Wege zur Entdeckung von Einbrüchen in Computernetze

Studienarbeit im Fachgebiet Datenverarbeitung
Institut für Informationstechnik
Abteilung Informatik, Informations- und Medientechnik
Fakultät 5 – Ingenieurwissenschaften
Universität Duisburg-Essen, Standort Duisburg

Betreuer: Prof.Dr.-Ing. W. Geisselhardt

Anmeldedatum: 30.01.2003

Abgabedatum: 30.04.2003

Martin Wodrich
<mail@martinwodrich.de>

19. April 2003

Inhaltsverzeichnis

Abbildungsverzeichnis

6 Abbildungsverzeichnis

Tabellenverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

Die Sicherheitsproblematik in Computernetzwerken wird zunehmend größer. Immer öfter liest man von erfolgreichen Einbrüchen in bzw. Angriffen auf Computernetze. Angriffe und Einbrüche die den Betrieb dieser Computernetze gefährden.
Als Betreiber eines Computernetzwerkes, ist man allerdings nicht schutzlos den Gefahren jedes Angriffes ausgeliefert. Man kann sehr wohl so einiges tun, um erfolgreiche Angriffe zu verhindern. Dabei geht es nicht nur um das Aussperren unerwünschter Zugriffe von aussen mittels Firewall, sondern auch das Erkennen von laufenden Angriffen (auch und gerade auch solche aus dem internen Netz, nicht bloß aus dem externen Netz, das zumeist das Internet ist). Dabei kann man bei Bedarf (Das heißt: Ein ernsthafter Angriffsversuch findet statt.) auch darauf reagieren, und so dann den Angriff erfolglos machen. Möglichst bevor ein echter Schaden eingetreten ist.

In dieser Studienarbeit soll anhand von verfügbarer Literatur, Informationen aus dem Internet, sowie eignen Tests, verschiedene *Wege zur Entdeckung von Einbrüchen in Computernetze* aufgezeigt werden. Dabei soll auch auf in der Praxis eingesetzte Verfahren eingegangen werden.

Die eignen Tests werden dabei durch Zugriff auf im Internet zu findende freie (PD-)[1] Intrusion-Detection-Systeme (IDS) und Installation in einem Versuchsnetz erfolgen.

Abschließend wird noch darauf eingegangen, welche Schwächen die vorhandenen IDS[2] haben, und wie weit sich diese Schwächen für gezielte Attacken gegen das IDS ausnutzen lassen. Vor allem jene Schwächen werden dabei betrachtet, die die Überwachung des Netzes durch das IDS so sehr behindern, dass Attacken gegen das Netz anschließend oder währenddessen ohne brauchbare Aufzeichnung durch das IDS möglich wären. Hierbei wird auch auf Tests in einem Versuchsnetz zurückgegriffen.

1 Public-Domain (“Gemeinschaftseigentum”)
2 Intrusion-Detection-System (“Einbruchsentdeckungssystem”)

Kapitel 2

Grundlagen der Netzwerk-Intrusion-Detection-Systeme

Die Entdeckung von Einbrüchen beginnt nicht erst bei Intrusion-Detection-Systemen für ganze Computernetze, sondern viel früher. Daher beginnt dieses Kapitel nach einer kurzen Begründung für Intrusion Detection, auch zuerst mit einer Auflistung der Hierachie-Ebenen von IDS, bevor ich auf Netzwerk-IDS näher eingehen werde.

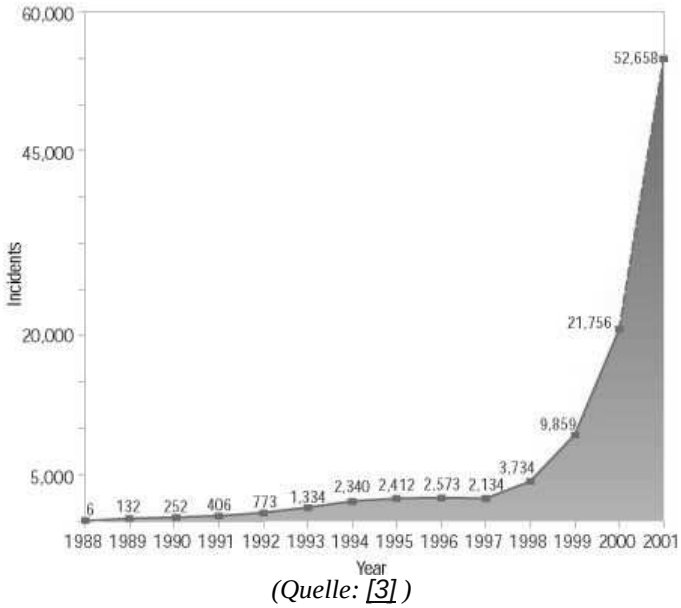
2.1 Warum Einbruchsentdeckung?

(Literatur:[3]), [17])

Wie man an nebenstehender Abbildung erkennt ist in den letzten Jahren die Einbruchs-aktivität in Computernetze immer dramatischer gestiegen.

Trotz Einsatz einer Firewall[1] bleibt oft eine vergessene Hintertür, die einen unerwünschten Zugang ermöglicht.

So bietet eine Firewall aufgrund ihres Prinzips keinen Schutz gegen Aktivitäten der eignen Nutzer gegen die eignen Systeme.



(Quelle: [3])

Abbildung 2.1: Ansteigende Intruderaktivität

Gegen diese Schwächen hilft nur ein Werkzeug, das das Netz und die daran angeschlossenen Computer überwacht und rechtzeitig Alarm auslöst. Oder vielleicht sogar selbstständig Maßnahmen einleitet, die diese Aktivitäten beendet.

Solche Werkzeuge nennt man Intrusion-Detection-Systeme[2].

1 “Brandmauer” , reglementiert den Netzwerkverkehr der aus dem lokalen Netz raus, bzw. in das lokale Netz reingeht.
2 “Einbruchsentdeckungssysteme”

2.2 Hierachie von Intrusion-Detection-Systemen

(Literatur: [1] S.10)

Man unterscheidet 4 grundsätzliche Typen von Intrusion-Detection-Systemen:

1. **Anwendung**
   Auf der Ebene der einzelnen Anwendung analysiert ein IDS[1] das Verhalten der einzelnen Anwendung. Meist durch das Anlegen von Logfiles.

2. **Rechner**
   Auf der Ebene eines ganzen Rechners analysiert ein IDS Daten z.B. aus Logfiles, Process Acounting, Benutzerverhalten oder kontrolliert die Einhaltung von Dateiänderungsregeln.

3. **Netzwerk**
   Auf der Ebene eines ganzen Netzwerks analysiert ein IDS den auftretenden Netzwerkverkehr nach Auffälligkeiten. Dabei können auch Informationen benutzt werden die ein IDS auf Rechner- oder Anwendungsebene ausgegeben hat.

4. **Multinetzwerk/Infrastruktur**
   Hierbei handelt es sich um ein IDS, das mehrere vom gleichen Administratorenteam betreute Netze analysiert. Im Allgemeinen wird ein solches IDS auf Informationen zurückgreifen, die von IDS auf Rechner- , Anwendungs- , Netzwerk- oder sogar anderen Multinetzwerk-IDS ausgegeben wurden.

2.3 Komponenten eines Netzwerk-IDS

(Literatur: [1] S.9)

Die Funktionalität eines Netzwerk-Intrusion-Detection-System lässt sich in 3-4 Komponenten einteilen:

2.3.1 Sensoren

Sensoren sammeln die Daten ein, die das IDS analysieren soll. Dabei können die Eingabedaten des Sensors alles sein, was sich dazu verwerten lässt, einen Angriff festzustellen. Beispielsweise kann es sich dabei um Netzwerkpakete handeln oder aber auch Logfiles. Diese Informationen werden zur Analysekomponente weitergeleitet.

1 Intrusion Detection System

2.3.2 Analysatoren

Die Analysekomponente des IDS empfängt ihre Informationen von den Sensoren und versucht aus diesen Informationen festzustellen, ob ein Angriff gerade abläuft oder nicht. Dabei gibt die Analysekomponente Beweise für ihren Entschluß, dass jetzt ein Angriff stattfindet, oder sogar Anleitungen für Aktionen, die jetzt zu tätigen sind, um den Angriff abzuwehren. Dabei ist die Analysekomponente meist in mehrere Analysatoren aufgeteilt, die jede für sich auf eine Angriffsart spezialisiert ist.

2.3.3 Benutzerschnittstelle

Die Benutzerschnittstelle ermöglicht dem Administrator des IDS die Ausgaben des Systems zu sehen und das Verhalten des IDS zu steuern.

2.3.4 Honeypot[1]

Als weitere Komponente, die aber nicht in allen Netzwerk-IDS vorhanden ist, kann ein Netzwerk-IDS einen so genannten „Honeypot“ enthalten. Dabei handelt es sich um einen Rechner, der für einen potenziellen Einbrecher wie ein Rechner erscheint, auf dem Serverdienste mit bekannten Sicherheitslücken installiert sind und bedeutsame Daten lagern. In Wirklichkeit befinden sich allerdings auf diesem Rechner keine echten wichtigen Daten.
Dafür schafft dieser Rechner allerdings eine Umgebung, die weitere Informationen liefert, um einen Einbruch in das Netz zu analysieren.
Ein Rechner, der als Honeypot fungiert, wartet dabei auf Einbrecher in das Netz, die sich auf den augenscheinlich unsicheren Rechner stürzen, um ihn anzugreifen. Dabei ist der Rechner, der als Honeypot dient, einfach nur ein weiterer Sensor des IDS, der Anzeichen und Warnungen eines Angriffs aufzeichnet. Dabei provitiert das IDS davon das alle Anfragen an den Honeypot Angriffe sind. Der Honeypot behält dabei den bekannten Zustand bei.

2.4 Grundlegende Analysearten von Netzwerk-IDS

(Literatur: [1] S.7-8 , [2] , [17] , [18])

Man unterscheidet zwischen zwei grundsätzlichen Arten von Vorgehensweisen bei IDS[2]-Software:

2.4.1 Signaturbasierte Angriffsentdeckung

Bekannte Angriffsarten können durch ihr Angriffsmuster erkannt werden. Diese signaturbasierten Erkennung basiert auf der Erkenntnis, dass jeder Einbruch Charakteristika aufweist, die für diesen Einbruch spezifisch sind und daher erkannt werden können.

1 Auf Deutsch übersetzt etwa: “Honigtopf”, hier also ein System, das verlockend erscheint es anzugreifen.
2 Intrusion-Detection-System

So verrät sich ein Angriff auf einen Web-Server[1] mittels des altbekannten 'phf-Fehler' (hierbei handelt es sich um einen Bug in einem CGI-Script[2], wodurch es möglich ist beliebige Kommados auf dem Web-Server auszuführen) durch den Inhalt der Netzwerkpakete. So handelt es sich hierbei um ein TCP-Paket mit Zielport 80 (auf diesem TCP-Port lauscht ein Web-Server normalerweise) und einem Inhalt der in etwa der Form:

```
GET /cgi-bin/phf?blafasel%0a<Befehl> HTTP/1.0
```

entspricht.
So was wäre eindeutig als Einbruchsversuch zu werten.
So eine Protokoll-Analyse kann dabei auch über mehr als ein Netzwerkpaket erfolgen. Doch lassen sich auf diese Weise nur bekannte Angriffe erkennen. Unbekannte Angriffsarten lassen sich so nicht erkennen.

2.4.2 Statistische Analyse des Netzwerkverkehrs

Da unbekannte Angriffsarten nicht, durch ihre Angriffssignatur, erkannt werden können, gibt es auch Verfahren, die auf statistische Auswertung des Netzwerkverkehrs basieren.
Dazu lernt das IDS im Netz erstmal welcher Netzwerkverkehr 'normal', das heißt also legitim ist. Nach dieser Lernphase erkennt das IDS dann Abweichungen vom erlernten 'Normalzustand'.
Hierbei ist allerdings zu beachten, dass abweichendes Verhalten im Netzwerkverkehr nicht zwangsläufig ein Angriff bedeutet, sondern es sich dabei auch um ein völlig legitimes Verhalten handeln kann, das einfach nur bisher (und damit in der Lernphase) noch nicht aufgetreten war und daher zum Profil des als 'normal' geltenden Verhaltens hinzuzufügen ist.
Im Allgemeinen muss bei diesen Verfahren der Administrator mit Hilfe von Regeln festlegen, was als Abweichung vom 'Normalen' zu gelten hat. Hierbei sind verschiedene Kriterien möglich:
Ein Beispiel wäre erhöhte Netzwerklast. Ein anderes Beispiel wäre die Verwendung von ungewöhnliche Übertragungsprotokollen z.B. IPX[3]-Pakete in einem Netz, das sonst nur die Protokolle der TCP/IP-Protokollfamilie[4] verwendet.
Die eigentliche Herausforderung dieser Systeme stellt dabei vor allem die Definition normalen Netzwerkverkehrs und das Einstellen der Grenzwerte, ab der der Netzwerkverkehr als nicht mehr 'normal' zu gelten hat und damit Alarm auszulösen ist.

1 Web-Server sind die Server, auf die man zumeist durch Eingabe einer URL in den Browser der Form http://www.firma.de zugreift.
2 CGI-Script: CGI (Common Gateway Interface) ist eine von mehreren Techniken, um auf einem Webserver nicht nur fertig abgelegte Seiten anzubieten, sondern auch Seiten bei jedem Aufruf neu erstellen zu lassen. Ein CGI-Script ist ein Programm, das in einer Scriptsprache (z.B. Perl) geschrieben ist und die Aufgabe des Neuerstellens für den Webserver übernimmt.
3 IPX ist eins der Protokolle der Novell-Netware-Software
4 TCP/IP besteht insgesamt aus TCP , UDP , ICMP und IP als Unterbau.

2.5 Mögliche automatische Reaktionen eines Netzwerk-IDS

(Literatur: [1] S.10, [17])

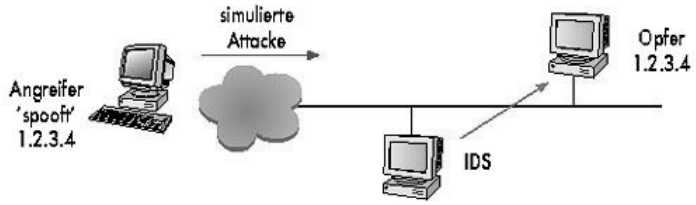
Typische automatische Reaktion eines Netzwerk-IDS auf einen erkannten Angriff stellt zumeist das Ändern von Regeln in Firewalls[1] oder Access-Control-Listen[2] in Routern dar. Dies geschieht mit dem Ziel, den zweifelhaften Netzwerkverkehr abzuschalten. Diese Vorgehensweise bezeichnet man als 'Active Security Policy'.

Weiterhin kann ein IDS die Angriffsaktivitäten an andere Rechner berichten, die von diesem Angriff ebenfalls betroffen sind.

2.6 Gefahren automatischer Aktionen eines Netzwerk-IDS

(Literatur: [17])

Die im letzten Unterkapitel aufgezählten Möglichkeiten sind zunächst verlockend, da sie scheinbar die Arbeit des Administrators stark vereinfachen, da das Netz ohne menschliches Zutun sich selbst schützt.



(Quelle: [17] , S.188)

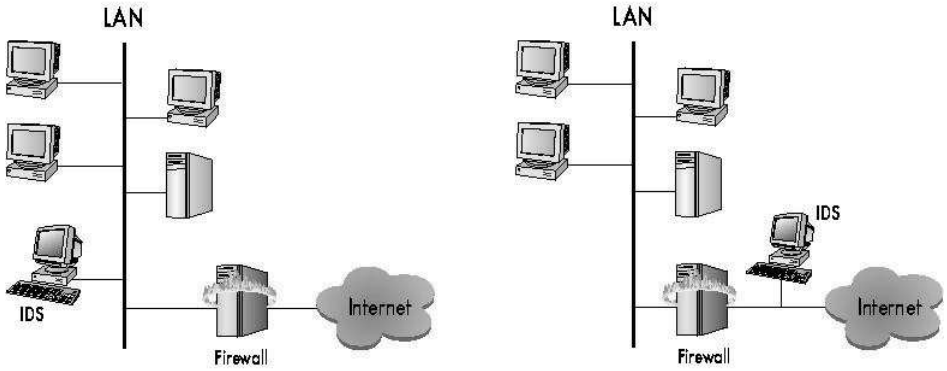
Abbildung 2.2 : Das IDS beendet fälschlicherweise Verbindungen des legitimen Inhaber der IP 1.2.3.4 aufgrund der vom Angreifer gefälschten Pakete.

Doch handelt man sich, unter Umständen, durch die vollautomatischen Aktionen neue Gefahren ein z.B. neue Formen von Denial-of-Service-Attacken[3] (so können vorgetäuschte Angriffe, mit falschen Absenderadressen[4], die Einleitung von Sicherheitsmaßnahmen bedeuten, die den normalen Betrieb stark behindern, siehe auch Abbildung 2.2).
Außerdem stellt sich die Frage, ob es für die Stabilität des Sicherheitssystems wirklich vorteilhaft ist, wenn IDS und Administrator munter an der Konfiguration von Router und Firewall rumeditieren.

1 "Brandmauer" , reglementiert den Netzwerkverkehr, der aus dem lokalen Netz raus, bzw. in das lokale Netz reingeht.
2 "Zugriffskontrolllisten"
3 Als Denial-of-Service-Attacke (kurz auch DoS genannt) bezeichnet man jegliche Attacken mit dem Ziel den normalen Betrieb eines Systems zu verhindern, oder zumindest massiv zu stören.
4 Bei TCP/IP nennt man sowas IP-Spoofing

2.7 Wo im Netz platziert man sein Netzwerk-IDS?

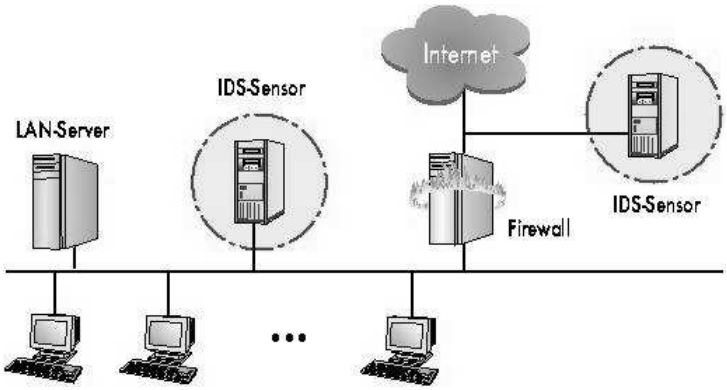
(Literatur: [17], [18])



(Quelle [17], S. 188)

Abbildung 2.3:
Rechts ein IDS, das alle Angriffsversuche erkennt, auch solche die die Firewall schon blockt. Links ein IDS das nur Angriffe erkennt die durch die Firewall durchkamen.

Je nach dem, was man mit dem Netzwerk-IDS erreichen will, gibt es verschiedene günstige Orte das IDS aufzustellen:
Will man vor allem wissen, welche bedrohlichen Aktivitäten von außen auf das eigene lokale Netz[1] eingehen, so platziert man sein IDS vor der Firewall (siehe Abbildung 2.3 rechts).



(Quelle: [18] , S. 213)

Abbildung 2.4:
IDS-Sensoren vor und hinter der Firewall.

Im lokalen Netz (also hinter der Firewall vom Internet aus gesehen, siehe Abbildung 2.3 links) sieht das IDS nur Aktivitäten, die die Firewall durchgelassen hat. Allerdings können hinter der Firewall auch suspekte Aktivitäten festgestellt werden, die aus dem eignen lokalen Netz kommen. Eine Firewall kann gegen Angreifer von innen nix tun, ein IDS kann solche Aktivitäten registrieren.
Auf einem Server, oder im Teilnetz bei einem Server, kann ein IDS die Zugriffe, auf z.B. den eignen Webauftritt kontrollieren und so Überraschungen vermeiden helfen.

