Ein Grid-basiertes, föderiertes Intrusion Detection System zur Sicherung der D-Grid Infrastruktur (GIDS)

GIDS – Prototypische Implementierung (MS 28)
Meilenstein zum Abschluss des Arbeitspakets 6

Aktualisierung vom 15.02.2012

Autoren:

Dr. Wolfgang Hommel	(Leibniz-Rechenzentrum)
Dr. Nils gentschen Felde	(Ludwig-Maximilians-Universität München)
Felix von Eye	(Leibniz-Rechenzentrum)
Jan Kohlrausch	(DFN-CERT GmbH)
Christian Szongott	(Regionales Rechenzentrum für Niedersachsen)

GEFÖRDERT VOM

Inhaltsverzeichnis

Kapitel 1

Einleitung

Dieses Dokument präsentiert zusammenfassend die Ergebnisse des Arbeitspakets 6 des Projekts „Ein Grid-basiertes, föderiertes Intrusion Detection System zur Sicherung der D-Grid Infrastruktur“ (GIDS). GIDS (`http://www.grid-ids.de`) ist ein Teilprojekt im Rahmen des D-Grid (`http://www.d-grid.de`) und wird vom Bundesministerium für Bildung und Forschung (BMBF, `http://www.bmbf.de`) gefördert. Weitere Projektinformationen und Unterlagen können der Projekt-Webseite entnommen werden.

1.1 Ziel

Ziel dieses Projekts ist die Bereitstellung eines GIDS-Dienstes für das D-Grid. Hierbei gilt es, soweit wie möglich bestehende Ansätze zu integrieren und ein domänen- und organisationsübergreifendes Gesamtsystem zu entwickeln. Insbesondere die Fähigkeit, mit Virtuellen Organisationen (VO) umzugehen und diese auch als Kunden in Betracht zu ziehen, ist dabei von entscheidender Bedeutung. Die Grundidee ist es, Angriffe durch die kooperative Nutzung und Auswertung von lokalen Sicherheitssystemen zu erkennen. Dazu ist der Austausch von Angriffsdaten und somit deren datenschutzkonforme Aufarbeitung, auch zur Wahrung individuell bestehender Sicherheits- und Informationsverbreitungsrichtlinien, notwendig. In einem kooperativen IDS besteht die Möglichkeit, Angriffe schneller zu erkennen, als dies mit unabhängigen und nur die lokale Sicht berücksichtigenden Sicherheitssystemen möglich ist. Somit kann eine Verkürzung der Reaktionszeit der beteiligten Parteien erzielt werden. Weiter können Vorwarnungen, an zum Zeitpunkt der Erkennung eines Angriffs noch nicht betroffenen Parteien, herausgegeben sowie gegebenfalls präventive Gegenmaßnahmen ergriffen werden.

Eine Auswertung der Daten kann sich zu großen Teilen auf bereits vorhandene Ansätze klassischer IDS stützen. Bei der Auswertung der verfügbaren Datengrundlage ist darauf zu achten, dass VO-spezifische Zugriffsrechte und Befugnisse eingehalten werden. Nach erfolgreicher Auswertung aller verfügbaren Informationen durch ein kooperatives und föderiertes GIDS, unter Beachtung individueller Sicherheits- und Datenschutz-Policies, erfolgt eine Berichterstattung über die erkannten Angriffe auf das Grid oder einzelne beteiligte Partner. Auch hier ist es von Bedeutung, dass eine VO-spezifische Sicht auf die bereitgestellten Informationen realisiert wird. Dazu ist eine Anbindung an die im D-Grid bestehenden VO Managementsysteme zu schaffen. Nach der Entwicklung einer geeigneten Architektur für ein kooperatives und föderiertes IDS in Grid-Umgebungen steht die Implementierung und Produktivführung des Systems. Es soll nach Abschluss der Projektlaufzeit ein produktives Intrusion Detection System als Grid-Dienst im D-Grid zu Verfügung stehen, das sowohl von Ressourcenanbietern als auch von Kunden (VOs, Communities etc.) genutzt werden kann.

1.2 Lösungsansatz von GIDS

Bereits einleitend in [1, 3, 4] ist die Idee aufgekommen, GIDS als Föderation aus bestehenden, für die Ressourcenanbieter eines Grids spezifischen, sicherheitsrelevanten Komponenten hin zu

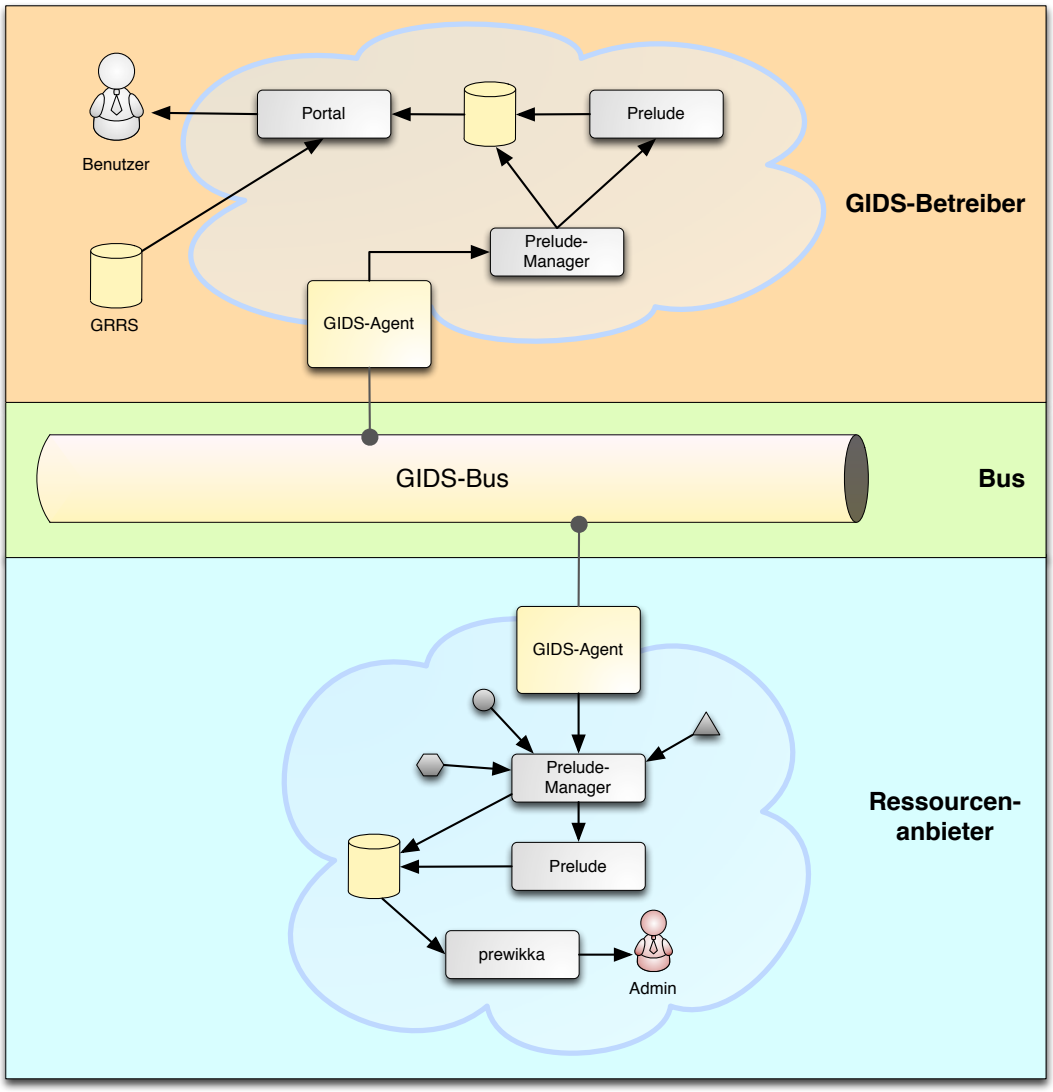

Abbildung 1.1: Überblick über den Aufbau von GIDS

einem Grid-weiten Frühwarnsystem zu konzipieren. Aus der Analyse in Kapitel 3 der genannten Arbeit [1] sind unter anderem Anforderungen abgeleitet worden, die die Autonomie der einzelnen Teilnehmer eines Grid-basierten IDS fordern, was nicht zuletzt für die Akzeptanz eines solchen Systems zwingend notwendig ist. Daraus abgeleitet bedingt sich eine verteilte Struktur und es entsteht eine lose Kopplung der am GIDS beteiligten Partner, die daraus folgend jeder für sich organisatorisch und administrativ wie auch technisch unabhängig und autonom agieren können und in vielen Bereichen sogar müssen.

Die grundsätzliche Idee von GIDS führt zu einem Grid-globalen Aufbau eines IDS wie es in Abbildung 1.1 wenig detailliert aus einer Vogelperspektive dargestellt ist. Jeder Teilnehmer des GIDS erhält eine zentrale Datenbank, in die alle verfügbaren, für die Sicherheit relevanten Informationen abgelegt werden können. Wie bereits zuvor angesprochen können dies zum einen Rohdaten (z. B. von versuchten Zugriffen auf gesperrte Ports an einer Firewall) oder auch bereits veredelte oder aggregierte Informationen wie Berichte lokal installierter IDS sein. An einen solchen zentralen Datenspeicher angeschlossen kann ein Agent unter Beachtung einiger notwendiger Randbedingungen Informationen an ein Grid-weites IDS weiterreichen.

1.3 Struktur des Dokuments

Dieses Dokument beschreibt die prototypische Implementierung des insbesondere in [1] vorgestellten Entwurfs für GIDS. Dazu wird im folgenden eine Gliederung in Anlehnung an den in Abbildung 1.1 dargestellten Entwurf vorgenommen. Kapitel 2 beschreibt den GIDS-Bus in seiner Implementierung und die notwendigen Schritte, um als Ressourcenanbieter Zugriff auf den GIDS-Bus zu erlangen. Nachfolgend beschreiben die Kapitel 3 und 4 den Aufbau, die Implementierung und die notwendigen Installations- und Konfigurationsschritte der GIDS-Komponenten auf Seiten eines Ressourcenanbieters bzw. des Betreibers von GIDS. Da im Rahmen von GIDS besonders sensible Informationen und Daten ausgetauscht werden, die besonders schützenswert sind, befasst sich Kapitel 5 mit Datenschutz und dessen (technischer) Gewährleistung, bevor abschließend Kapitel 6 die vorliegende Arbeit zusammenfasst.

Kapitel 2

Der GIDS-Bus

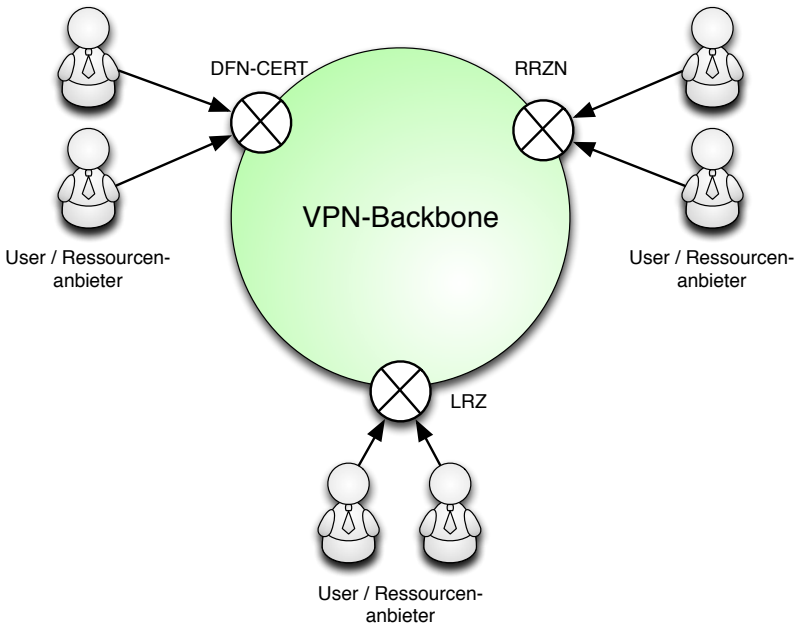

Abbildung 2.1: Grundidee zur Implementierung des GIDS-Bus auf Basis von VPN-Technologien

Die Grundidee von GIDS ist, dass vertrauliche Informationen (mit zum Teil privaten Daten) zu potentiellen Angriffen in einer Föderation (hier innerhalb des D-Grid) über unsichere Medien transportiert werden müssen, um global auf Hinweise zu Angriffen ausgewertet werden zu können. Hierfür gilt es eine Lösung zu finden, die einerseits die Sicherheit der Daten gewährleisten und andererseits alle Anforderungen, die sich durch die Infrastruktur selbst ergeben, erfüllen kann.

Über ein *Virtual Private Network* (VPN) können zwei oder mehr private Netze miteinander verbunden werden. Durch die zusätzliche Verwendung von kryptografischen Verfahren erfolgt die Kommunikation geschützt. Somit kann ein privates Netz auch auf unsicheren Leitungen bereitgestellt werden. In Bezug auf das GIDS-Projekt ist die Realisierung des GIDS-Bus mithilfe dieser Technik denkbar. Zur Nachbildung einer Bus-artigen Kommunikationsstruktur soll eine Multicast-fähige Infrastruktur implementiert werden, die die Vertraulichkeit, Authentizität und Integrität von übertragenen Daten gewährleistet. Auf Grund der Multicast-Fähigkeit kann dieser Ansatz dann im Weiteren als GIDS-Bus fungieren.

Bei den meisten VPN-Lösungen ist jedoch die redundante Auslegung der Zugangs-Server

problematisch. Für eine maximale Ausfallsicherheit im Rahmen von GIDS soll nicht nur eine redundante Auslegung, sondern vielmehr auch eine örtliche und organisatorische Verteilung der Zugangs-Server realisiert werden. Den Implementierungsansatz dazu illustriert Abbildung 2.1. Im Rahmen von GIDS ist dieser Ansatz unter Nutzung des freien *OpenVPN* (`http://openvpn.net/`) realisiert.

2.1 Implementierung

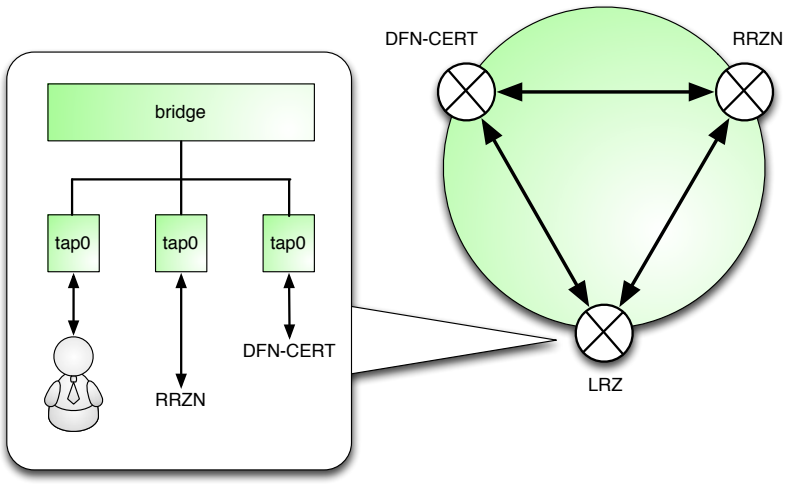

Abbildung 2.2: Implementierung des GIDS-Bus auf Basis von VPN-Technologien im Detail

Um ein Multicast-fähiges VPN mit redundanten und örtlich verteilten Zugangspunkten auf Basis von OpenVPN zu implementieren, bedarf es einer Vollvermaschung der existierenden VPN-Zugangs-Server im VPN-Backend (derzeit beim DFN-CERT, dem LRZ und dem RRZN installiert). Eine graphische Repräsentation der GIDS-Bus-Implementierung ist in Abbildung 2.2 zu finden.

Um eine Vollvermaschung der verfügbaren VPN-Server zu erreichen, werden jeweils paarweise VPN-Tunnel zwischen allen VPN-Servern eingerichtet. OpenVPN bietet neben der Tunnelung von ISO/OSI-Schicht 3 Verkehr (`tun device`) auch das Tunneln von ISO/OSI-Schicht 2 Verkehr unter Nutzung eines `tap device` an, von dem hier Gebrauch gemacht wird. Durch diesen Ansatz bedingt, entstehen in der derzeitigen Ausprägung mit insgesamt drei VPN-Servern an jedem Zugangspunkt drei `tap devices` – eines dient zur Entgegennahme neuer VPN-Verbindungen der Teilnehmer, zwei dienen der Kommunikation mit den anderen VPN-Servern.

Alle so entstehenden `tap devices` werden im Folgenden über ein *bridge device* miteinander verbunden. Wie aus Abbildung 2.2 jedoch hervorgeht, ist einer solchen Vernetzung Obacht zu zollen, da hierdurch ein Ringschluss im VPN-Backbone auf ISO/OSI-Schicht 2 entsteht. Entsprechend muss an den `bridge devices` der VPN-Server im Backbone das Spanning-Tree-Protocol (STP) aktiviert sein, um zum einen mit dem Ringschluss und zum anderen mit einem potentiellen Ausfall einer der VPN-Server geeignet umgehen zu können.

Um die zuvor stehend beschriebenen Konstellationen zu erzielen, bedarf es auf der einen Seite der Installation von OpenVPN, auf die an dieser Stelle nicht weiter eingegangen wird. Das Erzeugen und Vernetzen der `tap` und `bridge devices` ist jedoch ein wenig komplexer und kann z. B. mit folgenden Skript ausgeführt werden (die notwendigen Konfigurationsparameter werden aus einer Konfigurationsdatei (hier: `config`) gelesen):

```
#!/bin/bash
```

```
#################################
# Set up Ethernet bridge on Linux
# Requires: bridge-utils
#################################

# read config
. /etc/openvpn/scripts/config

# create tap-devices
for t in $tap; do
    openvpn --mktun --dev $t
done

brctl addbr $br
brctl addif $br $eth

brctl stp br0 on
brctl setageing br0 0

for t in $tap; do
    brctl addif $br $t
done

for t in $tap; do
    ifconfig $t 0.0.0.0 promisc up
done

ifconfig $eth 0.0.0.0 promisc up

ifconfig $br $eth_ip netmask $eth_netmask broadcast $eth_broadcast
```

Man beachte, bei der Erzeugung eines `bridge device` ist per Standard das Spanning-Tree-Protocol aktiviert. Mit dem Kommando `brctl show` lassen sich die Konfiguration aller auf einem Linux-System existierenden `bridge devices` anzeigen und überprüfen.

OpenVPN bietet die Möglichkeit der Authentifikation über X.509-Zertifikate. Um eine Authentifizierung eines GIDS-Teilnehmers an jedem der verfügbaren VPN-Server mit ein und demselben Zertifikat zu erlauben, müssen alle VPN-Server zur Nutzung derselben *Certificate Authority* (CA) konfiguriert sein. Dies lässt sich jedoch trivial erreichen, indem man die Konfigurationsparameter „`ca`“ wie nachfolgend setzt:

```
# SSL/TLS root certificate (ca), certificate
# (cert), and private key (key).  Each client
# and the server must have their own cert and
# key file.  The server and all clients will
# use the same ca file.
#
# See the "easy-rsa" directory for a series
# of scripts for generating RSA certificates
# and private keys.  Remember to use
# a unique Common Name for the server
# and each of the client certificates.
#
# Any X509 key management system can be used.
```

```
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca /etc/openvpn/ca.crt
cert /etc/openvpn/server.crt
key /etc/openvpn/server.key  # This file should be kept secret
```

Zusätzlich muss selbstverständlich unter dem Pfad `/etc/openvpn/ca.crt` das jeweils gleiche Zertifikat derselben CA vorliegen.

