *security audits* erstellen, die dem C2-Sicherheitsstandard (vgl. DOD (1985)) genügen und mit ihren detailierten Informationen eine ideale Datenquelle zur Erkennung von Sicherheitsverletzungen darstellen.

- Werden neben Betriebssystem-Audits auch Audits bzw. Logfiles von Applikationen (z.B. Datenbanken) als Datenquelle herangezogen, können auch Nutzern der Applikation unautorisierte Aktionen nachgewiesen werden. Dieser Nachweis ist auf Basis anderer Datenquellen kaum zu führen.

Dem Vorteil host-basierter IDS stehen auch einige Nachteile gegenüber:

- Bedingt durch ein Vielzahl möglicher Datenquellen und -formate (je Host, verschiedene Audit- und Logfile-Formate von Betriebssystemen und Applikationen) erhöht sich die Komplexität mit zunehmender Anzahl der zu überwachenden Systeme erheblich. Wegen dieser Vielfalt existieren nur für wenige Formate Sensoren bzw. Analyzer zur Auswertung.
- Da die Sensoren und Analyzer von host-basierten IDS oft auf dem zu überwachenden Host selbst ausgeführt werden, besteht die Gefahr der Deaktivierung des IDS. Bemerkt ein Angreifer ein IDS, wird er versuchen keine verwertbaren Spuren durch seinen Angriff zu hinterlassen. Anfällig für Manipulationen sind Logfiles (v.a. von Applikationen), die meist weniger gut als *security audits* gesichert sind. Eine weitere Möglichkeit ist die Deaktivierung des IDS mittels eines Angriffes (z.B. DoS) auf das IDS selbst.
- Host-basierte IDS benötigen Ressourcen in Form von Speicherkapazität und Rechenzeit auf dem überwachten Host. Dies muß bei der Konfiguration des Host berücksichtigt werden, um sowohl die erwartete Performanz der eigentlichen Anwendung als auch des IDS zu erreichen. Ebenso ist zu beachten, daß bei host-basierten IDS durch detailiertes Auditing schnell große Datenmengen anfallen können, die zu verarbeiten und zu speichern sind.
- Werden als Datenquelle Applikations-Logfiles herangezogen, erfolgt die Analyse der Daten auf einem relativ hohen Abstraktionsniveau. Angriffe auf darunterliegenden Ebenen (z.B. Systemebene) können dann nicht erkannt werden.

In der Praxis konzentriert sich die große Mehrzahl der IDS auf wenige Datenquellen (Ethernet-Frames bzw. TCP/IP-Protokolle bei netzwerk-basierten IDS und eine geringe Anzahl von Audit-Formate bei host-basierten IDS). Des weiteren ist ein Trend zu hybriden IDS festzustellen, die netzwerk-basierte Daten mit host-basierten Daten verknüpfen um ein vollständiges Bild der Systemumgebung zu erhalten und Angriffe damit effektiver erkennen zu können. Auf die dabei entstehende Problematik wird in Kapitel 4.2 eingegangen.
Beispiele für host-basierte IDS sind IDES/NIDES, für netzwerk-basierte IDS Cisco Secure Intrusion Detection System und für hybride Ansätze EMERALD (vgl. (Allen u. a. 2000, S. 18 ff.) und (von Helden u. a. 1998, S. 90 ff.)).

3.4.2. Detection Method

Die zweite bestimmende funktionale Charakteristik für ein IDS ist die Dimension *detection method*. Darunter wird der Ansatz zur Erkennung von *intrusions* verstanden. Diese Funktion eines IDS wird durch die Architektur-Komponente *analyzer* realisiert.
Im folgenden werden die beiden grundlegenden Ausprägungen dieser Dimension beschrieben. Für beide Ausprägungen können verschiedene Modelle und Realisierungsmöglichkeiten genutzt werden (vgl. (Axelsson 2000, S. 4 ff.) und (Alessandri u. a. 2001, S. 52 ff.)), die hier nicht vertiefend betrachtet werden.

In den Arbeiten von Debar u. a. (1999, 2000) und Axelsson (2000) werden die beiden grundlegenden Analysekonzepte *anomaly detection* (auch *behavior-based*) und *misuse detection* (auch *knowledge-based* oder *signature-based*) unterschieden.
Der Begriff *anomaly detection* wurde bereits durch Anderson verwendet, der von "characterization of computer use" (Anderson 1980, S. 17 ff.) spricht und *intrusions* durch anormale Verhaltensmuster in Audit-Daten erkennen will. Denning (1987) beschreibt ein statistisches Modell, das *intrusions* ebenfalls anhand von Anomalien in Audit-Daten erkennen soll. Sie stellt dabei besonders heraus, daß ein solches *anomaly detection system* die meisten Angriffe erkennen kann ohne spezielle Kenntnisse über die zugrundeliegenden Mechanismen des Angriffes zu besitzen. Ein *anomaly detection system* besitzt kein Wissen über bestimmte Schwachstellen oder Angriffe, sondern Wissen über das normale Verhalten des überwachten Systems. Die Parameter eines solchen normalen Verhaltensprofiles werden gesetzt oder vom IDS durch die Analyse von Datenproben "gelernt". Ein von den Verhaltensprofilen abweichendes Verhalten eines Nutzers oder einer Anwendung wird vom IDS dann als Angriff gewertet.
Die Analyse auf Anomalien kann durch verschiedene Techniken realisiert werden. In (Bace und Mell 2001, S. 19) werden beispielsweise *threshold detection*, *statistical measures* oder *rule-based measures* genannt. Weitere Techniken werden in (Debar u. a. 1999, S. 809 f.) sowie Halme und Bauer (2000) erläutert. Mögliche Parameter und Profile werden von (Anderson 1980, S. 17 ff) und Denning (1987) beschrieben.

Im Gegensatz zu *anomaly detection systems* benötigen *misuse detection systems* Informationen über charakteristische Merkmale eines Angriffes. Die Analyse-Daten werden zur Erkennung von *intrusions* nach kennzeichnenden Mustern (Signaturen) eines Angriffes durchsucht. Die Grundlage für ein *misuse detection system* bildet daher eine Datenbank von bekannten Angriffssignaturen. Paßt ein Muster auf die Analyse-Daten, wird ein Alarm ausgelöst. Dabei können detailierte Informationen, zu dem die Signatur erzeugenden Angriff, mitgeteilt werden.
Signatur-basierte Systeme können unter anderem durch Techniken des *pattern matching*, von Experten-Systemen oder neuronalen Netzen realisiert werden (vgl. (Debar u. a. 1999, S. 808 f.) sowie (Halme und Bauer 2000, Kap. 7.2)).

Im Kapitel 3.3 wurde darauf hingewiesen, daß die von Debar u. a. (2000) eingeführte Dimension *detection paradigma* in Verbindung mit der *detection method* behandelt wird. Nach dem *state-based* Paradigma wird versucht, bestimmte statische Zustände eines Systems als *intrusion* zu erkennen. Bei dem *transition-based* Ansatz werden Zustandsübergänge dahingehend untersucht, ob sie das System in einen kompromittierten Zustand versetzen. Angriffe können somit bereits in einer frühen Phase erkannt werden. Damit steigt die Wahrscheinlichkeit der erfolgreichen Abwehr des Angriffs und Schäden werden minimiert. (vgl. (Allen u. a. 2000, S. 21)).

Nach der Darstellung der Dimension *detection method* soll im folgenden auf Vor- und Nachteile der beiden Ausprägungen eingegangen werden (vgl. (Bace und Mell 2001, S. 18 ff.), (Allen u. a. 2000, S. 11 f.) und (Debar u. a. 1999, S. 808 ff.)).
Für *anomaly detection systems* lassen sich folgende Vorteile anführen:

- *Anomaly detection systems* sind unabhängig von einer Signatur-Datenbank, die in der Praxis nicht vollständig oder aktuell sein kann. Die Erkennung von *intrusions* anhand von anormalem Verhalten stellt damit einen wesentlichen Vorteil dar.
- Hierdurch sind *anomaly detection systems* in der Lage, Symptome eines Angriffs ohne genaues Wissen über den Angriff selbst zu erkennen. Damit können auch neue und unbekannte Angriffe erkannt werden. Ebenso können Angriffe unabhängig vom verwendeten Angriffswerkzeug (z.B. *exploit*, Trojanisches Pferd, Viren, *malicous code*) erkannt werden.
- Werden neue, unbekannte Angriffe identifiziert, kann ein *anomaly detection system* Informationen über den Angriff liefern. Mit Hilfe dieser Informationen lassen sich neuartige Angriffe analysieren und Angriffssignaturen erstellen.
- Weiterhin können *anomaly detection systems* den Mißbrauch von Privilegien (Angreifer vom Typ *masquerader* oder *misfeasor*) feststellen, der sich in Abweichungen vom normalen Verhaltensprofil zeigt. Damit können sie effektiv gegen *insider misuse* eingesetzt werden.

Den aufgeführten Vorteilen von *anomaly detection systems* stehen aber auch einige Nachteile gegenüber:

- Die Konfiguration von *anomaly detection systems* ist komplex, da das erwartet "normale" Verhalten ermittelt und mit geeigneten Parametern beschrieben werden muß. Es ist ein hoher Aufwand notwendig, um aus vorliegenden Daten in einer Lernphase ein Profil des "normalen" Verhaltens zu generieren. Die Trainingsdaten müssen sich über einen ausreichend langen Zeitraum erstrecken und alle gewöhnlichen Aktionen wiederspiegeln. Dies erfordert eine aufwendige und sorgfältige Erfassung der Trainingsdaten.
- Werden die Profile des normalen Verhaltens nicht sorgfältig genug erstellt oder ändert sich das Nutzerverhalten, so lösen *anomaly detection systems* eine hohe Zahl von falsch positiven Alarmen aus. Insbesondere Änderungen im Verhalten (z.B. neue Nutzer, verändertes

Verhalten von Nutzern durch neue Aufgabenbereich, neue Applikationen) erfordern eine ständige Anpassung der Profile, um die Anzahl der falsch positiven Alarme gering zu halten.

- Ein weiterer Nachteil ist die fehlende bzw. ungenaue Diagnose zu ausgelösten Alarmen. Das IDS kann nur ein "anormales" Verhalten und das auslösende Ereignis erkennen und melden. Die eigentliche Ursache für das anormale Verhalten kann ein *anomaly detection system* nicht ermitteln. Dies ist Aufgabe eines Mitarbeiters mit entsprechendem Fachwissen.
- IDS mit dem Ansatz der *anomaly detection* produzieren meist umfangreiche Berichtsdaten. Dies ist durch die höhere Zahl von falschen Alarmen bedingt und verhindert oft eine sorgfältige Auswertung der gemeldeten Angriffe durch entsprechendes Fachpersonal.