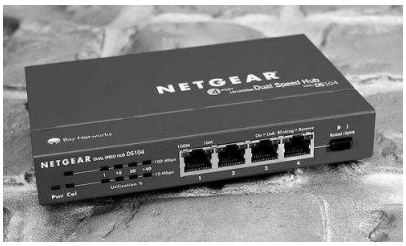
1 LAN – Lokal Area Network

Ist das Netz größer, so kann es auch Sinn machen, mehr als ein IDS-Sensor aufzustellen und die Meldungen dieser Sensoren zentral zu sammeln und auszuwerten.

2.8 Technische Voraussetzungen des Netzes für den Einsatz eines Netzwerk-IDS

(Literatur: [17], [18], [19] Kasten: Unfallverhütung – von Verstärkern und Schaltern)

Wichtig für den Betrieb eines Netzwerk-IDS ist es, dass das IDS den interessierenden Netzwerkverkehr auch wirklich zu sehen bekommt. Dies ist nicht in jedem Fall gegeben.
Kein Problem ist es, den interessanten Netzwerkverkehr durch das IDS überwachen zu lassen, wenn die Vernetzung mittels Hub[1] geschied. Dann braucht das IDS die Netzwerkkarte des Rechners, auf dem es installiert ist, nur in den so genannten "Promiscous Mode"[2] versetzen und kann dann den ganzen Netzwerkverkehr empfangen.



(Quelle [19] S. 107)

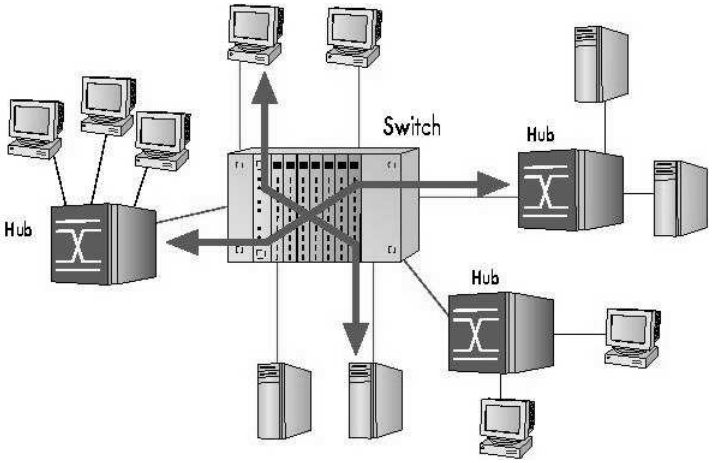
Abbildung 2.5: Ethernet-Hub

(Quelle [19] , S.107)

Abbildung 2.6: Ethernet-Switch

Bei Einsatz eines Switch[3] dagegen ist es nicht mehr so einfach, den Netzwerkverkehr durch das IDS mitlauschen zu lassen. Ein Switch, der aus seiner Lernphase raus ist, leitet Netzwerkverkehr nur noch in das Teilnetz[4] weiter, an dem das Gerät angeschlossen ist, das mit seiner Hardware-Zieladresse angesprochen wurde.
Daher ist es nicht so einfach möglich, dass ein Rechner, auf dem das IDS läuft, den ganzen Netzwerkverkehr mitlauscht.
Eine für das IDS günstige Ausnahme ist es wenn der Switch einen Monitoring-Port bzw. Promiscous-Mode-Port hat. An so einer speziellen Schnittstelle lässt sich der ganze Netzwerkverkehr mitlauschen. Aus zweckmäßigen Gründen ist so eine Schnittstelle, dann natürlich die Stelle, an der man den Rechner anschließt, auf dem der IDS-Sensor läuft.

1 Hub oder auch Repeater genannt: Netzwerkverteiler der grundsätzlich alle Signale die auf einer Schnittstelle reinkommen an alle anderen Schnittstellen ausgibt.
2 Netzwerkkarten verwerfen normalerweise alles was nicht für den eignen Rechner bestimmt ist oder eine Rundsendung (Broadcast) ist.
3 Switch oder auch Bridge genannt: intelligenter Netzwerkverteiler der allen Netzwerkverkehr der kein Broadcast ist nach Möglichkeit nur auf der Schnittstelle ausgibt an dem der Empfänger angeschlossen ist.
4 Man kann an einem Switch auch Hubs oder weitere Switche als Netzwerkverteiler anschließen.



(Quelle [19] S. 110)

Abbildung 2.7: Großes Ethernet mit zentralem Switch und Hubs als Unterverteiler

2.9 Historische Entwicklung der Intrusion Detection Systeme

(Literatur: [15] S.27)

Begonnen hat die Einbruchsentdeckung damit, dass System-Administratoren an einer Konsole saßen und Benutzeraktivitäten beobachtet haben. Sie haben dabei Einbrüche in ihr System entdeckt, in dem sie ungewöhnliche Aktivitäten bemerkten.
Diese Lösung war spontan und nicht skalierbar.
Der nächste Schritt der Einbruchsentdeckung in Computersysteme waren Logs die von den System-Administratoren nachträglich auf unübliches oder schädliches Verhalten durchgesehen wurden. In den späteren 1970ern und frühen 1980ern wurden diese Logs typischerweise auf Endlospapier gedruckt. Oft wurde so ein Stapel 4 bis 5 Fuß hoch nach einer typischen Woche. So einen Stapel zu untersuchen war sehr zeitaufwendig und wurde daher vor allem zum Feststellen der Ursache eines Sicherheitsproblems durchgeführt. Einen Angriff festzustellen, während er durchgeführt wird, bestand nur geringe Hoffnung.
Nachdem Speicherplatz billiger geworden war, wurden die Logs gespeichert und Forscher entwickelten Programme zur Analyse dieser Daten. Noch immer war die Analyse langsam und rechenzeitintentsiv. Daher wurde so eine Analyse meist Nachts, wenn die Systemlast gering war, durchgeführt. Weiterhin wurden daher die meisten Einbrüche erst nachträglich festgestellt.
Erst in den frühen 1990ern wurden Echtzeit-Intrusion-Detection-Systeme entwickelt, die Logs analysieren konnten während sie entstehen. Dies ermöglichte zum ersten Mal Reaktionen auf Einbruchsversuche in Echtzeit und manchmal auch Angriffsverhinderung.
Weitere Schritte waren Intrusion Detection Systeme für größere Computernetzwerke.

Kapitel 3

Überblick über bekannte Verfahren und Programme

Nachdem ich in Kapitel 2 einige Grundlagen der Netzwerk-Intrusion-Detection-Systeme erläutert habe, gehe ich ihn diesem Kapitel näher auf verschiedene bekannte Verfahren ein und stelle fertige Produkte, zur Entdeckung von Einbruchsversuchen in Computernetzwerke, kurz vor. Dieser Überblick erhebt aufgrund der Vielzahl an IDS-Lösungen keinen Anspruch auf Vollständigkeit. Ich denke aber dass ich die wichtigsten Netzwerk-IDS aufgezählt habe.

3.1 Programme aus der Forschung

Beginnen will ich diesen Überblick mit Programmen die ihren Ursprung aus dem Bereich der Forschung haben.

3.1.1 Bro

(Literatur: [1] S.22-23)

Bro wurde vom Lawrence Livermore National Laboratory entwickelt, zum Teil mit dem Ziel die Robustheit von Intrusion-Detection-Systemen zu erforschen. So war bei der Entwicklung, auch die Fähigkeit des IDS resistent gegen Angriffe auf sich selbst zu sein, wichtig.

Weitere Ziele die bei der Entwicklung von Bro wichtig waren, sind u.a.:

- Überwachung großen Netzwerkverkehrs:
  Es ist sehr wichtig, das auch bei sehr viel Netzwerkverkehr es nicht passiert, das Netzwerk-Pakete verworfen werden. Ein Einbrecher in das Computernetz könnte sonst die Tatsache das Pakete verworfen werden ausnutzen, indem er das Netz mit unwessendlichen Paketen flutet. Und damit provoziert das das IDS Pakete verwirft.

- Echtzeitbenachrichtigung:
  Dies wird benötigt um zeitnah auf die Aktivitäten des Einbrechers zu reagieren.

- Erweiterbarkeit des Systems:
  Die große Menge der bekannten Angriffsarten, die noch dazu ständig steigt, macht es nötig das neue Angriffsarten schnell in die Bibliothek eingetragen werden können.

Bro hat dabei eine dreistufige Hierachie an Funktionalität. In der untersten Stufe benutzt Bro ein Werkzeug namens *libpcap* um die Netzwerkpakete aufzusammeln. Dies entkoppelt die Hauptfunktion von Bro von den Funktionen des Netzwerks. Außerdem erlaubt dies das frühzeitige Verwerfen eines großen Teils der Netzwerkpakete.

Auf der nächsten Stufe der Funktionalität wird der Kopf der Netzwerkpakete auf Korrektheit überprüft. Ist der Kopf fehlerhaft so wird er nach Erstellen eines Ereignisberichts verworfen. Danach wird festgestellt ob der gesamte Inhalt des Netzwerkpakets für die weitere Analyse benötigt wird.

Die dritte Stufe der Funktionalität beinhaltet einen „Policy Script Interpreter“ der, in Form von Scripts in einer speziellen Programmiersprache, vorliegende Regeln interpretiert.

Bro kann die Datenübertragungen von 4 wichtigen Netzanwendungen überwachen: finger, ftp, portmapper und telnet.

3.1.2 HIDE

(Literatur: [16] S.77-82)

Während Bro ein Forschungs Netzwerk-IDS mit Einbruchsentdeckung per Angriffssignatur ist, handelt es sich bei HIDE um ein Netzwerk-IDS das ausschließlich über statistische Ansätze arbeitet.
Bei HIDE handelt es sich um ein hierarchisch arbeitendes System, dessen Komponenten sowohl verteilt als auch als Einzelsystem arbeiten können.
Es analysiert den Netzwerkverkehr, System-Logs und Hardwaremitteilungen.
Dabei entdeckt es Muster, von abweichenden Aktivitäten, anhand des Referenzmodels.
Dieses korrespondiert mit den erwarteten Benutzeraktivitäten. Dabei kann jeder Parameter einzeln, oder über in kombinierenden Gruppen, über ein Neuronales Netz analysiert werden. Aus dem Analyseergebnis werden dann Alarmmeldungen generiert. Außerdem wird anhand dieser Analyse das Systemmodel aktualisiert.
HIDE besteht normalerweise aus verschiedenen verteilten Komponten den IDA (Intrusion/Fault Detection Agents). Jeder dieser IDA besteht wiederum aus dem Tester , dem Ereignisvorverarbeiter , dem statistischen Prozessor, dem neuronalen Netzwerkklassivizierer und dem Nachbearbeiter.
Der Tester empfängt den Netzwerkverkehr und wandelt diesen in abstrakte statistische Variablen. Reports darüber werden periodisch an den Ereignisvorverarbeiter weitergeleitet.
Dieser empfängt nicht nur diese Reports, sondern auch die Reports von weiter unten gelegenden IDAs. Der Ereignisvorverarbeiter konvertiert diese Informationen in das Format das das statistische Modell benötigt.
Der statistische Prozessor vergleicht das Referenzmodel mit den Reports vom Erreignisvorverarbeiter und liefert den Eingabevektor für den neuronalen Netzwerkklassivizierer.
Der neuronale Netzwerkklassivizierer schließlich entscheidet ob der Netzwerkverkehr

normal ist oder nicht.
Der Nachbearbeiter generiert Reports für weiter oben gelegene IDAs und zeigt die Resultate auf einer Benutzerschnittstelle an.

3.2 Kommerzielle Intrusion Detection Systeme

(Literatur: [1] S.24-30, [4] , [5] , [6] , [7] , [8] , [20] , [22])

Programme die von Firmen zum Kauf angeboten werden gibt es ebenfalls.

3.2.1. NetProwler™

NetProwler™ wird üblicherweise im Verbund mit Intruder Alert (ein IDS auf Rechnerebene[1]) verkauft. NetProwler selbst ist dabei die Netzwerk-IDS-Komponente des Komplett-Pakets.

NetProwler testet die gesnifften Netzwerkpakete gegen vorher definierte Signaturen[2]. Diese Angriffssignaturen können während der Laufzeit des Systems aktualisiert werden. Viele Angriffssignaturen sind bereits im Programmpaket enthalten, es ist dem Benutzer des Systems allerdings möglich eigne Signaturen den vordefinierten hinzuzufügen. Hierzu enthält das Programm einen Signatur-Definititions-Assistenten.

Die Angriffs-Signaturen die von NetProwler unterstützt werden beinhalten Signaturen die Denial-of-Service-Attacken, unautorisierte Zugriffe, Tests auf ausnutzbare Sicherheitslücken, verdächtige oder schädliche Aktivitäten, sowie Aktivitäten die gegen die firmeneigene Sicherheitsregeln verstößt, erkennen.

NetProwler erlaubt auch automatische Reaktionen auf Angriffe. Zu den möglichen Reaktionen gehört neben Mitloggen einer Sitzung, das Beenden einer Sitzung[3] auch das Benachrichtung des Sicherheitspersonals über E-Mail , Pager oder ähnlichem.

3.2.2 NetRanger™

NetRanger™ von Cisco Systems in ein Netzwerk-IDS, das in einer früheren Version als *Cisco Secure Intrusion Detection System* bekannt war.

Es arbeitet in Echtzeit und ist zusammengesetzt aus Sensoren und einer oder mehrere Überwachungsstationen. Die Sensoren sind dabei durch Hardware von Cisco aufgebaut, während die Überwachungsstation auf Software basiert.

Die Sensoren werden dabei an strategischen Punkten im Netzwerk eingebaut, an denen Netzwerk-Verkehr durchkommt. Eine Analyse der Netzwerkpakete ist sowohl anhand des Paketkopfes als auch des Inhaltes möglich. Eine Analyse von mehreren

1 Dies nennt man auch „Host-Based-Intrusion-Detection-System“
2 Ist damit ein Netzwerk-IDS das auf die signaturbasierte Angriffsentdeckung setzt (siehe Kapitel 2.4.1).
3 Siehe auch Kapitel 2.6 (Gefahren automatischer Reaktionen)

Paketen, um Angriffe zu entdecken die über mehr als ein Paket gehen, ist auch möglich. Die Sensoren benutzen ein Regelbasiertes Expertensystem um die Art des Angriffs festzustellen.

Neben vielen vordefinierten Angriffssignaturen erlaubt es NetRanger auch dem Benutzer eigne Signaturen hinzuzufügen. Als Reaktionen auf eine Angriff gibt es neben der Möglichkeit des Loggings, auch das Beenden einer Sitzung , sowie das Abweisen zukünftigem Netzzugriffs.

Die Überwachungsstation erlaubt die zentrale Steuerung des NetRanger-Systems. Dazu gehört z.B. das Installieren neuer Signaturen in die Sensoren, aber auch das Sammeln und Analysieren von Sicherheitsdatensätzen. Außerdem kann die Überwachungsstation auch automatische Benachrichtungen an das Personal per E-Mail versenden. Benutzerdefinierte Benachrichtigungen sind auch möglich.

3.2.3 Centrax™

(früher als Entrax™ bekannt).

Entrax/Centrax war zu Anfang seiner Entwicklung ein reines Host-Based-Intrusion-Detection-System. Im Laufe der Zeit wurde die Funktionalität eines Netzwerk-IDS immer populärer und so enthält Centrax nun auch Funktionen zur Netzwerk-Überwachung.

Centrax besteht aus zwei Hauptkomponenten: Die Kommandokonsole und ein „Target Agent“. Dies ist vergleichbar mit der Überwachungsstation und den Sensoren im NetRanger-System. Der „Target Agent“ kann dabei allerdings einer von zwei Typen sein. Entweder sammelt er Informationen auf Rechnerebene oder auf Netzwerkebene.[1]

Die „Target Agents“ die auf Rechnerebene arbeiten befinden sich dabei direkt auf den zu überwachenden Rechnern. Aus Geschwindigkeitsgründen ist der Netzwerk-“Target-Agent“ als einzelner Rechner realisiert.

Von den auf den „Target Agents“ auf Rechnerebene werden über 170 Signaturen (z.B. Viren und Trojaner) unterstützt, während die „Target Agents“ auf Netzwerkebene 40 Signaturen unterstützen.

Die Kommandokonsole läuft unter Windows NT, während die „Target Agents“ auf Rechnerebene sowohl für Windows NT, als auch Solaris zu haben sind.
Der Netzwerk-“Target-Agent“ läuft nur unter Windows NT.

3.2.4 RealSecure

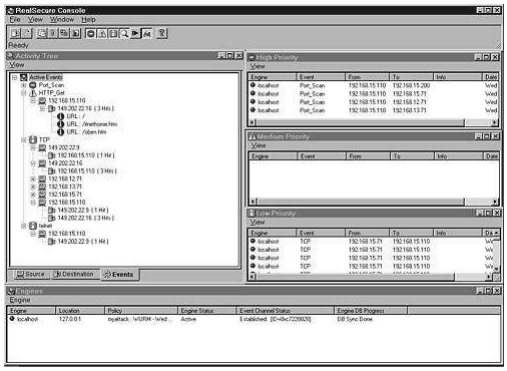
RealSecure ist ein weiteres Echtzeit IDS. Es besteht aus 3 Komponenten. Einer Netzwerkbasierten Erkennungs-Engine , einer Rechnerbasierten Erkennungs-Engine, sowie ein Administrationsmodul.

1 Centrax arbeitet schließlich in beiden Hierachiestufen.

Auch RealSecure ist zu automatischen Reaktionen fähig. Sie können darin bestehen das Netzwerkverbindungen beendet werden, aber auch in Benachrichtungen an den Administrator mittels E-Mail oder Pager. Auch ein Mitschneiden der gesamten Netzwerksitzung , oder eine Konfigurationsänderung in der Firewall ist möglich. Unterstützt werden die Firewalls Checkpoint Firewall-1 sowie Lucent Managed Firewall.

Weitere Benutzerdefinierte Aktionen sind auch möglich.
Alarmmeldungen erscheinen ebenfalls auf dem Administrationsmodul.
Die Kommunikation zwischen den Engines und dem Adminitrationsmodul findet verschlüsselt statt. Aus Geschwindigkeitsgründen wird empfohlen das Adminstrations-modul auf einem anderen Rechner zu installieren, als die Engines[1].

Die Rechnerbasierten Erkennungs-Engines analysieren Logfiles über festgestellte Angriffe und stellen fest ob der Angriff erfolgreich war oder nicht, sowie stellen weitere forensische Informationen zusammen, die aber nicht in Echtzeit verfügbar sind.



(Quelle: [20] S. 191)

Abbildung 3.1: RealSecure-Adminmodul

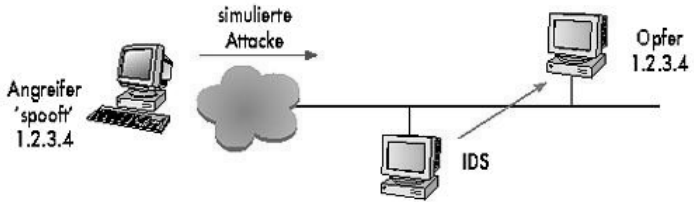
Verfügbar ist RealSecure für Windows NT , Solaris und Linux. Leider wird es für Linux nicht mehr weiterentwickelt.

RealSecure enthält vordefinierte Regeln die an die eignen Bedürfnisse angepasst werden können. Echte zusätzliche Regeln lassen sich nur mit Zusatzsoftware erstellen.

Im Test der c´t ([20]) gelang es sehr schnell Regeln aufzustellen, die erfolgreich einen Rechner der portscannt durch Änderung der Firewallkonfiguration auszusperren. Das Zurücknehmen dieser Netzblockade dagegen war aufwendiger, es musste nach Änderung der Konfiguration die Firewall rebootet werden.
Dabei ist auch aufgefallen, das sich nur schwer erkennen lässt, das die Konfiguration der Firewall durch das IDS geändert wurde. Nur in den Firewalllogs fand sich ein Hinweis, dass was geändert wurde. Nicht aber, was geändert wurde.