2.2 Installation & Konfiguration eines GIDS-Agenten

Um einen Zugang zum GIDS-Bus zu erhalten, ist die Installation eines GIDS-Agenten notwendig, der in wesentlichen Teilen aus einer entsprechend konfigurierten OpenVPN-Instanz besteht. Zur Anbindung eines neuen Ressourcenanbieters an den GIDS-Bus sind folgende Schritte notwendig:

Installation von OpenVPN. Die Installation des GIDS-Agenten und somit von OpenVPN lässt sich auf einer großen Vielzahl an Betriebssystemen einfach realisieren. Die Paketierung und Unterstützung durch die OpenVPN-Community ist sehr gut, viele Betriebssystemen-spezifische Pakete und Installationsanleitungen finden sich auf der Webseite zu OpenVPN unter `http://openvpn.net/`. In vielen aktuellen Distributionen von Betriebssystemen sind OpenVPN-Pakete ebenfalls direkt vom Hersteller verfügbar und unterstützt.

Erhalt eines X.509-Zertifikats. Eine Authentifikation eines GIDS-Agenten erfolgt durch die Nutzung eines X.509-Zertifikats. Dieses Zertifikat ist kein gewöhnliches Grid-Zertifikat, wie es im D-Grid verwendet wird, sondern wird von einer eigenen CA ausgestellt. Damit ist sichergestellt, dass der Zugriff auf den GIDS-Bus, der potentiell sehr sensible Informationen verbreitet, nur von einer kleinen und geschlossenen Nutzergruppe vorgenommen werden kann. Aus diesem Grund werden für die Authentifizierung keine regulären Grid-Zertifikate zugelassen, sondern nur Zertifikate der eigenen GIDS-CA zugelassen.

Um ein GIDS-Zertifikat zu erhalten, muss sich ein Ressourcenanbieter, der an GIDS teilnehmen möchte, an den Betreiber des GIDS wenden. Nach einem Feststellungsverfahren der Identität wird ein entsprechendes GIDS-Zertifikat ausgestellt und übergeben.

Konfiguration von OpenVPN. Nachdem ein GIDS-Zertifikat erfolgreich erteilt und übermittelt wurde, stellt dieses Zertifikat zusammen mit den darin signierten Schlüsseln den Kern der Konfiguration von OpenVPN im Sinne eines GIDS-Agenten dar. Nach der erfolgreichen Installtion von GIDS ist folgende Konfiguration (natürlich in einer dem Anwendungsfall angepassten Form) zu verwenden:

```
# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.
client

# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
dev tap0

# Are we connecting to a TCP or
# UDP server?  Use the same setting as
```

```
# on the server.
proto udp

# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote GIDS-OPENVPN-SERVER-1 1194
remote GIDS-OPENVPN-SERVER-2 1194
remote GIDS-OPENVPN-SERVER-3 1194

# Choose a random host from the remote
# list for load-balancing.  Otherwise
# try hosts in the order specified.
remote-random

# The keepalive directive causes ping-like
# messages to be sent back and forth over
# the link so that each side knows when
# the other side has gone down.
# Ping every 5 seconds, assume that remote
# peer is down if no ping received during
# a 15 second time period.
keepalive 5 15

# Keep trying indefinitely to resolve the
# host name of the OpenVPN server.  Very useful
# on machines which are not permanently connected
# to the internet such as laptops.
resolv-retry infinite

# Most clients don't need to bind to
# a specific local port number.
nobind

# Downgrade privileges after initialization (non-Windows only)
user nobody
group nogroup

# Try to preserve some state across restarts.
persist-key
persist-tun

# SSL/TLS parms.
# See the server config file for more
# description.  It's best to use
# a separate .crt/.key file pair
# for each client.  A single ca
# file can be used for all clients.
ca /etc/openvpn/ca.crt
cert /etc/openvpn/client.crt
key /etc/openvpn/client.key

# Verify server certificate by checking
# that the certicate has the nsCertType
# field set to "server".  This is an
# important precaution to protect against
```

```
# a potential attack discussed here:
#  http://openvpn.net/howto.html#mitm
#
# To use this feature, you will need to generate
# your server certificates with the nsCertType
# field set to "server".  The build-key-server
# script in the easy-rsa folder will do this.
ns-cert-type server

# Enable compression on the VPN link.
# Don't enable this unless it is also
# enabled in the server config file.
comp-lzo

# Set log file verbosity.
verb 3
```

2.3 Diskussion

Bei OpenVPN handelt es sich um ein Opensource-Projekt. Entsprechend fallen für GIDS bei der Nutzung der Community-Version von OpenVPN keine Kosten an. Weiterhin ist durch die Quelloffenheit gesichert, dass eine Weiterentwicklung der Software auch nach Ausscheiden des Hauptentwicklers noch möglich ist.

Neben finanziellen Aspekten sind die technischen und entscheidenden Vorteile von OpenVPN die Möglichkeit der Authentifikation über X.509-Zertifikate, die Broad- und Multicast-Fähigkeit und das in Version 2.1 eingeführte Port-Sharing.

Authentifizierung mittels X.509-Zertifikaten Um die Kommunikation durch ein VPN absichern zu können, ist beim Verbindungsaufbau eine Authentifizierung durchzuführen. Sowohl der Server als auch der jeweilige Client müssen sicher davon ausgehen können, dass es sich beim entsprechenden Kommunikationspartner zum einen nicht um einen Angreifer handelt und zum anderen um genau das System, an das die vertraulichen Daten auch übertragen werden sollen. Im D-Grid werden zur Authentifizierung bei der Abgabe von Gridjobs und zur verschlüsselten Kommunikation sogenannte X.509-Zertifikate verwendet. Hierbei handelt es sich um von einer *Certificate Authority* (CA) ausgestellte (signierte) Zertifikate, mithilfe welcher ein öffentlicher Schlüssel eindeutig einer Entität (Benutzer/Host) im Grid zugeordnet werden kann. Die gesicherte Kommunikation setzt voraus, dass alle im Grid verwendeten Zertifikate von einer CA ausgestellt (also signiert) werden, denen alle Gridressourcen explizit ihr Vertrauen aussprechen. OpenVPN erlaubt zum Aufbau einer authentifizierten Verbindung zum OpenVPN Access Server ebenfalls X.509-Zertifikate. Die bestehende Authentifizierungsstruktur kann also ebenfalls für den GIDS-Dienst verwendet werden.

Broadcast- und Multicast-Nachrichten Die sichere Verteilung von Informationen zwischen den GIDS-Agenten und dem GIDS-Provider ist ein wesentlicher Bestandteil der GIDS-Infrastruktur. Informationen müssen nicht nur von den GIDS-Agenten an einen zentralen GIDS-Provider übermittelt werden. Eine wesentlich größere Herausforderung stellt die Übermittlung von allgemeinen Informationen an alle beteiligten GIDS-Agenten dar. Allgemeine Informationen können beispielsweise aus einer Warnung über einen aktuellen Angriff bestehen, der so auf anderen Gridressourcen proaktiv verhindert werden kann. Die Übermittlung von Daten an mehr als einen Empfänger könnte mittels OpenVPN über Broadcast-Nachrichten erfolgen. Unter der Voraussetzung, dass dem zentralen GIDS-Betreiber alle beteiligten GIDS-Agenten bekannt sind, wäre auch die Verwendung von Multicast-Nachrichten denkbar.

Port-Sharing Das seit Version 2.1 in OpenVPN eingeführte Feature *Port-Sharing* erlaubt es, den selben Port sowohl für *HTTP over SSL* (HTTPS), als auch für den VPN-Dienst bereit zu stellen. Die Verwendung des gleichen Ports für beide Dienste ist vor allem für die Administration vorteilhaft. Die bereits konfigurierten Firewallregeln müssen nicht geändert werden. Insbesondere müssen keine weiteren Ports geöffnet werden, um die korrekte Funktion des Grid-basierten IDS zu ermöglichen.

Zentraler Server als Single Point of Failure Ein Nachteil der sich aus der Nutzung von OpenVPN ergibt, ist die Notwendigkeit eines zentralen OpenVPN Access Servers. Alle Clients, die über den GIDS-Bus Daten austauschen / empfangen wollen, müssen sich hier authentifizieren und ihre Verbindung zum GIDS-VPN aufbauen. Eine zentrale Authentifizierungsstelle erscheint zwar zunächst vorteilhaft, birgt aber die Gefahr eines Single Point of Failure. Ist der OpenVPN Access Server, aus welchen Gründen auch immer, nicht erreichbar, so wird die gesamte Kommunikation des GIDS-Systems nicht mehr möglich sein. Sollte die gesamte Kommunikation in der GIDS-Infrastruktur auf den mit OpenVPN gesicherten Bus ausgelegt sein, so wäre dies ein schwerwiegendes Sicherheitsproblem.

Aus diesem Grund ist eine Fallback-Lösung unumgänglich. Es besteht die Möglichkeit, einen redundanten, zweiten OpenVPN Access Server bereitzustellen, welcher die Aufgaben des primären Servers nahtlos übernehmen kann. Dieser Server muss, um den gleichzeitigen Ausfall beider Server zu verhindern, auf einer anderen Hardware betrieben werden. Idealerweise befinden sich Primärserver und Fallback-Server nicht im gleichen Netz, im Falle des D-Grid also nicht in der gleichen Site. Auf diese Weise muss selbst der vorrübergehende Verlust einer gesamten Site, z. B. durch Netzprobleme, für das GIDS keine Ausfälle bedeuten. Bei diesem Ansatz ist die Bereitstellung einer Liste von GIDS-VPN Access Servern (mitsamt den Fallback-Systemen) nötig sein.

Zusammenfassend bietet OpenVPN einige für GIDS sehr interessante Möglichkeiten. Die Nutzung eines eigenst für die streng vertrauliche Kommunikation zwischen den GIDS-Agenten und dem GIDS-Betreiber eingerichteten Netzes ist zur Wahrung der Privatsphäre der Kunden ein probates Mittel. Ebenso ist die Verwendung etablierter Authentifizierungsmechanismen ein Vorteil. Einziger bislang bekannter Nachteil ist die Entstehung eines Single Point of Failure beim OpenVPN Access Server. Die Lösung dieses Problems wurde jedoch in Abschnitt 2.1 aufgezeigt.

Kapitel 3

Ressourcenprovider

Die zentralen Komponenten des Ressourcenbetreibers basieren auf dem *Prelude-Framework*, das als Security Information and Event Management System (SIEM) Lösung für die Sammlung, Auswertung und Weitergabe der Alarmmeldungen auf Seiten der Ressourcenprovider verantwortlich ist. Abbildung 3.1 skizziert den Überblick über die Architektur auf Seiten der Ressourcenprovider. Dabei sind die Komponenten im oberen Drittel des Bildes für die Datenakquise verantwortlich, was in Abschnitt 3.1 näher erläutert wird. Alle so gewonnen Daten werden im mittleren Teil des Bildes neben der Speicherung durch eine lokale IDS-Instanz analysiert, was in Abschnitt 3.2 vertieft wird. Schlussendlich werden die aufgearbeiteten Daten entweder dem lokalen Administrator angezeigt oder an alle anderen GIDS-Teilnehmer versendet. Diese Schritte werden in Abschnitt 3.3 beschrieben.

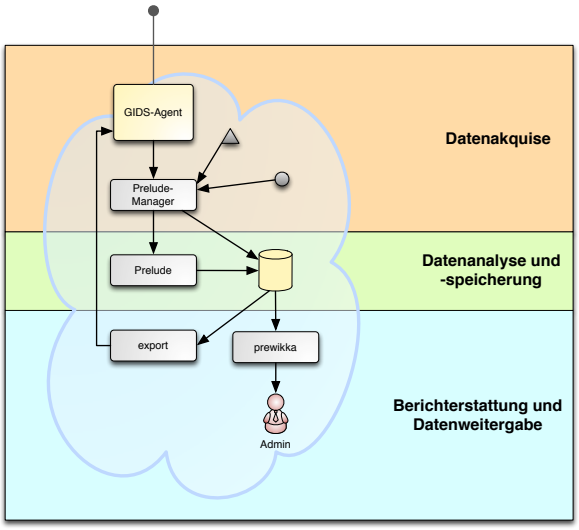

Abbildung 3.1: Überblick über die Komponenten auf Seiten des Ressourcenproviders

3.1 Datenakquise

3.1.1 libprelude

Das Prelude-Framework baut auf der Bibliothek *libprelude* auf, die Funktionalitäten, wie z. B. den verschlüsselten Transport von Daten zwischen den an libprelude angeschlossenen Komponenten und die Authentifizierung der Komponenten über X.509-Zertifikate, bietet.

3.1.1.1 Implementierung

Beim Einsatz der in C konzipierten Bibliothek libprelude wurde auf die vorhandenen Sourcen des unter GPLv2 stehenden Projekts Prelude zurückgegriffen. Für die Implementierung des GIDS-Prototypen bildet die Bibliothek einen wichtigen Baustein für die Realisierung der neu entwickelten GIDS-Komponenten. Ursprünglich konnte man unter `http://www.prelude-technologies.com/en/development/download/index.html` die Sourcen herunterladen. Um die Unabhängigkeit der Projektergebnisse zu gewährleisten, wurden die Sourcen als Git-Repository auf `http://git.lrz.de/prelude/` veröffentlicht. Anpassungen können so direkt einer großen Community zugutekommen.

3.1.1.2 Installation

Im Gegensatz zu einer Installation aus den vorhandenen Paketen muss bei einer Installation aus den Quelltexten eine ganze Reihe von Abhängigkeiten gelöst werden. Im Allgemeinen ist aus Aufwandsgründen eine Installation aus den Paketen vorzuziehen, jedoch existieren vor allem für SLES- und openSUSE-Systeme im Gegensatz zu beispielsweise Debian keine vorkonfigurierten Pakete.

Die Abhängigkeiten sind

- libgpg-error
- libgcrypt
- gnutls
- gtk-doc
- Python

```
# zypper install libgpg-error-devel
# zypper search libgpg-error
S | Name                       | Summary                                  | Type
--+--------------------------+------------------------------------------+--------
i | libgpg-error-devel         | Development package for libgpg-error | package
  | libgpg-error-devel-32bit | Development package for libgpg-error | package
i | libgpg-error0              | Library That Defines Common Error -> | package
i | libgpg-error0-32bit        | Library That Defines Common Error -> | package


# zypper install libgcrypt-devel
# zypper search libgcrypt
S | Name                     | Summary                 | Type
--+------------------------+-------------------------+--------
i | libgcrypt-devel          | The GNU Crypto Library | package
  | libgcrypt-devel-32bit | The GNU Crypto Library | package
i | libgcrypt11              | The GNU Crypto Library | package
i | libgcrypt11-32bit        | The GNU Crypto Library | package


# zypper install libgnutls-devel
# zypper search gnutls
S | Name                     | Summary                                  | Type
--+------------------------+------------------------------------------+--------
i | gnutls                   | The GNU Transport Layer Security Libr-> | package
i | libgnutls-devel          | Development package for gnutls           | package
  | libgnutls-extra-devel | The GNU Transport Layer Security Libr-> | package
  | libgnutls-extra26       | The GNU Transport Layer Security Libr-> | package
i | libgnutls26              | The GNU Transport Layer Security Libr-> | package
```

```
  | libgnutls26-32bit     | The GNU Transport Layer Security Libr-> | package


# zypper install gtk-doc
# zypper search gtk-doc
S | Name             | Summary                                  | Type
--+------------------+------------------------------------------+--------
i | gtk-doc          | GTK+ DocBook Documentation Generator     | package
  | python-gtk-doc   | Python bindings for the GTK+ widget set | package
  | python-wxGTK-doc | wxPython Documentation                   | package


# zypper search python
S | Name                         | Summary                                | Type
--+------------------------------+----------------------------------------+--------
i | apache2-mod_python           | A Python Module for the Apache 2->     | package
i | dbus-1-python                | Python bindings for D-Bus              | package
i | libpython2_6-1_0             | Python Interpreter shared library      | package
i | libxml2-python               | Python Bindings for libxml2            | package
i | python                       | Python Interpreter                     | package
i | python-base                  | Python Interpreter base package        | package
i | python-bibtex                | Python Interface to BibTeX Files       | package
i | python-cairo                 | Python Bindings for Cairo              | package
i | python-crypto                | Collection of cryptographic algo->     | package
i | python-devel                 | Include Files and Libraries Mand->     | package
i | python-doc                   | Additional Package Documentation->     | package
i | python-mysql                 | An Interface to the Popular MySQ->     | package
i | python-satsolver             | Python bindings for satsolver          | package
i | python-setuptools            | Download, build, install, upgrad->     | package
i | python-tk                    | TkInter - Python Tk Interface          | package
i | python-xml                   | A Python XML Interface                 | package
i | rpm-python                   | Python Bindings for Manipulating->     | package
i | yast2-python-bindings        | Python bindings for the YaST pla->     | package
...
```

Sind die Voraussetzungen erfüllt, so kann die Installationen beginnen. In der Standardinstallation wird für alle Installationsverzeichnisse der Prefix `/usr/local/` verwendet. Es müssen daher gegebenenfalls nach der Installation die Ordner, in denen die Shared Libraries zu finden sind, systemweit bekannt gegeben werden.

```
# ./configure --enable-easy-bindings --enable-gtk-doc
*** Dumping configuration ***
    - Generate documentation   : yes
    - LUA binding              : no
    - Perl binding             : yes
    - Python binding           : yes
    - Ruby binding             : no
    - Easy bindings            : yes

# make
# make check
# make install
```

In dem obigen Auszug aus der Konfiguration kann man erkennen, dass die Unterstützung für einige Programmiersprachen nicht gegeben ist. Dies ist der Fall, da die angezeigten Sprachen nicht auf dem System installiert waren. Da während des prototypischen Tests nicht zu erwarten ist, dass eine Unterstützung erforderlich ist, wurde auf dieses Feature verzichtet.

Es ist jedoch, falls erforderlich, möglich, diese Unterstützung durch Installation der nötigen Pakete zu aktivieren.

3.1.1.3 Konfiguration

Eine Konfiguration von libprelude ist nach einer erfolgten Installation nicht nötig und daher auch nicht möglich.

3.1.2 Prelude-Manager

Der Prelude-Manager ist die SIEM-Komponente des Prelude-Frameworks. Er ist daher verantwortlich für die Datensammlung und Weitergabe und stellt somit eine Schnittstelle zwischen den verschiedenen Komponenten her. Er basiert auf dem in Abschnitt 3.1.1 beschriebenen libprelude. Um andere Komponenten an den Prelude-Manager anzuschließen, gibt es zwei verschiedene Möglichkeiten: Entweder sind diese externe Komponenten, dann werden sie *Sensoren* genannt, oder sie sind fest in den Prelude-Manager integriert, dann heißen sie *Plugins*.

3.1.2.1 Implementierung

Da der Prelude-Manager die funktionalen und Sicherheitsanforderungen des GIDS-Projektes erfüllt, konnte auf die in C entwickelten Sourcen des unter GPLv2 stehenden Projekts Prelude zurückgegriffen werden. Ursprünglich konnte man unter `http://www.prelude-technologies.com/en/development/download/index.html` die Sourcen herunterladen. Um die Unabhängigkeit der Projektergebnisse zu gewährleisten, wurden die Sourcen als Git-Repository auf `http://git.lrz.de/prelude/` veröffentlicht. Anpassungen können so direkt einer großen Community zugutekommen.