Betrachtet man dagegen *misuse detection systems*, so lassen sich folgende spezifische Vorteile anführen:

- Die in *misuse detection systems* heute verwendeten Techniken zur Erkennung von Angriffen lassen sich gut in einfache Modelle fassen. Durch effiziente Implementierung von beherrschten Technologien (z.B. *pattern matching*) stellen signatur-basierte IDS in der Regel geringere Anforderungen an Ressourcen bzw. sind erheblich performanter als *anomaly detection systems*. Diese implementieren oft komplexe Modelle mit entsprechend höherem Ressourcenbedarf.
- *Misuse detection systems* sind im Gegensatz zu *anomaly detection systems* einfacher zu konfigurieren. Es ist kein komplexes und aufwendiges Erstellen von Verhaltentensprofilen notwendig, sondern lediglich das Einspielen und Anpassen der gewünschten Angriffssignaturen.
- Bekannte und durch Signaturen beschriebene Angriffe können durch *misuse detection systems* effektiv erkannt werden. Wurde das IDS mit einem angepaßten Satz an Signaturen ausgestattet, wird nur eine geringe Zahl von falsch-positiven Alarmen generiert. Diese können mit vertretbarem personellen Aufwand analysiert werden und entsprechende Maßnahmen oder Anpassungen der Signaturen vorgenommen werden.
- Da eine Signatur in der Regel genau einen Angriff oder eine eng abgegrenzte Gruppe von Angriffen identifiziert, kann dies bei einem Alarm zu einer besseren Diagnose genutzt werden. Zu der Signatur können detailierte Informationen über den Angriff, Hinweise auf Sofortmaßnahmen und Möglichkeiten zur Schließung der ausgenutzten Sicherheitslücke hinterlegt werden. Durch den höheren Informationsgrad des Personals wird die Entscheidungsfindung beschleunigt und *intrusions* kann zielgerichtet begegnet werden.

Neben diesen Vorteilen bringen *misuse detection systems* aber auch einige Nachteile mit sich:

- Für signatur-basierte IDS sind in der Regel keine auf die spezifischen Gegebenheiten angepaßten Signatur-Datenbanken verfügbar. Es ist nicht sinnvoll, Signaturen zur Analyse heranzuziehen, obwohl die damit beschriebenen Angriffe in der überwachten Systemumgebung wirkungslos sind. Beispielsweise macht es keinen Sinn eine reine Microsoft Windows Umgebung auf UNIX-spezifische *exploits* zu analysieren. Bei der Verwendung nicht benötigter Signaturen steigt lediglich die Zahl der unnötig ausgelösten Alarme, die wiederum Ressourcen zur Bearbeitung binden.
- Ein Problem für *misuse detection systems* ist die prinzipielle Unvollständigkeit und der hohe Pflegeaufwand der Signatur-Datenbank. Die Unvollständigkeit ergibt sich aus der Tatsache, daß es immer Angriffe geben wird, die noch nicht (öffentlich) bekannt oder beschrieben sind. Schon leicht abgewandelte Angriffe werden unter Umständen durch vorhandene Signaturen nicht erkannt.
- Um den Stand der Signatur-Datenbank aktuell zu halten, müssen Signaturen zu neuen Angriffen erstellt und verfügbar gemacht werden. Die Prozedur von der Analyse eines Angriffs bis zur Erstellung einer charakteristischen Signatur ist aber zeitaufwendig und erfordert erhebliches Wissen. Aus diesem Grund werden bekannte *vulnerabilities*, *exploits* und auch Signaturen von verschiedenen Organisationen erstellt, ausgetauscht und veröffentlicht (z.B. von CERT/CC (2002a); Whitehats (2001)).
- Weiterhin ist es für *misuse detection systems* nahezu unmöglich *insider misuse* zu erkennen. Für diesen Fall von Sicherheitsverletzungen lassen sich nämlich in den heutigen IDS keine charakteristischen Signaturen erstellen.

Betrachtet man die verschiedenen Vor- und Nachteile von *anomaly detection systems* und *misuse detection systems*, kann man für beide Ausprägungen geeignete Einsatzgebiete finden.
Die heutigen kommerziell verfügbaren IDS sind *misuse detection systems* (vgl. (Alessandri u. a. 2001, S. 11) und (Bace und Mell 2001, S. 51)). Signatur-basierte IDS werden als netzwerk-basierte IDS mit einer großen Zahl von Signaturen auf dem Markt angeboten. Die einfachere Realisierung, Konfiguration und eine geringere Anzahl von falsch-positiven Alarmen bei richtiger Konfiguration sprechen für diese IDS.
Anomaly detection systems eignen sich wegen der Komplexität der Konfiguration und Erstellung der Profile eher für Umgebungen, in denen genau definierte Aktivitäten über einen längeren Zeitraum unverändert ablaufen. Typischerweise trifft dies eher auf Mainframe- oder Minicomputer-Umgebungen (z.B. Kundenverwaltung einer Versicherung auf einem Mainframe) zu. Sind host-basierte *anomaly detection systems* einmal mit dem normalen Nutzerprofil des Hosts bekannt gemacht, generieren sie eine akzeptable Rate von falsch-positiven Alarmen. In heterogenen und dynamischen Umgebungen (z.B. Intranet mit ständig wechselnden Applikationen und Nutzern) ist es praktisch unmöglich ein dauerhaft "normales" Verhaltensprofil zu erstellen. Die daraus

resultierende hohe Zahl von falsch-positiven Alarmen verhindert einen sinnvollen Einsatz von *anomaly detection systems* in solchen dynamischen Umgebungen.
Es bleibt zu bemerken, daß beide Ausprägungen durch nicht sorgfältig durchgeführte und immer wieder angepaßte Konfiguration eine hohe Zahl von falsch-positiven als auch falsch-negativen Alarmen generieren.
Als effektive Lösung ist je nach Systemumgebung auch eine Kombination beider *detection methods* denkbar. Unbekannte Angriffe werden dabei durch das *anomaly detection system* erkannt und dokumentiert. Aus den bereitgestellten Daten können für das *misuse detection system* Signaturen erstellt werden.

Eine Übersicht von IDS mit Hinweisen zur *detection method* kann in den Arbeiten von Allen u. a. (2000), von Helden u. a. (1998) und detailiert auch in Axelsson (2000) gewonnen werden.

3.4.3. Behavior on Detection

Nach der verwendeten Taxonomie ist die dritte kennzeichnende Dimension für ein IDS die Art des Verhaltens bei einem erkannten Angriff (*behavior on detection*). Diese Funktion wird durch die Komponenten *management console* realisiert.

Für die Dimension *behavior on detection* lassen sich nach (Debar u. a. 1999, S. 811) und (Bace und Mell 2001, S. 20 ff.) zwei grundlegende Ausprägungen unterscheiden.
Wird durch das IDS bei einem erkannten Angriff ein Alarm ausgelöst, aber darüber hinaus keine weiteren (Gegen-)Maßnahmen getroffen, so spricht man von *passive alerting*. Das IDS informiert lediglich entsprechendes Personal über einen Angriff. Die Entscheidung über weitere Maßnahmen sind dann durch das Fachpersonal zu treffen. Die Benachrichtigung über eine *intrusion* kann auf verschiedene Art und Weise erfolgen: Durch Meldungen in einer Management-Konsole, per SNMP-Traps, Email oder SMS. Es muß sichergestellt sein, daß die Informationen die zuständigen Entscheidungsträger zuverlässig und ohne Zeitverzug erreichen.
In der Ausprägung *active responding* hingegen werden durch das IDS, neben der Generierung von Alarm-Meldungen, weitere Maßnahmen automatisch eingeleitet. Diese werden von (Bace und Mell 2001, S. 21) in drei Kategorien eingeteilt.
Die erste Maßnahme besteht darin, zusätzliche Informationen über den Angriff und den Angreifer zu sammeln. Damit werden evtl. wichtige Fakten als Beweise gesichert, die bei Nachforschungen bis hin zu strafrechtlichen Verfolgungen hilfreich sein können. Solche Informationen lassen sich beispielsweise durch ein ausführlicheres Auditing oder durch Logging des kompletten Netzwerkverkehrs gewinnen.
Als zweite Maßnahme kann die Systemumgebung des überwachten Systems verändert werden. Das IDS kann proaktiv oder korrektiv tätig werden, um Angriffe in Vorbereitung zu verhindern, laufende Angriffe zu unterbrechen und somit Schäden zu minimieren. Möglichkeiten dafür sind beispielsweise die Terminierung von Verbindungen und Umkonfiguration von Routern oder Paketfiltern um IP-Adressen bzw. Ports zu blockieren.

In einer dritten Stufe werden aktiv Gegenmaßnahmen gegen den Angreifer eingeleitet. Darunter fällt die aktive Gewinnung von Informationen über den Angreifer (z.B. durch *port scans*) oder die Einleitung eines Gegenangriffs. Diese Maßnahmen sind aber kritisch zu bewerten (vgl. Nachteile von *active responding* im folgenden).

Bei der Betrachtung der Ausprägung *passive alerting* lassen sich keine spezifischen Vorteile feststellen. *Passive alerting* ist die grundlegendste Reaktionsmöglichkeit und wird von allen IDS unterstützt. Es bringt aber einige Nachteile mit sich:

- Es besteht die Gefahr, daß Maßnahmen zur Minimierung von Schäden nicht rechtzeitig getroffen werden. Dies kann beispielsweise der Fall sein, wenn ein Techniker nicht erreichbar ist. Solche Probleme lassen sich allerdings durch organisatorische Regelungen mit entsprechendem Aufwand lösen.
- Die durch das IDS im Falle eines Alarmes übermittelten Informationen sind oft knapp gehalten. Damit ist es kaum möglich, qualifizierte Entscheidungen zu treffen. Bis genauere Informationen eingeholt sind, vergeht wertvolle Zeit, ohne daß geeignete Maßnahmen getroffen werden konnten.
- Ebenso ist es wegen fehlender Informationen oft nicht möglich, das Ausmaß des Schadens zu bestimmen, um adäquat reagieren zu können.

Einige dieser Nachteile werden bei *active responding* IDS durch folgende Vorteile kompensiert:

- Eine automatische Reaktion erfolgt in der Regel zeitnaher als eine manuelle Reaktion. Damit wird die Chance größer, Schäden zu minimieren oder ganz zu verhindern.
- Mit der Erhebung von Daten zur genauen Analyse des Vorfalls wird die Diagnose und Schadenfeststellung erheblich erleichtert. Zudem sind diese Daten Grundlage zur Erstellung neuer Signaturen und für Berichte über Sicherheitsvorfälle (*incident reports*).

Die aktive Reaktion auf erkannte Angriffe birgt aber auch Nachteile in sich:

- Automatisch eingeleitete Reaktionsmechanismen bergen die Gefahr, daß nicht adäquat reagiert wird. Die Reaktion kann unangemessen schwach (Schaden durch Angriff wird nicht verhindert) oder unangemessen hoch sein (Schaden durch Reaktion ist höher als Schaden durch eigentlichen Angriff). Reagiert ein IDS auf Angriffe beispielsweise mit der Umkonfiguration von Paketfiltern und Routern, kann dies zu einem *denial of service* von Netzen oder Anwendungen führen. Eine Abwägung der Auswirkungen eines Angriffes gegenüber möglichen Maßnahmen kann zur Zeit (noch) nicht von IDS geleistet werden. Hierfür wird ausgebildetes Personal mit technischem Know-how benötigt.