Leider lässt es sich ([20]) für DoS-Attacken missbrauchen, da gefälschte Absenderadressen nicht als solche erkannt werden. Ein Portscan mit gefälschtem Absender und erwünschte Verbindungen können blockiert sein.

1 Im C´t-Artikel ([20]) wird geschildert was passieren kann, wenn man Engine und Adminmodul auf demselben Rechner installiert: Das Adminmodul wird bei Installation auf einem Rechner der auch Engine ist unbedienbar, wenn z.B. durch einen Portscan erhöhte Netzlast herrscht.



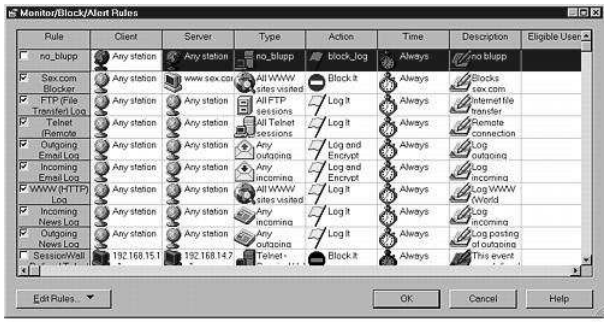
(Quelle: [17] , S.188)

Abbildung 3.2 : Das IDS beendet fälschlicherweise Verbindungen des legitimen Inhaber der IP 1.2.3.4 aufgrund der vom Angreifer gefälschten Pakete.

3.2.5 Session-Wall-3

Session-Wall-3 läuft unter Windows 95/98 und auch unter Windows NT und besteht nur aus einer einzigen Komponente, die sowohl Adminconsole als auch die Untersuchung des Netzwerkverkehrs durchführt. Session-Wall-3 lauscht dabei an allen Netzwerkschnittstellen die der Rechner hat auf dem es installiert ist.
Neben vordefinierten Signaturen lässt sich ebenfalls nur durch Zusatzsoftware eigne Signaturen erstellen.
Session-Wall-3 kann wenn es einen Angriff erkennt Benachrichtigungen per E-Mail, Pager oder sogar SNMP versenden. Leider erkannte es im c´t-Test ([20]) nicht alle Portscans in einem ausgelasteten Fast-Ethernet (100MBit/sek).

Neben der Funktionalität als Netzwerk-IDS beinhaltet Session-Wall auch weitere Sicherheitsfunktionen zu denen neben einem E-Mail-Virenscanner, einem Scanner auf bösartige Java-Applets und Active-X-Controls in HTTP-Verbindungen[1] auch die Möglichkeit besteht unerwünschte URLs zu verbieten.



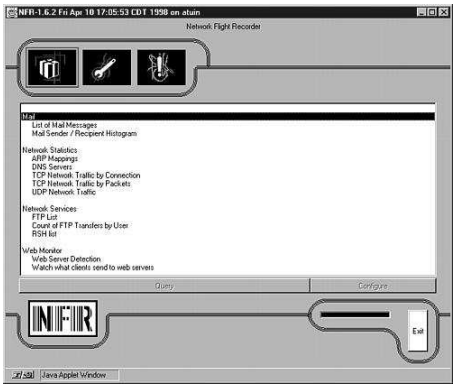
(Quelle: [20] S. 192)

Abbildung 3.3: Session-Wall-3

1 Also Verbindungen ins World Wide Web

3.2.6 Network Flight Recorder™

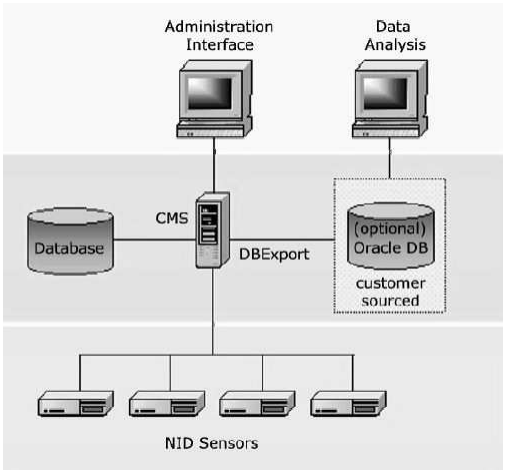
Der Network Flight Recorder diente als eine Art Werkzeugkasten für die Erstellung eines IDS. Lauffähig ist war es sowohl unter Solaris, als auch unter Linux.
Es konnten alle Netzwerk-Schnittstellen, des Rechners auf dem der NFR installiert war, überwacht werden. Die Analyse geschah durch in einer speziellen Programmiersprache geschriebene externe Programme.
Auch einige Module um Statistiken über den IP-Netzwerkverkehr zu erstellen waren vorhanden.
Um allerdings ein richtiges IDS aufzubauen war allerdings einiges an Eigenarbeit nötig.
Der NFR war für die nicht kommerzielle Verwendung frei.



(Quelle: [20] S. 193)

Abbildung 3.4: Network-Flight-Recorder (alt)

Seit einiger Zeit wird anstelle von der Firma NFR ein rein kommerzieller Nachfolger produziert. Dieser wird als NFR® Network Intrusion Detection (NFR NID) bezeichnet und überwacht das Netzwerk in Echtzeit. Dabei wird sowohl mit Signatur basierter Network Intrusion Detection als auch mit der Erkennung von abweichendem Verhalten gearbeitet.
NFR NID besteht dabei aus einem ganzen System von Netzwerkkomponenten.
Die Sensorstationen gibt es dabei sowohl als Hardwaresensoren (mit diverser Netzwerk-Schnittstellenbestückung, angefangen von 2*10/100Mbps Ethernet, über 3*10/100Mbps Ethernet , über 1*1000Mbps +2*10/100Mbps Ethernet bis zu 2*1000Mbps + 2* 10/100 Mbps Ethernet. Einer der Netzwerk-Schnittstellen ist dabei zum managen des Sensors, der Rest kann für die Netzwerküberwachung benutzt werden.)



(Quelle: [4])

Abbildung 3.5:

Komponenten des NFR NID

als auch als „Nur Software“ Sensoren.
Um das IDS zu administrieren, besteht das Komplettsystem auch noch aus einer Administratorschnittstelle als eigenständigen Rechner. Meldungen und Alarme werden auf einem oder mehreren davon getrennten zentralen Managment Servern abgelegt. Als optionale Komponente ist eine Oracle Datenbank mit Datenanalysestation möglich.

(Quelle: [4])

Abbildung 3.6:

NFR Hardwaresensor

Die Kommunikation zwischen den Komponenten des Systems läuft verschlüsselt ab und die Sensor-Software läuft direkt von der Produkt-CD , welche ein abgespecktes Unix-Betriebsystem enthält, welches nur die aller nötigsten Systemteile enthält um resistent gegen Angriffe zu sein.

Neue Angriffssignaturen können sowohl einfach selbst erstellt werden, als auch, von einer Webseite des Herstellers, heruntergeladen werden.

3.2.7 AirDefense

AirDefense IDS ist ein speziell für die kabellose Vernetzung mittels des 802.11 Standards (Wireless LAN[1]) entwickelte IDS[2] und Sicherheitslösung. Die Sensoren werden dabei in die Nähe eines Acess-Points[3] platziert, so das sie den ganzen kabellosen Netzwerkverkehr abgreifen können und an den Analyseserver senden, wo er dann in Echtzeit analysiert wird.

3.2.8 nPatrol

nPatrol ist ein adaptives Intrusion Detection System, das nicht nur das Ausnutzen bekannter Schwachstellen erkennen soll, sondern auch neue Arten von Angriffen. Dazu enthält es sowohl eine signaturbasierte Analyse als auch eine statistische Analyse auf Abweichungen vom Normalzustand. Zusätzlich zu der Funktion als Netzwerk-IDS enthält nPatrol auch einen Rechner basierten Datei Intrigritätstest und eine Dateiüberwachung als Rechner basiertes IDS.
Die verwendeten Angriffssignaturen können sowohl automatisch aktualisiert werden, als durch benutzerdefinierte Signaturen ergänzt werden.
nPatrol kann auf erkannte Angriffe automatisch mit einem TCP Reset reagieren. Das heißt mit dem Abbrechen der Verbindung die der Angriff darstellt.
Die Erkennung von Abweichungen vom Normalzustand erfordert das nPatrol eine Zeit lang lernt welcher Netzwerkverkehr als normal anzusehen ist. Es können dabei verschiedene Parameter von TCP, UDP und ICMP-Paketen analysiert werden.

3.2.9 PureSecure

PureSecure von Demarc Security Inc. ist eine Sicherheitslösung deren Netzwerk-IDS Komponente Snort (siehe Kapitel 3.3.1 und 4) mit einer zentralen Managment-Konsole verbindet. Außerdem enthält PureSecure auch eine Komponente die ein rechnerbasiertes IDS (als System Intrigritäts Test) ist.

3.2.10 Dragon IDS

Dragon IDS von Portcullis Computer Security Ltd ist ein Netzwerk-IDS das auf Unix-Systemen läuft. Es überwacht den Netzwerkverkehr und sucht darin nach Anzeichen für Computer-Kriminalität , Netzwerk Angriffen , Netzwerk-Mißbrauch und Anomalien. Dragon IDS kann wenn es etwas erkannt hat dies per Pager, E-Mail , SNMP und syslog melden und auch direkt dafür sorgen das das Bemerkte endet. Aufzeichnungen für eine spätere Analyse erfolgen ebenfalls.

1 Wireless-LAN , Wavelan u.ä. = kabellose lokale Netzwerke
2 IDS=Intrusion Detection System
3 Access-Point (Übergang zwischen einem kabelgebundenem Netz und einem kabellosen Netz, der auch als zentraler Netzwerkverteiler in kabellosen Netzwerken dient).

Alle Alarmmeldungen können in eine zentrale Datenbank geschrieben werden um verteilte Angriffsinformationen zu korrelieren.
Administriren lassen sich alle Komponenten des Dragon IDS per Kommandozeile und Weboberfläche.
Die Angriffssignaturen liegen als ASCII-Text vor und sind vollständig dokumentiert. Sie können daher leicht bearbeitet und aktualisiert werden.

3.3. Freie (z.B. „Public Domain“) Intrusion Detection Systeme

(Literatur [9] , [10], [11] , [12] , [18] ,)

Hier noch ein kurzer Überblick über freie Intrusion Detection Systeme für Netzwerke, bevor ich im nächsten Kapitel diese Testen werde.

3.3.1 Snort

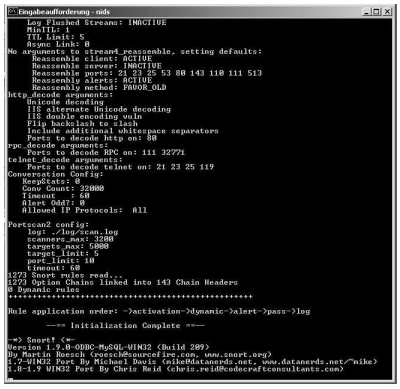


Abbildung 3.7:
Snort (Konsole)

Snort ist ein Netzwerk-IDS das kostenlos für Windows und Unix bzw. Linux im Internet zu bekommen ist (http://www.snort.org).
Seine Entwickler bezeichnen es als 'lightweight network intrusion detection system' .
Snort arbeitet hauptsächlich mit Signaturen bekannter Angriffe. Neben Protokollheadern lassen sich auch Nutzdaten auf verdächtige Eigenschaften analysieren. Neben der einfachen Möglichkeit eigene Regeln zu erstellen, liegen etwa 1000 fertige Regeln bereits fertig im Snort-Download-Paket bzw. auch in einem extra Download-Paket. Zusätzlich zu Paketeigenschaften können auch weitere verdächtige Aktivitäten von Snort erkannt werden. So löst der Portscan-Preprozessor bei Überschreitung einer eingestellten Anzahl von Verbindungsversuchen von einer IP-Adresse Alarm aus. Gemeldet werden dabei aber auch spezielle Pakettypen, die üblicherweise von so genannten Stealth-Scannern verwendet werden.
Neben der Verwendung von Angriffssignaturen enthält Snort seit einiger Zeit auch eine Erweiterung die statistische Analysen erlaubt und damit bei Abweichung des Netzwerkverkehrs von der Normalität Alarm auslöst.
Für Snort gibt es auch allerlei Zusätze mit denen man sich die Arbeit sehr erleichtern kann. So gibt es z.B. den *IDS Policy Manager* um in verteilten Umgebungen Snort IDS-Sensoren zu managen. Dies geschieht in dem die Textdateien zur Konfiguration sowie die Snort-Regeln-Dateien in ein einfach zu

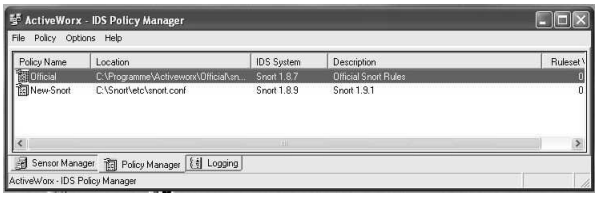


Abbildung 3.8:
IDS Policy Manager

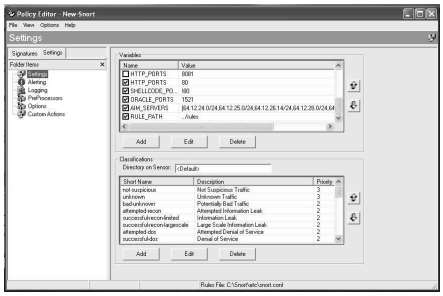


Abbildung 3.9:
Policy Editor
(gehört zum IDS Policy Manager)

benutzende grafische Oberfläche verpackt werden. Auch die sonstige Konfiguration die normalerweise über die Komandozeile bzw. eben über eine Textdatei erfolgt kann auch über eine grafische Oberfläche geschehen. Dazu gibt es z.B. IDS-Center.

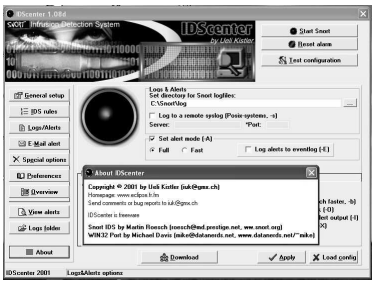
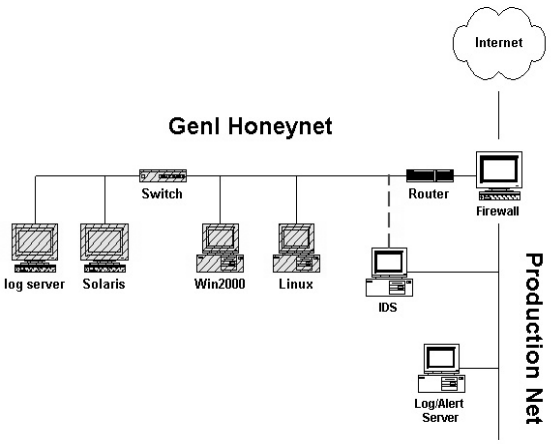


Abbildung 3.10: IDS Center

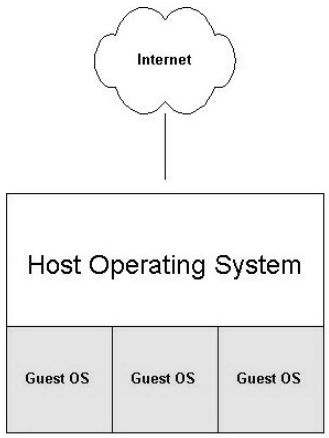
Neben solchen Konfigurations und Managmenthelfern gibt es auch ein Honeynet-Projekt auf Basis von Snort. Dieses ist auf http://www.honeynet.org zu finden.

Als Honeynet bezeichnet man dabei eine spezielle Form von Honeypot (siehe Kapitel 2.3.4). Während ein Honeypot normalerweise ein einzelner Rechner ist, der für Angreifer verlockend sein soll, ist ein Honeynet normalerweise ein ganzes Netzwerk für diesen Zweck. Es ist aber auch möglich ein so genanntes virtuelles Honeynet aufzustellen. Dieses ist ein auf einem einzelnen Rechner simuliertes Honeynet (das dabei scheinbar aus vielen Rechnern besteht). Das Honeynet-Projekt bietet ein eigenes Regelset für Snort



(Quelle: http://www.honeynet.org/papers/honeynet/)

Abbildung 3.11: Honeynet



(Quelle: http://www.honeynet.org/papers/virtual/)

Abbildung 3.12: Virtualisierung für ein virtuelles Honeynet

an. Dies ist allerdings nur für die Verwendung in einem Honeynet geeignet. Nicht aber für den allgemeinen Schutz eines normalen Netzwerks. Hierauf wird auch auf der Webseite zu Snort ([9]) hingewiesen.

3.3.2 Panoptis

Panoptis ist ein Projekt das mit dem Ziel gestartet ist Denial of Service[1] und Distributed Denial of Service[2] Angriffe zu stoppen. Im Augenblick benachrichtigt Panoptis erstmal nur den Administrator. Es existiert aber schon inaktiver Code der DoS/DDoS-Angriffe stoppen soll.

1 kurz auch : DoS , hiermit sind alle Angriffe gemeint die darauf Abzielen den regulären Betrieb eines Rechnersystems durch sinnlose Datenpakete zu stören.

2 kurz auch: DDoS, hiermit sind DoS-Angriffe gemeint die von sehr vielen (eben „verteiltes“ DoS) Rechnern ausgehen und damit auch Rechner in die Knie zwingen können die sehr leistungsfähig bzw. eine sehr leistungsfähige Netzanbindung haben.

3.3.3 Pakemon IDS

Pakemon IDS (Paket Monster) ist ein Netzwerk-IDS das entwickelt wurde um IDS-Komponenten auf OpenSource-Software-Basis zu teilen. Leider wird es offenbar nicht mehr weiterentwickelt (die letzte verfügbare Version 0.3.1 ist vom 7.Januar 2001) Diese kann allen Netzwerkverkehr überwachen und diesen auf das Vorkommen von Datenmustern durchsuchen. Außerdem kann Pakemon Sitzungslogs und Zusammenfassungen ausgeben. Pakemon läuft auf Unixsystemen und es sind 248 Signaturen für Pakemon verfügbar.
Auf der Webseite des Autors ist unter
http://web.sfc.keio.ac.jp/~keiji/backup/ids/petitmon/
mit Petitmon auch ein Programm verfügbar das nur den Netzwerkverkehr mitschneiden kann.