3.1.2.2 Installation

Im Gegensatz zu einer Installation aus den vorhandenen Paketen muss bei einer Installation aus den Quelltexten eine ganze Reihe von Abhängigkeiten gelöst werden. Im Allgemeinen ist aus Aufwandsgründen eine Installation aus den Paketen vorzuziehen, jedoch existieren vor allem für SLES- und openSUSE-Systeme im Gegensatz zu beispielsweise Debian keine vorkonfigurierten Pakete.

Die Abhängigkeiten sind

- libprelude (siehe Abschnitt 3.1.1)
- libpreludedb (siehe Abschnitt 3.2.2)
- libxml2
- tcpd
- MySQL (siehe Abschnitt 3.2.1)

```
# zypper install libxml2-devel
# zypper search libxml2
S | Name                | Summary                                    | Type
--+---------------------+--------------------------------------------+--------
i | libxml2             | A Library to Manipulate XML Files          | package
i | libxml2-32bit       | A Library to Manipulate XML Files          | package
i | libxml2-devel       | Include Files and Libraries mandatory f-> | package
  | libxml2-devel-32bit | Include Files and Libraries mandatory f-> | package
  | libxml2-doc         | A Library to Manipulate XML Files          | package
i | libxml2-python      | Python Bindings for libxml2                | package

# zypper install tcpd-devel
# zypper search tcpd
```

```
S | Name       | Summary                                                | Type
--+------------+--------------------------------------------------------+--------
i | tcpd       | A security wrapper for TCP daemons                     | package
i | tcpd-32bit | A security wrapper for TCP daemons                     | package
i | tcpd-devel | Include Files and Libraries for the TCP wrapper ->     | package
  | tcpdump    | A Packet Sniffer                                       | package
```

Sind die Voraussetzungen erfüllt, so kann die Installationen beginnen.

```
# ./configure
*** Dumping configuration ***
    - TCP wrapper support    : yes
    - XML plugin support     : yes
    - Database plugin support: yes

# make
# make check
# make install
```

3.1.2.3 Konfiguration

In der Standardinstallation wird unter `/usr/local/etc/prelude-manager/prelude-manager.conf` die Konfigurationsdatei des Prelude-Managers abgespeichert. Diese muss nach der Installation angepasst werden, um beispielsweise die Datenbankverbindung korrekt einzurichten. Die unveränderte Konfigurationsdatei ist in Anhang A unverändert mit angehängt. Im Folgenden werden wichtige Stellen in der Konfiguration hervorgehoben und kommentiert.

Eine wichtige Angabe ist, wie sich der Prelude-Manager mit einer anderen Komponente, im Allgemeinen mit einem Sensor verbinden soll. Dafür gibt es mehrere Möglichkeiten: Entweder über eine TCP-Verbindung oder es werden lokale Socket-Verbindungen benutzt. Um mehrere mögliche Szenarien abdecken zu können, erlaubt der Prelude-Manager die Angabe mehrerer „listen“-Angaben parallel. Bei den IP-Adressen können sowohl IPv4- als auch IPv6-Adressen angegeben werden.

```
# listen = address:port
# listen = unix:/tmp/prelude-manager.socket
# listen = unix
#
listen = 127.0.0.1
```

Das Betreiben des Prelude-Managers mit Root-Rechten kann unter Umständen ein Sicherheitsrisiko darstellen und auch durch lokale Sicherheitsrichtlinien untersagt sein. Daher kann hier ein alternativer Benutzername und Gruppenname angegeben. Diese müssen auf dem System natürlich schon vorhanden sein.

```
# Sets the user/group ID as which prelude-manager will run.
# In order to use this option, prelude-manager must be run initially as
# root
#
# user = prelude
# group = prelude
```

Standardmäßig hat ein Client 10 Versuche, um sich zu authentifizieren, bevor er vom Prelude-Manager ignoriert wird. Im Allgemeinen sollten solche Fehler erst gar nicht auftreten, daher ist eine Anpassung des Wertes nicht unbedingt nötig.

```
# Number of second prelude-manager wait for an incoming client to
# successfully authenticate before droping the connection.
#
# connection-timeout = 10
```

Der Prelude-Manager besitzt für jeden angebundenen Sensor einen Buffer, in dem empfangene Nachrichten zwischengespeichert werden. Sollte eine Überlastsituation eintreten, werden aus jedem Buffer nur eine bestimmte Anzahl von Nachrichten verarbeitet, bevor aus dem nächsten Buffer Sensordaten entnommen wird. Diese Rotation verhindert, dass ein Sensor nicht durch ein übermäßiges Schreiben von Nachrichten das Empfangen von Nachrichten von anderen Sensoren verhidert. Die Größe des Zwischenspeichers, das heißt, wieviele Meldungen im Arbeitsspeicher gehalten werden, bevor die letzten auf der Festplatte gespeichert werden, kann durch die Variable `sched-buffer-size` gewählt werden. Eine weitere Wahl hat man durch die Variable `sched-priority`, die besagt, wie viele Nachrichten von welcher Priorität verarbeitet werden sollen, bevor aus dem nächsten Buffer Sensordaten entnommen werden.

```
# Scheduler settings for Prelude-Manager
#
# On systems with many concurrent sensors sending events to
# Prelude-Manager, Prelude-Manager might have an hard time keeping up
# with the demand for events reporting.
#
# The Prelude Manager scheduler allocate reporting time per sensor,
# allowing to define the maximum number of events processed for one
# sensor before processing others sensors events (in case a sensor is
# sending a continuous events burst, this prevent other sensors
# starvation).
#
# By default, for each sensor connected, a maximum of 100 events will
# be processed before processing others sensors events.
#
# Additionally, priority will be given to events depending on their
# priority. Assuming there is enough events of each priority, 50 high
# priority message will be processed, 30 medium, and 20 low (totalling
# the maximum of 100 described above).
#
# You might use the sched-priority option in order to change this
# setting:
#
# sched-priority = high:50 medium:30 low:20
#
#
# When the number of events waiting to be processed exceed the defined
# amount of reserved memory (default is 1 Megabyte), Prelude-Manager
# will start storing events on disk:
#
# sched-buffer-size = 1M
```

Diese Einstellungen betreffen die Sicherheit bei der Verschlüsselung. An dieser Stelle kann man eine Entscheidung treffen, ob das System sicherer oder schneller sein soll, da jede Verbesserung auf Seiten der Verschlüsselung die Performanz des Systems einschränkt.

Im prototypischen Testbetrieb wurden die vorgeschlagenen Standardparameter verwendet. Dabei sind auch bei Lasttests an dieser Stelle keine Einschränkungen in der Verarbeitungsgeschwindigkeit auffällig geworden.

```
# TLS options (only available with GnuTLS 2.2.0 or higher):
# sets availables ciphers, key exchange methods, macs and compression
# methods.
#
# "NORMAL" option enables all "secure" ciphersuites, 256-bit ciphers
# included.
#
```

```
# "SECURE128" flag enables all "secure" ciphersuites with ciphers up to
# 128 bits.
#
# "SECURE256" flag enables all "secure" ciphersuites including the 256
# bit ciphers.
#
# "EXPORT" all the ciphersuites are enabled, including the low-security
# 40 bit ciphers.
#
# "NONE" nothing is enabled. This disables even protocols and
# compression methods.
#
# Note that much more settings might be enabled or disabled using this
# option: please see gnutls_priority_init(3) for more details.
#
# The default settings is "NORMAL".
# tls-options = NORMAL

# Number of bits of the prime used in the Diffie Hellman key exchange.
# Note that the value should be one of 768, 1024, 2048, 3072 or 4096.
# The default is 1024.
#
# dh-prime-length = 1024
# How often to regenerate the parameters used in the Diffie Hellman key
# exchange. These should be discarded and regenerated once a day, once
# a week or once a month. Depending on the security requirements.
#
# Generation is a CPU intensive operation. The value is in hours,
# 0 disables regeneration entirely. The default is 24 hours.
#
# dh-parameters-regenerate = 24
```

Prelude bietet die Möglichkeit, dass mehrere Instanzen des Prelude-Managers in einer hierarchischen oder heterarchischen Baumstruktur angeordnet werden. Daher ist es möglich, dass neben typischen Sensoren auch eine weitere Instanz des Prelude-Managers als Sensor mit angebunden wird. Die Angabe zu diesem Kind-Manager kann an dieser Stelle angegeben werden.

```
# If you want this Manager to retrieve message from another Manager
# (useful if the other Manager is located within a DMZ):
#
# child-managers = x.x.x.x
#
# This mean the messages should be gathered from x.x.x.x
```

Im Falle eines Fehlers werden die Daten, die durch ein Plugin verarbeitet wurden, nur dann durch ein Failover-Mechanismus geschützt, wenn diese Konfigurationseinstellung eingestellt ist. Leider ist es nicht möglich, einen Failover-Mechanismus für Sensoren zu aktivieren. Dies müsste man in einer Weiterentwicklung von Prelude berücksichtigen.

```
# If you want a given reporting plugin to be protected against possible
# failure, use the failover option. Failover will prevent data sent to
# the report plugin to be lost in case this one fail.
#
# You might use this option multiple time for different plugins.
#
# failover = name_of_plugin
```

Beim Umstieg von einem reinen IPv4-Netz auf einen IPv4-/IPv6-Mischbetrieb oder auf ein reines IPv6-Netz ist es gegebenenfalls gewünscht, dass Alarmmeldungen entweder global als IPv6-Adressen angezeigt werden oder dass ein vorherig durchgeführtes Mapping von IPv4- auf IPv6-Adressen rückgängig gemacht wird.

```
# Events normalization parameters
#
# Un-comment the following section in case you want to define any
# normalization parameters:
#
# [normalize]
#
# For each incoming events, Prelude-Manager will run a number of
# normalization routine: sanitize address, services information, etc.
#
# When the normalizer see an incoming IPv4 mapped IPv6 address, the
# default behavior is to map it back to raw IPv4. For example,
# ::ffff:192.168.0.1 will be mapped back to 192.168.0.1
#
# If you do not want IPv4 mapped IPv6 addresses, un-comment the
# following option:
#
# keep-ipv4-mapped-ipv6
#
# Alternatively, if you wish for any input IPv4 addresses to be
# converted to IPv6, un-comment the following option:
#
# ipv6-only
```

Wie oben schon beschrieben, ist es im Prelude-Framework möglich, dass Eltern-Kind-Beziehungen zwischen verschiedenen Prelude-Managern aufgebaut werden können. Analog zur obige Angabe `child-managers = x.x.x.x`, die die Kind-Manager beschreibt, kann man hier angeben, welche Eltern-Manager verwendet werden sollen. Um Redundanzen herzustellen, können mehrere Adressen angegeben werden, die entweder mit „und“ oder „oder“ verbunden werden.

```
# [relaying]
#
# If you want the message caught by this manager to be relayed.
# You can use boolean AND and OR to make the rule.
#
# parent-managers = x.x.x.x || y.y.y.y && z.z.z.z
#
# This mean the emission should occur on x.x.x.x or, if it fail, on
# y.y.y.y and z.z.z.z (if one of the two host in the AND fail, the
# emission will be considered as failed involving saving the message
# locally).
```

Einer der zentralen Angaben, sollte der Prelude-Manager nicht nur als Durchgangsstation verwendet werden, ist die Angabe der Datenbank, in der die empfangenen Nachrichten abgespeichert werden. Dies ist der Grund, warum Abhängigkeiten zu den Paketen *libpreludedb* und *MySQL* bestehen. Sollen keine Daten in die Datenbank gespeichert werden, so braucht man diese Pakete auch nicht installieren.

Wichtig zu wissen, ist, dass das Passwort für den Datenbankzugriff im Klartext in der Konfigurationsdatei abgespeichert werden muss. Daher muss man bei den Zugriffsregelungen für die Konfigurationsdatei aufpassen, damit das Passwort nicht unberechtigterweise ausgelesen werden kann.

```
# [db]

# The type of database: mysql, pgsql or sqlite3.
# type = mysql

# Only if you use sqlite3.
# file = /your/path/to/your/db/idmef-db.sql

# Host the database is listening on.
# host = localhost

# Port the database is listening on.
# port = 3306

# Name of the database.
# name = prelude

# Username to be used to connect the database.
# user = prelude

# Password used to connect the database.
# pass = xxxxxx
```

Neben der Speicherung von Alarmen kann man sich Alarme auch in eine Datei oder Standardausgaben, wie `stdout` oder `stdin`, ausgeben lassen. Diese Ausgabe ist jedoch im Gegensatz zur Datenbankausgabe eine reine Ausgabe, da die Bibliothek libpreludedb, die in Abschnitt 3.2.2 bschrieben wird, auch das Einlesen von Daten aus der Datenbank erlaubt. Eine solche Lesefunktion ist für Text- oder XML-Dateien nicht vorgesehen.

```
# [XmlMod]
#
# The Xmlmod plugin allow to report alert as IDMEF XML in a file,
# or to dump theses alert to stderr.
#
# The default behavior is to write output to stderr.
#
# Tell Xmlmod to disable output file buffering.
# This will prevent XML alerts to be truncated and thus make real-time
# parsing easier:
#
# disable-buffering
#
#
# Tell Xmlmod to check generated XML against IDMEF DTD:
# validate
#
# Tell Xmlmod to produce a pretty, human readable xml output:
# format
#
# logfile = stderr
# logfile = /var/log/prelude-xml.log

#
# [Debug]
#
# The Debug plugin allow to report alert as text in a file,
# or to dump theses alert to stderr.
```

```
#
# The default behavior is to write output to stderr.
#
# logfile = stderr
# logfile = /var/log/prelude.log
#
# You can specify the name of the IDMEF object to print (you might
# select multiple objects). If no object are provided, 'Debug' will
# print out the entire message.
#
# object = alert.classification.text, alert.source(0).node.address(0).address

#
# [TextMod]
#
# The Debug plugin allow to report alert as text in a file,
# or to dump theses alert to stderr.
#
# The default behavior is to write output to stderr.
#
# logfile = stderr
# logfile = /var/log/prelude.log
```

Sollen Nachrichten zusätzlich per E-Mail versendet werden, können hier die nötigen Angaben gemacht werden.

```
#[smtp]
#
# Sender to use for the mail message.
# sender = prelude@myhostname.
#
# Who the mail should be sent to.
# recipients = recipient1@hostname, recipient2@hostname
#
# SMTP server to use for sending mail
# smtp-server = localhost
#
# By default, the SMTP plugin send mail containing the whole IDMEF
# event. If you wish to send a subset of the information, you may
# customize the content of the generated mail through several options:
#
# You can define a specific subject to use with mail notification.
# The subject can include information from the event using IDMEF path.
# subject = Alert: $alert.classification.text
#
# You can define a specific message body to use for mail notification.
# As with the "subject" option, the template can include information
# from the event using IDMEF path.
#
# (Template example available in @DOCDIR@/smtp/template.example)
# template = /path/to/my/template
#
# You can provide your database settings here, so that the SMTP plugin
# retrieve alert linked to received CorrelationAlert from the database.
#
# dbtype = mysql
# dbname = prelude
```

```
# dbuser = prelude
# dbpass = passwd
# dbhost = localhost
# Other database options available include dbport, and dbfile (for
# sqlite3 database).
#
# If you have specified your database settings above, you can also
# use the correlated-alert-template option, which is like the "template"
# option but is specific to Correlated Alerts retrieved from database.
#
# (Template example available in @DOCDIR@/smtp/template.example)
# correlated-alert-template = /path/to/my/template
```

Die hier gemachten Angaben erhöhen die Nützlichkeit der oben eingeführten Plugins wesentlich. Im Gegensatz zur Standardeinstellung, in der alle Alarme beispielsweise per E-Mail versandt werden, kann man hier ein Kriterium angeben, wann eine E-Mail versendet werden soll. Beispielsweise erzeugt die Angabe

```
[idmef-criteria=UserLoginSuccess]
rule = alert.classification.text == 'Remote Login'
hook = smtp[default]
```

jedes Mal, wenn ein Alarm mit der Klassifikation *Remote Login* auftritt, was im Allgemeinen auf einen erfolgreichen Login-Vorgang hinweist, eine E-Mail an die in `smtp[default]` gemachten Angaben. In jedem anderen Fall wird keine E-Mail generiert. Dadurch kann man sich über betriebskritische Meldungen schnell informieren lassen.

```
# The idmef-criteria filtering plugin allow you to filter events based
# on specific IDMEF-Criteria.
#
# [idmef-criteria]
# rule = alert.classification.text == 'User login successful'
# hook = relaying[default]
#
# Will forward any events that match the defined criteria to the
# default instance of the relaying reporting plugin. The rule argument
# might also be a filename containing the rules. Example:
#
# rule = /path/to/rule.file
```

Wie auch die Angabe des IDMEF-Kriteriums die Nützlichkeit der Benachtichtigungen stark erhöhen kann, ist auch die Angabe eines Threshold in vielen Anwendungsgebieten sinnvoll, da man dabei dem System vorgeben kann, dass Events unterdrückt werden, wenn es zu viele werden. So ist es beispielsweise möglich, für eine zu definierende Zeitspanne die Anzahl der Nachrichten vorzugeben, die maximal gesendet werden sollen. Treten noch mehr Nachrichten auf, so werden diese ignoriert. Alternativ ist es möglich, zum Beispiel innerhalb von fünf Minuten nur jede 1000. Nachricht weiterzusenden. Somit wird man nur auf die wirklich wichtigen Meldungen aufmerksam gemacht.

```
# The thresholding filtering plugin allow you to suppress events based
# on their value.
#
# [thresholding]
# path = alert.classification.text, alert.source.node.address.address
# limit = 3600
# count = 1
# hook = relaying[default]
#
```

```
# Will forward one event with the unique alert.classification.text,
# alert.source.node.address.address value combination to the 'default'
# instance of the 'relaying' reporting plugin. Further events with the
# same value will be suppressed for 3600 seconds.
#
#
# [thresholding]
# path = alert.classification.text, alert.source.node.address.address
# threshold = 3600
# count = 10
# hook = relaying[default]
#
# Will forward every tenth event per 3600 seconds with the unique
# alert.classification.text, alert.source.node.address.address value
# combination to the 'default' instance of the 'relaying' reporting
# plugin.
#
# Note that limit and threshold might be combined, allowing to setup a
# limit as soon as the first threshold is reached.
```

Schlussendlich gibt es noch eine Konfigurationsmöglichkeit, ab wann ein Sensor, der scheinbar nicht mehr arbeitet und daher auf Verbindungsversuche nicht mehr reagiert, als nicht mehr erreichbar eingestuft werden soll. Diese Einstellung ist interessant, da Meldungen, die einmal zu einem Sensor gesendet wurden, werden als „abgearbeitet“ eingestuft und werden daher kein zweites Mal gesendet. Ist der Sensor nicht mehr erreichbar, so gehen daher Meldungen verloren. Auf der anderen Seite ist ein voreiliges Einstufen des Sensors als nicht erreichbar nicht sinnvoll, da es ein manuelles Eingreifen erfordert, um die Verbindung zum Prelude-Manager wiederherzustellen.

```
# [prelude]
#
# This is the global prelude section, where you can define Prelude
# related options. Option of matter for Prelude-Manager, are, most
# specifically, in the context of relaying, the connection options:
#
# The following settings instruct the operating system when to consider
# a connection dead in case sent data is left unacknowledged.
#
# Theses option are operating system specific, and might not work on
# certain platform. In case you modify these settings on an unsupported
# system, a warning message will be issued when the agent starts.
#
# Under Linux, the default system wide configuration is:
# tcp-keepalive-time   = 7200
# tcp-keepalive-probes = 9
# tcp-keepalive-intvl  = 75
#
# tcp-keepalive-time represents the number of seconds the connection
# needs to be idle before TCP begins sending out keep-alive probes.
#
# tcp-keepalive-probes represent the number of not acknowledged probes
# to send before considering the connection dead.
#
# tcp-keepalive-intvl represents the interval between subsequent
# keepalive probes.
#
# The average time to notice a dead connection can be calculated using:
```

```
# tcp-keepalive-time + (tcp-keepalive-probes * tcp-keepalive-intvl)
#
# Here is an example configuration:
# tcp-keepalive-time   = 60
# tcp-keepalive-probes = 3
# tcp-keepalive-intvl  = 10
#
# Using the above settings, a dead connection will be detected within
# 90 seconds.
```

3.1.3 Sensoriken

Im Allgemeinen sind die Sensoriken, die für die Erkennung von Angriffen verantwortlich sind, nicht Teil der Pakete, die zur Installation an andere D-Grid-Sites weitergegeben werden, da diese im Allgemeinen schon eigene Systeme zur Erkennung missbräuchlicher Nutzung haben. Im Folgenden werden einige Beispiele gegeben, die zusammen eine weitestgehend vollständige Überwachung der Ressourcen ermöglichen.