- Wird auf einen Angriff mit aktiven Gegenmaßnahmen reagiert, legt man die Existenz eines IDS offen. Dies kann zwar abschreckend wirken, birgt aber auch Gefahren. Bei einem folgenden Angriff des gleichen Angreifers wird dieser versuchen das IDS zu umgehen bzw. zu deaktivieren. Zudem kann ein IDS von Angreifern als zu überwindende Herausforderung angesehen werden und somit weitere Angriffe provozieren.
- Die Einleitung von aktiven Gegenmaßnahmen ist auch aus rechtlicher Sicht sehr problematisch. Ein Gegenangriff ist keinesfalls durch bestehende Gesetze gedeckt, kann einen Angreifer provozieren und trifft in der Regel nicht den wahren Angreifer (*IP spoofing*). Möglicherweise werden so an unbeteiligten Systemen Schäden angerichtet, für die der Betreiber des IDS haftbar ist.

Alle kommerziell verfügbaren Systeme bieten heute eine ganze Reihe von Mechanismen im Rahmen des *passive alerting*. Der Trend entwickelt sich aber immer mehr zu IDS, die auch aktiv reagieren. Zwar werden nicht aktiv Gegenangriffe ausgeführt, aber zumindest wird versucht, alle Informationen zu sammeln und die Systemumgebung zu verändern. Dies stellt als erste Reaktion sicherlich eine gute Maßnahme dar. Allerdings muß auch nach einer automatischen Reaktion möglichst rasch ein Fachman hinzugezogen werden. Dieser kann dann Entscheidungen über weitere Maßnahmen treffen. Mit diesem Vorgehen können Schäden durch zu langsame/schwache als auch durch zu starke Reaktionen minimiert werden.

3.4.4. Weitere Klassifikationskriterien

Die funktionalen Klassifikationsmerkmale von IDS wurden in den Kapiteln 3.4.1 bis 3.4.3 ausführlich dargestellt. In diesem Kapitel soll auf weitere Merkmale von IDS eingegangen werden, die Axelsson (2000) unter dem Begriff "Systemcharakteristiken" zusammengefaßt hat.

Zeitliche Dimensionen

In den zeitlichen Dimensionen sind Merkmale mit Zeitbezug zusammengefaßt, die zur Klassifikation eines IDS herangezogen werden können. Nach (Axelsson 2000, S. 10) sind dies *usage frequency*, *granularity of data-processing* und *time of detection*.

Die Dimension *usage frequency* trifft eine Aussage über das Nutzungsintervall eines IDS (vgl. (Debar u. a. 1999, S. 816)). IDS im engeren Sinne (*dynamic tools*) fallen in die Kategorie *continuous monitoring*. Sie ermöglichen eine kontinuierliche Überwachung der Zielsysteme. Dabei werden natürlich entsprechende Ressourcen beansprucht.
Im Gegensatz dazu steht die Ausprägung *periodic analysis*, zu der auch IDS im weiteren Sinne (*static tools*) gehören. Sie analysieren den Momentzustand eines Systems z.B. auf Konfigurationsfehler oder andere Schwachstellen (vgl. *security scanner* in Kap. 4.1.1). Diese Werkzeuge sind gut geeignet, präventiv Sicherheitslücken zu erkennen, zu schließen oder nachträglich Spuren eines Angriffs zu sichern. Problematisch sind dabei relativ lange Laufzeiten von komplexen

Überprüfungen (*security scan*). Außerdem können erkannte Sicherheitslücken unter Umständen nicht behoben werden weil z.B. keine Patches verfügbar sind.

Die Dimension *granularity of data-processing* unterscheidet in welchen Zeitabständen die erhobenen Daten analysiert werden. Es existieren die beiden Ausprägungen *continuously* und *batch mode / intervall-based*. Während in frühen IDS die Analyse wegen fehlender Ressourcen vielfach im *batch mode* erfolgen mußte, wird in praktisch allen aktuellen IDS die Analyse der Daten kontinuierlich betrieben. Dies ist die Voraussetzung, um eine möglichst kurze *time of detection* zu erzielen.

Die Dimension *time of detection* gibt an, wie groß der Zeitraum zwischen einer auftretenden *intrusion* und deren Erkennung ist. Das Ziel für IDS besteht darin, die Erkennung *(near) real-time* anstatt *non-real-time* zu leisten. Je früher ein Alarm ausgelöst werden kann, desto größer sind die Chancen durch planvolle Maßnahmen den Schaden zu minimieren. Der Begriff *real-time* muß kritisch betrachtet werden, da es systembedingt immer eine Verzögerung zwischen einem Angriff und dessen Erkennung gibt. Erst wenn nachweisbare Spuren eines Angriffs von Sensoren erfaßt und von Analyzern ausgewertet wurden, wird günstigenfalls ein Alarm generiert. Die Berücksichtigung dieser unvermeidbaren Zeitspanne wird durch das Adjektiv *near* vor *real-time* zum Ausdruck gebracht.

Heutige kommerzielle IDS überwachen überwiegend kontinuierlich, analysieren die Daten *continuously* und versuchen die Erkennung in *(near) real-time* zu leisten. Diese Ausprägungen sind vor allem für netzwerk-basierte IDS unumgänglich.

Räumliche Dimensionen

Unter räumliche Dimensionen fallen die Charakteristiken *location of data-collection* und *location of data-processing*.

Die Dimension *location of data-collection* gibt an, an welchem Ort die Analysedaten durch Sensoren erfaßt werden. Werden Daten nur durch einen Sensor zentral erfaßt, spricht man von *centralized* (*single source*) *location of data-collection*. Werden die Daten hingegen durch mehrere Sensoren an unterschiedlichen Orten erfaßt, nennt man diese Ausprägung *distributed* (*many sources*). Die Plazierung der Sensoren hat entscheidenden Einfluß auf die Quantität und Qualität der erhobenen Daten. Die Sensoren müssen so eingesetzt werden, daß sie alle relevanten Daten erfassen können. Dabei sollte die Erfassung von nicht benötigten Daten vermieden werden um die Performanz des IDS zu steigern und falsch-positive Alarme zu minimieren. Im Falle von host-basierten IDS sollten Sensoren auf allen kritischen Hosts installiert werden. Im Fall von netzwerk-basierten IDS gibt es verschiedene Möglichkeiten die Sensoren zu plazieren (vgl. (Bace und Mell 2001, S. 36 f.) und (Northcutt 1999, S. 41 ff.)).

Die Plazierung eines Sensors "hinter" einer äußeren Firewall[2] in einer DMZ (Location 1 in Abb. 3-3) bietet zwei Vorteile. Zum ersten sind nur relevante, und damit meist deutlich weniger, Daten

[2] Dies werden in der Regel Paketfilter sein. Es sind aber auch Application Level Firewalls, Proxies u.a. denkbar.

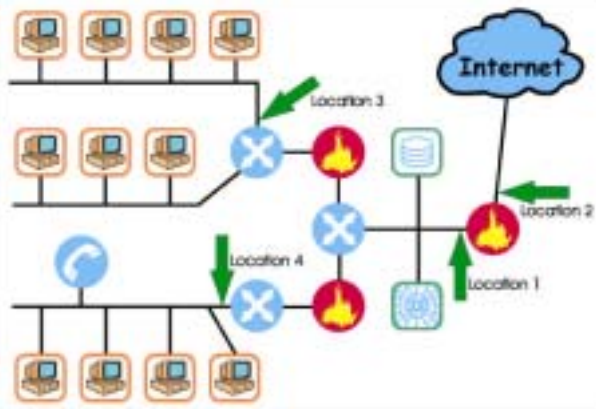


Abb. 3-3: "Locations of Network-based IDS sensors" aus Bace und Mell (2001)

zu analysieren. Dies wird durch die Filterfunktion der Firewall erreicht. Damit steigt die Performanz der Analyse. Zudem kann eine geringere Datenmenge kritischer analysiert werden, ohne die Anzahl der falsch-positiven Alarme erheblich zu erhöhen. Einfache Angriffe mittels bekannter *exploits* können beispielsweise bereits durch die Firewall verhindert werden. Zum zweiten können Fehler in der *security policy* und deren Umsetzung nachgewiesen werden, wenn nicht erwünschter oder erlaubter *traffic* durch den Sensor erfaßt wird.
Aber auch die Plazierung des Sensors "vor" der äußeren Firewall (Location 2 in Abb. 3-3) bietet einen Vorteil. Das IDS kann in diesem Fall alle Angriffsversuche aus dem Internet gegen das eigene Netz erfassen. Damit kann die Gefahr für das eigene Netzwerk realistischer dokumentiert werden. Die höhere und heterogene Menge an *traffic* lassen aber eine verminderte Performanz und eine höhere Anzahl von Fehlalarmen erwarten.
Weitere Lokationen für Sensoren sind *backbones* oder kritische Teilnetze (vgl. Location 3 und 4 in Abb. 3-3). An diesen Stellen können große Mengen *traffic* mit einem Sensor beobachtet werden. Das erhöht die Wahrscheinlichkeit, einen Angriff zu entdecken. Außerdem kann *insider misuse* oder Verhalten entgegen vorgegebener Sicherheitsrichtlinien festgestellt werden.

Mit der Dimension *location of data-processing* wird beschrieben, an welchem Ort die Analyse der erfaßten Daten stattfindet.
In der Ausprägung *central location* werden die Daten von den Sensoren an einen zentralen Analyzer übermittelt und dort ausgewertet. Dies ermöglicht eine Verknüpfung von Daten aus verschiedenen Quellen. Damit ergibt sich ein vollständigeres Bild der Systemumgebung und es lassen sich ansonsten unerkannte Angriffe nachweisen (vgl. Bass (2000)). Von Nachteil ist, daß sich die zu analysierende Datenmenge auf dem Analyzer rasch summiert und dieser mit entsprechenden Ressourcen ausgestattet sein muß, um eine performante Analyse zu ermöglichen. Außerdem müssen die Daten zum Analyzer übertragen werden. Dies beansprucht Bandbreite im Netzwerk und macht einen Schutz der Analysedaten vor Manipulationen notwendig. Weiterhin ergeben sich Probleme bei der Korrelation der Daten, wenn von Sensoren unterschiedliche Datenformate geliefert werden.

Dagegen werden in der Ausprägung *distributed* die Daten dezentral ausgewertet. Werden Sensor und *analyzer* gemeinsam in einer Komponente implementiert, sind entsprechende Ressourcen notwendig, um keine Einbrüche der Performanz zu erleiden. Es wird kaum Bandbreite beansprucht, da nur Statusmeldungen und Alarme zu einer zentralen Konsole übertragen werden. Erfolgt die Analyse dagegen verteilt auf mehreren *analyzers*, die nicht zwingend zusammen mit dem Sensor implementiert sein müssen, kann eine hohe Performanz erreicht werden. Allerdings wird Bandbreite zur Übertragung der Daten an den *analyzer* beansprucht und es müssen Vorkehrungen zum Schutz der Analysedaten getroffen werden. Bei einer verteilten Analyse der Daten kann eine Verknüpfung der einzelnen (Teil-)Ereignissen nur noch über Meldungen der *analyzer* in einer zentralen Management-Konsole durchgeführt werden. Da bereits eine Reduzierung der Datenmenge erfolgt ist, ist eine Verknüpfung wegen des Informationsverlustes schwieriger als bei einer zentralen Analyse der Daten.