3.4. Zusammenfassung über die kurz vorgestellten Programme

	Netzwerk-IDS aus der Forschung		Kommerzielle Netzwerk-IDS			
Name	Bro	HIDE	NetProwler	NetRanger	Centrax	RealSecure
Verwendet Angriffssignaturen	Ja	Nein	Ja	Ja	Ja	Ja
Verwendet Statistische Analyse	Nein	Ja	Nein	Nein	Nein	Nein
Eigne Angriffs-Signaturen möglich	Ja	Nein, da reiner Statistikansatz	keine Angaben	Ja	keine Angaben	Nur mit Zusatz-Software
Automatische Reaktionen	keine	keine	Beenden der Verbindung	Beenden der Verbindung	keine	Aussperren des Angreifers

Tabelle 3.1: Zusammenfassung der vorgestellten Programme (Teil 1)

Wie aus dieser, und den beiden nachfolgenden Tabellen, zu entnehmen ist, benutzen die allermeisten, für den allgemeinen Schutz eines Netzwerkes entwickelten, Netzwerk-IDS Angriffssignaturen um ihre Entscheidung, über die Existens eines aktuell im Gange befindlichen Angriffes, zu fällen. Einige benutzen zusätzlich eine statistische Analyse um auch unbekannte Angriffe zu bemerken. Eine alleinige Entscheidung aufgrund statistischer Parameter des Netzwerkverkehrs, kommt bei den hier vorgestellten Netzwerk-IDS Programmen, zum allgemeinen Schutz eines Netzwerkes, nur bei HIDE vor. Als häufigste automatische Reaktion, bei den Programmen die zu automatischen

	Kommerzilele Netzwerk-IDS					
Name	Session-Wall-3	Network Flight Recorder (alt)	Network Flight Recorder (neu)	AirDefence	nPatrol	PureSecure
Verwendet Angriffssignaturen	Ja	Ja	Ja	Speziallösung für Wireless LAN	Ja	Lösung basiert auf Snort
Verwendet Statistische Analyse	Nein	Nein	Ja		Ja	
Eigne Angriffs-Signaturen möglich	Nur mit Zusatz-Software	Ja	Ja		Ja	
Automatische Reaktionen	keine	eventuell mit Zusatzmodul	keine		Beenden der Verbindung	

Tabelle 3.2: Zusammenfassung der vorgestellten Programme (Teil 2)

Reaktionen in der Lage sind, kommt offenbar das reine Abbrechen der aktuellen Netzwerksitzung, die gerade einen Angriff durchführt, zum Einsatz. Nur ein einziges Programm blockiert einen vermeintlichen Angreifer ganz.

	K. NIDS	Freie Netzwerk-IDS		
Name	Dragon IDS	Snort	Panoptics	Pakemon
Verwendet Angriffssignaturen	Ja	Ja	Speziallösung zur Abwehr von DoS und DDoS-Angriffen	Ja
Verwendet Statistische Analyse	Ja	Ja, optional Möglich		Nein
Eigne Angriffs-Signaturen möglich	Ja	Ja		Ja
Automatische Reaktionen	Beenden der Verbindung	Beenden der Verbindung		keine

Tabelle 3.3: Zusammenfassung der vorgestellten Programme (Teil3)

Kapitel 4

Test interessanter freier IDS-Software

Nachdem ich in Kapitel 3 einige Netzwerk-Intrusion Detection Systeme kurz vorgestellt habe, werde ich in diesem Kapitel einige vorgestellte kostenlose Netzwerk Intrusion Detection Systeme testen. Dabei werde ich zunächst das für diesen Test benutzte Test-Netzwerk beschreiben, bevor ich dann auf die Herausforderungen eingehe, bei denen sich die eingesetzten Netzwerk-IDS bewähren mussten.

4.1 Das Test-Netzwerk

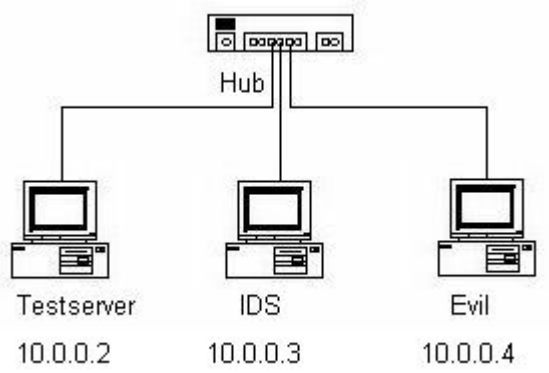


Abbildung 4.1: Das Test-Netzwerk

Das von mir verwendete Test-Netzwerk bestand aus 3 PCs die mit einem Hub verbunden wurden. Die Wahl des Netzwerk Verteilers fiel auf einen Hub, da ein Switch das Mitlesen des Netzwerkverkehrs durch das Intrusion Detection System verhindert hätte (siehe auch Kapitel 2.8).
Beim Hub handelte es sich um einen einfachen 10 Mbit/sek 4-Port Ethernet-Hub.
Die Rechner wurden anhand ihrer jeweiligen Aufgabe im Netzwerk benannt. So erhielt der Rechner auf dem einige Serverdienste liefen den Namen „Testserver“. Auf dem Rechner „IDS“ habe ich das, jeweils zu testende, Intrusion Detection System installiert. Der Rechner „Evil“[1] war für die Angriffe im Netz zuständig.

Rechnername	IP	Aufgabe
Testserver	10.0.0.2	Serverdienste die durch das IDS geschützt werden sollen
IDS	10.0.0.3	Intrusion Detection System
Evil	10.0.0.4	Angreifer

Tabelle 4.1: Die Rechner des Test-Netzwerkes mit ihren IP-Adressen und Aufgaben

Der Rechner „Testserver“ war ein AMD K6-3D mit 256 MB Arbeitsspeicher.
Bei „IDS“ handelte es sich um einen PII mit 128 MB Arbeitsspeicher.
„Evil“ musste mit einem PentiumMMX und 32 MB Arbeitsspeicher auskommen.

1 Evil = engl. Böse

Alle 3 Rechner wurden zunächst mit Windows 2000 Professional als Betriebsystem ausgestattet. Da ich auf einen Domain-Name-Server(DNS) verzichtet habe, habe ich die Hosts-Datei (c:\winnt\system32\drivers\etc\hosts) so angepasst, das die genannten Namen darüber aufgelöst werden konnten. Diese Datei hatte daher auf allen 3 Rechnern den folgenden Inhalt:

```
127.0.0.1       localhost
10.0.0.2        testserver.testnetz
10.0.0.3        ids.testnetz
10.0.0.4        evil.testnetz
```

Als Serverdienste auf „Testserver“ kamen Apache 2.0.44 (Webserver) und CesarFTP(FTP-Server), sowie der eingebaute NetBIOS-Server (alias Datei- und Druckerfreigabe) zum Einsatz.

4.2 Die Herausforderungen an die Intrusion Detection Systeme

Die folgenden Angriffe bzw. Angriffsvorbereitungen sollten erkannt werden.

4.2.1 Portscanner

(Literatur: [13] S.38-40 , [18] , [21])

Portscanner führen einen so genannten Portscan durch. Dies ist das Verbinden mit TCP oder UDP-Ports auf dem Zielsystem um festzustellen, welche Dienste auf diesem laufen und auf Verbindungen warten. Auf diesem Wege lässt sich oft auch feststellen welches Serverdienstprogramm oder Betriebsystem auf dem Zielsystem läuft.
So ist es auch möglich darüber festzustellen, dass ein Serverdienst mit bekannten Sicherheitslücken auf dem Zielsystem läuft. Portscans können daher eine Angriffsvorbereitung sein.

Portscanner können dabei verschiedene Portscan-Techniken anwenden:

1. *TCP Connect Scan*: Dieser Portscantyp führt einen kompletten 3-Wege Handshake durch. (SYN,SYN/ACK , ACK).

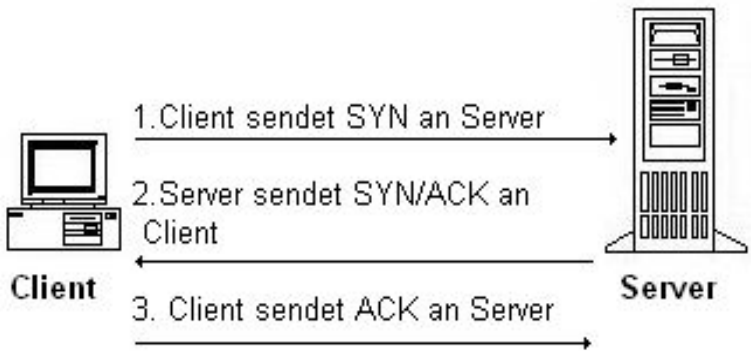


Abbildung 4.2: TCP 3 Wege Handshake

2. *TCP SYN Scan* : Diese Portscan-Technik wird auch Half-Open scanning genannt, da eine vollständige TCP-Verbindung nicht aufgebaut wird. Stattdessen antwortet der Client auf das SYN/ACK des Servers mit RST/ACK. Lauscht auf dem angefragten Port kein Dienst so sendet der Server statt SYN/ACK ein RST/ACK.

3. *TCP FIN Scan* : Der Client sendet ausschließlich ein FIN Paket an den betreffenden Port des Servers. Dies würde normalerweise eine TCP-Verbindung beenden. Basierend auf RFC793 sendet der Server ein RST-Paket zurück, wenn kein Dienst auf diesem Port lauscht. Lauscht auf dem angefragten Port ein Dienst so wird das RST-Paket ignoriert. Dies funktioniert meist nur mit Unix-Systemen , da der IP-Stack von Microsoft auf diese Art von Anfragen immer mit RST antwortet.

4. *TCP Xmas Tree*[1] *Scan*: Bei dieser Portscan-Technik wird ein TCP-Paket an den Server gesendet in dem alle TCP-Flags gesetzt sind. Basierend auf RFC793 sollte das Zielsystem auf so ein TCP-Paket mit einem RST antworten, wenn kein Dienst auf dem angefragten Port lauscht.

5. *TCP Null Scan*: Hierbei wird an den Server ein TCP-Paket gesendet, in dem alle Flags aus sind. Das Zielsystem sollte darauf mit RST Antworten, wenn kein Dienst lauscht.

6. *UDP Scan*: Diese Portscan-Technik sendet ein UDP-Paket zum Zielsystem. Dieses antwortet mit „ICMP Port unreachable“ wenn kein Dienst lauscht. Somit kann davon ausgegangen werden, das wenn kein „ICMP Port unreachable“-Paket zurückkommt, ein Dienst auf dem UDP-Port lauscht.

Nicht jeder der oben aufgeführten Portscan-Techniken funktioniert mit jeder Art von Zielsystem, da manche TCP/IP-Implementierungen auf manche Anfragen immer mit RST antworten.

Folgende für Windows existierende Portscanner kamen im Test zum Einsatz:

1. Superscan: Dieser Portscanner führt ausschießlich normale *TCP Connect Scans* durch.

2. WUPS: Der Windows UDP Port Scanner führt UDP Scans durch.

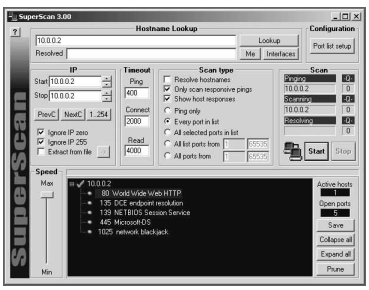


Abbildung 4.3: Superscan

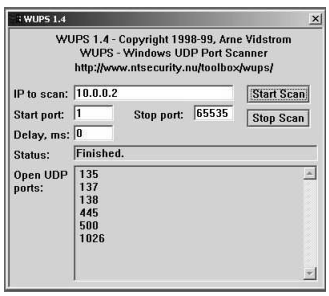


Abbildung 4.4: WUPS

1 Xmas Tree: engl. „Weihnachtsbaum“

3. IPEYE: Dieses Kommandozeilentool für Windows kann SYN,FIN,Null und Xmas-Tree Scans durchführen.

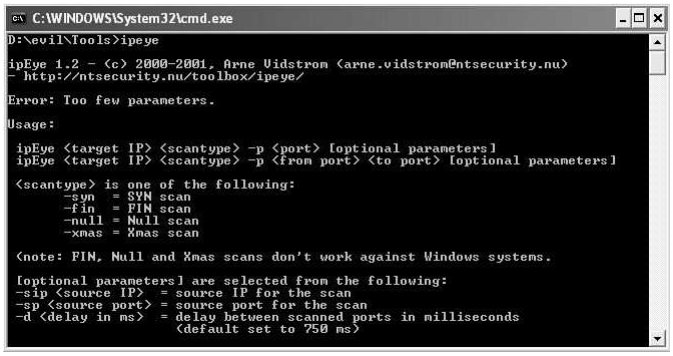


Abbildung: 4.5: IPEYE

4.2.2 NetBIOS Freigaben auskundschaften

(Literatur: [13] S.66-67+74)

Unter Windows NT/2000/XP ist es möglich über eine so genannte Null-Session an einige nützliche Informationen zu kommen. Eine Null-Session lässt sich durch Zugriff auf die versteckte Freigabe IPC$ und den immer vorhandenen anonymen User (`/u:""`) und einem leeren Passwort (`""`) aufbauen z.B. so:

```
c:\>net use  \\10.0.0.2\IPC$  "" /u:""
```

Oder auch mittels eines der zahlreichen Tools, die gleich direkt die Informationen die man über diese Null-Session gewinnen kann anzeigen z.B. WINFO.

Abbildung 4.6: WINFO

Diese Möglichkeit ist auch ohne den NetBIOS-Server komplett abzuschalten, weitgehend auszuschalten, indem man mit `regedt32` unter `HKEY_LOKAL_MASCHINE\SYSTEM\CurrentControlSet\Control\LSA` das `REG_DWORD` „`RestrictAnonymous`“ auf `1` (WindowsNT) oder `2` (Win2000/XP) setzt. Ansonsten lassen sich z.B. diese Informationen gewinnen (Null-Session von Evil nach Testserver):

```
D:\evil\Tools>winfo 10.0.0.2 -n

winfo 1.5 - copyright (c) 1999-2001, Arne Vidstrom
          - http://www.ntsecurity.nu/toolbox/winfo/

Trying to establish null session...

Null session established.
```

```
USER ACCOUNTS:

* Admin

* Administrator

  (This account is the built-in administrator account)

* Gast

  (This account is the built-in guest account)

WORKSTATION TRUST ACCOUNTS:

INTERDOMAIN TRUST ACCOUNTS:

SERVER TRUST ACCOUNTS:

SHARES:

* IPC$

* smbserver

* ADMIN$

* C$
```

4.2.3 Passwörter erraten

Kurze Passwörter lassen sich durch Ausprobieren finden.
Dazu gibt es für einige beliebte Internetdienste fertige Programme.
So z.B. Brutus – AET2 (kann HTTP Basic Auth,HTTP Form , FTP , POP3, Telnet und SMB (NetBIOS) Passwörter raten. Und dies sowohl per BruteForce[1], als auch per Wörterbuchattacke.

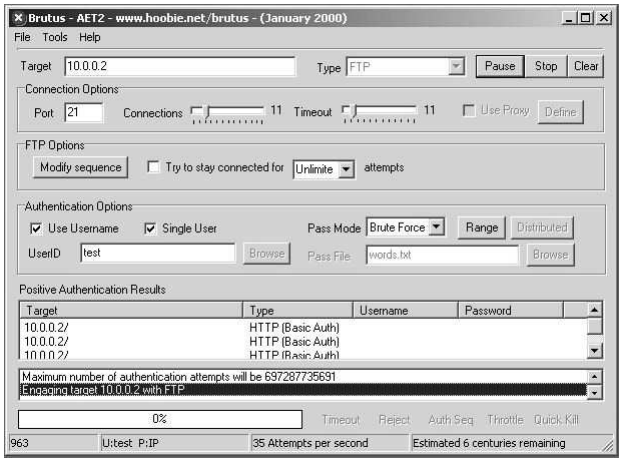


Abbildung 4.7: Brutus AET2

1 BruteForce meint hier pures Ausprobieren aller möglichen Kombinationen

Unsecure – Ein Tool das innerhalb kurzer Zeit viele Passwörter ausprobieren kann. Unsecure kann dies sowohl per Wörterbuchattacke als auch per BruteForce. Unsecure kann dabei verschiedene Server angreifen, da die Auswahl des Servers nicht vorgegeben ist, sondern durch Eingabe einer Portnummer geschieht. Weiterhin kann

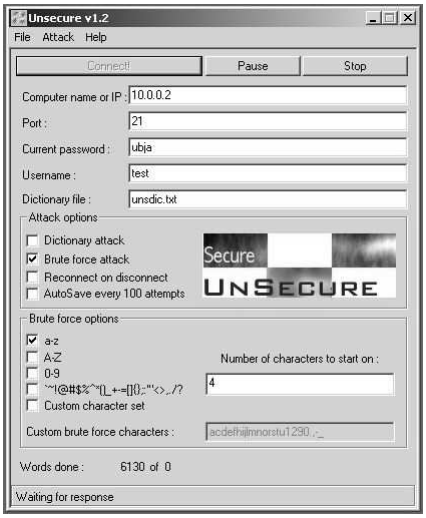


Abbildung 4.8a: Unsecure in Aktion

Abbildung 4.8b: Unsecure hat das Passwort gefunden.

nur ein Acount auf einmal getestet werden. Dieser muss daher auch vor dem Start des Passwortratens eingegeben werden. Danach ist aber nach relativ kurzer Zeit ein Passwort erraten, sofern das Passwort zu kurz gewählt ist.
Als letztes Passwortrateprogramm möchte ich kurz Webcracker vorstellen. Dieses Programm ist spezialisiert auf das Raten von http-Passwörtern. Zum Raten der Passwörter nimmt es eine Liste von Usernamen und Passwörtern (wahlweise als eine Liste mit nur Usernamen und eine mit nur Passwörtern, oder eine Kombinierte Liste mit beidem zugleich). Im Lieferumfang befindet sich bereits eine kleine Liste mit Usernamen und Passwörtern.

Abbildung 4.9: Webcracker

4.2.4 Denial of Service(DoS)

(Literatur: [13] S.504-507, [23])

Eine Denial of Service Attacke stört oder unterbricht Dienste, die somit für die berechtigten Benutzer, Netzwerke oder Systeme nicht, oder nur eingeschränkt, zur Verfügung stehen.
Alle diese Angriffe werden zumeist mit dem Vorsatz Schaden anzurichten beganen, und erfordern nur geringe Kenntnisse, da fertige Tools dafür verfügbar sind.
In meinen Tests benutzte ich `RPCNuke` , `Meliksak` , `HackNuker` , `CGSi-OOB` und `IGMPNuke` um die Erkennung solcher Angriffe durch ein IDS festzustellen. `Meliksak` hat sich leider als komplett für Tests unbrauchbar erwiesen, da es sich selbst und den TCP/IP-Stack des *angreifenden* Rechners durcheinander bringt, so das ein vollständiger Netzzugang erst nach einem Systemneustart wieder möglich war. Daher werde ich auf `Meliksak` nur kurz eingehen und in den Unterkapiteln über den eigentlichen IDS-Test nichts mehr schreiben.

Bevor ich aber auf die im Test eingesetzte DoS-Tools eingehe hier erstmal noch etwas zu den verschiedenen Arten von DoS-Attacken.

- **Bandbreitenverbrauch:**
  Dies ist die häufigste Art von DoS-Attacken. Hierbei versucht der Angreifer alle verfügbare Bandbreite eines Teilnetzwerkes zu verbrauchen. Dies kann in einem lokalen Netzwerk geschehen, es ist aber üblicher in einem entfernt gelegenden Netzwerk. Dies kann entweder dadurch geschehen das der Angreifer eine höhere Bandbreite als der Angegriffene besitzt oder durch Nutzen verschiedener anderer Rechner, die dann gleichzeitig den Zielrechner angreifen (Distributed Denial of Service).

- **Resourcen erschöpfen lassen:**
  Diese Form von DoS-Angriffen unterscheidet von der erstgenannten Form darin, das es hierbei nicht um Netzwerkresourcen handelt, sondern um Resourcen eines einzelnen Rechners. Das heißt der Angreifer erreicht z.B. das er die CPU auslastet, den Speicher oder die Festplatten voll schreibt oder ähnliches. Dies ist z.B. möglich wenn der Angreifer Zugriffserlaubnis zu einem endlichen Teil der Resourcen hat, und irgendwie erreicht das er mehr als diese im zustehenden Resourcen in Anspruch nehmen kann.

- **Programmrisse:**
  Bei dieser Form von DoS-Angriffen werden Fehler in Programmen, Betriebsystemen oder eingebetteten Logikchips bei der Behandlung von Ausnahmezuständen ausgenutzt. Ausnahmezustände entstehen z.B. wenn ein Benutzer unerwartete Daten sendet. Benutzt ein Programm z.B. einen Zwischenspeicher fester Länge und ein Angreifer sendet eine sehr lange Eingabe, so verursacht dies einen so genannten „buffer overflow“ und das betroffene Programm stürzt ab. Schlimmstenfalls gelingt es dem Angreifer dabei privilegierte Programme auszuführen, wenn in der langen Eingabe ausführbarer Programmcode enthalten war. Unter Umständen kann ausführbarer Code auch dafür genutzt werden, nicht nur das Programm zum Absturz zu bringen sondern auch das Betriebsystem. Dies gelingt z.B. unter Ausnutzung des f00f-Bugs im Pentium. (Falsche Behandlung der ungültigen Anweisung 0xf00fc7c8).