3.1.3.1 Snort

Snort ist ein Netz-IDS, das den Netzverkehr überwacht und anhand eines gegebenen Regelsatzes entscheidet, ob der beobachtete Verkehr legitim war oder ob ein Alarm generiert werden soll.

3.1.3.1.1 Implementierung

Die Implementierung von Snort war nicht Teil des Projektes und wird im Wesentlichen von der Firma Sourcefire vorangetrieben. Allerdings wurde das NIDS an die Bedürfnisse des Projektes angepasst. Die Sourcen können unter `http://www.snort.org/snort-downloads` heruntergeladen werden.

3.1.3.1.2 Installation

Im Gegensatz zu einer Installation aus den vorhandenen Paketen muss bei einer Installation aus den Quelltexten eine ganze Reihe von Abhängigkeiten gelöst werden.

Die Abhängigkeiten sind

- libprelude (siehe Abschnitt 3.1.1)
- libpcap
- pcre
- libdnet
- barnyard
- daq

```
# zypper search pcre
S | Name       | Summary                                              | Type
--+------------+------------------------------------------------------+--------
i | pcre       | A library for Perl-compatible regular expressions | package
i | pcre-32bit | A library for Perl-compatible regular expressions | package
i | pcre-devel | A library for Perl-compatible regular expressions | package


# zypper install libdnet-devel
# zypper search libdnet
S | Name          | Summary                                           | Type
```

```
--+--------------+-------------------------------------------------+--------
i | libdnet-devel | Devel files for libdnet                         | package
i | libdnet1      | Library for Simple, Portable Interface to Low-> | package

# zypper search libpcap
S | Name           | Summary                          | Type
--+----------------+----------------------------------+--------
  | libpcap-devel  | A Library for Network Sniffers | package
  | libpcap0       | A Library for Network Sniffers | package
  | libpcap0-32bit | A Library for Network Sniffers | package
```

Die Pakete barnyard und daq sind nicht in der SLES-Paketverwaltung vorhanden, daher ist eine Installation nur aus den Sourcen möglich. Weiterhin ist das Paket libpcap in einer Version > 1.0 nötig, in der Paketverwaltung ist jedoch nur die Version 0.9.8 vorhanden. Daher ist auch hier die Installation aus den Sourcen nötig. Die nötigen Sourcen findet man unter:

- **libpcap**. `http://www.tcpdump.org/#latest-release`
- **barnyard**. `http://www.snort.org/downloads/462`
- **daq**. `http://www.snort.org/snort-downloads`

Alle Pakete werden jeweils mit `./configure`, `make` und `make install` installiert.

Sind alle Voraussetzungen erfüllt, kann die Installation beginnen. Wichtig ist die Option `--enable-prelude`, die für die Unterstützung von verantwortlich ist.

```
# ./configure --enable-prelude
# make
# make check
# make install
```

3.1.3.1.3 Konfiguration

In der Konfigurationsdatei snort.conf, die entweder unter `/etc/snort/` oder unter `/usr/local/etc/snort/` zu finden ist, muss die Zeile

```
# prelude
output alert_prelude
```

aktiviert werden.

Weiterhin muss die in der Datei `/usr/local/etc/prelude/default/client.conf` die Zeile

```
server-addr = 127.0.0.1
```

auf die IP-Adresse des Prelude-Managers geändert werden, sollte dieser nicht auf der gleichen Maschine laufen.

Schlussendlich muss Snort sich noch am zuständigen Prelude-Manager registrieren, damit Snort zum einen berechtigt ist, Daten mit den Prelude-Manager austauschen zu dürfen und zum anderen, damit der verschlüsselte Versand vom Meldungen vorbereitet wird. Dazu muss auf der Maschine, auf der Snort läuft, der Befehl

```
# prelude-admin register snort "idmef:w admin:r" localhost --uid 0 --gid 0
Generating 2048 bits RSA private key... This might take a very long time.
```

ausgeführt werden. Dabei muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft, mit dem man sich verbinden möchte. Weiterhin müssen die Angaben `uid` und `gid` auf die Werte, die Snort im laufenden Betrieb hat.

Ist die Generierung des Schlüsselpaares abgeschlossen, muss am Server des Prelude-Managers die Registrierung abgeschlossen werden. Dazu dient der Befehl

```
# prelude-admin registration-server prelude-manager --listen localhost
Generating 2048 bits RSA private key... This might take a very long time.
```

Auch hier muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft. Am Ende wird einem hier ein Passwort angezeigt, dass man auf Seiten der Snort-Ressource wiederholen muss. Damit ist die Konfiguration von Snort aus der Sichtweise von Prelude abgeschlossen. Snort-spezifische Anpassungen, wie das Laden der geeigneten Regelsätze ist in den einschlägigen Dokumentationen nachzulesen. Anhang H gibt ein Beispiel für Regelsätze, die in Snort Angriffe auf GSISSH, was üblicherweise auf Port 2222 läuft, erkennen können.

3.1.3.2 Prelude-LML

Der Prelude-LML oder Prelude Log Monitoring Lackey ist die Komponente des Prelude-Frameworks, die Logfiles von verschiedenen Systemen analysieren kann und anhand von regulären Ausdrücken Meldungen extrahieren kann.

3.1.3.2.1 Implementierung

Die Implementierung des in C entwickelten Prelude-LML war nicht Teil des Projektes, da auf die vorhandenen Sourcen des unter GPLv2 stehenden Projekts Prelude zurückgegriffen werden konnte. Ursprünglich konnte man unter `http://www.prelude-technologies.com/en/development/download/index.html` die Sourcen herunterladen. Um die Unabhängigkeit der Projektergebnisse zu gewährleisten, wurden die Sourcen als Git-Repository auf `http://git.lrz.de/prelude/` veröffentlicht. Anpassungen, wie verbesserte Regelsätze, können so direkt einer großen Community zugutekommen.

3.1.3.2.2 Installation

Im Gegensatz zu einer Installation aus den vorhandenen Paketen muss bei einer Installation aus den Quelltexten eine ganze Reihe von Abhängigkeiten gelöst werden. Im Allgemeinen ist aus Aufwandsgründen eine Installation aus den Paketen vorzuziehen, jedoch existieren vor allem für SLES- und openSUSE-Systeme im Gegensatz zu beispielsweise Debian keine vorkonfigurierten Pakete.

Die Abhängigkeiten sind

- libprelude (siehe Abschnitt 3.1.1)
- libicu

```
# zypper install libicu-devel
# zypper search libicu
S | Name               | Summary                                      | Type
--+--------------------+----------------------------------------------+--------
i | libicu             | International Components for Unicode (de-> | package
  | libicu-32bit       | International Components for Unicode (de-> | package
i | libicu-devel       | International Components for Unicode (de-> | package
  | libicu-devel-32bit | International Components for Unicode (de-> | package
  | libicu-doc         | International Components for Unicode  (h-> | package
```

Sind die Voraussetzungen erfüllt, so kann die Installationen beginnen.

```
# ./configure --enable-unsupported-rulesets
*** Dumping configuration ***
    - Favor libICU over Iconv    : yes
    - Enable unsupported rulesets: yes

# make
# make check
# make install
```

3.1.3.2.3 Konfiguration

In der Standardinstallation wird unter `/usr/local/etc/prelude-lml/prelude-lml.conf` die Konfigurationsdatei des Prelude-Managers abgespeichert. Diese muss nach der Installation angepasst werden, um das Einlesen der Logfiles korrekt einzurichten. Die unveränderte Konfigurationsdatei ist in Anhang C unverändert mit angehängt. Im Folgenden wird ein Beispieleintrag angegeben und kommentiert.

```
[format=syslog]
time-format = "%b %d %H:%M:%S"
prefix-regex = "^(?P<timestamp>.{15}) (?P<hostname>\S+) (?:(?P<process>\S+?)
               (?:\[(?P<pid>[0-9]+)\])?: )?"
file = /var/log/messages
# udp-server = 0.0.0.0
```

Hierbei wird unter `file` die Logdatei angegeben, die analysiert werden soll. Alternativ kann man unter der Einstellung `udp-server` Prelude-LML anweisen, auf eine UDP-Verbindung zu hören. Die Angaben `time-format` und `prefix-regex` geben reguläre Ausdrücke an, wie Teile der Einträge des Logfiles aufgebaut sind. Eine genaue Beschreibung, welche Aktion bei welchem Logeintrag durchgeführt werden soll, findet man in den Regelsätzen, die standardmäßig unter `/usr/local/etc/prelude-lml/ruleset/` zu finden sind. Sollen benutzerdefinierte Regelsätze hinzugefügt werden, so muss die dafür nötige Datei unter `pcre.rules` im gleichen Verzeichnis registriert werden

Schlussendlich muss Prelude-LML sich noch am zuständigen Prelude-Manager registrieren, damit Prelude-LML zum einen berechtigt ist, Daten mit den Prelude-Manager austauschen zu dürfen und zum anderen, damit der verschlüsselte Versand vom Meldungen vorbereitet wird. Dazu muss auf der Maschine, auf der Prelude-LML läuft, der Befehl

```
# prelude-admin register prelude-lml "idmef:w admin:r" localhost --uid 0 --gid 0
Generating 2048 bits RSA private key... This might take a very long time.
```

ausgeführt werden. Dabei muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft, mit dem man sich verbinden möchte. Weiterhin müssen die Angaben `uid` und `gid` auf die Werte, die Prelude-LML im laufenden Betrieb hat.

Ist die Generierung des Schlüsselpaares abgeschlossen, muss am Server des Prelude-Managers die Registrierung abgeschlossen werden. Dazu dient der Befehl

```
# prelude-admin registration-server prelude-manager --listen localhost
Generating 2048 bits RSA private key... This might take a very long time.
```

Auch hier muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft. Am Ende wird einem hier ein Passwort angezeigt, dass man auf Seiten der Prelude-LML-Ressource wiederholen muss. Damit ist die Konfiguration von Prelude-LML abgeschlossen.

3.1.3.3 Import

Die Komponente Prelude-Import ist für den Importvorgang von IDMEF-Nachrichten in das Prelude-Framework verantwortlich.

3.1.3.3.1 Implementierung

Die C basierte Implementierung von Prelude-Import basiert im Wesentlichen auf libxml, da diese Bibliothek die Verarbeitung des XML-basierte Formats IDMEF vereinfacht. Das Programm liest zeilenweise komplette IDMEF-Nachrichten ein und parst diese dann durch die von libxml bereitgestellten Funktionen. Danach wird bei jedem XML-Knoten die zugehörige Funktion der libprelude-API anprogrammiert, so dass am Ende eine Nachricht im Prelude-eigenen Binärformat an den Prelude-Manager geschickt werden kann.

Prelude-Import besitzt zwei Möglichkeiten, um neue IDMEF-Nachrichten zu empfangen. Entweder werden die Nachrichten aus dem `stdin` gelesen oder sie werden direkt aus dem

GIDS-Bus empfangen. Dafür besitzt Prelude-Import eine Anbindung an libemcast, das eine API für das in Kapitel 4.1.1 beschriebene Programm Emcast dient. Welcher Weg für die Dateneingabe benutzt wird, wird in der Konfigurationsdatei `prelude-import.conf` festgelegt. Ist dort `emcast = true` angegeben, wird eine Emcast-Verbindung aufgebaut und alle Nachrichten werden von dort gelesen. In jedem anderen Fall wird `stdin` verwendet.

In der weiteren Entwicklung des Programms ist angedacht, die zu parsenden Meldungen beim Importvorgang manipulieren zu können. Dies ist jedoch noch nicht implementiert worden und es muss sich noch im praktischen Betrieb zeigen, ob eine solche Manipulationsmöglichkeit gewünscht und sinnvoll ist. Die Entwicklung des Programmes kann unter `http://git.lrz.de/gitweb/?p=Prelude-Im-_und_Export.git;a=summary` verfolgt werden.

3.1.3.3.2 Installation

Momentan existiert noch keine Paketierung des Programms, so dass man um eine manuelle Kompilierung nicht herumkommt.

Die Abhängigkeiten sind

- libprelude (siehe Abschnitt 3.1.1)
- libxml
- iniparser (`http://ndevilla.free.fr/iniparser/`)
- libemcast (`http://git.lrz.de/gitweb/?p=Emcast.git;a=summary`)

Die oben genannten Pakete iniparser und libemcast sind nicht Teil der Paketverwaltung von SLES. Daher müssen diese Pakete manuell installiert werden.

Sind die Voraussetzungen erfüllt, so kann die Installationen beginnen, die in diesem Fall durch Kompilierung des Quelltextes besteht.

```
# gcc prelude-import.c -o prelude-import
    ‘xml2-config --cflags --libs‘
    ‘emcast-config --cflags --libs‘
    ‘libprelude-config --cflags --libs‘
    -I./iniparser/iniparser-3.0/src -L./iniparser/iniparser-3.0 -liniparser
```

3.1.3.3.3 Konfiguration

Wie schon eingangs erwähnt, gibt es die Konfigurationsdatei `prelude-import.conf`, die in Anhang E abgedruckt ist. In dieser kann man festlegen, ob man Nachrichten direkt aus dem GIDS-Bus mittels emcast empfangen will und wenn ja, wie die Parameter dafür aussehen.

```
#########
# Spezielle Einstellungen für die Verbindung zum GIDS-Bus
#########
[Emcast]
emcast = true        ; Schaltet Emcast-Unterstützung ein.
                     ; Die Alarme werden dann an den GIDS-Bus weitergeleitet.

url = 224.1.2.3:1234 ; Die Broadcastadresse des GIDS-Bus

buffer = 16384       ; Die Größe des Buffers zum Empfang der Daten

loopback = false     ; Wenn Nachrichten, die an den GIDS-Bus versendet
                     ; werden sollen, auch an die absendende Site
                     ; geschickt werden soll, muss hier "true"
                     ; eingetragen werden. ACHTUNG: Diese Einstellung
                     ; kann bei unsachgemäßer Verwendung eine Schleife
                     ; und damit eine Überlastung produzieren!!!
```

Schlussendlich muss Prelude-Import sich noch am zuständigen Prelude-Manager registrieren, damit Prelude-Import zum einen berechtigt ist, Daten mit den Prelude-Manager austauschen zu dürfen und zum anderen, damit der verschlüsselte Versand vom Meldungen vorbereitet wird. Dazu muss auf der Maschine, auf der Prelude-Import läuft, der Befehl

```
# prelude-admin register prelude-import "idmef:w admin:r" localhost --uid 0 --gid 0
Generating 2048 bits RSA private key... This might take a very long time.
```

ausgeführt werden. Dabei muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft, mit dem man sich verbinden möchte. Weiterhin müssen die Angaben `uid` und `gid` auf die Werte, die Prelude-Import im laufenden Betrieb hat.

Ist die Generierung des Schlüsselpaares abgeschlossen, muss am Server des Prelude-Managers die Registrierung abgeschlossen werden. Dazu dient der Befehl

```
# prelude-admin registration-server prelude-manager --listen localhost
Generating 2048 bits RSA private key... This might take a very long time.
```

Auch hier muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft. Am Ende wird einem hier ein Passwort angezeigt, dass man auf Seiten der Prelude-Import-Ressource wiederholen muss. Damit ist die Konfiguration von Prelude-Import abgeschlossen.

3.1.3.4 generischer Ansatz

Da weder alle möglichen Sensoren an dieser Stelle beschrieben werden können, wird im Folgenden ein generischer Ansatz beschrieben, wie noch weitere Sensoren an das Prelude-Framework mit angeschlossen werden können.

Je nach Art, wie der Sensor angeschlossen werden soll, gibt es verschiedene Möglichkeiten:

- **Der Sensor ist nativ mit Prelude kompatibel.** In diesem Fall ist die nötige Abhängigkeit die Bibliothek libprelude, die auf der Sensor-Ressource installiert sein muss. Zumeist muss bei der Installation des Sensors zusätzlich die Option `./configure --enable-prelude` oder ähnliches aktiviert sein. Hier sei auf die Dokumentation des jeweiligen Sensors verwiesen. Weiterhin ist es nötig, dass sich der neue Sensor am Prelude-Manager registriert, damit der neue Sensor zum einen berechtigt ist, Daten mit den Prelude-Manager austauschen zu dürfen und zum anderen, damit der verschlüsselte Versand vom Meldungen vorbereitet wird. Dazu muss auf der Maschine, auf der der neue Sensor läuft, der Befehl

  ```
  # prelude-admin register <new_sensor> "idmef:w admin:r" localhost --uid 0 --gid 0
  Generating 2048 bits RSA private key... This might take a very long time.
  ```

  ausgeführt werden. Dabei muss die Angabe `<new_sensor>` durch den Namen des Sensors ersetzt werden, was im Allgemeinen auch dem Namen der ausführbaren Datei entspricht. Weiterhin muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft, mit dem man sich verbinden möchte. Weiterhin müssen die Angaben `uid` und `gid` auf die Werte, die der neue Sensor im laufenden Betrieb hat.

  Ist die Generierung des Schlüsselpaares abgeschlossen, muss am Server des Prelude-Managers die Registrierung abgeschlossen werden. Dazu dient der Befehl

  ```
  # prelude-admin registration-server prelude-manager --listen localhost
  Generating 2048 bits RSA private key... This might take a very long time.
  ```

  Auch hier muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft. Am Ende wird einem hier ein Passwort angezeigt, dass man auf Seiten der Ressource des neuen Sensors wiederholen muss. Damit ist die Konfiguration des neuen Sensors abgeschlossen.

- **Der Sensor schreibt seine Meldungen in eine syslog-ähnliche Datei.** Eine direkte Anbindung des Servers ist in diesem Fall nicht zwingend notwendig. Es kann hier vielmehr der schon vorhandene Prelude-LML zum Einsatz kommen. Dabei wird Prelude-LML entweder dezidiert auf der Ressorce des Sensors installiert und analysiert lokal die Logdatei. Eine weitere Möglichkeit wäre, Prelude-LML als UDP-Server zu konfigurieren, so dass die Meldungen des Sensors über UDP an Prelude-LML gesendet werden, falls das die Sensor-Ressource unterstützt. Diese Möglichkeit hätte den Vorteil, dass auf der Sensor-Ressource keine weiteren Dienste installiert sein müssten.
- **Der Sensor exportiert seine Meldungen in ein beliebiges Textformat.** Ähnlich zum obigen Fall, wo auf de Prelude-LML zurückgegriffen werden konnte, kann man hier auf den Prelude-Import zurückgreifen. Dafür ist es nötig, dass die Daten vom Sensor zuerst ins IDMEF-Format konvertiert werden. Anschließend kann der IDMEF-String durch den Importvorgang in das Prelude-Framework geschrieben werden. Im Gegensatz zur Lösung mit Hilfe des Prelude-LML, unterstützt der Prelude-Import noch kein Empfang von Nachrichten über UDP, so dass eine Installation auf der Sensor-Ressource unvermeidlich bleibt.