In der Praxis implementierte IDS verfügen heute meist über mehrere und auch unterschiedliche Typen von Sensoren. Die Analyse der Daten erfolgt in der Regel dezentral und es werden nur Meldungen der Sensor/Analyzer-Komponente an eine zentrale Konsole übermittelt (vgl. z.B. ISS (2002) oder SRI (2002)).

Sicherheit und Interoperabilität

Der Aspekt der Sicherheit von IDS selbst ist erst seit kurzer Zeit Gegenstand der Forschung (vgl. (Axelsson 2000, S. 11)). Dabei ist die Widerstandsfähigkeit eines IDS im Falle eines Angriffs wichtig, um die Funktionsfähigkeit zu garantieren. Von Axelsson (2000) werden alle in seiner Arbeit aufgeführten IDS als wenig sicher eingestuft. Aktuelle Forschung beschäftigt sich beispielsweise mit DoS-resistenten IDS (vgl. Mell u. a. (2000)).

Als letzte Systemcharakteristik muß die Interoperabilität mit anderen Komponenten einer Sicherheitsarchitektur genannt werden. Dies könnte beispielsweise den Austausch von Audit-Daten und Ereignismeldungen zwischen Firewalls, Routern und einem IDS betreffen. Weiterhin ist die Verarbeitung von Ereignismeldungen verschiedener Sensoren in einer Management-Konsole zur Verknüpfung von Ereignissen denkbar. Bisher sind aber Ansätze zu einer solch engen Integration von IDS in die Sicherheitsarchitektur nur mit Komponenten eines Herstellers möglich (vgl. z.B. Cisco (2000)). Eine herstellerübergreifende Interoperabilität ist bisher kaum vorhanden, da offene Frameworks (wie z.B. Porras u. a. (1999), IDWG (2002)) bis heute nicht die notwendige Akzeptanz gefunden haben. Mit OPSEC (vgl. OPSEC (2002)) existiert mittlerweile ein offenes, herstellerunabhängiges *security framework*, welches auch IDS integriert. Es ist zu erwarten, daß in Zukunft ein höherer Interoperabilitätsgrad von IDS mit weiteren Komponenten einer Sicherheitsarchitektur erreicht wird.

4. Einsatzmöglichkeiten und Grenzen von IDS

4.1. Einsatz von IDS im Rahmen eines Sicherheitskonzeptes

In diesem Kapitel sollen IDS im Kontext einer ausformulierten Sicherheitspolitik betrachtet werden. Dazu wird zuerst eine Klassifizierung von Maßnahmen zur Vorbeugung und Abwehr von *intrusions* vorgestellt und IDS darin eingeordnet. Anschließend werden Gründe für den Einsatz von IDS als Komponente eines Sicherheitskonzeptes dargestellt.

4.1.1. Anti-Intrusion Taxonomie

Im Kapitel 2.1 wurde skizziert, wie durch die Entwicklung eines Sicherheitskonzeptes und der Implementierung von Sicherheitsfunktionen ein hoher Grad an IT-Sicherheit erreicht werden soll. Dabei werden die zu schützenden Systeme in der Regel durch abgestufte Sicherheitsfunktionen geschützt. Die Strategie mehrerer abgestufter Verteidigungslinien wird auch als "*defense in depth strategy*" bezeichnet (vgl. McHugh u. a. (2001)).
In ihrer Arbeit "AINT Misbehaving: A Taxonomy of Anti-Intrusion Techniques" stellen Halme und Bauer (2000) eine Klassifikationsmöglichkeit für *anti-intrusion* Techniken vor. Diese wird auch von (Axelsson 1998, S. 2) aufgegriffen und zur Einordnung von IDS im engeren Sinne in das breite Spektrum von Maßnahmen zur Verhinderung und Abwehr von *intrusions* genutzt.

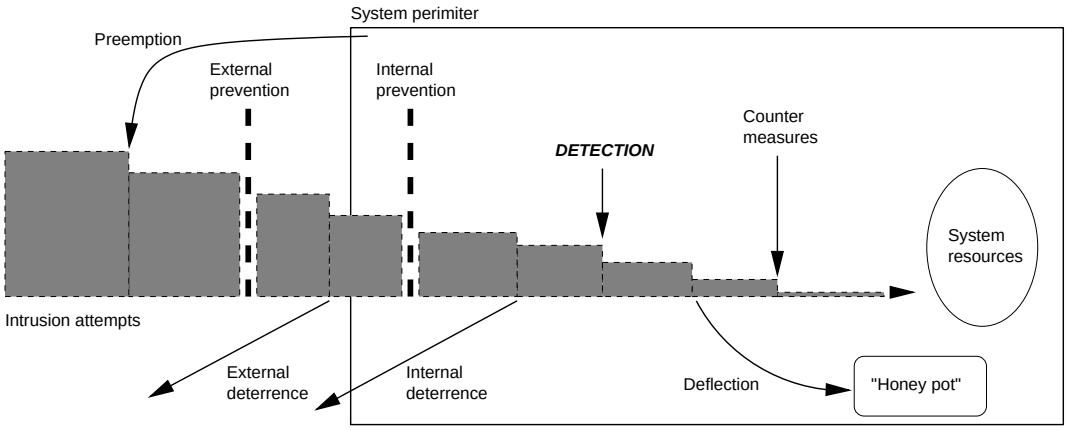


Abb. 4-1: "Anti-Intrusion Approaches" aus Axelsson (1998)

In der Abb. 4-1 werden sechs allgemein und sich nicht ausschließende Ansätze zur Abwehr von *intrusions* dargestellt. Die Verminderung der Zahl erfolgreicher Angriffe wird durch von links nach rechts schmaler werdende graue Säulen symbolisiert. Die Kategorien sind im Einzelnen:

1. **Prävention** (*prevention*)
   Unter Prävention werden geeignete Maßnahmen verstanden, die ein Eindringen im Vorhinein ausschließen oder möglichst unwahrscheinlich machen. Dazu zählt das sicherheitsbewußte Design von Systemen, der Einsatz von Paketfiltern und Firewalls sowie alle Arten von Tools zur Überprüfung der Sicherheit. Periodisch oder kontinuierlich können etwa Dateisystem oder Emails auf Viren und *malicous code* überprüft werden. Es existieren Tools um Konfigurationen auf Verwundbarkeit zu prüfen (vgl. z.B. Deraison (2000): Nessus *remote security scanner* oder Farmer und Spafford (2002): COPS für lokale Konfigurationen) oder um die Integrität von Dateien (vgl. Tripwire (2002): Tripwire) zu sichern.

2. **aktive Prävention** (*preemption*)
   Mit *Preemption* meinen Halme und Bauer (2000) aktive Maßnahmen zur Verhinderung eines Angriffs. Darunter fallen etwa der Aufbau von Sicherheitsbewußtsein bei Nutzern, die Beschaffung von Informationen über Sicherheitslücken oder aktuelle Angriffe sowie demonstrative Aufmerksamkeit für potentielle Angriffe.

3. **Abschreckung** (*deterrence*)
   Abschreckende Maßnahmen sollen einen Angreifer davon überzeugen, den Angriff abzubrechen oder gar nicht erst durchzuführen. Dazu wird im Verhältnis zu dem möglichen Erfolg ein erhebliches Risikopotential für den Angreifer aufgebaut. Solche Maßnahmen sind beispielsweise offensichtliche Warnungen vor Mißbrauch von Systemen mit Hinweisen auf die konsequente Verfolgung. Ebenfalls abschreckend wirken entsprechende Gesetze, die Verletzungen der IT-Sicherheit verbieten und strikt durchgesetzt werden. Durch Verbergen von wichtigen Systemen bzw. das Verschweigen und Tarnen von wichtigen Daten (Camouflage) kann ebenfalls versucht werden, möglichst kein Interesse bei potentiellen Angreifern zu wecken. All diese Methoden sind aber dann relativ wirkungslos, wenn der erwartete Nutzen für den Angreifer höher ist als das Risikopotential.

4. **Ablenkung** (*deflection*)
   Als Ablenkung werden Maßnahmen bezeichnet, die einen Angreifer in eine speziell kontrollierte Umgebung locken, in der er möglichst geringen Schaden verursachen kann. Halme und Bauer (2000) unterscheiden dabei *controlled faux accounts*, *quarantined faux systems* und *honey pots*.
   Ein *controlled faux account* ist eine speziell gesicherte Umgebung auf dem Zielsystem (z.B. `chroot`-Umgebung in UNIX oder BSD-Jails). Ein *quarantined faux system* dagegen ist ein dediziertes System mit einer kontrollierten Umgebung zu der ein Angreifer möglichst unbemerkt weitergeleitet wird. *Honey pots* sollen Angriffe auf sich ziehen und damit weg von schützenswerten Systemen (vgl. Honeynet (2002)).
   Die Kosten-Nutzen-Relation dieser Methoden muß im Einzelfall genau geprüft werden, da auf der einen Seite fraglich ist, ob sich ambitionierte Angreifer von solchen Systemen vom eigentlichen Ziel ablenken lassen. Auf der anderen Seite fallen Kosten zur Konfiguration und das Betreiben solcher Systeme an.

5. **Erkennung** (*detection*)
   In die Kategorie Erkennung und Nachweis von *intrusions* sind die in Kapitel 3 ausführlicher klassifizierten IDS im engeren Sinne einzuordnen.

6. **Gegenmaßnahmen** (*countermeasures*)
   In diese Kategorie fallen nach Halme und Bauer (2000) aktive und autonome Gegenmaßnahmen als Reaktion auf eine *intrusion*. Die damit verbundene Problematik wurde ebenfalls im Rahmen der Klassifizierung von IDS im engeren Sinne in Kapitel 3.4.3 erläutert.

4.1.2. Spezifische Funktionen von IDS im Rahmen eines Sicherheitskonzeptes

Im letzten Kapitel wurden die verschiedenen Möglichkeiten zur Vorbeugung und Abwehr von *intrusions* in einer Taxonomie dargestellt. In diesem Kapitel sollen Gründe für den Einsatz von IDS im engeren Sinne im Rahmen eines Sicherheitskonzeptes erläutert werden (vgl. (Bace und Mell 2001, S. 5 ff.), (Allen u. a. 2000, S. 43) und (Bace und Mell 2001, S. 38)).

Ein Grund für den Einsatz eines IDS stellt die Erhöhung des Risikos einer Entdeckung und Bestrafung für einen potentiellen Angreifer dar. Damit trägt der Einsatz eines IDS zur Abschreckung (*deterrence*) eines potentiellen Angreifers bei.