- **Routing und Domain-Name-Service-Angriffe:**
  Eine Routing basierte DoS-Attacke erfolgt durch Manipulieren von Routing-Tabelleneinträgen, durch den Angreifer. Dies wird sehr oft durch Routingprotokolle die

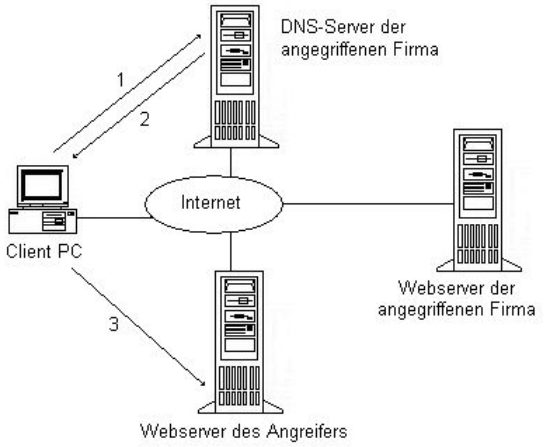


Abbildung 4.10: DNS Server vergiftet

keine, oder nur eine sehr schwache, Authentifizierung haben, erleichtert. Und wo Authentifizierung implementiert ist, wird sie oft nicht genutzt. Daher ist es zuweilen leicht, für einen Angreifer, die Routingeinträge zu ändern z.B. das alles durch das Netz des Angreifers geroutet wird oder in ein so genanntes *black hole*[1], ein Netzwerk das erst gar nicht existiert.
Angriffe auf den Domain Name Service verursachen ähnlichen großen Ärger wie Routing basierte Angriffe. Hierbei wird der DNS-Server einer Firma dazu gebracht falsche Adressinformationen zu speichern, und somit auf Anfragen die IP-Adresse z.B. des Webservers zu liefern (http://www.firma.de) stattdessen die IP-Adresse des Angreifers zu liefern. Damit werden die Zugriffe auf den Webserver faktisch auf einen Server des Angreifers umgeleitet. (Siehe Abbildung 4.10).

Die folgenden in meinem Test verwendeten Tools gehören alle in die Kategorie der Programmrisse:

- **RPC-Nuke:**
  Dieses Tool versucht durch unsinnige Remote Procedure Aufrufe (RPC) den Zielrechner zum Absturz zu bringen. Windows 2000 ist dagegen weitgehend abgesichert. Es stürzt nicht ab, sondern zeigt nur eine relativ unbedeutende Fehlermeldung.

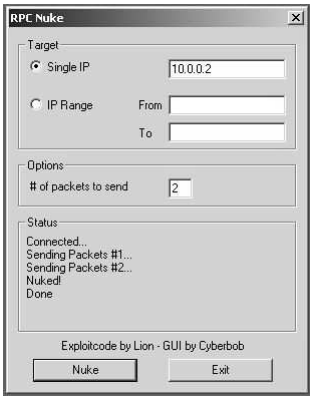


Abbildung 4.11: RPCNuke

Abbildung 4.12:
Fehlermeldung nach Nukeversuch

- **Meliksah Nuke:**
  Dieses Tool versucht unsinnige Anfragen an den NetBios-Server (auf Port 139) zu senden. Leider ist es wie eingangs schon geschrieben ziemlich ungeeignet für Tests.

Abbildung 4.13:
Meliksah Nuke

1 black hole = „Schwarzes Loch“ , es wird also alles verschluckt.

- **Hacknuker:**
  Auch dieser Tool versucht Anfragen an den NetBios-Server(auf Port 139) zu senden und damit den Zielrechner zum Absturz zu bringen. Windows 2000 ist dagegen abgesichert, so das es keinerlei Folgen hat. Hacknuker kann weiterhin auch Webserver und FTP-Server versuchen zum Absturz zu bringen. Auch dies klappt mit aktuellen Serverversionen nicht mehr.

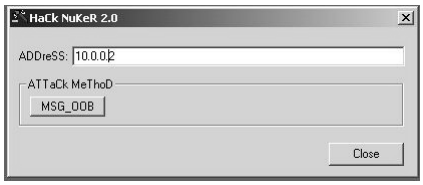


Abbildung 4.14a: Hacknuker Startfenster

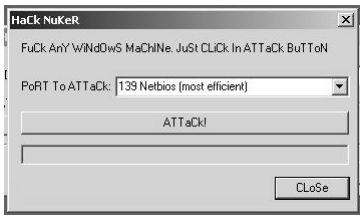


Abbildung 4.14b: Hacknuker Portauswahl

- **CGSi_OOB:**
  Und noch ein Tool das es auf den NetBios-Server abgesehen hat. Und wieder ist Windows 2000 immun dagegen.

Abbildung 4.15: CGSi_OOB

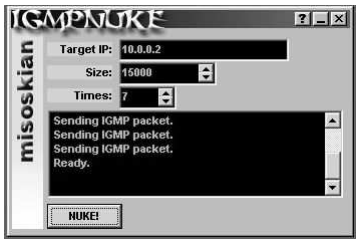


Abbildung 4.16: IGMPNuke

- **IGMP-Nuke:**
  Dieses Tool versucht einen Buffer Overflow im TCP/IP-Stack zu bewirken. Auch hier führt es unter Windows 2000 nicht zum Erfolg.

- **DDoS-Ping:**
  Dieses Tool ist ein spezialisierter Portscanner, der nur die von DDoS-Clients (Trinoo, Stacheldraht und Tribe Flood) verwendeten Ports durchsucht und damit eben auf das Vorhandensein der genannten DDoS-Clients testet.

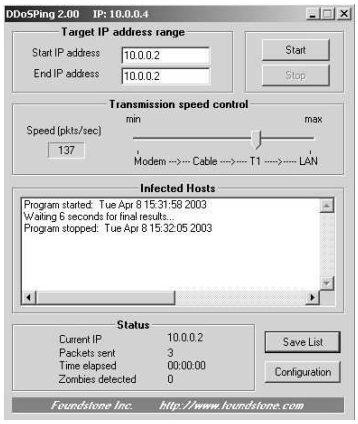


Abbildung 4.17: DDoS-Ping

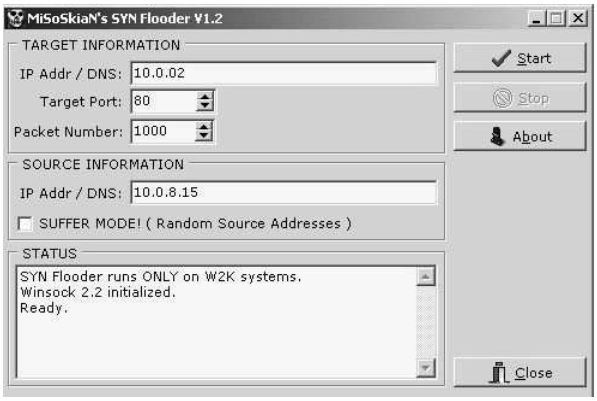


Abbildung 4.18: SYN-Flooder

- **SYN-Flooder**
  attackiert das Zielsystem mit Verbindungsanfragen. Dabei ist es möglich neben einer gewählten falschen IP-Adresse auch rein zufällige falsche Ip-Adressen zu verwenden. IP-Spoofing also. Die Anzahl der Verbindungsanfragen ist einstellbar.

4.2.5 Security-Scanner

(Literatur: [23])

Hierbei handelt es sich um Programme, die ein ganzes Netzwerk nach verschiedenen Sicherheitslücken absuchen können. Diese Programme lassen sich aber auch dazu nutzen um ein Netzwerk nach lohnenswerten Zielen für Angriffe abzusuchen.

Es gibt verschiedene derartige Programme, ich stelle daher nur die im Test verwendeten kurz vor:

- **LANguard Network Scanner:**
  Dieser Freeware-Securityscanner durchsucht ein ganzes Netzwerk oder auch nur einen ausgewählten Rechner nach Netzwerkfreigaben, Benutzerkontennamen, Win32-Dienste, Laufwerke, Passwortregeln, installierte Hotfixe , offene Ports , sowie bekannte Sicherheitsprobleme.
  LANguard Network Scanner erstellt dabei nicht nur eine direkte Ausgabe in seinem Bildschirmfenster, sondern auf Wunsch auch eine Ausgabe in eine HTML-Seite.

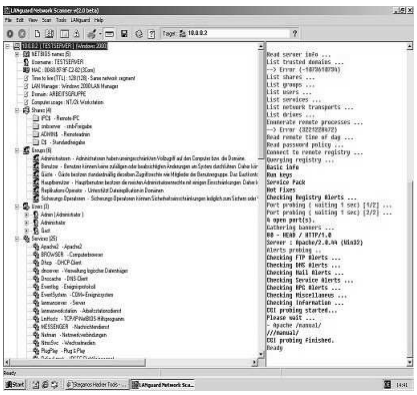


Abbildung 4.19: LANGuard Network Scanner

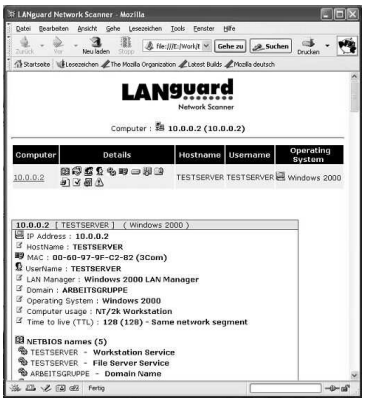


Abbildung 4.20: LANGuard HTML-Report

- **Cerberus Internet Scanner:**
  Dieser Securityscanner führt umfangreiche Tests in allen Bereichen durch, und kann ebenfalls auch eine HTML-Datei als Report ausgeben.

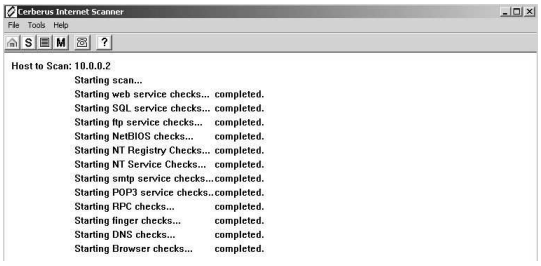


Abbildung 4.21: Cerberos Internet Scanner

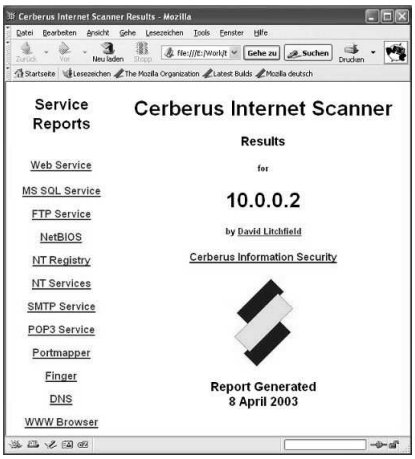


Abbildung 4.22: Cerberos Internet Scanner HTML-Report

Abbildung 4.23: Winfingerprint

- **Winfingerprint:**
  erkennt das auf dem Zielrechner installierte Betriebsystem samt Servicepacks und Hotfixe, sowie Netzwerkfreigaben (NetBIOS) , Benutzerkonten u.a.

4.2.6: Trojanische Pferde/Trojaner-Scanner/Hintertüren

(Literatur: [13] S.578 , [23])

Als Trojanisches Pferd bezeichnet man Programme, die vorgeben eine nützliche Funktion zu erfüllen, aber anderseits auch unbeabsichtigte und oft auch unautorisierte Aktionen durchführen oder schädliche oder zerstörerische Software im Hintergrund installieren.
Viele so genannte „Remote Control“[1]-Software lässt sich in so ein Paket schnüren. Damit wird aus der Fernwartungssoftware schnell eine Hintertür ins System, die sich der normale Nutzer ohne es zu wissen installiert.
Traurige Berümtheit hat hierbei das Programm „Back Orifice“ erlangt.

Das Vorhandensein des Back Orifice lässt sich unter anderem mit dem Trojaner-Scanner „Back Orifice Ping“ feststellen. Dies ist ein spezieller Portscanner der nur den von Back Orifice verwendeten Port untersucht.

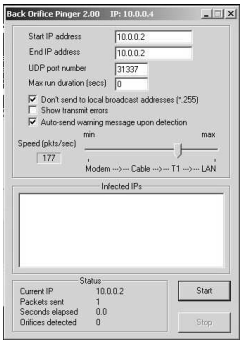


Abbildung 4.24: Bo-Ping

1 Remote-Control-Software bedeutet hier schlicht „Fernwartungssoftware“

4.3 Test der freien IDS-Software

Und nun gehe ich auf den eigentlichen Test ausgewählter freier Intrusion Detection Systeme ein. Als erstes testete ich das Programm Snort welches ich in Kapitel 3.3.1 vorgestellt habe.

4.3.1 Test von Snort

Direkt schon beim Test wie Snort auf Portscans reagiert, ist mir aufgefallen, das die Bildschirmausgabe der Snort-Konsole, sowie das Logging in die Logdateien, einige Zeit in Anspruch nimmt und nicht unmittelbar abgeschlossen ist. Vor allem fiel mir dies auf, als ich den Portscanner beendet hatte und noch Minuten später immer noch Meldungen auf der Snort-Konsole neu aufgetaucht sind.
Beim Beenden von Snort wird noch eine Zusammenfassung auf der Konsole ausgegeben. Diese zeigte im Falle der Portscantests das in meinem Test ca. 28 % aller Pakete von Snort gar nicht analysiert wurden (dropping xxxx (x%) packets) . Und dies in einem relativ langsamen 10BaseT-Ethernet[1]. In einem 100BaseT-Ethernet[2] dürfte dies noch schlimmer aussehen (Abbildung 4.25 zeigt das Ergebnis eines UDP-Portscans mit WUPS).

```
===============================================================================
Snort analyzed 131144 out of 182616 packets, dropping 51472(28.186%) packets

Breakdown by protocol:                Action Stats:
    TCP: 16         (0.009%)          ALERTS: 13
    UDP: 39844      (21.818%)         LOGGED: 6
   ICMP: 39812      (21.801%)         PASSED: 0
    ARP: 0          (0.000%)
  EAPOL: 0          (0.000%)
   IPv6: 0          (0.000%)
    IPX: 0          (0.000%)
  OTHER: 0          (0.000%)
DISCARD: 0          (0.000%)
===============================================================================
Wireless Stats:
Breakdown by type:
Management Packets: 0          (0.000%)
   Control Packets: 0          (0.000%)
      Data Packets: 0          (0.000%)
===============================================================================
Fragmentation Stats:
Fragmented IP Packets: 0          (0.000%)
    Fragment Trackers: 0
   Rebuilt IP Packets: 0
   Frag elements used: 0
Discarded(incomplete): 0
   Discarded(timeout): 0
  Frag2 memory faults: 0
===============================================================================
TCP Stream Reassembly Stats:
     TCP Packets Used: 16         (0.009%)
      Stream Trackers: 1
       Stream flushes: 0
        Segments used: 0
Stream4 Memory Faults: 0
===============================================================================
```

Abbildung 4.25: Snort-Endbildschirm

Es wurden alle Arten von Portscans als solche , mit den Standard-IDS-Regeln, erkannt. Der TCP-Connect-Scan mittels Superscan erzeugte in der Logdatei alert.ids Einträge der folgenden Art:

```
[**] [1:474:1] ICMP superscan echo [**]
[Classification: Attempted Information Leak] [Priority: 2]
02/11-12:22:29.652622 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3C
10.0.0.4 -> 10.0.0.2 ICMP TTL:128 TOS:0x0 ID:89 IpLen:20 DgmLen:36
Type:8  Code:0  ID:512   Seq:256  ECHO
```

1 10BaseT ist Ethernet mit Datenübertragungsraten bis 10MBit/sek
2 100BaseT ist Ethernet mit Datenübertragungsraten bis 100MBit/nsec (auch Fast-Ethernet genannt)

```
[**] [117:1:1] (spp_portscan2) Portscan detected from 10.0.0.4: 2 targets 11 ports in 31 seconds [**]
02/11-12:22:30.524955 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3E
10.0.0.4:1035 -> 10.0.0.2:7 TCP TTL:128 TOS:0x0 ID:96 IpLen:20 DgmLen:48 DF
******S* Seq: 0xBB69C0A7  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

Woraus also hier sogar der verwendete Portscanner hervorgeht.
Die UDP-Scans mit WUPS erzeugten die folgenden Einträge in der alert.ids :

```
[**] [117:1:1] (spp_portscan2) Portscan detected from 10.0.0.4: 1 targets 11 ports in 1 seconds [**]
01/30-13:24:24.861507 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3C
10.0.0.4:4118 -> 10.0.0.2:10 UDP TTL:128 TOS:0x0 ID:23056 IpLen:20 DgmLen:29
Len: 9
```

Die Portscans mit IPEYE (alle Typen) werden alle als Steath-Portscan erkannt, da sie keine komplette TCP/IP-Verbindung aufbauen, sondern den Zielrechner (hier also den Rechner „Testserver“) nur dazu bringen zu verraten ob und auf welchem Port es keinen Dienst gibt (siehe Kapitel 4.2.1). Dennoch benennt Snort sogar das verwendete Portscanverfahren. Die so genannten Steath-Portscans sind also für Snort nicht wirklich stealth[1]. Die folgenden Einträge finden sich jeweils mehrfach in der alert.ids .

Xmas-Scan mit IPEYE:

```
[**] [111:10:1] (spp_stream4) STEALTH ACTIVITY (XMAS scan) detection [**]
01/30-13:56:58.648868 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3E
10.0.0.4:57893 -> 10.0.0.2:2 TCP TTL:128 TOS:0x0 ID:0 IpLen:20 DgmLen:48
**U*P**F Seq: 0x74C11307  Ack: 0x0  Win: 0x4000  TcpLen: 28  UrgPtr: 0x0
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

NULL-Scan mit IPEYE:

```
[**] [111:9:1] (spp_stream4) STEALTH ACTIVITY (NULL scan) detection [**]
01/30-14:04:50.242514 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3E
10.0.0.4:57893 -> 10.0.0.2:3 TCP TTL:128 TOS:0x0 ID:0 IpLen:20 DgmLen:48
******** Seq: 0x74C11307  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

FIN-Scan mit IPEYE:

```
[**] [111:8:1] (spp_stream4) STEALTH ACTIVITY (FIN scan) detection [**]
01/30-14:33:05.267351 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3E
10.0.0.4:57893 -> 10.0.0.2:3 TCP TTL:128 TOS:0x0 ID:0 IpLen:20 DgmLen:48
*******F Seq: 0x74C11307  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

SYN-Scan mit IPEYE:

```
[**] [1:622:2] SCAN ipEye SYN scan [**]
[Classification: Attempted Information Leak] [Priority: 2]
01/30-14:35:31.472744 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3E
10.0.0.4:57893 -> 10.0.0.2:3 TCP TTL:128 TOS:0x0 ID:0 IpLen:20 DgmLen:48
******S* Seq: 0x74C11307  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
[Xref => arachnids 236]
```