3.2 Datenanalyse und -speicherung

3.2.1 MySQL

Zur lokalen Speicherung von Daten unterstützt das Prelude-Framework drei verschiedene Datenbanksysteme: MySQL, PostgreSQL und SQLite3. Aus Kompatibilitätsgründen wird jedoch auf die Ausnutzung von Features der verschiedenen Datenbanksysteme verzichtet, somit ist das verwendete Datenbankschema in weiten Teilen universell einsetzbar. Da wir in der prototypischen Implementierung nur den Fall betrachtet haben, dass wir den Prelude-Manager mit einer MySQL-Datenbank verbinden, ist im Folgenden stellvertretend für alle Datenbanksysteme nur von MySQL-Datenbanken die Rede.

3.2.2 libpreludedb

Unterstützend für einen effizienten Zugriff auf die Meldungen in der Datenbank wurde die Bibliothek *libpreludedb* bereitgestellt, die neben den Datenbankschemata, das in in der Standardinstallation unter `/usr/local/share/libpreludedb/classic/mysql.sql` zu finden ist, die IDMEF in ein SQL-Schema abbilden, auch eine Möglichkeit bereitstellt, auf in der Datenbank gespeicherte Meldungen direkt zugreifen zu können, so dass nötige SQL-Queries automatisch erstellt werden.

3.2.2.1 Implementierung

Die Implementierung der in C konzipierten Bibliothek libpreludedb ist war nicht Teil des Projektes, da auf die vorhandenen Sourcen des unter GPLv2 stehenden Projekts Prelude zurückgegriffen werden konnte. Ursprünglich konnte man unter `http://www.prelude-technologies.com/en/development/download/index.html` die Sourcen herunterladen. Um die Unabhängigkeit der Projektergebnisse zu gewährleisten, wurden die Sourcen als Git-Repository auf `http://git.lrz.de/prelude/` veröffentlicht. Anpassungen können so direkt einer großen Community zugutekommen.

3.2.2.2 Installation

Im Gegensatz zu einer Installation aus den vorhandenen Paketen muss bei einer Installation aus den Quelltexten eine ganze Reihe von Abhängigkeiten gelöst werden. Im Allgemeinen ist aus Aufwandsgründen eine Installation aus den Paketen vorzuziehen, jedoch existieren vor allem für SLES- und openSUSE-Systeme im Gegensatz zu beispielsweise Debian keine vorkonfigurierten Pakete.

Die Abhängigkeiten sind

- libprelude (siehe Abschnitt 3.1.1)
- MySQL (siehe Abschnitt 3.2.1)

```
# zypper install libmysqlclient-devel
# zypper search mysql
S | Name                      | Summary                                | Type
--+---------------------------+----------------------------------------+--------
  | bytefx-data-mysql         | Database connectivity for Mono         | package
  | libgda-3_0-mysql          | mySQL Provider for GNU Data Access->   | package
  | libgda-4_0-mysql          | MySQL Provider for GNU Data Access->   | package
i | libmysqlclient-devel      | MySQL Development Header Files and->   | package
i | libmysqlclient15          | MySQL Shared Libraries                 | package
  | libmysqlclient15-32bit    | MySQL Shared Libraries                 | package
i | libmysqlclient_r15        | A True Multiuser, Multithreaded SQ->   | package
  | libmysqlclient_r15-32bit  | A True Multiuser, Multithreaded SQ->   | package
  | libqt4-sql-mysql          | Qt 4 MySQL support                     | package
  | libqt4-sql-mysql-32bit    | Qt 4 MySQL support                     | package
  | lighttpd-mod_mysql_vhost  | MySQL based virtual hosts (vhosts)->   | package
i | mysql                     | A True Multiuser, Multithreaded SQ->   | package
  | mysql-Max                 | MySQL - Server with Berkeley DB        | package
i | mysql-client              | MySQL Client                           | package
  | mysql-connector-java      | Official JDBC Driver for MySQL         | package
  | mysql-tools               | MySQL tools                            | package
  | perl-DBD-mysql            | Interface to the MySQL database        | package
i | php5-mysql                | PHP5 Extension Module                  | package
  | postfix-mysql             | Postfix plugin to support MySQL ma     | package
i | python-mysql              | An Interface to the Popular MySQL ->   | package
  | ruby-mysql                | MySQL bindings for Ruby                | package
```

Sind die Voraussetzungen erfüllt, so kann die Installationen beginnen. In der Standardinstallation wird für alle Installationsverzeichnisse der Prefix `/usr/local/` verwendet. Es müssen daher gegebenenfalls nach der Installation die Ordner, in denen die Shared Libraries zu finden sind, systemweit bekannt gegeben werden.

```
# ./configure --enable-gtk-doc --with-postgresql=no --with-sqlite3=no
*** Dumping configuration ***
    - Generate documentation      : yes
    - Enable MySQL plugin         : yes
    - Enable PostgreSQL plugin    : no
    - Enable SQLite3 plugin       : no
    - Perl binding                : yes
    - Python binding              : yes

# make
# make check
# make install
```

In dem obigen Auszug aus der Konfiguration kann man erkennen, dass die Unterstützung für einige Datenbanksysteme nicht gegeben ist. Sollten jedoch PostgreSQL- oder SQLite3-Datenbanken unterstützt werden, müssen die entsprechenden Parameter bei der Installation abgeändert werden.

3.2.2.3 Konfiguration

Im Prinzip erfordert die Bibliothek libpreludedb keine Konfiguration. Es muss jedoch die Datenbank entsprechend vorbereitet werden, was hier am Beispiel von MySQL gezeigt wird.

```
# mysql -u root -p

# mysql> CREATE database prelude;
Query OK, 1 row affected (0.03 sec)
# mysql> exit

# mysql -u root prelude -p < /usr/local/share/libpreludedb/classic/mysql.sql

# mysqlshow -u root -p
+--------------------+
|     Databases      |
+--------------------+
| information_schema |
| mysql              |
| prelude            |
| test               |
+--------------------+

# mysqlshow -u root -p prelude
Database: prelude
+-------------------------------+
|            Tables             |
+-------------------------------+
| Prelude_Action                |
| Prelude_AdditionalData        |
| Prelude_Address               |
| Prelude_Alert                 |
| Prelude_Alertident            |
| Prelude_Analyzer              |
| Prelude_AnalyzerTime          |
| Prelude_Assessment            |
| Prelude_Checksum              |
| Prelude_Classification        |
| Prelude_Confidence            |
| Prelude_CorrelationAlert      |
| Prelude_CreateTime            |
| Prelude_DetectTime            |
| Prelude_File                  |
| Prelude_FileAccess            |
| Prelude_FileAccess_Permission |
| Prelude_Heartbeat             |
| Prelude_Impact                |
| Prelude_Inode                 |
| Prelude_Linkage               |
| Prelude_Node                  |
| Prelude_OverflowAlert         |
| Prelude_Process               |
| Prelude_ProcessArg            |
| Prelude_ProcessEnv            |
| Prelude_Reference             |
| Prelude_Service               |
| Prelude_SnmpService           |
| Prelude_Source                |
| Prelude_Target                |
| Prelude_ToolAlert             |
| Prelude_User                  |
```

```
| Prelude_UserId                |
| Prelude_WebService            |
| Prelude_WebServiceArg         |
| _format                       |
+-------------------------------+
```

Es sollte aus Sicherheitsgründen noch erwogen werden, für den Zugriff auf die Datenbank noch einen dezidierten Nutzer anzulegen, was aber optional ist.

3.2.3 Löschroutine

Wie im Datenschutzkonzept [2] gefordert, hat jeder Ressourcenprovider das Recht vorzuschreiben, nach welchem Zeitraum eine Meldung von ihm bei allen anderen beteiligten Partnern gelöscht werden soll. Dafür verpflichten sich alle teilnehmenden Partner, einen Cronjob täglich laufen zu lassen, der für das Löschen verantwortlich ist.

3.2.3.1 Implementierung

Im Wesentlichen besteht der Cronjob aus einem Konfigurationsteil und einem Shell-Aufruf der libpreludedb-Managementkomponenten.

3.2.3.2 Installation

Die Abhängikeit ist

- libpreludedb (siehe Abschnitt 3.2.2)

3.2.3.3 Konfiguration

Im Wesentlichen besteht die Konfiguration aus drei Teilen. Im ersten Teil muss man dem System sagen, an welcher Stelle das Binary von preludedb-admin zu finden ist. Wurde bei der Installation von libpreludedb keine spezielle Anpassung getroffen, so muss im Allgemeinen an dieser Stelle keine Änderung durchgeführt werden.

```
###
# Nötige Pfad-Variable, um das Programm "preludedb-admin" zu finden.
###
PATH="/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin"
```

Der nächste Block von Einstellung ist die Angabe der Datenbank, in der die Meldungen, die gelöscht werden sollen, gespeichert sind.

```
###
# Nötige Variablen, um die Datenbank anzusprechen.
###

DB_TYPE="mysql"     # Typ der Datenbank. Kann [mysql|pgsql|sqlite] sein.
DB_HOST="localhost" # IP-Adresse des Datenbankservers oder localhost.
DB_PORT="3306"      # Port, an dem der Datenbankserver lauscht.
DB_NAME="prelude"   # Name der Datenbank.
DB_USER="prelude"   # Benutzername, der benutzt wird,
                    # um auf die Datenbank zuzugreifen.
DB_PASS="xxx"       # Das zum obigen Benutzername passende Passwort.
```

Schlussendlich muss jeder Ressourcenprovider für sich entscheiden, wie viele Tage, Wochen oder Monate die eigenen Meldungen aufbewahrt werden sollen. Diese Angabe ist unabhängig davon, wann die Meldungen bei anderen Partnern gelöscht werden sollen. So ist es legitim, wenn die Meldungen bei allen anderen Partnern schon nach sieben Tagen gelöscht werden, wohingegen die Meldungen innerhalb der eigenen Domäne einen Monat lang aufbewahrt werden, solange dies den eigenen Informationssicherheitsrichtlinien entspricht.

```
###
# Datenschutzrichtline.
###

KEEP_INTERVAL_ALERT="7 days"
                    # Angabe, nach wie lange spätestens ein Alarm
                    # gelöscht werden soll.
                    # Kann auf [1 day|[2-..] days|1 month|[2-..]months|..]
                    # gesetzt werden.
KEEP_INTERVAL_HEARTBEAT="1 day"
                    # Angabe, nach wie lange spätestens ein Heartbeat
                    # gelöscht werden soll.
                    # Kann auf [1 day|[2-..] days|1 month|[2-..]months|..]
                    # gesetzt werden.
```

3.2.4 Prelude-Correlator

Der Correlator ist Teil des Prelude Rahmenwerkes und dient zur Korrelation von Alarmen aus den am Prelude angebundenen IDS. Dies sind im Fall des GIDS das NIDS Snort und das lokale IDS Prelude-LML. Ziel ist die Erkennung zusammengehöriger Angriffsmeldungen im GIDS und die Aggregation von mehreren IDS-Meldungen zu einer höherwertigen (beispielsweise brute-force Angriff).

3.2.4.1 Implementierung

Der Correlator ist in der Programmiersprache Python implementiert worden und besteht aus einem Kern, der die grundlegende Funktionalität zur Verfügung stellt. Dies ist die Erzeugung von korrelierten Alarmen im Format IDMEF (Correlation Alert) und die Funktionalität zur Korrelation von Alarmen, wobei diese auf der Basis gemeinsamer Eigenschaften (zum Beispiel IP-Adressen) erfolgen kann. Die Korrelation erfolgt durch Plug-ins, die die Funktionen des Kerns verwenden und die Kriterien implementieren, nach denen Korreliert werden soll. Fertig vorhanden sind Plug-ins für die Korrelation von Alarmen, die zu Port-Scans und brute-force Angriffen zusammengefasst werden. Weiterhin werden externen Quellen von abgefragt, die bekannte Quellen von Angriffen zur Verfügung stellen.

Da insbesondere die Plug-ins für die Korrelation von Portscans und brute-force Angriffen für das GIDS unbrauchbar waren, wurden diese re-implementiert und an die Bedürfnisse des GIDS angepasst. Weiterhin wurde ein Plug-in zur Korrelation von kritischen Angriffen entworfen.

3.2.4.1.1 Installation

Momentan existiert noch keine Paketierung des Programms, so dass man um eine manuelle Installation nicht herumkommt.

Die Abhängigkeiten sind:

- libprelude (siehe Abschnitt 3.1.1)
- Python 2.4 oder spätere Versionen

Die Installation erfolgt durch Aufruf des Python Skriptes zur Installation; als Benutzer “root”

```
# python setup.py install
```

oder als unprivilegierter Benutzer:

```
$ python setup.py install --prefix /prefix/
```

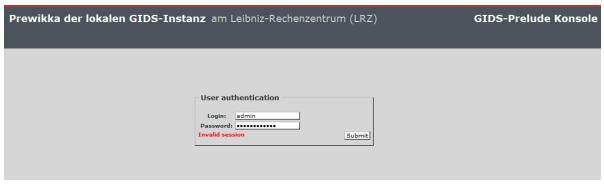

Abbildung 3.2: Das Login-Fenster von Prewikka

3.2.4.1.2 Konfiguration

In der Konfigurationsdatei `prelude-correlator.conf`, die in Anhang G abgedruckt ist, können die Plug-ins des Correlators einzeln konfiguriert und aktiviert werden.

Da der Correlator als Client im Prelude-Rahmenwerk implementiert ist, muss dieser sich noch am zuständigen Prelude-Manager registrieren, damit der Correlator zum einen berechtigt ist, Daten mit den Prelude-Manager austauschen zu dürfen und zum anderen, damit der verschlüsselte Versand vom Meldungen vorbereitet wird. Dazu muss auf Maschine, auf der Prelude-Correlator läuft, der Befehl

```
# prelude-admin register prelude-correlator "idmef:w admin:r" localhost --uid 0 --gid 0
Generating 2048 bits RSA private key... This might take a very long time.
```

ausgeführt werden. Dabei muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft, mit dem man sich verbinden möchte. Weiterhin müssen die Angaben `uid` und `gid` auf die Werte, die Prelude-Correlator im laufenden Betrieb hat.

Ist die Generierung des Schlüsselpaares abgeschlossen, muss am Server des Prelude-Managers die Registrierung abgeschlossen werden. Dazu dient der Befehl

```
# prelude-admin registration-server prelude-manager --listen localhost
Generating 2048 bits RSA private key... This might take a very long time.
```

Auch hier muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft. Am Ende wird einem hier ein Passwort angezeigt, dass man auf Seiten der Prelude-Correlator-Ressource wiederholen muss. Damit ist die Konfiguration abgeschlossen.

3.3 Berichterstattung und Datenweitergabe

3.3.1 Prewikka

Um Meldungen aus der Datenbank Site-lokal anzeigen zu können, verwenden wir die in Python geschriebene Software *Prewikka*. Diese dient als Schnittstelle zwischen dem Prelude-Framework und dem Administrator, da sie neben der Anzeige von Alarmen und Korrelationsalarmen, wie in den Abbildungen 3.3 und 3.4 gezeigt, auch eine Löschung der Alarme vorsieht. Weiterhin bietet Prewikka eine Übersicht über den Status der Sensoren, so dass ein Ausfall sofort bemerkt wird, wie in Abbildung 3.5 ersichtlich. Allerdings ist die Skalierbarkeit von Prewikka sehr begrenzt, was sowohl an dem aufwändigen Datenbankschema als auch an der Darstellung der Alarme liegt. Des weiteren ist Prewikka nicht mandantenfähig. Aus diesen Gründen verbietet sich der direkte Einsatz auf der Seite des Betreibers. Da der Aufwand für die Anpassung an die Anforderungen höher als eine Neuwentwicklung ist, wurde des GIDS-Portals vollständig neu entwickelt. Diese wurde auch durch Nutzung von vorhandenem Codeund Strukturen des DFN-CERT Portals deutlich vereinfacht.

3.3.1.1 Implementierung

Bei der Implementierung des in Python entwickelten Prewikka wurde auf die vorhandenen Sourcen des unter GPLv2 stehenden Projekts Prelude zurückgegriffen. Ursprünglich konnte man unter `http://www.prelude-technologies.com/en/development/download/index.`

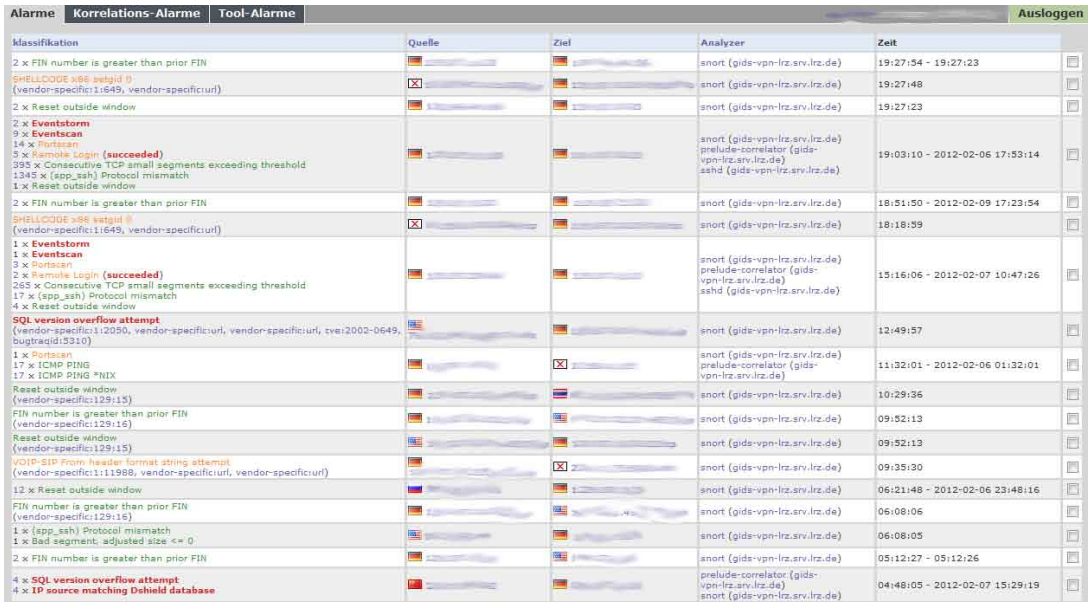

Abbildung 3.3: Übersicht über die Alarmmeldungen

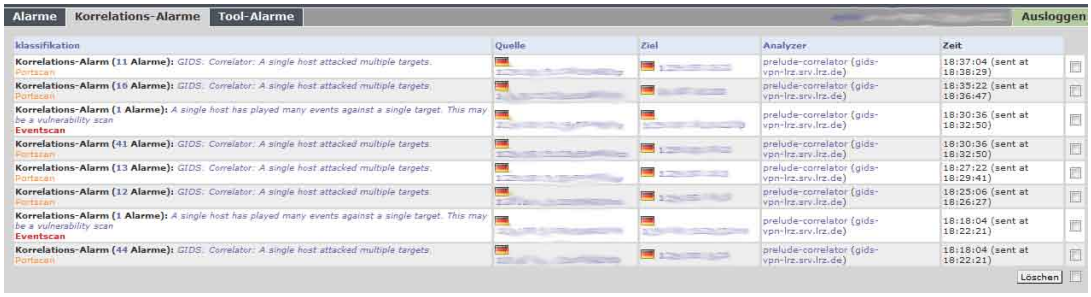

Abbildung 3.4: Übersicht über die Korrelationsalarme

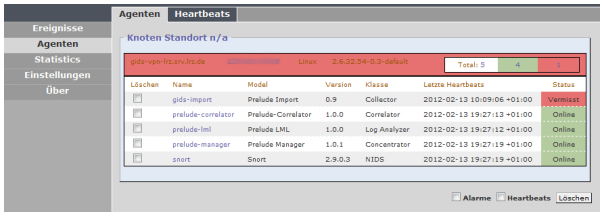

Abbildung 3.5: Sind Sensoren ausgefallen, wird dies deutlich angezeigt.