Die frühzeitige Erkennung von Angriffsvorbereitungen ist ein weiterer Grund ein IDS zu nutzen. Die Angriffsvorbereitung dient dazu Informationen über das Zielsystem zu erhalten und dessen Struktur und Schwachstellen zu erkunden. Die rasche Erkennung solcher *scans* im Netzwerk oder auf dem Host durch ein IDS eröffnet Reaktionsspielräume um aktiv Schutz- und Gegenmaßnahmen zu treffen (*preemtion*).

Ein weiterer wichtiger Grund für den Einsatz von IDS mit aktiven Reaktionsmöglichkeiten ist der Schutz von Systemen, die aus verschiedenen Gründen nicht durch andere Maßnahmen zu schützen sind. Beispielsweise können Fehler in der Konfiguration von Systemen gemacht oder bekannte *vulnerabilities* wider besseren Wissens nicht behoben werden. Oftmals werden bekanntermaßen verwundbare Anwendungen oder Protokolle (z.B. Telnet, IMAP) vom Anwender unbedingt benötigt. In vielen Fällen müssen auch (Legacy)-Systeme oder Anwendungen mit bekannten Vulnerabilities betrieben werden, da Patches oder Updates nicht bzw. nicht rechtzeitig zur Verfügung stehen. Darüberhinaus werden Patches oft nur zögerlich eingespielt, da Nebenwirkungen und Beeinträchtigungen der Systemstabilität befürchtet werden.
Aus all diesen Gründen kann es günstiger sein ein System mit einer bekannten Schwachstelle durch ein IDS zu schützen. So können *exploits* für bekannte *vulnerabilities* registriert und entsprechende aktive Reaktionen des IDS ausgelöst werden.

Konnte eine erfolgreiche *intrusion* nicht verhindert werden, können mit Unterstützung eines IDS meist umfangreiche Informationen über den Verlauf des Angriffs bereitgestellt werden (Forensische Analyse). Diese Informationen ermöglichen eine detailliertere Diagnose, Recovery und nach Möglichkeit die Behebung der Schwachstelle. Es können Informationen über Schwachstellen im

Sicherheitskonzept bzw. dessen Umsetzung erkannt werden. Zudem können die Daten zur Lokalisierung und Identifizierung des Angreifers sowie zur straf- oder zivilrechtlichen Verfolgung herangezogen werden.

Weiterhin kann durch ein IDS die reale Bedrohung für Systeme oder eine Organisation dokumentiert werden. Es können Häufigkeit, Charakter und Ziel der Angriffsversuche dokumentiert werden. Daraus lassen sich zielgerichtet weitere Sicherheitsmaßnahmen ableiten und umsetzen.

Ein letztes Argument für den Einsatz eines IDS ist, daß insbesondere in großen und komplexen Systemumgebungen in Teilbereichen ein "Qualitätskontrolle" des Sicherheitskonzeptes durchgeführt werden kann. Schwachstellen im Design und der Umsetzung können festgestellt und beseitigt werden (z.B. Design und Umsetzung eines Regelsatzes für Paketfilter). Zudem können IDS Muster in der Nutzung der Systeme oder bei auftretenden Problemen deutlich machen.
All diese angeführten Gründe zeigen, daß der Einsatz eines IDS wichtige Funktionen zur Gewährleistung einer umfassenden IT-Sicherheit beiträgt.

4.2. Herausforderungen für IDS und aktuelle Entwicklungsrichtungen

4.2.1. Herausforderungen für IDS

In diesem Abschnitt sollen zuerst ungelöste Probleme und Herausforderungen als Gründe für nicht ausgereifte IDS genannte werden. Auf eine Wiederholung der bereits in Kapitel 3.4 erläuterten spezifischen Nachteile unterschiedlicher Ausprägungen wird dabei verzichtet. In Anlehnung an (Allen u. a. 2000, S. 47 ff.) soll anschließend in den fünf Bereichen externe Faktoren, Netzwerkinfrastruktur, Test und Datenanalyse, organisatorische-menschlicher Bereich und funktionale Grenzen treibende Faktoren für die Weiterentwicklung von IDS aufgezeigt werden.

Das jährlich stattfindende "International Symposium on Recent Advances in Intrusion Detection (RAID)" (vgl. RAID (2002)) dient dem Informationsaustausch zur Forschung im Gebiet der *intrusion detection*. Im Beitrag "Recent advances in intrusion detection systems" (Frincke und Huang 2000, S. 542) werden die folgenden ungelösten Probleme und Herausforderungen als Gründe für relativ unreife IDS genannt.
Zum ersten werden Rechner und Rechnernetze von immer mehr Personen und Organisationen genutzt. Damit ergibt sich eine größere Anzahl von potentiellen Angriffszielen. Zudem hat ein Großteil dieser Nutzer in der Regel nur ein relativ geringes technisches Verständnis der eingesetzten Technologien und der damit verbundenen Risiken. Zusammen mit der hohen Verfügbarkeit von einfach anzuwendenden Angriffs-Skripten oder Exploit-Tools ist die Anzahl der Angriffe bereits deutlich gestiegen und wird in Zukunft sicher noch weiter steigen (vgl. CERT/CC (2002c)).
Zum zweiten wird durch aktuelle IDS das Problem des internen Mißbrauchs kaum adressiert. Vermutungen und Erfahrungswerten zufolge stellt *insider misuse* jedoch eine beachtliche Bedrohung der IT-Sicherheit dar.

Drittens wird Sicherheit oftmals noch nicht als integraler Bestandteil von Systemen begriffen. Design-, Implementierungs- und Konfigurationsfehler kommerzieller Systeme führen zu Schwachstellen, die dann für Angriffe ausgenutzt werden können.
Als viertes wird die immer noch hohe Anzahl von falschen Alarmen der heutigen IDS als ungelöstes Problem angesehen. Falsch-positive Alarme erzeugen Kosten durch Analyse der Daten während falsch-negative Alarme eine nicht vorhandene Sicherheit vorspiegeln.

In den folgenden fünf Abschnitten werden angelehnt an (Allen u. a. 2000, S. 47 ff.) Herausforderungen für IDS in verschiedenen Bereichen dargestellt. Im Bereich "externe Faktoren" werden Herausforderungen für IDS beschrieben, die sich aus der Änderung der äußeren Rahmenbedingungen ergeben und die Entwicklung von IDS vorantreiben.
Durch immer komplexere Angriffsstrategien (z. B. verteilte Angriffe) und mächtigere Werkzeugen (z. B. Angriffs-Skripte, benutzerfreundliche *exploit tools*) verbessern Angreifer ihre Möglichkeiten in IT-Systeme einzudringen. Beide Faktoren erfordern eine ständige Weiterentwicklung von IDS und der zugrundeliegenden Technologien.
Neue Herausforderungen für IDS erwachsen auch aus dem Einsatz neuer Anwendungen und Infrastrukturkomponenten (z. B. Verzeichnisdiensten (LDAP), Public Key Infrastrukturen (PKI)). Diese relativ jungen Anwendungen sind oft wenig ausgereift, enthalten Sicherheitslücken, werden noch nicht ausreichend beherrscht und stellen damit ein Ziel für Angriffe dar.
Ein weiterer externer Faktor sind Angriffe, die gegen das IDS selbst gerichtet sind. Bei einem Angriff werden IDS nach Möglichkeit zuerst attackiert. Ziel ist eine Deaktivierung des IDS. Um IDS gegen solche Angriffe resistent zu machen sind weitere Forschungsarbeiten notwendig.
Weiterhin gelten die potentiellen Gefahren durch mobilen Code als Herausforderung für IDS. Mobiler Code kann heute kaum automatisch und sorgfältig hinsichtlich seiner möglichen Wirkungen auf das Zielsystem überprüft werden.

Die immer weiter anwachsende Komplexität von Netzwerkinfrastrukturen stellt ebenfalls neue Anforderungen an IDS.
Mit wachsender Größe von vernetzten Systemen nimmt der Aufwand zur Skalierung des IDS zu. Es ergeben sich Probleme wenn unterschiedliche IDS in Umgebungen mit heterogenen Technologien, Datenformaten oder Sicherheitspolitiken eingesetzt werden sollen.
Durch eine bisher mangelhafte Standardisierung wird zudem eine Interoperabilität zwischen verschiedenen IDS und Netzwerk-Management Systemen (NMS) verhindert. Erst mit weithin akzeptierten Standards können Daten von allen beteiligten Systemkomponenten ausgetauscht und verarbeitet werden.
In der Arbeit (Iheagwara und Blyth 2002, S. 109 f.) werden ähnliche Herausforderungen für IDS in großen und verteilten Infrastrukturen gesehen. Es wird die Verknüpfung lokaler Informationen zu einer globalen Sicht angestrebt, um unterschiedlichste Angriffsmuster zu erkennen. Dazu werden neue Architekturansätze benötigt, die den Austausch und die Aggregation von Informationen, Arbeitsteilung und Koordination autonomer lokaler IDS-Komponenten ermöglichen. Dies erfordert die Entwicklung geeigneter Sprachen und Protokolle. Gleichfalls muß durch diese Ar-

chitektur sichergestellt werden, daß die nötige Performanz in komplexen Infrastrukturen erreicht wird. Letztlich müssen IDS vollständig in NMS integriert werden.

Im Bereich "Tests und Datenanalyse" werden weitere Herausforderungen für IDS gesehen.
Bis heute wurden erst wenige Anstrengungen unternommen, IDS nach anerkannten und aussagekräftigen Kriterien zu evaluieren. Auf die Problematik der Evaluierung von IDS wird beispielsweise in Lippmann u. a. (2000) eingegangen.
Die Reduktion und Analyse von Audit-Daten ist ebenfalls eine Herausforderung für IDS. Hier fehlen IDS oft Werkzeuge um große Datenmengen effektiv und nach spezifischen Erfordernissen auswerten zu können. Als bedeutsame Herausforderung wird z.B. von Bass (2000) zukünftig die Verknüpfung von Daten verschiedenster Sensoren sowie die Einbeziehung von weiteren Wissensquellen gesehen.
Als weiterer Schwachpunkt aktueller IDS wird die fehlende oder unzureichende Unterstützung der forensische Analyse gesehen. Darunter wird die Sicherung und Auswertung relevanter Daten einer erkannten *intrusion* verstanden um Angreifer und ihre Ziele zu identifizieren. Diese Beweismittel sollen Gegenmaßnahmen (z.B. Strafverfolgungsmaßnahmen) gegen den Angreifer unterstützen. Die Bildung eines Angreifer-Profils anhand der Vorgehensweise und der genutzten Werkzeuge ist noch nicht möglich. Auch die die Ermittlung des Ausmaßes und die Behebung des Schadens werden durch IDS praktisch nicht unterstützt.