In der scan.log finden sich jeweils die Details, über die während des Portscans angefragten Ports und verwendeten TCP-Flags. Auszugsweise sieht dies z.B. so aus:

```
...
01/30-14:35:31.723055  TCP src: 10.0.0.4 dst: 10.0.0.2 sport: 57893 dport: 28 tgts: 1 ports: 28 flags: ******S* event_id: 22
01/30-14:35:31.733075  TCP src: 10.0.0.4 dst: 10.0.0.2 sport: 57893 dport: 29 tgts: 1 ports: 29 flags: ******S* event_id: 22
01/30-14:35:31.743083  TCP src: 10.0.0.4 dst: 10.0.0.2 sport: 57893 dport: 30 tgts: 1 ports: 30 flags: ******S* event_id: 22
01/30-14:35:31.753107  TCP src: 10.0.0.4 dst: 10.0.0.2 sport: 57893 dport: 31 tgts: 1 ports: 31 flags: ******S* event_id: 22
01/30-14:35:31.763124  TCP src: 10.0.0.4 dst: 10.0.0.2 sport: 57893 dport: 32 tgts: 1 ports: 32 flags: ******S* event_id: 22
01/30-14:35:31.773131  TCP src: 10.0.0.4 dst: 10.0.0.2 sport: 57893 dport: 33 tgts: 1 ports: 33 flags: ******S* event_id: 22
01/30-14:35:31.783140  TCP src: 10.0.0.4 dst: 10.0.0.2 sport: 57893 dport: 34 tgts: 1 ports: 34 flags: ******S* event_id: 22
01/30-14:35:31.793228  TCP src: 10.0.0.4 dst: 10.0.0.2 sport: 57893 dport: 35 tgts: 1 ports: 35 flags: ******S* event_id: 22
01/30-14:35:31.803169  TCP src: 10.0.0.4 dst: 10.0.0.2 sport: 57893 dport: 36 tgts: 1 ports: 36 flags: ******S* event_id: 22
...
```

NetBios-Nullsessions wurden von Snort nicht erkannt und daher können ohne Snort-Alarmmeldung Informationen über vorhandene NetBios-Freigaben gewonnen werden. Die Aktivitäten der Passwortrateprogramme wurden mit den Standardregeln fälschlich als normaler Portscan (Aktivitäten von Brutus AET2 und Webcracker) oder gar nicht erkannt (Aktivitäten von Unsecure). Irritierender weise werden diese Aktivitäten als

1 steath – engl heimlich

Portscans vom Server zum Client erkannt. In meinem Test also als Portscan von Testserver (10.0.0.2) nach Evil (10.0.0.4). Wie sich an folgendem Beispiel sehen lässt (scan.log):

```
...
02/18-12:52:02.901115  TCP src: 10.0.0.2 dst: 10.0.0.4 sport: 80 dport: 1080 tgts: 1 ports: 15 flags: ***A**** event_id: 2
02/18-12:52:13.846627  TCP src: 10.0.0.2 dst: 10.0.0.4 sport: 80 dport: 1086 tgts: 1 ports: 16 flags: ***A**** event_id: 2
02/18-12:52:13.856442  TCP src: 10.0.0.2 dst: 10.0.0.4 sport: 80 dport: 1087 tgts: 1 ports: 17 flags: ***A**** event_id: 2
02/18-12:52:13.867968  TCP src: 10.0.0.2 dst: 10.0.0.4 sport: 80 dport: 1088 tgts: 1 ports: 18 flags: ***A**** event_id: 2
02/18-12:52:13.886315  TCP src: 10.0.0.2 dst: 10.0.0.4 sport: 80 dport: 1089 tgts: 1 ports: 19 flags: ***A**** event_id: 2
02/18-12:52:13.942197  TCP src: 10.0.0.2 dst: 10.0.0.4 sport: 80 dport: 1092 tgts: 1 ports: 20 flags: ***A**** event_id: 2
02/18-12:52:13.952088  TCP src: 10.0.0.2 dst: 10.0.0.4 sport: 80 dport: 1091 tgts: 1 ports: 21 flags: ***A**** event_id: 2
...
```

In der alert.ids fanden sich diese Einträge:

```
[**] [117:1:1] (spp_portscan2) Portscan detected from 10.0.0.2: 1 targets 11 ports in 11 seconds [**]
02/18-12:52:02.776191 0:60:97:9F:C2:82 -> 0:60:8:51:19:99 type:0x800 len:0x5EA
10.0.0.2:80 -> 10.0.0.4:1074 TCP TTL:128 TOS:0x0 ID:717 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x514DE459  Ack: 0x26F99A3B  Win: 0x43B0  TcpLen: 20

[**] [117:1:1] (spp_portscan2) Portscan detected from 10.0.0.2: 1 targets 11 ports in 22 seconds [**]
02/18-12:54:59.305846 0:60:97:9F:C2:82 -> 0:60:8:51:19:99 type:0x800 len:0x3E
10.0.0.2:80 -> 10.0.0.4:1258 TCP TTL:128 TOS:0x0 ID:1634 IpLen:20 DgmLen:48 DF
***A**S* Seq: 0x5477D60E  Ack: 0x2A224B89  Win: 0x4470  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

Der Angriff wird also nicht nur falsch klassifiziert, (eben als Portscan statt als Passwort-Rateversuch) sondern auch als Verbindung in der falschen Richtung erkannt. Nur durch die für einen normalen Portscan unsinnigen TCP-Flags, (***A***S) (SYN-ACK ist nun mal die Antwort eines aktiven Servers auf ein eingehendes SYN) kann dies vom Benutzer des Intrusion Detection Systems als etwas anderes als einen einfachen Portscan erkannt werden.
Die Denial-of-Service-Attacken mit Hacknuker, CGSi-OOB und IGMPNuke wurden von Snort erkannt. Die Aktivitäten von RPCNuke hat Snort 1.9 nicht als DoS-Angriff erkannt. Ebenso versagte die Erkennung der Aktivitäten mit SYN-Flooder.
Denial-of-Service-Attacken erzeugten in der alert.ids Einträge der folgenden Art:

HackNuker:

```
[**] [1:1257:3] DOS Winnuke attack [**]
[Classification: Attempted Denial of Service] [Priority: 2]
02/04-12:03:03.414098 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3C
10.0.0.4:1030 -> 10.0.0.2:139 TCP TTL:128 TOS:0x0 ID:3531 IpLen:20 DgmLen:43 DF
**UAP*** Seq: 0x5BD7F457  Ack: 0xA39043C2  Win: 0x4470  TcpLen: 20  UrgPtr: 0x3
[Xref => cve CVE-1999-0153][Xref =>  bugtraq 2010]
```

CGSi-OOB:

```
[**] [1:1257:3] DOS Winnuke attack [**]
[Classification: Attempted Denial of Service] [Priority: 2]
02/06-13:43:55.413256 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3C
10.0.0.4:1029 -> 10.0.0.2:139 TCP TTL:128 TOS:0x0 ID:126 IpLen:20 DgmLen:43 DF
**UAP*** Seq: 0x386AC094  Ack: 0xEE571713  Win: 0x4470  TcpLen: 20  UrgPtr: 0x3
[Xref => cve CVE-1999-0153][Xref =>  bugtraq 2010]
```

IGMPNuke:

```
[**] [1:273:2] DOS IGMP dos attack [**]
[Classification: Attempted Denial of Service] [Priority: 2]
02/06-14:23:05.320784 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x5EA
10.0.0.4 -> 10.0.0.2 PROTO002 TTL:128 TOS:0x0 ID:2738 IpLen:20 DgmLen:1500 MF
Frag Offset: 0x0000   Frag Size: 0x05C8
```

Die Erkennung des versuchten Denial-of-Service-Angriffs hat sich als unzuverlässig erwiesen, wenn gleichzeitig mit dem DoS-Angriff ein Portscan im Gange war. Es wurde dann zuweilen nur der Portscan erkannt, aber nicht der DoS-Angriff. Dies hat sich bereits beim alleinigen Portscan angedeutet, wo ich bemerkt hatte das Pakete verworfen wurden. Das nicht Erkennen des DoS-Angriffs hatte auch hier die Ursache in verworfenen Paketen, die somit nicht analysiert wurden.

Ansonsten tauchten in allen genannten Dateien auch viele Meldungen auf, die wenig mit dem tatsächlich ausgeführten Angriff zu tun haben, und daher also falsche bzw. verwirrende Alarmmeldungen sind.

Bei den Tests die mit DDoS-Ping durchgeführt wurden (Scanner auf Distributed Denial of Service Clients) tauchte nur die Abfrage auf den DDoS-Client „Stacheldraht“ im Snort-Alert.ids-Log auf:

```
[**] [1:236:1] DDOS Stacheldraht client-check-gag [**]
[Classification: Attempted Denial of Service] [Priority: 2]
04/08-15:16:36.413380 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3C
10.0.0.4 -> 10.0.0.2 ICMP TTL:128 TOS:0x0 ID:17046 IpLen:20 DgmLen:43
Type:0  Code:0  ID:668  Seq:0  ECHO REPLY
[Xref => arachnids 194]
```

DDoS-Ping untersucht allerdings den oder die Zielhost(s) auf das Vorhandensein von Clients von „Trinoo“, „Stacheldraht“ und „Tribe Flood Network“ (siehe Readme.txt im DDoS-Paket auf der CD zu [23]). Somit hat Snort nicht alle Aktivitäten von DDoS-Ping gemeldet. Es können Angriffe übersehen werden.

Die Aktivitäten der Securityscanner „LANguard Network Scanner“, „Cerberos Internet Scanner“ und „Winfingerprint“ fanden sich in den Snort-Logs nur durch sehr viele, teilweise verwirrende (Web-IIS, obwohl kein IIS auf „Testserver“ installiert ist), teilweise doch wenigstens auf das richtige hindeutende Meldungen „[Classification: Attempted Information Leak]“ wieder. Direkt auf das genaue und korrekte Loch hindeutend war keine der Meldungen.

Der Scan des „Testservers“ mit „BO-Ping“ (Trojaner-Scanner) wurde als Aktivität des Trojaners „Back Orifice“ erkannt:

```
[**] [105:1:1] spp_bo: Back Orifice Traffic detected (key: 31337) [**]
[Classification: Misc Attack] [Priority: 2]
04/08-15:38:17.175071 0:60:8:51:19:99 -> 0:60:97:9F:C2:82 type:0x800 len:0x3D
10.0.0.4:1449 -> 10.0.0.2:31337 UDP TTL:128 TOS:0x0 ID:21582 IpLen:20 DgmLen:47
Len: 27
[Xref =>  cve CVE-2000-0666][Xref => bugtraq 1480]
```

Zusammenfassung der Tests mit Snort:

Erkennung von Portscans:

TCP-Connect	TCP-SYN	TCP-FIN	TCP-XMAS	TCP-Null	UDP-Scans
Die Erkennung aller Portscantypen gelingt zuverlässig, auch die Art des Scans wird zuverlässig klassifiziert.					

Erkennung von SMB-Nullsessions:
keine Erkennung

Erkennung von Passwortrateaktivitäten:

BrutusAET2	UNSECURE	Webcracker
fälschlich als Portscan	keine Erkennung	fälschlich als Portscan

Erkennung von Denial-of-Service-Attacken

RPC-Nuke	Meliksah Nuk	Hacknuker	CGSi-OOB	IGMPNuke	SYN-Flooder
keine Erkennung	Test nicht durchführbar (1	Erkennung erfolgt	Erkennung erfolgt	Erkennung erfolgt	keine Erkennung
-	-	Erkennung bei hoher Last eingeschränkt			-

(1) siehe Text

Tabelle 4.2: Zusammenfassung der Tests mit Snort (Teil1)

Aktivitäten von Security-Scannern:

LANguard Network Scanner	Cerberos Internet Scanner	Win-Fingerprint
Viele verschiedene Alarmmeldungen mit zum Teil verwirrendem Inhalt.		

Trojaner-Scanner:

Bo-Ping
Aktivität erkannt

Tabelle 4.3: Zusammenfassung der Tests mit Snort (Teil2)

4.3.2 Test von Pakemon-IDS

Da Pakemon-IDS nicht mehr weiterentwickelt wird, und die vorhandenen Angriffssignaturen einen sehr übersichtlichen Umfang haben (siehe Abbildung 4.26) erübrigen sich ausführliche Tests.
Pakemon kann mit seinen mitgelieferten Signaturen fast keine der beschriebenen Herausforderungen erkennen.

```
# signature definition file for pakemon0.3.0
# format:
#    name transport_protocol src_port dest_port payload_pattern
#
# payload_pattern
#   "..." : case sensitive text pattern
#   '...' : case insensitive text pattern
#   \x or |...| between "" or '' : binary data
#
# You can find more signatures from contributers sites such as ...
#
#  Whitehats http://whitehats.com/ids/vision-pakemon.conf


# FTP

FTP-exploit1                            tcp * 21 "|50 57 44 0A 2F 69|"
FTP-exploit2                            tcp * 21 "|58 58 58 58 58 2F|"
# FTP-nopassword                        tcp * 21 'pass |0d|'
# FTP-incorrect-login                   tcp 21 * "Login incorrect"

# TELNET

# TELNET-Incorrect-login                tcp * 23 "Login incorrect"

# SMTP some of these are converted from snort's smtp-lib

SMTP-exploit1                           tcp * 25 'Croot|09090909090909|Mprog,P=/bin'
SMTP-exploit2                           tcp * 25 'rcpt to|3a207c| sed '1,/^$/d'|7c|'
SMTP-exploit3                           tcp * 25 'mail from|3a20227c|'
SMTP-exploit4                           tcp * 25 'Croot|0d0a|Mprog, P=/bin/'
SMTP-exploit6                           tcp * 25 'rcpt to|3a| decode'
SMTP-exploit7(CVE-1999-0204)            tcp * 25 '|0a|C|3a|daemon|0a|R'
SMTP-exploit8(CVE-1999-0204)            tcp * 25 '|0a|Croot|0a|Mprog'
SMTP-exploit9(CVE-1999-0204)            tcp * 25 '|0a|Croot|0d0a|Mprog'
SMTP-exploit10(CVE-1999-0095)           tcp * 25 '|7c 73 65 64 20 2d 65 20 27 31 2c 2f 5e 24 2f 27|'
SMTP-exploit11(CVE-1999-0204)           tcp 113 25 '|0a|D/'
SMTP-expn-decode                        tcp * 25 'expn decode'
SMTP-expn-root                          tcp * 25 'expn root'
SMTP-vrfy-decode                        tcp * 25 'vrfy decode'
SMTP-Pikachu(Pokey)-Worm                tcp * 25 'pikachupokemon.exe'

# DNS
DNS-named-exploit                       tcp * 53 "|CD 80 E8 D7 FF FF FF|"
DNS-zone-transfer                       tcp * 53 "|01 00 00 01 00 00 00 00 00 00|"

# HTTP - IIS

IIS-administrator.pwd                   tcp * 80 '/_vti_pvt/administrators.pwd'
IIS-authors.pwd                         tcp * 80 '/_vti_pvt/authors.pwd'
IIS-service.pwd                         tcp * 80 '/_vti_pvt/service.pwd'
IIS-users.pwd                           tcp * 80 '/_vti_pvt/users.pwd'
IIS-iisadmpwd                           tcp * 80 'iisadmpwd/aexp3.htr'
IIS-carbo.dll                           tcp * 80 '/carbo.dll'
IIS-admin-default                       tcp * 80 'scripts/iisadmin/'
IIS-ISM.DLL-Exploit                     tcp * 80 '%20%20%20%20%20.htr'

# HTTP - CGI

CGI-aglimpse                            tcp * 80 '/cgi-bin/aglimpse'
CGI-args.bat                            tcp * 80 '/cgi-dos/args.bat'
CGI-bash                                tcp * 80 '/cgi-bin/bash'
CGI-csh                                 tcp * 80 '/cgi-bin/csh'
```

```
CGI-AnyForm                             tcp * 80 '/cgi-bin/AnyForm'
CGI-AnyForm2                            tcp * 80 '/cgi-bin/AnyForm2'
CGI-AT-admin.cgi                        tcp * 80 '/cgi-bin/AT-admin.cgi'
CGI-bash                                tcp * 80 '/cgi-bin/bash'
CGI-bnbform.cgi                         tcp * 80 '/cgi-bin/bnbform.cgi'
CGI-campas                              tcp * 80 '/cgi-bin/campas'
CGI-classifieds.cgi                     tcp * 80 '/cgi-bin/classifieds.cgi'
CGI-environ.cgi                         tcp * 80 '/cgi-bin/environ.cgi'
CGI-faxsurvey                           tcp * 80 '/cgi-bin/faxsurvey'
CGI-filemail.pl                         tcp * 80 '/cgi-bin/filemail.pl'
CGI-files.pl                            tcp * 80 '/cgi-bin/files.pl'
CGI-formmail                            tcp * 80 'formmail'
CGI-fpexplore.exe                       tcp * 80 '/cgi-bin/fpexplore.exe'
CGI-GuestBook(CVE-1999-0237)            tcp * 80 '/cgi-bin/guestbook.cgi'
CGI-handler(CVE-1999-0148)              tcp * 80 '/cgi-bin/handler'
CGI-htmlscript                          tcp * 80 '/cgi-bin/htmlscript'
CGI-info2www                            tcp * 80 '/cgi-bin/info2www'
CGI-jj                                  tcp * 80 '/cgi-bin/jj'
CGI-MachineInfo                         tcp * 80 '/cgi-bin/MachineInfo'
CGI-maillist.pl                         tcp * 80 '/cgi-bin/maillist.pl'
CGI-man.sh                              tcp * 80 '/cgi-bin/man.sh'
CGI-newdsn.exe                          tcp * 80 '/scripts/tools/newdsn.exe'
CGI-nph-test-cgi(CVE-1999-0045)         tcp * 80 '/cgi-bin/nph-test-cgi'
CGI-perl                                tcp * 80 '/cgi-bin/perl'
CGI-pfdispaly.cgi                       tcp * 80 '/cgi-bin/pfdispaly.cgi'
CGI-phf(CVE-1999-0067)                  tcp * 80 '/cgi-bin/phf'
CGI-php.cgi                             tcp * 80 '/cgi-bin/php.cgi'
CGI-rsh                                 tcp * 80 '/cgi-bin/rsh'
CGI-rksh                                tcp * 80 '/cgi-bin/rksh'
CGI-survey.cgi                          tcp * 80 '/cgi-bin/survey.cgi'
CGI-tcsh                                tcp * 80 '/cgi-bin/tcsh'
CGI-test-cgi                            tcp * 80 '/cgi-bin/test-cgi'
CGI-textcounter.pl                      tcp * 80 '/cgi-bin/textcounter.pl'
CGI-unlg1.1                             tcp * 80 '/cgi-bin/unlg1.1'
CGI-upload                              tcp * 80 '/cgi-bin/upload.pl'
CGI-uploader.exe                        tcp * 80 '/cgi-bin/uploader.exe'
CGI-view-source                         tcp * 80 '/cgi-bin/view-source'
CGI-webdist.cgi                         tcp * 80 '/cgi-bin/webdist.cgi'
CGI-webgais                             tcp * 80 '/cgi-bin/webgais'
CGI-websendmail                         tcp * 80 '/cgi-bin/websendmail'
CGI-whois_raw.cgi                       tcp * 80 '/cgi-bin/whois_raw.cgi'
CGI-win-c-sample.exe                    tcp * 80 '/cgi-shl/win-c-sample.exe'
CGI-wrap                                tcp * 80 '/cgi-bin/wrap'
CGI-wwwadmin                            tcp * 80 '/cgi-bin/wwwadmin.pl'
CGI-wwwboard.pl                         tcp * 80 '/cgi-bin/wwwboard.pl'
CGI-www-sql                             tcp * 80 '/cgi-bin/www-sql'
CGI-wwwuploader.exe                     tcp * 80 '/cgi-win/wwwuploader.exe'
CGI-mlog.phtml                          tcp * 80 '/mlog.phtml'
CGI-mylog.phtml                         tcp * 80 '/mylog.phtml'
CGI-search97.vts                        tcp * 80 '/search97.vts'
CGI-snork.bat                           tcp * 80 '/scripts/snork.bat'

# HTTP - other

HTTP-../../../                          tcp * 80 "../../../"
HTTP-/....                              tcp * 80 "/...."
HTTP-ApacheDOS                          tcp * 80 "|2f2f2f2f2f2f2f2f|"
HTTP-PiranhaPasswd.php3                 tcp * 80 "passwd.php3"