`html` die Sourcen herunterladen. Um die Unabhängigkeit der Projektergebnisse zu gewährleisten, wurden die Sourcen als Git-Repository auf `http://git.lrz.de/prelude/` veröffentlicht. Anpassungen können so direkt einer großen Community zugutekommen.

Im Projekt wurden diverse Verbesserungen und Fehlerkorrekturen durchgeführt. Weiterhin wurde die Anzeige um weitere Punkte erweitert, so dass Informationen schneller visuell erfassbar werden.

3.3.1.2 Installation

Im Gegensatz zu einer Installation aus den vorhandenen Paketen muss bei einer Installation aus den Quelltexten eine ganze Reihe von Abhängigkeiten gelöst werden. Im Allgemeinen ist aus Aufwandsgründen eine Installation aus den Paketen vorzuziehen, jedoch existieren vor allem für SLES- und openSUSE-Systeme im Gegensatz zu beispielsweise Debian keine vorkonfigurierten Pakete.

Die Abhängigkeiten sind

- libprelude (siehe Abschnitt 3.1.1)
- libpreludedb (siehe Abschnitt 3.2.2)
- MySQL (siehe Abschnitt 3.2.1)
- Python Cheetah (`http://cheetahtemplate.org/`)
- Python Cairo
- GeoIP C Library (`http://www.maxmind.com/app/c`)
- GeoIP Python (`http://www.maxmind.com/app/python`)
- Apache2

```
# zypper install python-cairo-devel
# zypper search python-cairo
S | Name               | Summary                    | Type
--+--------------------+----------------------------+--------
i | python-cairo       | Python Bindings for Cairo | package
i | python-cairo-devel | Headers for python-cairo  | package
```

Sind die Voraussetzungen erfüllt, so kann die Installationen beginnen.

```
# python setup.py install
```

3.3.1.3 Konfiguration

In der Standardinstallation werden an alle Installationspfade der Präfix `/usr/local/` angehängt.

Als erstes muss die nötige Datenbank erstellt werden, damit die nötigen Daten, wie Berechtigungen, gespeichert werden können.

```
# mysql -u root -p

# mysql> CREATE database prewikka;
Query OK, 1 row affected (0.03 sec)
# mysql> exit

# mysql -u root prelude -p < /usr/local/share/prewikka/database/mysql.sql

# mysqlshow -u root -p
+--------------------+
|     Databases      |
+--------------------+
| information_schema |
| mysql              |
| prelude            |
| prewikka           |
| test               |
```

```
+--------------------+

# mysqlshow -u root -p prelude
Database: prewikka
+-----------------------------+
|           Tables            |
+-----------------------------+
| Prewikka_Filter             |
| Prewikka_Filter_Criterion   |
| Prewikka_Permission         |
| Prewikka_Session            |
| Prewikka_User               |
| Prewikka_User_Configuration |
| Prewikka_Version            |
+-----------------------------+
```

Es sollte aus Sicherheitsgründen noch erwogen werden, für den Zugriff auf die Datenbank noch einen dezidierten Nutzer anzulegen, was aber optional ist.

In der Standardinstallation wird unter `/usr/local/etc/prewikka/prewikka.conf` die Konfigurationsdatei von Prewikka abgespeichert. Diese muss nach der Installation angepasst werden, um beispielsweise die Datenbankverbindung korrekt einzurichten. Die unveränderte Konfigurationsdatei ist in Anhang D unverändert mit angehängt. Im Folgenden werden wichtige Stellen in der Konfiguration hervorgehoben und kommentiert.

Als erstes kann die Default-Sprache eingestellt werden. Diese ist in der Grundeinstellung Englisch.

```
# Default locale to use (default is English):
# default_locale: fr

# Default encoding to use (default is UTF8):
# encoding: utf8
```

An der oberen Seite werden, wie in Abbildung 3.2 zu sehen, eine lokalisierte Headerzeile angezeigt. Diese Angaben können hier individualisiert werden.

```
[interface]
software: Prewikka
place: company ltd.
title: Prelude console
```

Eine sehr zentrale Angabe ist die Folgende, in der zwei Angaben gemacht werden müssen. Zum einen muss die oben erzeugte Prewikka-Datenbank referenziert werden und zum anderen müssen Angaben dazu gemacht werden, welche Zugangsdaten für die Datenbank nötig sind, in der die IDMEF-Nachrichten des Prelude-Managers gelten.

```
[idmef_database]
#
# if your database is a sqlite file, please use:
#
# type: sqlite3
# file: /path/to/your/sqlite_database
#
type: mysql
host: localhost
user: prelude
pass: prelude
name: prelude
```

```
[database]
type: mysql
host: localhost
user: prelude
pass: prelude
name: prewikka
```

Wurde noch nichts angepasst, gilt für den administrativen Zugang die Benutzname/Passwort-Kombination `admin/admin`, was schleunigst geändert werden muss, da man über den Prewikka-Zugang jegliche Meldung aus der Datenbank löschen kann.

```
# Standard login / password authentication:
[auth loginpassword]
expiration: 60
# If there is no user with administrative right defined in the database,
# the initial user will be created according to these settings:
initial_admin_user: admin
initial_admin_pass: admin
```

Alternativ kann man die in Prewikka integrierte Authentifikation ausschalten und dafür diejenige des Webservers verwenden.

```
# Rely on webserver for user authentication:
#
# User that authenticate for the first time won't have any permission.
# If the "default_admin_user" option is provided, the specified user will
# be granted ALL access, allowing to edit other users permissions.
#
# [auth cgi]
# default_admin_user: myuser


# Disable Prewikka authentication:
# [auth anonymous]
```

Sollen Fehler oder Warnungen mitgeloggt werden, können hier dafür Angaben gemacht werden.

```
# Logging configuration:
# - You can activate several log section.
# - Log level might be set to all/debug, info, warning, error, critical.
#   If unspecified, the default level is "warning".

# [log stderr]
# level: info

# [log file]
# level: debug
# file: /tmp/prewikka.log

# [log syslog]
# level: info

# [log nteventlog]
# level: info

# [log smtp]
# level: warning
```

```
# host: mail.domain.com
# from: user@address
# to: recipient1@address, recipient2@address, recipientN@address
# subject: Subject to use
```

Damit ist die Grundkonfiguration abgeschlossen.

Am Ende muss man noch den Webserver so konfigurieren, dass er auch bei Anfragen die Prewikka-Seiten ausliefern kann.

```
<VirtualHost *:80>
 ServerName my.server.org
 Setenv PREWIKKA_CONFIG "/usr/local/etc/prewikka/prewikka.conf"

<Location "/">
        AllowOverride None
        Options ExecCGI

        <IfModule mod_mime.c>
                AddHandler cgi-script .cgi
        </IfModule>

        Order allow,deny
        Allow from all
</Location>

Alias /prewikka/ /usr/local/share/prewikka/htdocs/
ScriptAlias / /usr/local/share/prewikka/cgi-bin/prewikka.cgi

</VirtualHost>
```

Alternativ ist in Anhang D eine Möglichkeit beschrieben, wie man auch eine verschlüsselte Verbindung mit Prewikka herstellen kann.

3.3.2 Export

Die Komponente Prelude-Export ist für den Exportvorgang von IDMEF-Nachrichten aus dem Prelude-Framework verantwortlich.

3.3.2.1 Implementierung

Die C basierte Implementierung des Prelude-Exports verwendet die libxml, da diese Bibliothek die Verarbeitung des XML-basierte Formats IDMEF vereinfacht. Das Programm liest aus dem Prelude-Manager kommende komplette Meldungen im Prelude-eigenen Binärformat ein. Für jeden Teil, die in dieser Nachricht vorhanden ist, wird durch die entsprechende Funktion der libprelude-API die entsprechende Information extrahiert. Schlussendlich wird mit Hilfe der von libxml bereitgestellten Funktionen ein kompletter XML-String generiert.

Prelude-Export besitzt zwei Möglichkeiten, um neue IDMEF-Nachrichten zu senden. Entweder werden die Nachrichten in das `stdout` geschrieben oder sie werden direkt auf den GIDS-Bus gesendet. Dafür besitzt Prelude-Export eine Anbindung an libemcast, das eine API für das in Kapitel 4.1.1 beschriebene Programm Emcast dient. Welcher Weg für die Dateneingabe benutzt wird, wird in der Konfigurationsdatei `prelude-export.conf` festgelegt. Ist dort `emcast = true` angegeben, wird eine Emcast-Verbindung aufgebaut und alle Nachrichten werden von dort gelesen. In jedem anderen Fall wird `stdout` verwendet.

Um den in [2] beschriebenen und ausgeführten Datenschutzkonzept genüge zu tun, werden alle Meldungen vor einem Exportvorgang geeignet manipuliert. Dabei werden alle XML-Knoten, die potentiell den Datenschutz verletzen könnten gelöscht. Welche Felder das sind oder aus welchen Gründen eine Löschung nötig sein kann, wird in Kapitel 5 beschrieben.

3.3.2.2 Installation

Momentan existiert noch keine Paketierung des Programms, so dass man um eine manuelle Kompilierung nicht herumkommt.

Die Abhängigkeiten sind

- libprelude (siehe Abschnitt 3.1.1)
- libxml
- iniparser (`http://ndevilla.free.fr/iniparser/`)
- libemcast (`http://git.lrz.de/gitweb/?p=Emcast.git;a=summary`)

Die oben genannten Pakete iniparser und libemcast sind nicht Teil der Paketverwaltung von SLES. Daher müssen diese Pakete manuell installiert werden.

Sind die Voraussetzungen erfüllt, so kann die Installationen beginnen, die in diesem Fall durch Kompilierung des Quelltextes besteht.

```
# gcc prelude-export.c -o prelude-export
    ‘xml2-config --cflags --libs‘
    ‘emcast-config --cflags --libs‘
    ‘libprelude-config --cflags --libs‘
    -I./iniparser/iniparser-3.0/src -L./iniparser/iniparser-3.0 -liniparser
```

3.3.2.3 Konfiguration

Wie schon eingangs erwähnt, gibt es die Konfigurationsdatei `prelude-export.conf`, die in Anhang F abgedruckt ist. In dieser kann man festlegen, ob man Nachrichten direkt aus dem GIDS-Bus mittels emcast empfangen will und wenn ja, wie die Parameter dafür aussehen.

```
#########
# Spezielle Einstellungen für die Verbindung zum GIDS-Bus
#########
[Emcast]
emcast = true        ; Schaltet Emcast-Unterstützung ein.
                     ; Die Alarme werden dann an den GIDS-Bus weitergeleitet.

url = 224.1.2.3:1234 ; Die Broadcastadresse des GIDS-Bus

buffer = 16384       ; Die Größe des Buffers zum Empfang der Daten

loopback = false     ; Wenn Nachrichten, die an den GIDS-Bus versendet
                     ; werden sollen, auch an die absendende Site
                     ; geschickt werden soll, muss hier "true"
                     ; eingetragen werden. ACHTUNG: Diese Einstellung
                     ; kann bei unsachgemäßer Verwendung eine Schleife
                     ; und damit eine Überlastung produzieren!!!
```

Die Angaben im Abschnitt `Anonymisieren` beschreiben die in Kapitel 5 abgedruckte Tabelle. Dabei entsprechen alle mit `D` gekennzeichneten XML-Knoten denjenigen, die potentiell den Datenschutz betreffen und alle mit `LS` gekennzeichneten, die potentiell eine lokale Sicherheitsrichtlinie verletzen. In der Standardkonfiguration werden alle Daten, die den Datenschutz betreffen, ausgesondert, während alle Daten, die potentiell die lokale Sicherheitsrichtlinie verletzen können, weiter gegeben werden.

```
[Anonymisieren]
Action.Inhalt = true                        ; D / LS
AdditionalData.Inhalt = true                ; D / LS
Address.address = true                      ; D / LS
Address.netmask = false                     ; LS
Address.vlan-name = false                   ; LS
Address.vlan-num = true                     ; D / LS
Analyzer.name = false                       ; LS
Analyzer.manufacturer = false               ; LS
Analyzer.model = false                      ; LS
Analyzer.version = false                    ; LS
Analyzer.ostype = false                     ; LS
Analyzer.osversion = false                  ; LS
Checksum.key = false                        ; LS
File.name = true                            ; D / LS
File.path = true                            ; D / LS
File.disk-size = false                      ; LS
File.fstype = false                         ; LS
File.file-type = false                      ; LS
FileAccess.Permission = false               ; LS
Impact.Inhalt = true                        ; D / LS
Linkage.name = true                         ; D / LS
Linkage.path = true                         ; D / LS
Node.location = false                       ; LS
Node.name = true                            ; D / LS
OverflowAlert.buffer = true                 ; D / LS
Process.name = false                        ; LS
Process.path = true                         ; D / LS
Process.arg = true                          ; D / LS
Process.env = true                          ; D / LS
Service.name = false                        ; LS
Service.port = false                        ; LS
Service.portlist = false                    ; LS
Service.protocol = false                    ; LS
Service.ip_version = false                  ; LS
Service.iana_protocol_number = false        ; LS
Service.iana_protocol_name = false          ; LS
SNMPService.oid = false                     ; LS
SNMPService.messageProcessingModel = false ; LS
SNMPService.securityModel = false           ; LS
SNMPService.securityName = false            ; LS
SNMPService.securityLevel = false           ; LS
SNMPService.contextName = false             ; LS
SNMPService.contextEngineID = false         ; LS
SNMPService.command = true                  ; D / LS
Source.interface = false                    ; LS
Target.interface = false                    ; LS
UserId.name = true                          ; D / LS
UserId.number = true                        ; D / LS
UserId.tty = false                          ; LS
WebService.arg = true                       ; D / LS
WebService.cgi = false                      ; LS
WebService.url = false                      ; LS
```

Für die Zuordnung einer Meldung zu einer Ressource bzw. zu einer Institution, ist es erforderlich, dass man die Herkunft der Meldung kenntlich macht. Für diesen Zweck benutzen wir die

im D-Grid etablierte GRRS-Datenbank, bei der eine eindeutige Zuordung einer Grid-Ressource zu einem Verantwortlichen gegeben ist. Weiterhin gibt man hier an, nach wievielen Tagen eine Meldung bei allen anderen Partnern gelöscht werden muss.

```
[SiteSpezifika]
gidsRessource = lrz ; Die gleiche Angabe wie in der GRRS-Datenbank
validUntil = 7 ; Anzahl der Tage, die die Meldung bei anderen Sites
               ; gespeichert werden kann.
```

Neben diesen global interessanten Einstellungsmöglichkeiten gibt es noch einige, die Site-lokal interessant sind. Dazu gehört, nach wieviel Sekunden ein Heartbeat gesendet werden soll oder wie die Prelude-Manager-Adresse lautet, von dem die Meldungen empfangen werden sollen.

```
[Prelude}
heartbeat = 600 ; Anzahl der Sekunden,
                ; nach denen ein Heartbeat geschickt werden soll.
timeout = 1 ; Schlafe eine vorgegebene Anzahl von Millisekunden
            ; zwischen der Abfrage,
            ; ob neue Meldungen im Nachrichtenpool vorhanden sind.
manager = 127.0.0.1 ; Die Adresse des Prelude-Managers,
                    ; von dem man die Nachrichten empfängt.
importName = gids-import ; Name des Sensors, der für den
                         ; Importvorgang aus dem GIDS-Bus zuständig ist.
```

Schlussendlich muss Prelude-Export sich noch am zuständigen Prelude-Manager registrieren, damit Prelude-Export zum einen berechtigt ist, Daten mit den Prelude-Manager austauschen zu dürfen und zum anderen, damit der verschlüsselte Versand vom Meldungen vorbereitet wird. Dazu muss auf der Maschine, auf der Prelude-Export läuft, der Befehl

```
# prelude-admin register prelude-export "idmef:w admin:r" localhost --uid 0 --gid 0
Generating 2048 bits RSA private key... This might take a very long time.
```

ausgeführt werden. Dabei muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft, mit dem man sich verbinden möchte. Weiterhin müssen die Angaben `uid` und `gid` auf die Werte, die Prelude-Export im laufenden Betrieb hat.

Ist die Generierung des Schlüsselpaares abgeschlossen, muss am Server des Prelude-Managers die Registrierung abgeschlossen werden. Dazu dient der Befehl

```
# prelude-admin registration-server prelude-manager --listen localhost
Generating 2048 bits RSA private key... This might take a very long time.
```

Auch hier muss die Angabe `localhost` durch den Server ersetzt werden, auf dem der Prelude-Manager läuft. Am Ende wird einem hier ein Passwort angezeigt, dass man auf Seiten der Prelude-Export-Ressource wiederholen muss. Damit ist die Konfiguration von Prelude-Export abgeschlossen.

Kapitel 4

Der GIDS-Betreiber

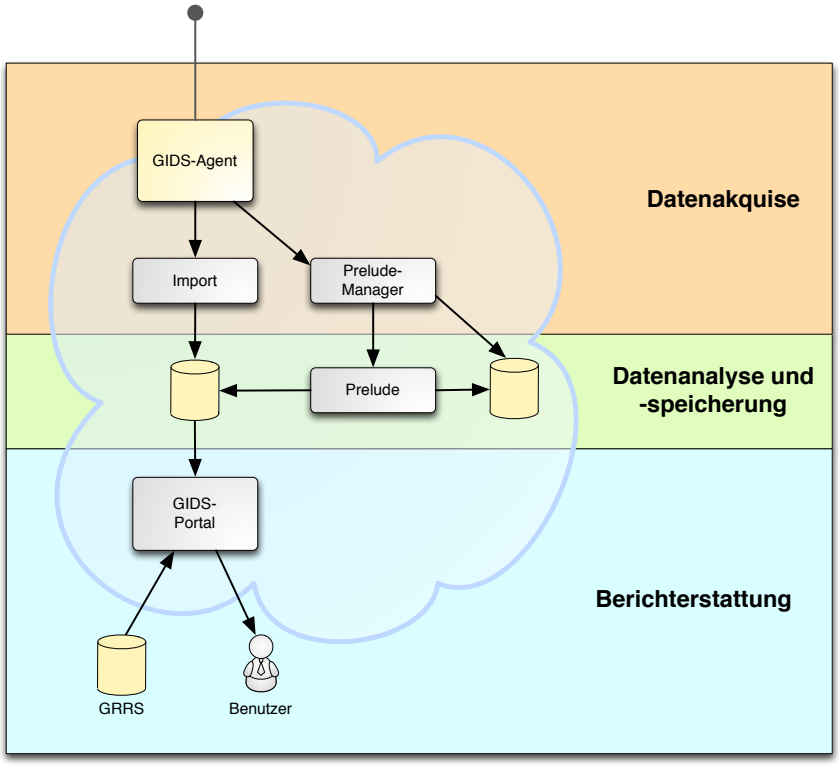

Abbildung 4.1: Überblick über den Aufbau von GIDS auf Betreiberseite

Den Betreiber des GIDS zeichnet in besonderem Maße aus, dass seine konzeptuelle Rolle vorsieht, dass er selber keine Daten zur Analyse erhebt, sondern vielmehr alle auf dem GIDS-Bus verfügbar gemachten Informationen liest (Abschnitt 4.1), speichert und auswertet (Abschnitt 4.2) und darüber berichtet (Abschnitt 4.3). Entsprechend verfügt der GIDS-Betreiber lediglich über einen GIDS-Agenten und daran angeschlossene Import-Routinen, die eine Speicherung zur darauf folgenden Analyse und Anzeige realisieren. Abbildung 4.1 stellt einen Überblick über die auf Seiten des Betreibers des GIDS installierten Komponenten und ihre Interaktion miteinander dar. Die Abbildung dient zugleich der Strukturierung dieses Kapitels.