Im organisatorischen und menschlichen Bereich werden ebenfalls einige Herausforderungen beim Einsatz von IDS identifiziert.
Die Kooperation im Bereich IDS ist zwischen konkurrierenden Organisationen oftmals wegen fehlender Regelungen nicht möglich. Damit wird der Austausch und die Publikation von Informationen über *intrusions* behindert. Diese Kooperation soll aber zum Vorteil aller Beteiligten gefördert werden, um rasch auf neue Bedrohungen reagieren zu können. Die Zusammenarbeit zwischen Organisationen wird außerdem durch das Fehlen einer weithin akzeptierte ID Terminologie (vgl. die verschiedensten Taxonomien für *intrusions* oder *exploits*) erschwert.
Der Einsatz von IDS in Unternehmen wirft auch Fragen bezüglich des Datenschutzes und des Arbeitsrechts auf. Gerade in Deutschland ist zu klären, in wie weit die erhobenen Daten personalisiert ausgewertet werden dürfen und damit zur Überwachung von Angestellten eingesetzt werden können. Ausführlich mit Aspekten des Datenschutzes beschäftigt sich Sobirey (1999).
Eine weitere Schwachstelle heutiger IDS ist die zielgerichtete Interaktion von Mensch und Maschine im Fall eines Angriffs. Heute werden von IDS Alarme generiert, die wenige (technische) Detailinformationen enthalten. Der Anwender muß auf dieser Basis den Grad der Gefährdung einschätzen und daraufhin kritische Entscheidungen treffen. Zur Diagnose und Entscheidungsfindung werden deshalb so schnell als möglich umfangreiche Informationen in angemessener Darstellung sowie Verknüpfungen benötigt. Damit können die analytisch-kreativen Möglichkeiten des Menschen (Erkennung von vorher unbekannten Mustern, Herstellung von Beziehungen) besser in den Prozeß der ID einbezogen werden.

Dies ist auch unter dem Aspekt wichtig, daß zur ID qualifiziertes Personal notwendig ist und notwendiges Wissen entsprechend gepflegt werden muß (vgl. dazu (Allen u. a. 2000, S. 14)). Sowohl qualifizierte Mitarbeiter als auch Know-how sind begrenzte Ressourcen. Diese müssen gepflegt und weiterentwickelt werden, da auch in Zukunft menschliche Aktionen im Rahmen des *incident handling* notwendig sein werden.

Im fünften und letzten Bereich werden in Anlehnung an (Allen u. a. 2000, S. 68 ff.) funktionale Grenzen von IDS aufgeführt, zu deren Überwindung eine Weiterentwicklung von ID-Technologien stattfinden muß.
Aktuelle IDS sind meist nicht in der Lage einen Angriff zu erkennen, bevor Schaden entstanden ist. Viele Angriffe können nämlich von einem IDS in frühen Anfangsstadien nicht sicher oder überhaupt nicht erkannt werden. Genauso können unbekannte Angriffsstrategien von aktuellen IDS meist nicht erkannt werden. Hier sind weitere Arbeiten in Richtung von *anomaly-based*- bis hin zu *computer immunology*-Technologien notwendig.
Ein weiteres Problem heutiger IDS ist die oft unzureichende Performanz. Die Performanz eines IDS zu messen und aussagekräftig zu vergleichen ist schwierig, da keine allgemein akzeptierten Kriterien existiert. Bei *network-based* IDS treten oftmals bei hoher Bandbreite Performanzprobleme (z.B. nicht erfaßte Pakete) auf. Weiterhin sind heutige IDS nicht in der Lage einen (DoS-)Angriff auf sich selbst zu erkennen und entsprechend darauf zu reagieren.
Ebenso problematisch ist die Beurteilung der Schutzwirkung bzw. der Qualität und Effektivität eines eingesetzten IDS. Aus verschiedenen Gründen können unbemerkte *intrusions* (*false negativ alarms*) in der Praxis nie vollständig ausgeschlossen werden. Beispielsweise besteht bei der Generierung von Signaturen gegen neue Angriffe fast immer ein Konflikt zwischen Qualitätstests und zeitnaher Veröffentlichung der Signatur.
Als letzte Herausforderung für zukünftige IDS soll an dieser Stelle die Problematik durch zunehmend mobile Nutzer und Geräte (Laptop, PDA, Mobiltelefon) genannt werden. Diese Endgeräte werden zunehmend in Netze integriert, bieten neue Schwachstellen, öffnen Möglichkeiten für Angriffe und sind selbst Angriffsziele. Mit zunehmender Vernetzung von Geräten aller Art und deren Steuerung mittels Mikroprozessoren (z.B. Hausnetzwerke und Geräte in "intelligenten Häusern") ergeben sich weitere Angriffsmöglichkeiten. Alle diese Herausforderungen sind durch IDS zukünftig zu berücksichtigen.

4.2.2. Forschung und Entwicklung im Bereich IDS

Im vorangehenden Kapitel wurden die aktuellen Herausforderungen für IDS dargestellt. In diesem Kapitel sollen zuerst Konstanten und Trends in der Entwicklung im Bereich IDS genannt werden. Anschließend werden Empfehlungen zur Umsetzung bekannter Technologien sowie neue Forschungsansätze vorgestellt werden. Abschließend werden aktuelle Forschungs- und Entwicklungsrichtungen im Bereich IDS aufgezeigt.

In der bisherigen Entwicklung von IDS lassen sich nach (Axelsson 1998, S. 12 ff.) die folgenden Konstanten erkennen: Zur Erkennung von *intrusions* werden hybride Ansätze verwendet, es wird versucht die Erkennung nahezu in Realzeit durchzuführen und der für IDS benötigte Ressourcenanteil nimmt eher zu als ab. Weiterhin wurden bisher Angriffe gegen IDS selbst vernachlässigt, kaum Studien über die Effektivität und Effizienz von IDS erstellt und organisatorische Fragen kaum untersucht.
Demgegenüber werden Trends von host- zu netzwerk-basierten IDS und von zentralisierten hin zu verteilten Ansätzen ausgemacht. Es zeichnet sich ebenfalls ein Trend zu größerer Interoperabilität von IDS-(Komponenten) und einer höheren Widerstandsfähigkeit gegen Angriffe auf das IDS selbst ab.

Aufgrund ihrer Studie zum Stand der Technik von ID-Technologien geben (Allen u. a. 2000, S. 109) einige Empfehlungen, die sich aus den im Kapitel 4.2.1 beschriebenen Herausforderungen ableiten. Nach diesen Empfehlungen sollen vorhandene Technologien und Ansätze in konkrete IDS umgesetzt werden.
Es wird gefordert, die Anzahl der falschen Alarme drastisch zu senken und neue Bedrohungen zuverlässig zu erkennen. Möglichkeit dazu sind die Einbeziehung vieler Sensoren, der Verknüpfung der Analysedaten und eine gemeinsame Auswertung. Die Verwendung von "*state-transition*"- oder Petri-Netz-Modellen verspricht gleichfalls eine Reduktion der falschen Alarme. Darüberhinaus sollen neue Technologien entwickelt werden, um bisher unbekannte Bedrohungen zu erkennen. Beispielsweise führen (Allen u. a. 2000, S. 83 ff.) Lernen durch Klassifikation von Beispielen, Neuronale Netze oder genetische Algorithmen an. Zudem soll mobiler Code künftig stärker in den Fokus von IDS rücken und detailierter analysiert werden.
Eine andere Empfehlung ist, Angriffe auf IDS selbst angemessen zu berücksichtigen und die Erkennung sowie Abwehr solcher Angriffe zu verbessern.
Weiterhin werden das Fehlen von Testumgebungen und aussagekräftige Qualitäts-Tests für IDS kritisiert. In ihrer Arbeit konnten Allen u. a. (2000) beispielsweise keine produktive Umgebung finden,in der kommerzielle IDS intensiv getestet wurden. Geeignete Tests sind zu entwickeln, um IDS ausreichend testen und sowohl ihre Effektivität als auch ihre Effizienz beurteilen zu können.
Weitere Ansatzpunkte zur Weiterentwicklung von IDS werden in der besseren Einbindung des Menschen in den Prozeß der ID gesehen. Nach Möglichkeit soll vorhandene menschliche Erfahrung effektiver in den Prozeß der ID integriert werden. Die Schnittstelle zwischen Mensch und IDS soll so gestaltet werden, daß die menschliche Analyse der Daten effektiv unterstützt wird.
Als letzten Punkt empfehlen Allen u. a. (2000) die engere Kooperation und den Informationsaustausch zwischen allen Beteiligten um die Entwicklung von IDS zu fördern.

Zu ähnlichen Ergebnissen und Empfehlungen kommen (Lippmann u. a. 2000, S. 593 f.) in ihrer Studie "The 1999 DARPA off-line intrusion detection evaluation". Die Autoren schlagen vor, neue und bisher nicht einbezogene Datenquellen zukünftig bei der Analyse zu berücksichtigen. Beispielsweise können Ansätze zur Auswertung von UNIX-Audit-Daten auf Windows NT-Audit-Daten übertragen werden und eine größere Bandbreite von Netzwerkprotokollen und -diensten

analysiert werden. In dieser Studie wird ebenfalls auf fehlende Detailinformationen bei einem Alarm hingewiesen. Deshalb sollen IDS in Zukunft ausreichende forensische Informationen liefern können. Zur Entdeckung neuer Angriffe sollen entsprechende Ansätze (wie beispielsweise *anomaly detection* weiterentwickelt werden.

Nach der ausführlichen Darstellung der Herausforderungen an IDS soll nun abschließend kurz auf die aktuelle Forschung im Gebiet IDS eingegangen werden.
Heute konzentriert sich die jüngere Forschung im Bereich IDS nach (Frincke und Huang 2000, S. 543 f.) vor allem auf die folgenden Gebiete: Neben der Evaluierung von IDS wird an gemeinsamen Kommunikationsstandards und -formaten zur Integration von IDS gearbeitet. In diesem Kontext werden Architekturen für IDS in großen und verteilten Netzwerken entwickelt. Außerdem werden Möglichkeiten zur Erkennung neuartiger Angriffe gesucht.
Zum Zeitpunkt der RAID 1999 waren weitere Forschungsschwerpunkte aktuell. Es wird an dem Problem der Identifikation von *insider misuse* gearbeitet und die Schwierigkeiten diskutiert, mit IDS rechtliche Verfahren zu unterstützen. Da immer häufiger die verschiedenartigsten Gerätschaften miteinander vernetzt werden, zeichnet sich ein Wechseln von klassischen Computer- und Netzwerksystemen hin zu Systemen und Netzwerken von Geräten ab. Die Auswirkungen dieses Wechsels auf Sicherheitsfragen und IDS werden untersucht. Die aktuelle Forschung beschäftigt sich weiterhin mit aktiven Funktionen zur Selbstverteidigung eines IDS.
In (Bace und Mell 2001, S. 46 f.) wird die Zukunft von IDS in IDS-Appliances gesehen. Diese werden zunehmenden transparent in Netzwerkkomponenten integriert werden.