# RPC
RPC-portmap-request-rusers              udp * 111 "|01 86 A2 00 00|"

# IMAP
IMAP-exploit1                           tcp * 143 "|bf ff|"
IMAP-exploit2                           tcp * 143 "|E8 C0 FF FF FF|"

# MISC

DOS-Trin00-killme                       tcp * 27665 "killme"
DOS-Trin00-gOrave                       tcp * 27665 "gOrave"
DOS-Trin00-144dsl                       udp * 27444 "l44adsl"
DOS-Trin00-PONG                         udp * 31335 "PONG"
DOS-Trin00-144                          udp * 31335 "l44"
DOS-Trin00-HELLO                        udp * 31335 "*HELLO*"
# DOS-TFN                               icmp 0 0 "|73 68 65 6C 6C 20 62 6F 75 6E 64 20 74 6F 20 70 6F 72 74|"
BD-BackOrifice                          udp * 31337 ""
```

Abbildung 4.26: Pakemon-Signaturen
(Quelle: signatur.txt in der pakemon-0.3.1.tar.gz)

Einzig der Traffic des Trojaners BackOrifice bzw. des BO-Pingers, sowie nach Auskommentieren der vorletzten Zeile auch Aktivitäten des DDoS „Tribe Flood Network“, würde erkannt werden.

Hieran zeigt sich auch, wie wichtig es ist, das immer aktuelle Angriffssignaturen bereitgestellt werden, den ohne eine vorhandene Angriffssignatur kann ein IDS das sich auf Signaturen vorhandener Angriffe verlässt, nichts feststellen.

Pakemon IDS 0.3.1 ist vom 7 Januar 2001 (siehe [12]) und somit sind auch die Signaturen mindestens 2 Jahre alt.

Zudem wurde Pakemon nicht fertig entwickelt, worauf die mit 0 beginnende Versionnummer hindeutet.

Kapitel 5

Grenzen von Intrusion Detection Systemen

Abschließend möchte ich noch einiges über aktuelle Grenzen von Intrusion Detection Systeme schreiben, auch wenn ich dies schon in den vorherigen Kapiteln angerissen habe.

5.1 Weiterentwickelte Angriffsstrategien

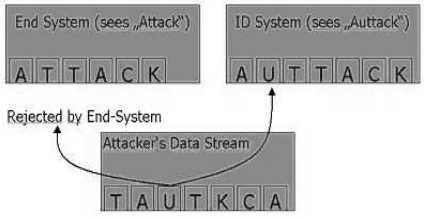
(Literatur: [1] S.48-55+S.72-73 , [14])

- Angreifer verbessern laufend ihre Fähigkeiten in Netzwerke einzudringen. Insbesondere ihre Fähigkeit unentdeckt zu bleiben und unerwartete Strategien anzuwenden. Intrusion Detection Systeme haben es schwer immer mitzuhalten.

- Angreifertools werden laufend weiterentwickelt und sind eine immer größere Herausforderung für Intrusion Detection Systeme

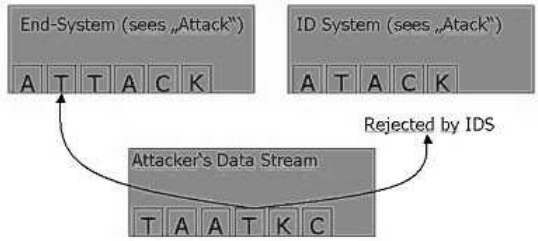
- Verschlüsselte Nachrichten können schädliche Information enthalten, die von einem Netzwerk Intrusion Detection System nicht gefunden werden kann, da diese darauf angewiesen sind den Paketinhalt auf Muster zu untersuchen. Anderseits macht eine gute Verschlüsselung es schwieriger Informationen zu stehlen. Dieses Problem lässt sich nur durch Intrusion Detection Systeme auf Rechner- und Anwendungebene lösen. Die sich den Inhalt der Nachricht ansehen, wenn sie entschlüsselt wurde.

- Es gibt Angriffe die das Intrusion Detection System dazu bringen, Netzwerkeinbrüche nicht mehr zu registrieren. Dies kann z.B. geschehen wenn das System für die Daten bestimmt sind andere Daten sieht als das Netzwerk Intrusion Detection System. Hierbei gibt es zwei Möglichkeiten. Entweder akzeptiert das Netzwerk-IDS Pakete die das Zielsystem verwirft (dies nennt man „Insertion Attack“ da



(Quelle: [14])
Abbildung 5.1: Insertion Attacke

Netzwerkpakete in den vom Netzwerk-IDS analysierten Datenstrom eingefügt sind. (Abbildung 5.1 zeigt es an einem einfachen Beispiel mit Ein-Zeichen-Paketen)



(Quelle: [14])
Abbildung 5.2: Evasion Attacke

Eine weitere Möglichkeit ist die so genannte „Evasion Attacke“ bei der das Netzwerk-IDS Pakete verwirft die das Endsystem nicht verwirft.
(Abbildung 5.2 zeigt ein einfaches Beispiel).

- Intrusion Detection Systeme sind nicht in der Lage neue Angriffsstrategien zu erkennen. Einige Intrusion Detection Systeme versuchen diese Lücke durch eine statistische Untersuchung des Netzwerkverkehrs (und damit einer Untersuchung auf Anormalitäten) zu schließen. Dies ist aber schwierig da:

  1. Es schwierig ist Parameter normalen Netzwerkverkehrs zu definieren.

  2. Benutzerverhalten sich so schnell ändern kann, das das lernende System nicht so schnell in der Lage ist sein Verhalten zu ändern.

  3. Geschickte Angreifer können in der Lage sein, das System so zu trainieren, das es ihr Verhalten als normal ansieht.

5.2 Undokumentierte Modems

(Literatur: [1] S.56, [13] S.396-399)

Ein Modem, das an einen am Netzwerk angeschlossenen Rechner angeschlossen ist, kann ein unbemerkter verletzbarer Punkt sein. Angreifer können mit einem „Wardialer“[1] so ein Modem finden und damit einen Eintrittspunkt in das Netzwerk erhalten, der nicht durch die Firewall geschützt ist ,und unter Umständen, auch nicht durch das Netzwerk-IDS überwacht wird. IDS auf Rechnerebene könnten dies bemerken, allerdings kann dies vom normalen Benutzer des undokumentierten Modems abgeschaltet worden sein.

1 Wardialer: Hardware/Software die eine größere Anzahl von Telefonnummern z.B. innerhalb eines Firmentelefonetzes von außen anruft um herauszufinden, ob und unter welcher Telefonnummer sich ein Modem meldet.

5.3 Netzwerkstrukturen

(Literatur: [1] S.58-63)

- Intrusion Detection Systeme arbeiten nicht über Netze mit verschiedenen Techniken oder Sicherheitsregeln zusammen. Jedes aktuelle Intrusion Detection System gibt seine Informationen in einem eignen Datenformat aus. Es ist daher schwierig für Organisationen die zusammenarbeiten, aber sich nicht vertrauen wichtige Informationen über Einbrüche bzw. Angriffe sich mitzuteilen.

- Intrusion Detection Systeme sind nicht in der Lage ihre Informationen auszutauschen. Es ist daher schwierig, ihre Ergebnisse miteinander zu korrelieren.

- Netzwerk-Intrusion Detection Systeme sehen nicht immer jeden Netzwerkverkehr. In Netzen wo ein Switch zum Einsatz kommt, können sie keinen Netzwerkverkehr aufnehmen. Der Trend bei Vernetzungen geht immer mehr zum Einsatz von Switchen.

5.4 Menschliche Faktoren

(Literatur: [1] S.65)

Intrusion Detection Systeme sind nicht in der Lage Angreifer zu identizifizieren und ihre Ziele zu benennen. Profiling wie es in Kriminalfällen, vor allem bei Serienmördern, zuweilen durchgeführt wird, kann durch ein Intrusion Detection System, derzeit nicht durchgeführt werden.

5.5 Funktionale Lücken

(Literatur: [1] S.68-71+S.75+S.78, [14], [24])

- Intrusion Detection Systeme können viele Angriffsarten nicht in ihren frühen Schritten identifizieren. Damit ist es schwer Schaden abzuwehren bevor er aufgetreten ist.

- Intrusion Detection Systeme haben es schwer korrekte automatische Aktionen durchzuführen. Einige Intrusion Detection Systeme haben die Möglichkeit auf bestimmte Angriffe mit Beenden der TCP-Verbindung oder Blockieren des zweifelhaften Netzwerkverkehrs in der Firewall zu reagieren. Dies macht das Netzwerk verwundbar für Angriffsarten die genau diese automatischen Aktionen des IDS ausnutzen. Diese Angriffe geschehen durch IP-Spoofing des Angreifers. Das heißt der Angreifer fälscht seine eigne IP-Adresse. (siehe auch Kapitel 2.6).

- Intrusion Detection Systeme unterstützen kaum bei der Systemwiederherstellung nachdem ein Schaden eingetreten ist.

- Intrusion Detection Systeme bieten kaum bis gar keine Anleitung, wie auf einen identifizierten Angriff reagiert werden kann.

- Intrusion Detection Systeme können Angreifer nicht bemerken, wenn sie erst die Resourcen des Intrusion Detection Systems erschöpfen lassen. Also einen Denial-of-Service-Angriff (siehe Kapitel 4.2.4) durchführen. Dies geschied dadurch, dass das Intrusion Detection System, gezielt mit nutzloser Arbeit überlastet wird und damit gezwungen ist Pakete ohne sie zu analysieren weg zuwerfen.

  Resourcenerschöpfung kann auf 3 Arten geschehen:

  1. CPU Rechenzeiterschöpfung: Das IDS wird dazu gebracht eine Menge unsinnige Rechenarbeit zu leisten z.B. durch fragmentierte IP-Pakete.

  2. Arbeitsspeichererschöpfung: Das IDS wird gezwungen eine Menge TCP-Verbindungen zu speichern um sie einer so genannten „stateful inspection“, also einer Zustandsuntersuchung zu unterziehen.

  3. Erschöpfung von Bandbreite: Intrusion Detection Systeme haben Schwierigkeiten allen Netzwerkverkehr zu analysieren wenn die Netzwerklast sehr hoch ist (siehe auch Kapitel 4.3.1).

- Die Richtigkeit und Angemessenheit der Angriffssignaturen von Intrusion Detection Systeme ist nicht bestimmbar. Meistens scheinen diese Signaturen, auf einem sehr konkreten Niveau, nicht sehr weit entfernt von einfachen Zeichenkettenvergleichen zu sein. Dies ist aber nur für sehr einfachen Angriffe brauchbar, aber nicht bei mehrschrittigen Angriffen brauchbar .

- Das Intrusion Detection System kann selbst Sicherheitslöcher haben.
  Aktuell ist am 16.04.2003 eine Sicherheitslücke in Snort bekannt geworden, die es erlaubt beliebigen Programmcode auf dem Rechner auf dem Snort läuft auszuführen und außerdem das IDS lahm legen kann. Siehe [24] .

- Berechtigter Netzwerkverkehr enthält oft auch gewisse Pakettypen, die auch bei Angriffen auftreten. Dadurch kommt es oft zu falschen Alarmen. Es gibt daher auch Meinungen die das gesamte Konzept der Netzwerk Intrusion Detection Systeme für zum Scheitern verurteilt sehen (siehe [1] S.78). Einige Beispiele für seltsamen aber berechtigten Netzwerkverkehrs sind:

  1. Das Auftreten vieler FIN- und RST Pakete.

  2. Fragmentierte Pakete mit gesetztem „Nicht fragmentieren“-Flag.

  3. Besonders kleine Pakete, aber berechtigte Pakete

  4. Daten die beim Neuversand verändert übertragen werden.

  Dies führte zu der Meinung, das System-Administratoren, aufgrund der hohen Rate an falschen Alarmen, alle Warnungen des Netzwerk Intrusion Detection Systems ignorieren.

Kapitel 6

Abschließende Beurteilung

Die verschiedenen Netzwerk basierten Intrusion Detection Systeme sind *Wege zur Entdeckung von Einbrüchen in Computernetze*.

Mit dem Kurzüberblick, über einige der bekannten Intrusion Detection Systeme, dürfte klar geworden sein, das es im Grunde zwei Verfahren gibt nach dem ein IDS arbeiten kann. Die Untersuchung des Netzwerkverkehrs auf bekannte Angriffssignaturen, sowie die statistische Analyse auf Abweichungen gegenüber dem normalen Netzwerkverkehr. Dabei dürfte auch klar geworden sein, das das überwiegend eingesetzte Verfahren die Verwendung von Angriffssignaturen ist.
Von den vorgestellten Programmen dürfte insbesondere das kostenfreie Open Source Programm Snort favorisiert werden.

Die Möglichkeiten eines Intrusion Detection Systems sollten allerdings nicht überschätzt werden. Eine vollständige Sicherheit das alle Einbrüche und Angriffe auf das eigne Computernetzwerk mit einem IDS erkannt werden gibt es nicht.
Weder bei Anwendung der Methode der Angriffssignaturen, noch bei der Verwendung einer statistischen Analyse zur Erkennung von Abweichungen. Dazu ist der Unterschied zwischen berechtigtem Netzwerkverkehr und Einbruch bzw. Angriff zu schwer zu definieren.

Außerdem sollte der Administrator, der ein Netzwerk Intrusion Detection System einsetzt, sein von ihm verwendete System so individuell anpassen, so das es nicht zu viele falsche Alarmmeldungen ausgibt, die er nicht mehr nachgehen will oder kann. Die Gefahr die es bedeuten würde, aufgrund zu vieler Fehlalarme, gar keine Warnmeldungen mehr zu beachten ist nicht einzuschätzen.

Einbrecher und Angreifer auf ein Computernetz entwickeln immer neue Vorgehensweisen. Netzwerk Intrusion Detection Systeme müssen sich daher ständig weiterentwickeln, um mit den immer raffinierter werdenden Methoden der Einbrecher und Angreifer mithalten zu können.

Klar sollte aber auch sein, das es Angriffsarten gibt die prinzipel nicht durch ein Netzwerk basiertes Intrusion Detection System feststellbar sind. Verschlüsselter Netzwerkverkehr kann nicht durch eine Technik analysiert werden, die darauf angewiesen ist am Netzwerk zu lauschen. Dies impliziert auch, dass der Einsatz, eines Netzwerk basierten Intrusion Detection System, in einem geswitchten Netzwerk nur schwierig sinnvoll zu realisieren ist.

In dieser Studienarbeit, in der ich Beispielhaft einige kostenfreie Netzwerk basierte Intrusion Detection Systeme gestetet habe, ist aber zumindest klar geworden, das es Vorbereitungen für Einbrüche und Angriffe auf Computernetze, sowie Angriffsarten gibt die ein Intrusion Detection System sehr gut feststellen kann.

Am Ende ist festzuhalten, das Netzwerk basierte Intrusion Detection Systeme zwar keine 100%ige Sicherheit bieten können, aber zumindest einige Arten von Einbrüchen und Angriffen auf Computernetze feststellen können und damit eine gute Ergänzung zu einer Firewall sind.

Anhänge
A. Portliste

Auszugsweise aus [2] , eine etwas ausführlichere Liste als hier findet sich auch auf jedem PC mit Betriebsystem das den Netzwerkzugang unterstützt.
(/etc/services unter Linux und anderen unixartigen Betriebsystemen,
C:\windows\services unter Windows 95/98/ME
C:\winnt\system32\drivers\etc\services unter WindowsNT/2000
C:\windows\system32\drivers\etc\services unter WindowsXP)
Vollständig ist dagegen nur die Liste unter [2] .

Schlüsselwort	Port	Beschreibung
	0/tcp	Reserved
	0/udp	Reserved
tcpmux	1/tcp	TCP Port Service Multiplexer
tcpmux	1/udp	TCP Port Service Multiplexer
compressnet	2/tcp	Management Utility
compressnet	2/udp	Management Utility
compressnet	3/tcp	Compression Process
compressnet	3/udp	Compression Process
rje	5/tcp	Remote Job Entry
rje	5/udp	Remote Job Entry
echo	7/tcp	Echo
echo	7/udp	Echo
discard	9/tcp	Discard
discard	9/udp	Discard
systat	11/tcp	Active Users
systat	11/udp	Active Users
daytime	13/tcp	Daytime (RFC 867)
daytime	13/udp	Daytime (RFC 867)
qotd	17/tcp	Quote of the Day
qotd	17/udp	Quote of the Day
msp	18/tcp	Message Send Protocol
msp	18/udp	Message Send Protocol
chargen	19/tcp	Character Generator
chargen	19/udp	Character Generator
ftp-data	20/tcp	File Transfer [Default Data]
ftp-data	20/udp	File Transfer [Default Data]
ftp	21/tcp	File Transfer [Control]
ftp	21/udp	File Transfer [Control]
ssh	22/tcp	SSH Remote Login Protocol
ssh	22/udp	SSH Remote Login Protocol
telnet	23/tcp	Telnet
telnet	23/udp	Telnet
smtp	25/tcp	Simple Mail Transfer
smtp	25/udp	Simple Mail Transfer
nsw-fe	27/tcp	NSW User System FE
nsw-fe	27/udp	NSW User System FE
msg-icp	29/tcp	MSG ICP
msg-icp	29/udp	MSG ICP
msg-auth	31/tcp	MSG Authentication
msg-auth	31/udp	MSG Authentication
dsp	33/tcp	Display Support Protocol
dsp	33/udp	Display Support Protocol
time	37/tcp	Time
time	37/udp	Time
rap	38/tcp	Route Access Protocol
rap	38/udp	Route Access Protocol
rlp	39/tcp	Resource Location Protocol
rlp	39/udp	Resource Location Protocol
graphics	41/tcp	Graphics
graphics	41/udp	Graphics
name	42/tcp	Host Name Server
name	42/udp	Host Name Server
nameserver	42/tcp	Host Name Server
nameserver	42/udp	Host Name Server
nicname	43/tcp	Who Is
nicname	43/udp	Who Is
mpm-flags	44/tcp	MPM FLAGS Protocol
mpm-flags	44/udp	MPM FLAGS Protocol
mpm	45/tcp	Message Processing Module [recv]
mpm	45/udp	Message Processing Module [recv]
mpm-snd	46/tcp	MPM [default send]
mpm-snd	46/udp	MPM [default send]
ni-ftp	47/tcp	NI FTP
ni-ftp	47/udp	NI FTP
auditd	48/tcp	Digital Audit Daemon
auditd	48/udp	Digital Audit Daemon
tacacs	49/tcp	Login Host Protocol (TACACS)
tacacs	49/udp	Login Host Protocol (TACACS)
re-mail-ck	50/tcp	Remote Mail Checking Protocol
re-mail-ck	50/udp	Remote Mail Checking Protocol
la-maint	51/tcp	IMP Logical Address Maintenance
la-maint	51/udp	IMP Logical Address Maintenance
xns-time	52/tcp	XNS Time Protocol
xns-time	52/udp	XNS Time Protocol
domain	53/tcp	Domain Name Server