4.1 Datenakquise

Die Datenakquise beschränkt sich im Falle des GIDS-Betreibers auf das Lesen und Importieren von IDMEF-Nachrichten, die über den GIDS-Bus verbreitet werden. Entsprechend bedarf es eines Zugriffs auf den GIDS-Bus (vgl. Abschnitt 4.1.1) sowie entsprechenden Import-Routinen (vgl. Abschnitt 4.1.2) zur Speicherung der Informationen.

4.1.1 Empfangen und senden von IDMEF-Nachrichten per emcast

Nach der erfolgreichen Installation eines GIDS-Agenten (vgl. Kap. 2.2) verfügt auch der Betreiber des GIDS über Zugriff auf den GIDS-Bus. Da die Bus-artige Struktur über eine Multicast-Gruppe realisiert ist, bedarf es eines Werkzeugs, das das Binden und Lesen der Multicast-Gruppe übernimmt. Hierfür kommt `emcast` zum Einsatz.

4.1.1.1 Implementierung

`emcast` ist ein einfaches, Kommandozeilen-basiertes Werkzeug, das Multicast-basierte Kommunikation sowohl lesend, als auch schreibend ermöglicht. `emcast` ist freie Software und von der *Regents of the University of Michigan* entwickelt. Der Quellcode ist unter `http://www.gizmolabs.org/~dhelder/junglemonkey/emcast/` verfügbar.

4.1.1.2 Installation

Die Installation von `emcast` ist lediglich ordentlich durch Übersetzen seines Quellcodes möglich. Zwar existieren ein paar wenige Paketierungen von `emcast`, die jedoch wenig stabil und daheer nicht weiter empfehlenswert sind. Eine Übersetzung und Installation ist durch die gewohnte Befehlsfolge

```
./configure
make
make install
```

trivial möglich.

4.1.1.3 Konfiguration

Eine Konfiguration von `emcast` selber ist nicht notwendig. Alle notwendigen Einstellungen wie z. B. das Binden an eine Multicast-Gruppe. Somit kann `emcast` zum Beispiel mit dem Aufruf `emcast 224.1.2.3:1234` gestartet werden. Dabei werden Daten, die auf der Standardeingabe an emcast übergeben werden, an die Multicast-Gruppe versendet und alle eingehenden Informationen auf der Standardausgabe ausgegeben. Diese Implementierung bietet sich entsprechend dafür an in Kombination mit *named pipes* eingesetzt zu werden.

4.1.2 Import-Routinen

Nachdem per `emcast` Daten vom GIDS-Bus gelesen werden können, werden die verfügbaren Informationen auf zweierlei Wegen zur weiteren Verarbeitung gespeichert (vgl. auch Abbildung 4.1). Zum einen kommt auch auf der Betreiberseite von GIDS *prelude* mit seiner zugehörigen Datenbank zum Einsatz (siehe hierzu Kap. 3). Auf der anderen Seite wird zusätzlich eine Speicherung der empfangenen IDMEF-Nachrichten in einer GIDS-proprietären Datenbank gespeichert (vgl. dazu Abschnitt 4.2). Diese Datenbank bildet IDMEF-Nachrichten nicht in voller semantischer Tiefe ab, umfasst allerdings alle für das GIDS-Portal notwendigen Informationen und arbeitet erheblich viel performanter.

4.1.2.1 Implementierung

Um eingehende IDMEF-Nachrichten, die in einem XML-basierten Fomat eintreffen, in der GIDS-proprietären Datenbank zu speichern, kommt eine XSL-Transformation zum Einsatz. Hierzu existiert ein XSLT-Stylesheet, das eine IDMEF-Nachricht in eine SQL-Anfrage transformiert. Aufgrund seiner Länge wird dieses XSLT-Stylesheet an dieser Stelle nicht weiter aufgelistet.

Um das XSLT-Stylesheet auf eine IDMEF-Nachricht anwenden zu können, wird ein XSL-Prozessor benötigt. Im Rahmen von GIDS hat sich der Einsatz von `libxml` bzw. `xsltproc` dafür sehr bewährt. Andere XSL-Prozessoren können aber selbstverständlich ebenfalls verwendet werden.

4.1.2.2 Installation

Eine Installation ist neben dem Kopieren des XSLT-Stylesheet nicht weiter nötig.

4.1.2.3 Konfiguration

Das XSLT-Stylesheet bedarf keiner Konfiguration.

4.2 Datenanalyse und -speicherung

Wie aus Abbildung 4.1 hervorgeht, werden sämtliche vom GIDS-Bus gelesenen Daten in zwei Datenbanken gespeichert: Zum einen kommt eine Datenbank für das IDS *prelude* zum Einsatz (vgl. hierzu Kapitel 3), zum anderen wird ein GIDS-proprietäres Datenbankschema entworfen, das als Basis für die Anzeigen im GIDS-Portal dient.

4.2.1 Ein SQL-Schema zur effizienten Abbildung des IDMEF

Das eigens im Rahmen von GIDS entwickelte Datenbankschema bildet zwar das IDMEF nicht vollständig ab, enthält aber alle für das GIDS-Portal notwendigen Daten. Die Notwendigkeit für ein schlankes Datenbankschema ist aus dem Wunsch nach schnellen Antwortzeiten und der besseren Bedienbarkeit des GIDS-Portals erwachsen.

4.2.1.1 Implementierung

Abbildung 4.2 zeigt den Entwurf des speziell auf die Anforderungen und Notwendigkeiten, die durch das GIDS-Portal gegeben sind, angepassten Datenbankschemas, das das XML-basierte IDMEF im Rahmen einer relationalen Datenbank abbilden kann. Dafür stehen für ausgewählte Attribute des IDMEF, die in einer IDMEF-Nachricht mehrfach auftreten können und für GIDS und das Portal eine entsprechende Relevanz haben, eigene Tabellen zur Verfügung, die über Fremdschlüsselbeziehungen mit dem ursprünglichen Eintrag in Verbindung gebracht werden können.

4.2.1.2 Installation

Die Installation des proprietären Datenbankschemas ist insofern trivial, als dass ein in Listing 4.1 gezeigtes SQL-Skript zur Ausführung bereitsteht. Nach der Ausführung dieses Skripts steht das in Abbildung 4.2 modellierte Schema bereit.

Listing 4.1: idmef.sql

```
DROP TABLE IF EXISTS IDMEF_Message;
CREATE TABLE IDMEF_Message (
  id BIGINT UNSIGNED NOT NULL PRIMARY KEY AUTO_INCREMENT,
  type ENUM('A','H') NOT NULL                                  # A=Alert, H=↘
    →Heartbeat
);
```

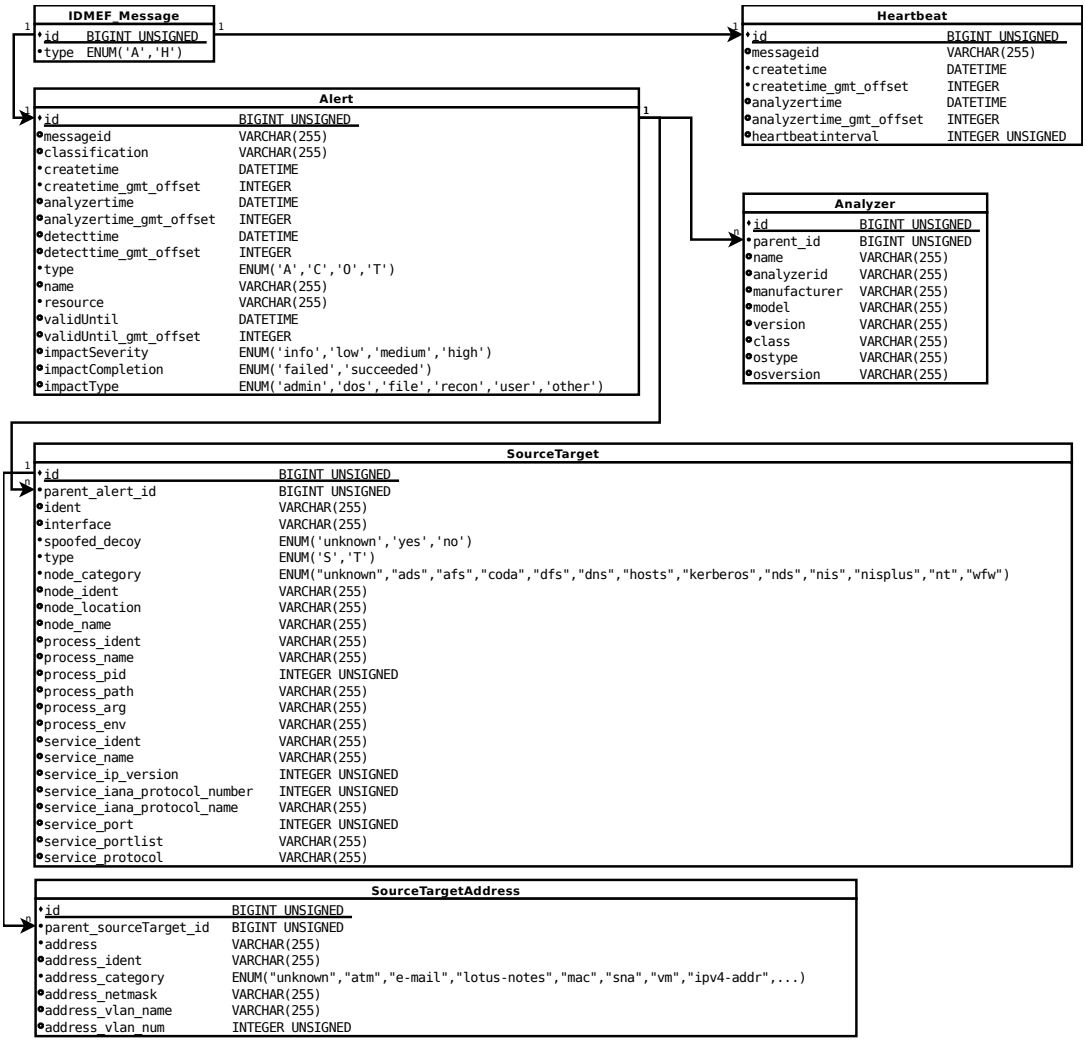

Abbildung 4.2: Entwurf eines eigenen SQL-Schmas zur Repräsentation von Basisinformationen des IDMEF in einer Datenbank.

```
DROP TABLE IF EXISTS Alert;
CREATE TABLE Alert (
  id BIGINT UNSIGNED NOT NULL PRIMARY KEY,
  messageid VARCHAR(255) NULL,
  classification VARCHAR(255) NULL,
  createtime DATETIME NOT NULL,
  createtime_gmt_offset INTEGER NOT NULL,
  analyzertime DATETIME NULL,
  analyzertime_gmt_offset INTEGER NULL,
  detecttime DATETIME NULL,
  detecttime_gmt_offset INTEGER NULL,
  type ENUM('A','C','O','T') NOT NULL,                    # A="plain alert", ↘
    →C=CorrelationAlert, O=OverflowAllert, T=ToolAlert
  name VARCHAR(255) NULL,                                 # only given/valid,↘
    → if type!=A
  resource VARCHAR(255) NOT NULL,                         # ressource, where ↘
    →the alarm was raised
  validUntil DATETIME NULL,
  validUntil_gmt_offset INTEGER NULL,
  impactSeverity ENUM('info','low','medium','high') NULL,
  impactCompletion ENUM('failed','succeeded') NULL,
  impactType ENUM('admin','dos','file','recon','user','other') NULL
);

DROP TABLE IF EXISTS Analyzer;
CREATE TABLE Analyzer (
  id BIGINT UNSIGNED NOT NULL PRIMARY KEY AUTO_INCREMENT,
  parent_id BIGINT UNSIGNED NOT NULL,
  name VARCHAR(255) NULL,
  analyzerid VARCHAR(255) NULL,
  manufacturer VARCHAR(255) NULL,
  model VARCHAR(255) NULL,
  version VARCHAR(255) NULL,
  class VARCHAR(255) NULL,
  ostype VARCHAR(255) NULL,
  osversion VARCHAR(255) NULL
);

DROP TABLE IF EXISTS SourceTarget;
CREATE TABLE SourceTarget (
  id BIGINT UNSIGNED NOT NULL PRIMARY KEY AUTO_INCREMENT,
  parent_alert_id BIGINT UNSIGNED NOT NULL,
  ident VARCHAR(255) NULL,
  interface VARCHAR(255) NULL,
  spoofed_decoy ENUM('unknown','yes','no') NOT NULL,
  type ENUM('S','T') NOT NULL, # S=Source, T=Target
  node_category ENUM("unknown","ads","afs","coda","dfs","dns","hosts","↘
    →kerberos","nds","nis","nisplus","nt","wfw") NOT NULL,
  node_ident VARCHAR(255) NULL,
  node_location VARCHAR(255) NULL,
  node_name VARCHAR(255) NULL,
  process_ident VARCHAR(255) NULL,
  process_name VARCHAR(255) NULL,
  process_pid INTEGER UNSIGNED NULL,
  process_path VARCHAR(255) NULL,
  process_arg VARCHAR(255) NULL,
  process_env VARCHAR(255) NULL,
  service_ident VARCHAR(255) NULL,
  service_name VARCHAR(255) NULL,
  service_ip_version INTEGER UNSIGNED NULL,
  service_iana_protocol_number INTEGER UNSIGNED NULL,
  service_iana_protocol_name VARCHAR(255) NULL,
  service_port INTEGER UNSIGNED NULL,
  service_portlist VARCHAR(255) NULL,
  service_protocol VARCHAR(255) NULL
);

DROP TABLE IF EXISTS SourceTargetAddress;
CREATE TABLE SourceTargetAddress (
  id BIGINT UNSIGNED NOT NULL PRIMARY KEY AUTO_INCREMENT,
```

```
  parent_sourceTarget_id BIGINT UNSIGNED NOT NULL,
  address VARCHAR(255) NOT NULL,
  address_ident VARCHAR(255) NULL,
  address_category ENUM("unknown","atm","e-mail","lotus-notes","mac","sna","vm↘
    →","ipv4-addr","ipv4-addr-hex","ipv4-net","ipv4-net-mask","ipv6-addr","↘
    →ipv6-addr-hex","ipv6-net","ipv6-net-mask") NOT NULL,
  address_netmask VARCHAR(255) NULL,
  address_vlan_name VARCHAR(255) NULL,
  address_vlan_num INTEGER UNSIGNED NULL
);


DROP TABLE IF EXISTS Heartbeat;
CREATE TABLE Heartbeat (
  id BIGINT UNSIGNED NOT NULL PRIMARY KEY,
  messageid VARCHAR(255) NULL,
  createtime DATETIME NOT NULL,
  createtime_gmt_offset INTEGER NOT NULL,
  analyzertime DATETIME NULL,
  analyzertime_gmt_offset INTEGER NULL,
  heartbeatinterval INTEGER UNSIGNED NULL
);
```

4.2.1.3 Konfiguration

Neben dem Anlegen eines Nutzers mit den entsprechenden Rechten für die Datenbank bedarf es keiner weiteren Konfiguration. Selbstverständlich müssen die festgelegten Zugangsdaten in den Konfigurationsdateien des GIDS-Portals angepasst werden (vgl. hierzu Kapitel 4.3.1).

4.2.2 Löschroutine für eigenes SQL

Im Sinne des Datenschutzes sowie lokaler Sicherheitsrichtlinies der Ressourcenanbieter ist eine Löschung von übermittelten Daten nach einer gewissen Zeit notwendig. Um dieser Anforderung nachzukommen, verfügt jede gespeicherte Alarmmeldung über einen zusätzlichen Zeitstempel `validUntil`, der den spätesten Zeitpunkt einer Löschung angibt. Entsprechend dieser Vorgabe muss nun in regelmäßigen Abständen eine Verschlankung des vorgehaltenen Datenbestandes um „zu alt gewordene“ Datensätze vorgenommen werden. Hierzu bietet sich das Einrichten eines `cronjob`s an, der z. B. stündlich alle überalterten Alarme und die damit in anderen Tabellen verbundenen Informationen (vgl. hierzu Abbildung 4.2, die das entsprechende SQL-Schema als Model darstellt) löscht.

4.3 Berichterstattung und Datenweitergabe

Ein zentraler Bestandteil der GIDS-Infrastruktur ist das GIDS-Portal. Diese Webschnittstelle ermöglicht es allen Benutzern des D-Grids, sich über die aktuellen Sicherheitsvorfälle im D-Grid zu informieren. Der Fokus liegt hierbei auf Sicherheitsvorfällen, die an Grid-Sites aufgetreten sind, auf denen der Benutzer Berechtigungen zur Nutzung der dortigen Ressourcen besitzt. Aber nicht nur die Benachrichtigung über lokal bei den Ressourcenbetreibern erkannte Sicherheitsvorfälle steht hierbei im Fokus. Vielmehr werden auch solche Alarmmeldungen dargestellt, die durch das zentral betriebene Grid-weite IDS erkannt wurden. Das schließt unter anderem auch Angriffe ein, die nicht alleine durch lokal implementierte Sicherheitssysteme erkannt werden können. Durch die Grid-weite Korrelation von Alarmmeldungen in der zentralen GIDS-Instanz können ebenso Angriffe erkannt und dargestellt werden, die sich über mehrere Grid-Sites erstrecken. Im folgenden Abschnitt wird das GIDS-Portal, seine Funktionen und die Nutzung beschrieben. Danach folgt eine Beschreibung der zugrundeliegenden Techniken und ihrer Implementierung. Die für den Betrieb notwendige Installation der Komponenten für das GIDS-Portal, sowie seine Konfiguration werden in einem weiteren Abschnitt beschrieben.

4.3.1 Portal

Das GIDS-Portal dient als zentrale Anlaufstelle für Gridbenutzer. Jeder Gridbenutzer soll in die Lage versetzt werden, sich über die aktuelle Sicherheitslage der Ressourcen, die er zur Bewältigung seiner Aufgaben nutzen möchte, zu informieren. Eine wesentliche Anforderung hierbei besteht in der freien Zugänglichkeit für alle Gridbenutzer. Aus diesem Grund wurde hierfür ein Webportal gewählt, das über jeden herkömmlichen Browser aufgerufen werden kann.