Einen weiteren Ausblick auf zukünftige IDS liefert Bass (2000). Die Erkenntnis, daß IDS und NMS integriert werden müssen, macht ein gemeinsames Modell und die Vereinigung zahlreicher heterogenen Datenquellen notwendig. Das "*multisensor data fusion*"-Modell stellt dazu ein funktionales Rahmenwerk zur Verfügung und bildet die Grundlage eines Entscheidungsunterstützungssystems. Dieses soll den Prozeß der Abstraktion von Daten über Informationen zu Wissen leisten und bedient sich dabei zahlreicher unterschiedlicher Methoden. Schließlich läßt sich auch Data Minig betreiben, um aus historischen Analysedaten Wissen zu gewinnen.

5. Konzeption eines Praktikumsversuches zum Thema IDS

Der praktische Teil dieser Arbeit besteht in der Konzeption eines Praktikumsversuches zum Thema IDS. In diesem Kapitel werden die Überlegungen zur Konzeption und Umsetzung eines solchen Praktikumsversuches erläutert. Anschließend wird das ausgewählte IDS kurz vorgestellt.

Der zu konzipierende Praktikumsversuch soll in die Lehrveranstaltungen "Telematik und Verteilte Anwendungen (Wi-Inf. 7. Sem)" und evtl. auch "Sicherheit in Rechnernetzen" (Inf. 9. Sem.) des Fachgebietes Telematik an der TU Ilmenau integriert werden.
Ziel des Praktikums ist es, Kenntnisse der grundlegenden Funktionsprinzipien und Komponenten eines IDS, von Klassifizierungsmerkmalen für IDS sowie Möglichkeiten und Grenzen eines IDS zu erarbeiten. Aus den Zielen wurde anschließend ein Konzept für das Praktikum entwickelt.
Zur Vorbereitung auf die praktische Übung muß sich der Praktikumsteilnehmer in die Thematik IDS einarbeiten. Hierzu stehen ihm die Praktikumsanleitung und die darin genannte Literatur zur Verfügung. Außerdem sollte er über Kenntnisse des Betriebssystemes Linux mit seinen Werkzeugen und Diensten, der TCP/IP-Protokollwelt und über Sicherheit in Rechnernetzen verfügen.
In der praktischen Übung soll der Student die Möglichkeiten und Grenzen eines IDS selbst austesten. Dazu werden *exploits* mit bzw. ohne Schutz durch ein IDS zur Ausführung gebracht und die Ergebnisse verglichen.

In der ersten Überlegung zur praktischen Realisierung sollten die Praktikumsteilnehmer selbst ein IDS implementieren. Allerdings kann die dafür notwendige tiefgehende Fachkenntnis nicht vorausgesetzt werden. Zudem sprechen praktische Probleme gegen diesen Ansatz (zu wenig Rechner mit physikalischem Root-Zugang, ...).
Die zweite Idee bestand im Entwurf eines sinnvollen Anwendungsszenarios, dessen verwundbare Stellen gefunden und gezielt angegriffen werden sollen. Hierbei stellte sich heraus, daß es sehr schwierig ist für aktuelle Implementierungen von Diensten ausführbare *exploits* zu finden.
Deshalb wurde schließlich gezielt nach ausführbaren *exploits* gesucht. Hierbei steht nicht die Praxisrelevanz der *exploits*, sondern vielmehr das Verständnis der zugrundeliegenden Prinzipien im Vordergrund. Die *exploits* sollen vom Praktikumsteilnehmer ausgeführt werden können und vom IDS erkannt werden. Für erfahrenere Teilnehmer bietet sich darüberhinaus die Möglichkeit weitere *exploits* zu suchen und anzuwenden.

Für den Praktikumsversuch wurde ein regelbasiertes *anomaly detection* IDS gewählt, das nach dem *transition-based detection paradigma* arbeitet. Das IDS ist als Linux Kernel Modul (vgl. Lawless (2002b)) für die 2.2er und 2.4er Kernel-Reihe implementiert und arbeitet nach dem Saint Jude Modell (vgl. Lawless (2002c) und Lawless (2002a)). Saint Jude ist ein Modell zur Erkennung von unzulässigen privilegierten Transitionen eines Prozesses und versucht Angriffe ohne genaueres Wissen über das Angriffsziel oder Angriffssignaturen zu erkennen und zu unterbinden. Die wesentlichen Punkte des Modells werden im folgenden beschrieben.

Prozesse sind entweder privilegiert (in der Implementierung ist dies identisch mit UID = 0, also Root) oder unprivilegiert. Privilegierte Prozesse können durch Systemaufrufe (*exit()*, *setuid(*$\neq 0$*)*, *setreuid(*$\neq 0$*)*) ihre Privilegien verlieren. Prozesse können mittels des Systemaufrufs *execve()* Applikationen ausführen. Ist die ausgeführte Applikation *setuid = 0* wird aus einem unprivilegierten ein privilegierter Prozeß. Eine Änderung des Privilegienstatus heißt im Modell Transition.
Für jeden privilegierten Prozeß wird vom IDS zur Laufzeit eine Liste von Restriktionen geführt. In dieser Liste ist für den Prozeß angegeben, welche Transitionen dieser Prozeß ausführen darf. Existiert zu einer Transition eine passende Regel, so wird die Transition ausgeführt. Existiert keine Regel, wird die Transition als unzulässig angesehen und nicht ausgeführt. Damit können beispielsweise durch *buffer overflow exploits* provoziert Transitionen verhindert werden.
Die Liste der Restriktionen wird für jeden privilegierten Prozeß auf Grund der Regelbasis generiert. Die Regelbasis wird während einer Lernphase durch das IDS aufgebaut. Sie enthält als Schlüssel den vollständigen Pfad und Argumente von privilegierte Prozessen. Zu jedem Schlüssel können zulässige Transitionen angegeben werden (d.h. welche Applikationen mit welchen Parametern der privilegierte Prozeß ausführen darf). Die *default rule* paßt auf alle privilegierten Prozesse und verbietet die Ausführung jeglicher anderer Applikationen, solange nicht vorher die Privilegien abgegeben werden. Die *unrestricted rule* ermöglicht privilegierten Prozessen jede andere Applikation aufzurufen und wird auch an die aufgerufene Applikation vererbt. Einträge werden in der Regelbasis nach der längsten Übereinstimmung (d.h. vollständiger Pfad und alle Argumente) ausgewählt.
Die Implementierung des Saint Jude Modells bietet bei umsichtiger Konfiguration guten Schutz gegen bekannte und unbekannte Angriffe. Sowohl die Rate der falsch positiven als auch der falsch negativen Alarme liegt nach Auskunft des Entwicklers sehr niedrig.
Einschränkungen ergeben sich einmal durch die Eigenschaften von *anomaly detection* IDS (vgl. Kap. 3.4.2). Die Lernphase muß ausreichend lange sein, alle "normalen" Vorgänge enthalten und nach Änderungen des Systemverhaltens ist in der Regel eine neue Lernphase notwendig um die Regelbasis anzupassen. Weiterhin müssen beispielsweise die ausführbaren Dateien im Dateisystem vor Austausch und Veränderungen geschützt werden. Sonst kann hier ein trojanisches Pferd plaziert werden und wird durch eine zulässige Transition mit privilegiertem Status ausgeführt.

Basierend auf diesen Grundlagen wurde eine Praktikumsanleitung (vgl. Anhang A) sowie Hilfestellungen für den Praktikumsbetreuer (vgl. Anhang B) erarbeitet.

Literatur

[**Alessandri u. a. 2001**] Alessandri, Dominique (Hrsg.) ; Christian, Cachin (Mitarb.) ; Marc, Dacier (Mitarb.) ; Oliver, Deak (Mitarb.) ; Klaus, Julisch (Mitarb.) ; Brian, Randell (Mitarb.) ; James, Riordan (Mitarb.) ; Andreas, Tscharner (Mitarb.) ; Andreas, Wespi (Mitarb.) ; Candid, Wüest (Mitarb.): Towards a Taxonomy of Intrusion Detection Systems and Attacks / IBM Research, Zurich Research Laboratory. URL http://www.newcastle.research.ec.org/maftia/deliverables/D3final.pdf. – Zugriffsdatum: 2002-07-17, September 2001 (Project IST-1999-11583 Malicious- and Accidental-Fault Tolerance for Internet Applications). – Forschungsbericht

[**Allen u. a. 2000**] Allen, Julia ; Christie, Alan ; Fithen, William ; McHugh, John ; Pickel, Jed ; Stoner, Ed: State of the Practice of Intrusion Detection Technologies / Carnegie Mellon Software Engineering Institute, Carnegie Mellon University. Pittsburgh, PA, USA, Januar 2000 (CMU/SEI-99-TR-028). – Forschungsbericht. – URL http://www.sei.cmu.edu/pub/documents/99.reports/pdf/99tr028.pdf. – Zugriffsdatum: 2002-07-23

[**Anderson 1980**] Anderson, James P.: Computer Security Threat Monitoring and Surveillance / James P. Anderson Co. Fort Washington, PA, USA, April 1980 (Contract 79F296400). – Forschungsbericht. – URL http://seclab.cs.ucdavis.edu/projects/history/papers/ande80.pdf. – Zugriffsdatum: 2002-08-16

[**Axelsson 1998**] Axelsson, Stefan: Research in Intrusion-Detection Systems: A Survey / Department of Computer Engineering, Chalmers University of Technology. Göteborg, Sweden, Dezember 1998 (TR: 98-17). – Forschungsbericht. – URL http://www.ce.chalmers.se/staff/sax/survey.ps. – Zugriffsdatum: 2002-07-24

[**Axelsson 2000**] Axelsson, Stefan: Intrusion Detection Systems: A Survey and Taxonomy / Department of Computer Engineering, Chalmers University of Technology. Göteborg, Sweden, März 2000. – Forschungsbericht. – URL http://www.ce.chalmers.se/staff/sax/taxonomy.ps. – Zugriffsdatum: 2002-07-17

[**Bace und Mell 2001**] Bace, Rebecca ; Mell, Peter: Intrusion Detection Systems / National Institution of Standards and Technology. USA, November 2001 (NIST Special Publication (SP 800-31)). – Forschungsbericht. – URL http://csrc.nist.gov/publications/nistpubs/800-31/sp800-31.pdf. – Zugriffsdatum: 2002-07-16

[**Bass 2000**] Bass, Tim: Intrusion Detection Systems And Multisensor Data Fusion. In: *Communications Of The ACM* 43 (2000), April, Nr. 4, S. 99–105

[**Britannica 1997–1999**] Encyclopædia Britannica: *Encyclopaedia Britannica CD : knowledge for the information age*. Chicago, Ill., USA : Encyclopaedia Britannica, 1997–1999. – CD-ROM-Ausg.