domain	53/udp	Domain Name Server
xns-ch	54/tcp	XNS Clearinghouse
xns-ch	54/udp	XNS Clearinghouse
isi-gl	55/tcp	ISI Graphics Language
isi-gl	55/udp	ISI Graphics Language
xns-auth	56/tcp	XNS Authentication
xns-auth	56/udp	XNS Authentication
xns-mail	58/tcp	XNS Mail
xns-mail	58/udp	XNS Mail
ni-mail	61/tcp	NI MAIL
ni-mail	61/udp	NI MAIL
acas	62/tcp	ACA Services
acas	62/udp	ACA Services
whois++	63/tcp	whois++
whois++	63/udp	whois++
covia	64/tcp	Communications Integrator (CI)
covia	64/udp	Communications Integrator (CI)
tacacs-ds	65/tcp	TACACS-Database Service
tacacs-ds	65/udp	TACACS-Database Service
sql*net	66/tcp	Oracle SQL*NET
sql*net	66/udp	Oracle SQL*NET
bootps	67/tcp	Bootstrap Protocol Server
bootps	67/udp	Bootstrap Protocol Server
bootpc	68/tcp	Bootstrap Protocol Client
bootpc	68/udp	Bootstrap Protocol Client
tftp	69/tcp	Trivial File Transfer
tftp	69/udp	Trivial File Transfer
gopher	70/tcp	Gopher
gopher	70/udp	Gopher
netrjs-1	71/tcp	Remote Job Service
netrjs-1	71/udp	Remote Job Service
netrjs-2	72/tcp	Remote Job Service
netrjs-2	72/udp	Remote Job Service
netrjs-3	73/tcp	Remote Job Service
netrjs-3	73/udp	Remote Job Service
netrjs-4	74/tcp	Remote Job Service
netrjs-4	74/udp	Remote Job Service
deos	76/tcp	Distributed External Object Store
deos	76/udp	Distributed External Object Store
vettcp	78/tcp	vettcp
vettcp	78/udp	vettcp
finger	79/tcp	Finger
finger	79/udp	Finger
http	80/tcp	World Wide Web HTTP
http	80/udp	World Wide Web HTTP
www	80/tcp	World Wide Web HTTP
www	80/udp	World Wide Web HTTP
www-http	80/tcp	World Wide Web HTTP
www-http	80/udp	World Wide Web HTTP
hosts2-ns	81/tcp	HOSTS2 Name Server
hosts2-ns	81/udp	HOSTS2 Name Server
xfer	82/tcp	XFER Utility
xfer	82/udp	XFER Utility
mit-ml-dev	83/tcp	MIT ML Device
mit-ml-dev	83/udp	MIT ML Device
ctf	84/tcp	Common Trace Facility
ctf	84/udp	Common Trace Facility
mit-ml-dev	85/tcp	MIT ML Device
mit-ml-dev	85/udp	MIT ML Device
mfcobol	86/tcp	Micro Focus Cobol
mfcobol	86/udp	Micro Focus Cobol
kerberos	88/tcp	Kerberos
kerberos	88/udp	Kerberos
su-mit-tg	89/tcp	SU/MIT Telnet Gateway
su-mit-tg	89/udp	SU/MIT Telnet Gateway
dnsix	90/tcp	DNSIX Securit Attribute Token Map
dnsix	90/udp	DNSIX Securit Attribute Token Map
mit-dov	91/tcp	MIT Dover Spooler
mit-dov	91/udp	MIT Dover Spooler
npp	92/tcp	Network Printing Protocol
npp	92/udp	Network Printing Protocol
dcp	93/tcp	Device Control Protocol
dcp	93/udp	Device Control Protocol
objcall	94/tcp	Tivoli Object Dispatcher
objcall	94/udp	Tivoli Object Dispatcher
supdup	95/tcp	SUPDUP
supdup	95/udp	SUPDUP
dixie	96/tcp	DIXIE Protocol Specification
dixie	96/udp	DIXIE Protocol Specification
swift-rvf	97/tcp	Swift Remote Virtural File Protocol
swift-rvf	97/udp	Swift Remote Virtural File Protocol
tacnews	98/tcp	TAC News
tacnews	98/udp	TAC News
metagram	99/tcp	Metagram Relay
metagram	99/udp	Metagram Relay
newacct	100/tcp	[unauthorized use]
hostname	101/tcp	NIC Host Name Server
hostname	101/udp	NIC Host Name Server
iso-tsap	102/tcp	ISO-TSAP Class 0
iso-tsap	102/udp	ISO-TSAP Class 0
gppitnp	103/tcp	Genesis Point-to-Point Trans Net

Schlüsselwort	Port	Beschreibung
gppitnp	103/udp	Genesis Point-to-Point Trans Net
acr-nema	104/tcp	ACR-NEMA Digital Imag. & Comm. 300
acr-nema	104/udp	ACR-NEMA Digital Imag. & Comm. 300
cso	105/tcp	CCSO name server protocol
cso	105/udp	CCSO name server protocol
csnet-ns	105/tcp	Mailbox Name Nameserver
csnet-ns	105/udp	Mailbox Name Nameserver
3com-tsmux	106/tcp	3COM-TSMUX
3com-tsmux	106/udp	3COM-TSMUX
rtelnet	107/tcp	Remote Telnet Service
rtelnet	107/udp	Remote Telnet Service
snagas	108/tcp	SNA Gateway Access Server
snagas	108/udp	SNA Gateway Access Server
pop2	109/tcp	Post Office Protocol - Version 2
pop2	109/udp	Post Office Protocol - Version 2
pop3	110/tcp	Post Office Protocol - Version 3
pop3	110/udp	Post Office Protocol - Version 3
sunrpc	111/tcp	SUN Remote Procedure Call
sunrpc	111/udp	SUN Remote Procedure Call
mcidas	112/tcp	McIDAS Data Transmission Protocol
mcidas	112/udp	McIDAS Data Transmission Protocol
ident	113/tcp	
auth	113/tcp	Authentication Service
auth	113/udp	Authentication Service
audionews	114/tcp	Audio News Multicast
audionews	114/udp	Audio News Multicast
sftp	115/tcp	Simple File Transfer Protocol
sftp	115/udp	Simple File Transfer Protocol
ansanotify	116/tcp	ANSA REX Notify
ansanotify	116/udp	ANSA REX Notify
uucp-path	117/tcp	UUCP Path Service
uucp-path	117/udp	UUCP Path Service
sqlserv	118/tcp	SQL Services
sqlserv	118/udp	SQL Services
nntp	119/tcp	Network News Transfer Protocol
nntp	119/udp	Network News Transfer Protocol
cfdptkt	120/tcp	CFDPTKT
cfdptkt	120/udp	CFDPTKT
erpc	121/tcp	Encore Expedited Remote Pro.Call
erpc	121/udp	Encore Expedited Remote Pro.Call
smakynet	122/tcp	SMAKYNET
smakynet	122/udp	SMAKYNET
ntp	123/tcp	Network Time Protocol
ntp	123/udp	Network Time Protocol
ansatrader	124/tcp	ANSA REX Trader
ansatrader	124/udp	ANSA REX Trader
locus-map	125/tcp	Locus PC-Interface Net Map Ser
locus-map	125/udp	Locus PC-Interface Net Map Ser
locus-con	127/tcp	Locus PC-Interface Conn Server
locus-con	127/udp	Locus PC-Interface Conn Server
gss-xlicen	128/tcp	GSS X License Verification
gss-xlicen	128/udp	GSS X License Verification
pwdgen	129/tcp	Password Generator Protocol
pwdgen	129/udp	Password Generator Protocol
cisco-fna	130/tcp	cisco FNATIVE
cisco-fna	130/udp	cisco FNATIVE
cisco-tna	131/tcp	cisco TNATIVE
cisco-tna	131/udp	cisco TNATIVE
cisco-sys	132/tcp	cisco SYSMAINT
cisco-sys	132/udp	cisco SYSMAINT
statsrv	133/tcp	Statistics Service
statsrv	133/udp	Statistics Service
ingres-net	134/tcp	INGRES-NET Service
ingres-net	134/udp	INGRES-NET Service
epmap	135/tcp	DCE endpoint resolution
epmap	135/udp	DCE endpoint resolution
profile	136/tcp	PROFILE Naming System
profile	136/udp	PROFILE Naming System
netbios-ns	137/tcp	NETBIOS Name Service
netbios-ns	137/udp	NETBIOS Name Service
netbios-dgm	138/tcp	NETBIOS Datagram Service
netbios-dgm	138/udp	NETBIOS Datagram Service
netbios-ssn	139/tcp	NETBIOS Session Service
netbios-ssn	139/udp	NETBIOS Session Service
emfis-data	140/tcp	EMFIS Data Service
emfis-data	140/udp	EMFIS Data Service
emfis-cntl	141/tcp	EMFIS Control Service
emfis-cntl	141/udp	EMFIS Control Service
bl-idm	142/tcp	Britton-Lee IDM
bl-idm	142/udp	Britton-Lee IDM
imap	143/tcp	Internet Message Access Protocol
imap	143/udp	Internet Message Access Protocol
uma	144/tcp	Universal Management Architecture
uma	144/udp	Universal Management Architecture
uaac	145/tcp	UAAC Protocol
uaac	145/udp	UAAC Protocol
iso-tp0	146/tcp	ISO-IP0
iso-tp0	146/udp	ISO-IP0
iso-ip	147/tcp	ISO-IP
iso-ip	147/udp	ISO-IP
jargon	148/tcp	Jargon
jargon	148/udp	Jargon
aed-512	149/tcp	AED 512 Emulation Service
aed-512	149/udp	AED 512 Emulation Service
sql-net	150/tcp	SQL-NET
sql-net	150/udp	SQL-NET
...		
https	443/tcp	http protocol over TLS/SSL
https	443/udp	http protocol over TLS/SSL
snpp	444/tcp	Simple Network Paging Protocol
snpp	444/udp	Simple Network Paging Protocol
microsoft-ds	445/tcp	Microsoft-DS
microsoft-ds	445/udp	Microsoft-DS
ddm-rdb	446/tcp	DDM-RDB
ddm-rdb	446/udp	DDM-RDB
ddm-dfm	447/tcp	DDM-RFM
ddm-dfm	447/udp	DDM-RFM
...		
uucp	540/tcp	uucpd
uucp	540/udp	uucpd
uucp-rlogin	541/tcp	uucp-rlogin
uucp-rlogin	541/udp	uucp-rlogin
commerce	542/tcp	commerce
commerce	542/udp	commerce
klogin	543/tcp	
klogin	543/udp	
kshell	544/tcp	krcmd
kshell	544/udp	krcmd
appleqtcsrvr	545/tcp	appleqtcsrvr
appleqtcsrvr	545/udp	appleqtcsrvr
dhcpv6-client	546/tcp	DHCPv6 Client
dhcpv6-client	546/udp	DHCPv6 Client
dhcpv6-server	547/tcp	DHCPv6 Server
dhcpv6-server	547/udp	DHCPv6 Server
afpovertcp	548/tcp	AFP over TCP
afpovertcp	548/udp	AFP over TCP
idfp	549/tcp	IDFP
idfp	549/udp	IDFP
new-rwho	550/tcp	new-who
new-rwho	550/udp	new-who
cybercash	551/tcp	cybercash
cybercash	551/udp	cybercash
devshr-nts	552/tcp	DeviceShare
devshr-nts	552/udp	DeviceShare
pirp	553/tcp	pirp
pirp	553/udp	pirp
rtsp	554/tcp	Real Time Stream Control Protocol
rtsp	554/udp	Real Time Stream Control Protocol
dsf	555/tcp	
dsf	555/udp	
remotefs	556/tcp	rfs server
remotefs	556/udp	rfs server
openvms-sysipc	557/tcp	openvms-sysipc
openvms-sysipc	557/udp	openvms-sysipc
sdnskmp	558/tcp	SDNSKMP
sdnskmp	558/udp	SDNSKMP
teedtap	559/tcp	TEEDTAP
teedtap	559/udp	TEEDTAP
rmonitor	560/tcp	rmonitord
rmonitor	560/udp	rmonitord
monitor	561/tcp	
monitor	561/udp	
chshell	562/tcp	chcmd
chshell	562/udp	chcmd
nntps	563/tcp	nntp protocol over TLS/SSL (was snntp)
nntps	563/udp	nntp protocol over TLS/SSL (was snntp)
9pfs	564/tcp	plan 9 file service
9pfs	564/udp	plan 9 file service
whoami	565/tcp	whoami
whoami	565/udp	whoami
streettalk	566/tcp	streettalk
streettalk	566/udp	streettalk
banyan-rpc	567/tcp	banyan-rpc
banyan-rpc	567/udp	banyan-rpc
ms-shuttle	568/tcp	microsoft shuttle
ms-shuttle	568/udp	microsoft shuttle
ms-rome	569/tcp	microsoft rome
ms-rome	569/udp	microsoft rome
meter	570/tcp	demon
meter	570/udp	demon
meter	571/tcp	udemon
meter	571/udp	udemon
sonar	572/tcp	sonar
sonar	572/udp	sonar
banyan-vip	573/tcp	banyan-vip
banyan-vip	573/udp	banyan-vip
ftp-agent	574/tcp	FTP Software Agent System
ftp-agent	574/udp	FTP Software Agent System
vemmi	575/tcp	VEMMI
vemmi	575/udp	VEMMI
ipcd	576/tcp	ipcd
ipcd	576/udp	ipcd
vnas	577/tcp	vnas
vnas	577/udp	vnas
ipdd	578/tcp	ipdd
ipdd	578/udp	ipdd
...		
	1024/tcp	Reserved
	1024/udp	Reserved
blackjack	1025/tcp	network blackjack
blackjack	1025/udp	network blackjack
cap	1026/tcp	Calender Access Protocol
cap	1026/udp	Calender Access Protocol
iad1	1030/tcp	BBN IAD
iad1	1030/udp	BBN IAD
iad2	1031/tcp	BBN IAD
iad2	1031/udp	BBN IAD
iad3	1032/tcp	BBN IAD
iad3	1032/udp	BBN IAD
netinfo-local	1033/tcp	local netinfo port
netinfo-local	1033/udp	local netinfo port
pcg-radar	1036/tcp	RADAR Service Protocol
pcg-radar	1036/udp	RADAR Service Protocol
netarx	1040/tcp	Netarx
netarx	1040/udp	Netarx
fpitp	1045/tcp	Fingerprint Image Transfer Protocol
fpitp	1045/udp	Fingerprint Image Transfer Protocol
neod1	1047/tcp	Sun's NEO Object Request Broker
neod1	1047/udp	Sun's NEO Object Request Broker
neod2	1048/tcp	Sun's NEO Object Request Broker
neod2	1048/udp	Sun's NEO Object Request Broker
...		

B. Bezugsquellen der im Test verwendeten Programme

Herausforderungen an die IDS-Systeme

1. Portscanner

Superscan	CD zum Buch „Hacking Exposed : Network Security Secrets & Solutions“ (siehe Literaturverzeichnis [13]) oder Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])
WUPS	http://www.ntsecurity.nu
IPEYE	http://www.ntsecurity.nu

2. NetBIOS-Freigaben auskundschaften

`net use ...`	Windows Boardmittel
WINFO	http://www.ntsecurity.nu/toolbox/winfo/

3. Passwörter erraten

Brutus AET2	Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])
Unsecure	Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])
Webcracker	Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])

4. Denial of Service

rpcnuke	http://toolz.pluto01.de/page/index.php?site=nuker
Meliksah Nuke	http://www.Hacker-Toolz.info
Hacknuker	http://www.Hacker-Toolz.info
CGSi_OOB	http://www.Hacker-Toolz.info
IGMP-Nuke	http://www.Hacker-Toolz.info
DDoS-Ping	Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])
SYN-Flooder	Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])

5.Security-Scanner

LANguard Network Scanner	Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])
Cerberos Internet Scanner	Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])
Winfingerprint	Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])

6.Trojaner-Scanner

Back Orifice Pinger (BO-Ping)	Steganos Anti Hacker 1.5 (siehe Literaturverzeichnis [23])

Die oben und auf der letzten Seite aufgeführten Programme sind ausschließlich für den Test eigener Netze und Systeme gedacht. Ein Einsatz in öffentlichen Netzen könnte strafrechtliche Folgen nach sich ziehen.

Freie Network Intrusion Detection Systeme

Snort	http://www.snort.org
Pakemon IDS	http://www.sfc.keio.ac.jp/ ~keiji/backup/ids/pakemon/
Panoptis	http://panoptis.sourceforge.net

Eingesetzte Server

NetBIOS	Windows Boardmittel
Apache 2.0.44	http://www.apache.org
CesarFTP	z.B. http://www.webattack.com/get/cesarftp.shtml

C. Literaturverzeichnis

1. Englischsprachige Literatur:

[1] Julia Allen, Alan Christie, William Fithen, John McHugh, Jed Pickel und Ed Stoner, Januar 2000 , State of the Practice of Intrusion Technologies Technical Report, Software Engineering Institute, Carnegie Mellon University , Pittsburgh , PA 15213
http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html

[2] IANA, 17.01.2003 ,Port-Numbers , IANA
http://www.iana.org/assignments/port-numbers

[3] Allen Householder, Kevin Houle , Chad Dougherty, 2002 , Computer Attack Trends Challenge Internet Security, CERT Coordination Center
http://www.computer.org/security/supplement1/hou/

[4] NFR® , *Data Sheet* und *Product Overview , NFR®*
http://www.nfr.com

[5] AirDefense Overview , AirDefence INC.
http://www.airdefense.net

[6] nPatrol (Webseite)
http://www.nsecure.net

[7] Demarc , PureSecure Software , Demarc Security Inc.
http://www.demarc.com

[8] Intrusion Detection: The Dragon Intrusion Detection Software System
http://www.securityware.co.uk/intrusion-detection

[9] Snort – The Open Source Network Intrusion Detection System
http://www.snort.org

[10] The Honeynet Projekt
http://www.honeynet.org

[11] Panoptis: A project to detect and block DoS/DDoS attacks
http://panoptis.sourceforge.net

[12] Pakemon
http://www.sfc.keio.ac.jp/~keiji/backup/ids/pakemon/

[13] Stuart McClure, Joel Scambray , George Kurtz , 2001, Hacking Exposed - Network Security Secrets & Solutions , Third Edition , Osborne/McGraw-Hill

[14] Manfred Bortenschlager, Martin Essl , Christopher Trauner , u.a. , Project ´OtO` - Observing the Observers -Kapitel Attacks , FH Salzburg/Interphasetech .
http://www.students.fh-sbg.ac.at/~messl/

[15] Richard A. Kemmerer , Giovanni Vigna, 2002. Intrusion Detection: A Brief History and Overview. Reliable Software Group, Computer Science Deparment, University of California Santa Barbara, Security & Privacy 2002.

[16] Constantine Manikopoulos , Symeon Papavassiliou, 2002. Network Intrusion and Fault Detection: A Statistical Anomaly Approach. New Jersey Institute of Technology. IEEE Communications Magazine – October 2002.

2. Deutschsprachige Literatur:

[17] Martin Freiss , 1999. Alarmanlagen fürs Netz – Intrusion Detection Systems erkennen Angriffe. C´t – Magazin für Computertechnik. Ausgabe 03/1999 , S. 186-189

[18] Martin Freiss, Jürgen Schmidt , 2001. Einbrecheralarm – Intrusion Detection mit Snort. C´t – Magazin für Computertechnik.
Ausgabe 08/2001 , S. 212-219

[19] Jürgen Kuri, 1999. Spinne im Netz – Vom Hub bis zum Switch.C't – Magazin für Computertechnik. Ausgabe 17/1999 , S. 106-112

[20] Bernd Rudack, Martin Freiss, 1999. Schriller die Glocken nie klingen – Intrusion Detection Systeme im Test. C´t – Magazin für Computertechnik. Ausgabe 03/1999, S.190-193

[21] Jürgen Schmidt , 1998. Netcat – ein Tool nicht nur für Netzwerker.
C´t – Magazin für Computertechnik. Ausgabe 8/1998 , S.192-193
Kasten „Die schwarze Kunst des Port-Scans“ S.193

[22] Hendrik Busch (hb@ping.de) , 2002 . Funknetzwerke und Sicherheit in Funknetzwerken, Verein zur Förderung der privaten Internetnutzung e.V. (PING e.V.) , S.16
http://www.ping.de/~hb/download/vortrag_funknetze.pdf

[23] Steganos Hacker Tools 1.5 .
com-Online. Ausgabe 5/2003 , Heft-CD

[24] pab , 16.04.2003. Sicherheitsloch in Snort.
Heise Online Meldung vom 16.04.2003 12:57
http://www.heise.de/newsticker/data/pab-16.04.03-000/