[**Bruneau 2001**] Bruneau, Guy: *The History and Evolution of Intrusion Detection* / The SANS Institute (Hrsg.). Oktober 2001. – URL http://rr.sans.org/intrusion/evolution.php. – Zugriffsdatum: 2002-08-01

[**CERT/CC 2002a**] CERT/CC (Hrsg.): *CERT Coordination Center* / Carnegie Mellon Software Engineering Institute, Carnegie Mellon University. Juni 2002. – URL http://www.cert.org/nav/index.html. – Zugriffsdatum: 2002-07-23

[**CERT/CC 2002b**] CERT/CC (Hrsg.): *CERT Coordination Center 2001 Annual Report* / Carnegie Mellon Software Engineering Institute, Carnegie Mellon University. Februar 2002. – URL http://www.cert.org/annual_rpts/cert_rpt_01.html. – Zugriffsdatum: 2002-07-23

[**CERT/CC 2002c**] CERT/CC (Hrsg.): *CERT/CC Statistics 1988-2002* / Carnegie Mellon Software Engineering Institute, Carnegie Mellon University. Oktober 2002. – URL http://www.cert.org/stats/cert_stats.html. – Zugriffsdatum: 2002-09-23

[**Cisco 2000**] Cisco Systems, Inc. (Hrsg.): *Cisco Intrusion Detection*. 2000. – URL http://www.cisco.com/warp/public/cc/pd/sqsw/sqidsz/index.shtml. – Zugriffsdatum: 2002-08-30

[**Debar u. a. 1999**] Debar, Hervé ; Dacier, Marc ; Wespi, Andreas: Towards a Taxonomy of Intrusion Detection Systems. In: *Computer Networks* 31 (1999), S. 805–822

[**Debar u. a. 2000**] Debar, Hervé ; Dacier, Marc ; Wespi, Andreas: A Revised Taxonomy for Intrusion-Detection Systems. In: *Annales des Télécommunications* 55 (2000), S. 361–378

[**Denning 1987**] Denning, Dorothy E.: An Intrusion Detection Model. In: *IEEE Transactions on Software Engineering* 13 (1987), Februar, Nr. 2, S. 222–232

[**Deraison 2000**] Deraison, Renaud (Hrsg.): *The Nessus Project*. 2000. – URL http://www.nessus.de/intro.html. – Zugriffsdatum: 2002-08-12

[**DOD 1985**] Department Of Defense (Hrsg.): Department Of Defense Trusted Computer System Evaluation Criteria. USA, December 1985 (DOD 5200.28-STD). – Department of Defense Standard. – URL http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.html. – Zugriffsdatum: 2002-08-10

[**Einwechter 2002**] Einwechter, Nathan: *Implementing Networks Taps with Network Intrusion Detection Systems* / Fate Research Labs. Juni 2002. – URL http://online.securityfocus.com/infocus/1594. – Zugriffsdatum: 2002-07-23

[**Escamilla 1998**] Escamilla, Terry: *Intrusion Detection : Network Security Beyond the Firewall*. New York : Wiley, 1998 (Wiley computer publishing). – ISBN 0-471-29000-9

[**Farmer und Spafford 2002**] Farmer, Daniel ; Spafford, Eugene H.: The COPS Security Checker System / Purdue University, Software Engineering Research Center. West Lafayette, Indiana, USA, November 2002. – Technical Report CSD-TR-993. – URL https://www.cerias.purdue.edu/techreports-ssl/public/94-01.pdf. – Zugriffsdatum: 2002-08-12

[**Feiertag u. a. 1999**] Feiertag, Rich ; Kahn, Cliff ; Porras, Phil ; Schnackenberg, Dan ; Staniford-Chen, Stuart ; Tung, Brian (Hrsg.): *A Common Intrusion Specification Language (CISL)*. Juni 1999. – URL http://www.isi.edu/gost/cidf/drafts/language.txt. – Zugriffsdatum: 2002-08-24

[**Frincke und Huang 2000**] Frincke, Deborah A. ; Huang, Ming-Yuh: Recent advances in intrusion detection systems. In: *Computer Networks* 34 (2000), S. 541–545

[**Halme und Bauer 2000**] Halme, Lawrence R. ; Bauer, R. K.: *AINT Misbehaving: A Taxonomy of Anti-Intrusion Techniques* / SANS Institute. 2000. – URL http://www.sans.org/newlook/resources/IDFAQ/aint.htm. – Zugriffsdatum: 2002-08-10

[**von Helden u. a. 1998**] Helden, Dr. J. von ; Karsch, Dr. S. ; debis IT Security Services (Hrsg.): Grundlagen, Forderungen und Marktübersicht für Intrusion Detection Systeme (IDS) und Intrusion Response Systeme (IRS). Bonn, Oktober 1998 (IDS-10-03). – Forschungsbericht. – URL http://www.bsi.de/literat/studien/ids/ids-stud.pdf. – Zugriffsdatum: 2002-08-13

[**Honeynet 2002**] The Honeynet Project: *Honeynet Project*. 2002. – URL http://honeynet.slashgod.info/. – Zugriffsdatum: 2002-09-23

[**Howard 1997**] Howard, John D.: *An Analysis Of Security Incidents On The Internet 1989 –1995*. Pittsburgh, PA, USA, Carnegie Mellon University, Dissertation, April 1997. – URL http://www.cert.org/research/JHThesis/Start.html. – Zugriffsdatum: 2002-08-14

[**IBM 2002**] IBM: *AIX 4.3.1 TCSEC Evaluated C2 Security (PRPQ)*. Produktinformation. 2002. – URL http://www-1.ibm.com/servers/aix/products/ibmsw/security/c2brief.html. – Zugriffsdatum: 2002-10-10

[**IDWG 2002**] Intrusion Detection Working Group (Hrsg.): *Intrusion Detection Exchange Format (idwg) Charter* / Internet Engineering Task Force. 2002. – URL http://www.ietf.org/html.charters/idwg-charter.html. – Zugriffsdatum: 2002-08-01

[**Iheagwara und Blyth 2002**] Iheagwara, Charles ; Blyth, Andrew: Evaluation of the performance of ID systems in a switched and distributed environment : the RealSecure case study. In: *Computer Networks* 39 (2002), S. 93–112

[**ISS 2002**] Internet Security Systems, Inc. (Hrsg.): *Enterprise Protection*. Produktinformation. 2002. – URL http://www.iss.net/products_services/enterprise_protection/. – Zugriffsdatum: 2002-07-30

[**Kersten 1995**] Kersten, Heinrich: *Sicherheit in der Informationstechnik : Einführung in Probleme, Konzepte und Lösungen*. 2., völlig überarb. Aufl. München : Oldenbourg, 1995. – ISBN 3-486-23179-0

[**Lawless 2002a**] Lawless, Tim: *On Intrusion Resiliency*. Mai 2002. – URL http://prdownloads.sourceforge.net/stjude/OIR.pdf?download. – Zugriffsdatum: 2002-10-23

[**Lawless 2002b**] Lawless, Tim: *SourceForge.net: Project Info - Saint Jude*. 2002. – URL http://sourceforge.net/projects/stjude. – Zugriffsdatum: 2002-11-10

[**Lawless 2002c**] Lawless, Timothy: *Saint Jude, the Model*. Juni 2002. – URL http://prdownloads.sourceforge.net/stjude/StJudeModel-1.1.pdf?download. – Zugriffsdatum: 2002-10-23

[**Lippmann u. a. 2000**] Lippmann, Richard ; Haines, Joshua W. ; Fried, David J. ; Korba, Jonathan ; Das, Kumar: The 1999 DARPA off-line intrusion detection evaluation. In: *Computer Networks* 34 (2000), S. 579–595

[**McHugh u. a. 2001**] McHugh, John ; Christie, Alan ; Allen, Julia: Intrusion Detection: Implementation and Operational Issues. In: *Crosstalk : The Journal of Defense Software Engineering* 14 (2001), Januar, Nr. 1, S. 27–31. – URL http://www.stsc.hill.af.mil/crosstalk/2001/01/jan01.pdf. – Zugriffsdatum: 2002-08-10

[**Mell u. a. 2000**] Mell, Peter ; Marks, Donald ; McLarnon, Mark: A denial-of-service resistant intrusion detection architecture. In: *Computer Networks* 34 (2000), S. 641–658

[**Mukherjee u. a. 1994**] Mukherjee, B. ; Heberlein, T.L. ; Levitt, K.N.: Network Intrusion Detection. In: *IEEE Network* 8 (1994), Mai/Juni, Nr. 3, S. 26–41

[**Northcutt 1999**] Northcutt, Stephen: *Network Intrusion Detection : An Analyst´s Handbook*. 1. Auflage. Indianapolis, Ind, USA : New Riders, 1999. – ISBN 0-7357-0868-1

[**OPSEC 2002**] Check Point Software Technologies Ltd. (Hrsg.): *Check Point OPSEC.com*. 2002. – URL http://www.opsec.com/. – Zugriffsdatum: 2002-08-23

[**Porras u. a. 1999**] Porras, Phil ; Schnackenberg, Dan ; Staniford-Chen (Hrsg.), Stuart ; Stillman, Maureen ; Wu, Felix: *The Common Intrusion Detection Framework Architecture*. 1999. – URL http://www.isi.edu/gost/cidf/drafts/architecture.txt. – Zugriffsdatum: 2002-07-24

[**RAID 2002**] RAID (Hrsg.): *RAID : International Symposium on Recent Advances in Intrusion Detection*. 2002. – URL http://www.raid-symposium.org/. – Zugriffsdatum: 2002-09-11

[**Sobirey 1999**] Sobirey, Michael: *Datenschutzorientiertes Intrusion Detection : Grundlagen, Realisierung, Normung*. Braunschweig : Vieweg, 1999 (DuD-Fachbeiträge). – ISBN 3-528-05704-1

[**Sobirey 2000**] Sobirey, Michael: *Michael Sobirey's Intrusion Detection Systems page*. 2000. – URL http://www-rnks.informatik.tu-cottbus.de/~sobirey/ids.html. – Zugriffsdatum: 2002-07-17

[**SRI 2002**] SRI International (Hrsg.): *Programs : Intrusion Detection* / SRI International System Design Laboratory. 2002. – URL http://www.sdl.sri.com/programs/intrusion/. – Zugriffsdatum: 2002-08-30

[**Sun 2002**] Sun Microsystems, Inc. (Hrsg.): *Security Auditing*. Produktinformation. 2002. – URL http://wwws.sun.com/software/security/audit/. – Zugriffsdatum: 2002-08-10

[**Tanenbaum 1998**] Tanenbaum, Andrew S.: *Computernetzwerke*. 3., revidierte Auflage. München [u.a.] : Prentice Hall, 1998. – ISBN 3-8272-9568-8

[**Tripwire 2002**] Tripwire, Inc. (Hrsg.): *Tripwire.org - Home of the Tripwire Open Source Project*. 2002. – URL http://www.tripwire.org/. – Zugriffsdatum: 2002-08-12

[**Whitehats 2001**] Whitehats, Inc. (Hrsg.): *Whitehats Network Security Resource*. 2001. – URL http://www.whitehats.com/index.shtml. – Zugriffsdatum: 2002-08-11

[**Wolf 2002**] Wolf, Jürgen: *Konzeption und Sicherheitsanalyse eines Institutsnetzwerkes und Umsetzung der Ergebnisse*. Ilmenau, TU Ilmenau, Fakultät für Informatik und Automatisierung, Institut für Praktische Informatik, Fachgebiet Telematik, Dipl.-Arb., 2002