Ein Gridbenutzer, der Ressourcen des D-Grids verwenden kann, ist im Besitz eines Gridzertifikats. Ohne ein solches Zertifikat ist die Nutzung von Ressourcen innerhalb des D-Grids nicht vorgesehen. Diese Tatsache macht sich das GIDS-Projekt zunutze. Das GIDS-Portal ist daher nur mit einem gültigen Gridzertifikat zugänglich. Ein solches Zertifikat kann bei den folgenden Einrichtungen beantragt werden:

- DFN-Verein
  `https://www.pki.dfn.de/grid`
- GridKA-CA des KIT
  `http://www.gridka.de/cgi-bin/frame.pl?seite=/ca/d_inhalt.html`

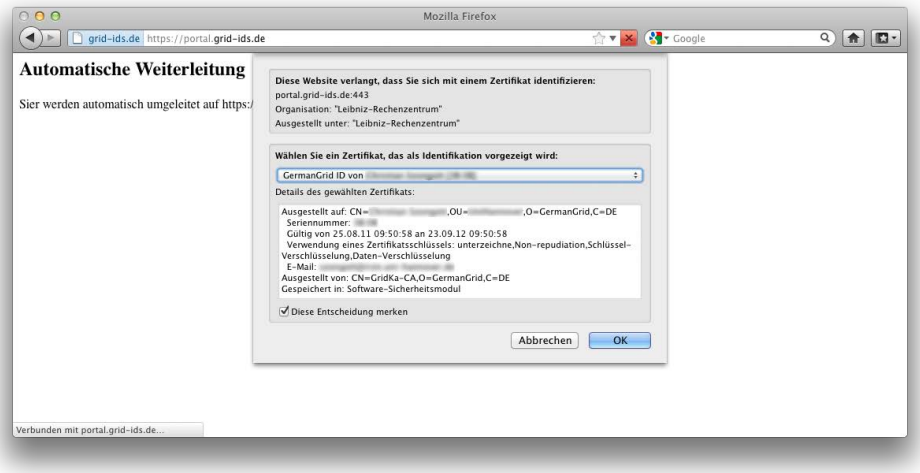

Abbildung 4.3: Authentifizierung mit Hilfe des Gridzertifikats beim Aufruf des GIDS-Portals

Beim Aufruf der URL des GIDS-Portals mit einem Webbrowser wird man dazu aufgefordert, ein entsprechendes Zertifikat vorzuweisen (siehe Abbildung 4.3). Einmal angemeldet erhält der Benutzer die Möglichkeit, sich eine tabellarische Übersicht über die aktuell aufgetretenen Sicherheitsvorfälle und Statistiken über eben jene Alarme anzeigen zu lassen. Zusätzlich können individuelle Einstellungen im Management-Bereich vorgenommen werden.

Alarme Unter dem Menüpunkt "Current Alerts"gelangen Benutzer zur Anzeige von korrelierten Alarmmeldungen – also Alarmmeldungen, die durch die Korrelation einer Vielzahl einzelner Alarme von Prelude erzeugt wurden (siehe Abbildung 4.4).

Die folgenden Felder werden in dieser tabellarischen Darstellung angezeigt:

- **Alert ID:** Eine eindeutige ID der Alarmmeldung des Korrelationsalarms (Jede einzelne Alarmmeldung, die in einem Korrelationsalarm enthalten ist, besitzt ihrerseits wieder eine eigene eindeutige ID).
- **Analyzer Time:** Ein Zeitstempel, der angibt zu welchem Zeitpunkt der Analyzer den korrelierten Alarm erkannt hat.

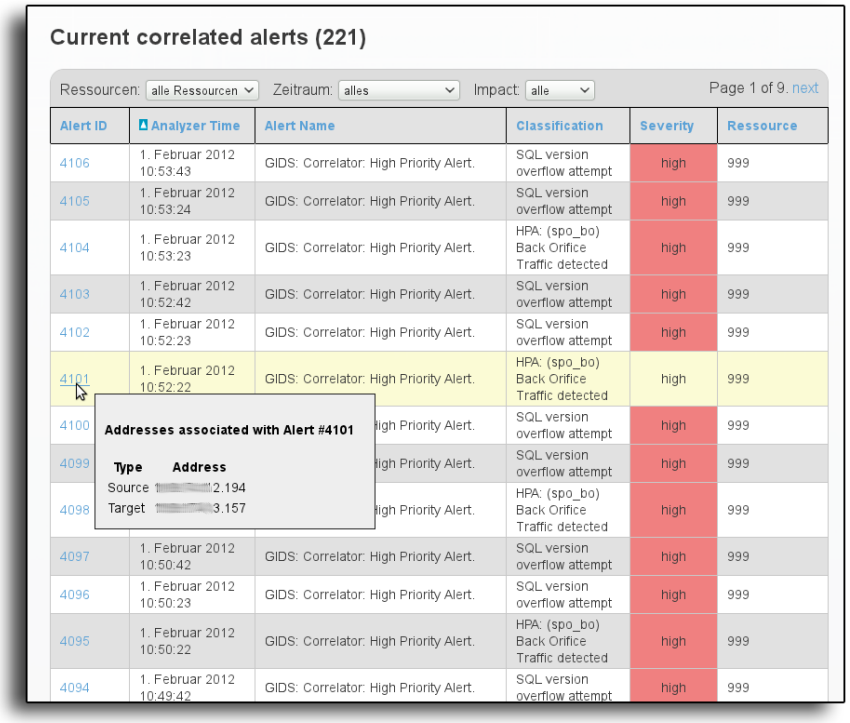

Abbildung 4.4: Tabellarische Darstellung aktueller Korrelationsalarme

- **Alert Name:** Eine textuelle Beschreibung der Alarmmeldung.
- **Classification:** Alarmmeldungen werden in Kategorien eingeteilt. Der Name der entsprechenden Kategorie wird hier dargestellt.
- **Severity:** Die Severity gibt die Schwere eines korrelierten Alarms an.
- **Ressource:** An dieser Stelle wird angezeigt, auf welche Ressource innerhalb des D-Grids sich die Alarmmeldung bezieht.

An dieser Stelle ist es dem Benutzer möglich, sich nur ausgewählte Alarmmeldungen bestimmter Ressourcen anzeigen zu lassen. Hierfür kann in der entsprechenden Liste eine der Ressourcen ausgewählt werden. Eine zeitliche Filterung der Alarmmeldungen ist ebenfalls vorgesehen. Es kann ein Wert zwischen "allen verfügbaren Meldungen" bis hin zu Meldungen "der letzten Stunde" gewählt werden. Weiterhin ist eine Filterung nach der Schwere der Alarme möglich. Um die Übersichtlichkeit der Seite zu wahren, werden pro Seite nur 25 Alarmmeldungen angezeigt. Für das Blättern zwischen den einzelnen Seiten stehen Schaltflächen oberhalb, sowie unterhalb der Tabelle bereit.

Bewegt der Benutzer den Mauszeiger über die ID eines Alarms, so werden involvierte IP-Adressen für den schnellen Überblick eingeblendet. Durch die Auswahl einer Alarmmeldung bzw. den Klick auf eine der angezeigten Alert IDs gelangt der Benutzer zu einer detailliertere Anzeige der ausgewählten Alarmmeldung (siehe Abbildung 4.5). An dieser Stelle werden neben weiteren allgemeinen Informationen zur Alarmmeldungen auch Tabellen mit den entsprechenden Quellen und Zielen des zur Alarmmeldung gehörenden Angriffs aufgelistet.

Innerhalb dieser Tabellen ist es seinerseits wieder möglich, sich detailliertere Information zu jeder einzelnen Quelle bzw. jedem einzelnen Ziel des Angriffs anzeigen zu lassen (siehe Abbildung 4.6

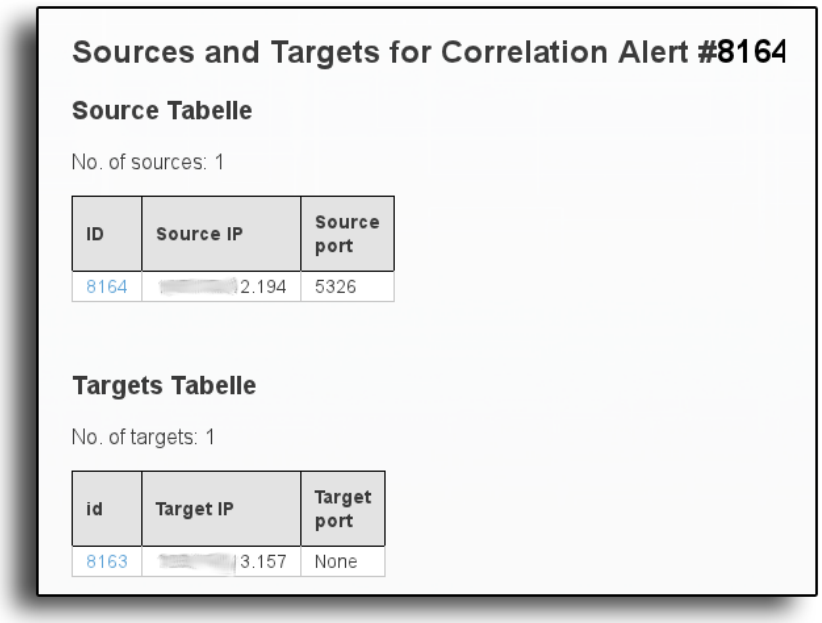

Abbildung 4.5: Detailinformationen zu einer Alarmmeldung

Hinter dem Punkt "Monitored resources"verbirgt sich die Möglichkeit, sich eine Liste aller Ressourcen anzeigen zu lassen, auf denen man durch sein Gridzertifikat Zugriffsrechte besitzt. Hier wird neben dem eindeutigen Ressourcenschlüssel (siehe GRRS-Datenbank) das Ressourcen-Kürzel und der vollständige Name der Ressource angezeigt.

Statistiken Die Statistik-Seite soll einen Überblick über den Gesamtzustand des D-Grid bieten. Hierfür werden diverse statistische Betrachtungen über die gespeicherten Alarme angewendet und entsprechend grafisch aufbereitet. Dies umfasst die Häufigkeit der am meisten auftretenen Klassifizierungen, Ports und eine Verteilung der Schwere (severity) eines Alarms. Zusätzlich können Benutzer individuelle Statistiken für Ressourcen, auf die sie Zugriff haben, durch entsprechende Auswahl einsehen. Jene Statistiken werden analog zur Gesamtstatistik erzeugt.

Management Der Management-Bereich (siehe Abbildung 4.7) bietet Ressourcenanbietern und Benutzern die Möglichkeit, diverse Konfigurationen vorzunehmen. So können Ressourcenanbieter einstellen, ob Mitglieder bestimmter VOs der Ressource zugeordnete Alarme nicht einsehen können sollen. Alle Benutzer können außerdem Abonnements anlegen, zu denen automatische Warnmeldung gesendet werden sollen. Eine weitere, benutzerseitige Filterung von Meldungen nach Ressource kann hier ebenfalls vorgenommen werden.

4.3.1.1 Implementierung

In diesem Abschnitt wird auf die technische Umsetzung der oben beschriebenen Lösung eingegangen. Eingesetzte Technologien, Software und Frameworks, die zur Implementierung des GIDS-Portals beitragen, werden beschrieben.

Grundlegende Komponenten Das GIDS-Portal ist in einer eigenen virtuellen Maschine betrieben. Eventuelle Sicherheitslücken in anderen von GIDS verwendeten Komponenten haben so keinen direkten Einfluß auf die Sicherheit und die Funktion des GIDS-Portals. Selbiges

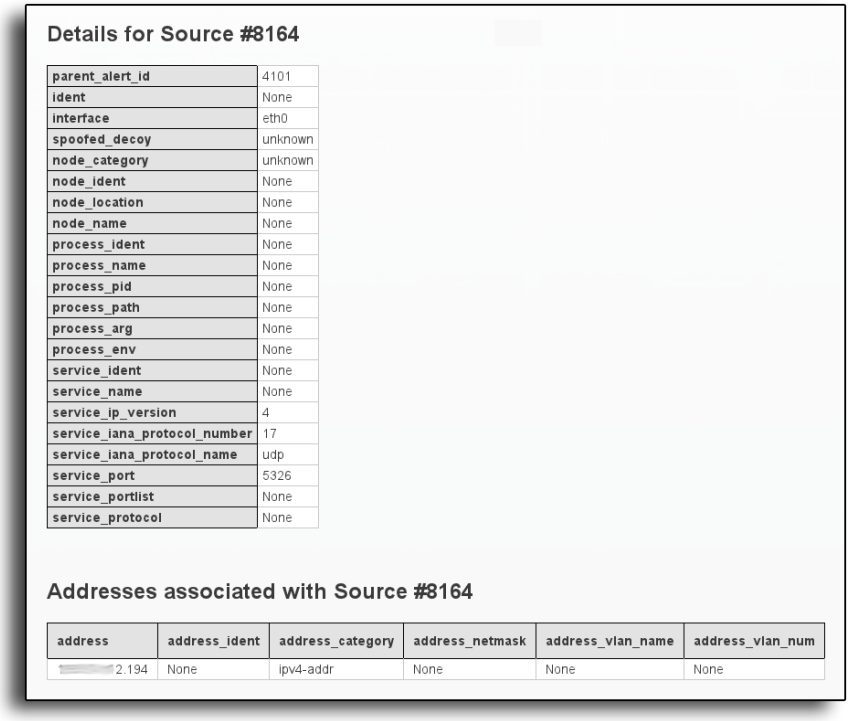

Abbildung 4.6: Detailinformationen zu einer Quelle bzw. einem Ziel eines Angriffs

gilt entsprechend ebenfalls umgekehrt. Als Betriebssystem wird ein Standard Debian Linux System (Squeeze) eingesetzt.

Als Basis des Portals dient der Webserver Apache2.[1] Da das GIDS-Portal nach Projektende als Dienst vom DFN-CERT angeboten werden soll, ist die nahtlose Integration in die bestehende Infrastruktur des DFN-CERTs unumgänglich. Aus diesem Grund wurde auf das Webframework Django[2] zurückgegriffen, welches bereits als Basis für das DFN-CERT Portal eingesetzt wird. Django seinerseits ist ein high-level Python Web Framework, welches die schnelle und komfortable Entwicklung von Webseiten erlaubt und ein Objekt-relationales Mapping zwischen Datenbanktabellen und Python-Objekten ermöglicht. Als weitere Grundlage ergibt sich somit Python als Programmiersprache. Wie bereits o.g. werden innerhalb der GIDS-Infrastruktur eine Vielzahl von Daten in MySQL-Datenbanken gespeichert und verarbeitet. Unter anderem betrifft dies auch die zur Anzeige benötigten Alarmmeldungen. Aus diesem Grund wird ebenfalls auf MySQL-Bibliotheken zurückgegriffen, die von Python in Verbindung mit dem Django-Framework genutzt werden, um aktuelle Datensätze aus entsprechenden Datenbanktabellen abzufragen und aufbereitet im GIDS-Portal anzuzeigen. Zur Darstellung von statistischen Daten wird die JavaScript-Bibliothek flot[3] verwendet, welche eine einfache, grafische Aufbereitung von statischen Daten ermöglicht.

Da bereits einige der Komponenten im DFN-CERT Portal realisiert wurden, sind diese Komponenten in das GIDS-Portal übernommen worden. Das betrifft auf der einen Seite die Authentifizierung und Autorisierung von Benutzern, als auch die Zugriffskontrolle der Benutzergruppen auf die Daten. Technisch sind die bestehenden Komponenten aus dem DFN-CERT Portal extrahiert worden und wurden in das GIDS-Portal in Form einer Bibliothek integriert.

Eine weitere essentielle Anforderung an das GIDS-Portal ist die Sicherung des Zugangs zu den vom GIDS gesammelten und verarbeiteten Daten. Um die verschlüsselte Kommuni-

[1] The Apache Software Foundation – `http://www.apache.org/`
[2] Django Webframework –`https://www.djangoproject.com/`
[3] flot –`http://code.google.com/p/flot/`

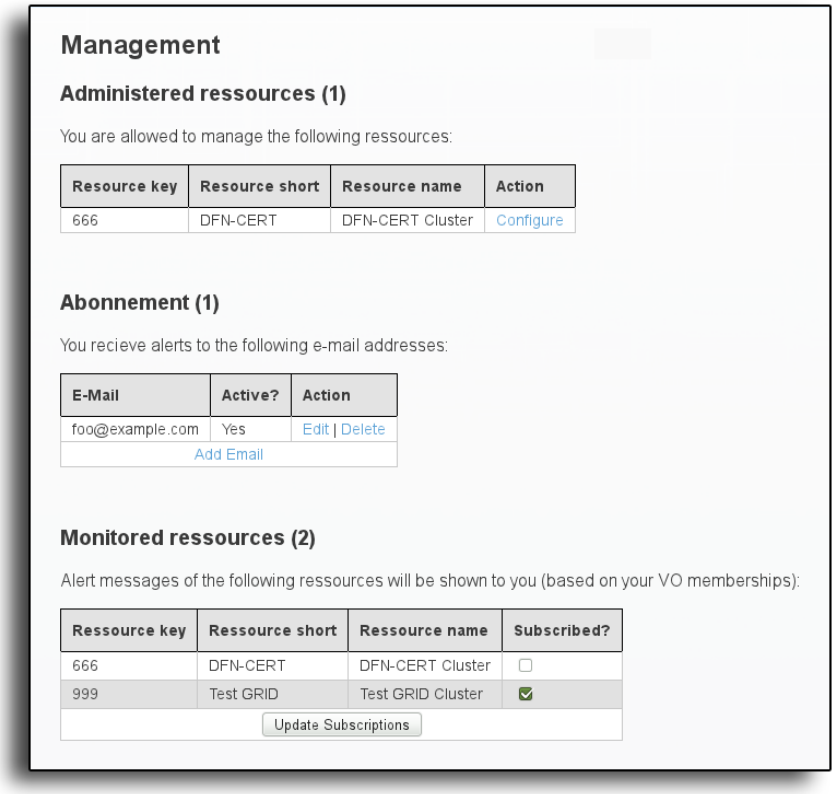

Abbildung 4.7: Management Bereich des GIDS-Portals

kation zwischen den Gridbenutzern und dem GIDS-Portal sicherzustellen, wird auf TLS als Verschlüsselungsprotokoll zurückgegriffen. Dem Webserver Apache2 wird über entsprechende Bibliotheken ermöglicht, verschlüsselte Verbindungen zwischen den Browsern der Benutzer und dem GIDS-Portal zur Verfügung zu stellen.

Für den Betrieb notwendige Pakete und Abhängigkeiten werden in Abschnitt 4.3.1.2 näher erläutert.

Erzwungener verschlüsselter Zugriff auf das GIDS-Portal Aus zweierlei Gründen ist der unverschlüsselte Zugriff auf die von GIDS gesammelten und verarbeiteten Daten zu verhindern.

1. Informationen, die im GIDS-Portal veröffentlicht werden, sind nur für Benutzer des D-Grid bestimmt. Es werden unter Umständen Sicherheitslücken von Ressourcenanbietern aufgezeigt, die auf keinen Fall weiter veröffentlicht werden dürfen. Aus diesem Grund muss der Zugang zum GIDS-Portal unter allen Umständen gesichert erfolgen. Auch Man-in-the-middle-Attacken können auf diese Weise vermieden werden, die die Veröffentlichung sicherheitskritischer Probleme von Ressourcenanbietern zur Folge haben könnten.

2. Für jede im GIDS-Portal angezeigte Alarmmeldung ist festgelegt, welche D-Grid Benutzer zur Ansicht berechtigt sind. Somit muss die Identität des Benutzers sichergestellt werden, der auf die Seiten des GIDS-Portals zugreift. Durch die Verwendung von X.509-Zertifikaten (Gridzertifikaten) wird nicht nur die Authentifizierung des Benutzers, sondern auch die Autorisierung innerhalb des GIDS-Portals vorgenommen. Der genaue Ablauf wird im folgenden Abschnitt beschrieben.

Die Notwendigkeit zur ausschließlich verschlüsselten Verbindung zum GIDS-Portal hat die folgende zwingende Anforderung an den Webserver zur Folge: Verbindungen über HTTP sind nicht zulässig. Lediglich SSL-gesicherte Verbindungen (HTTPS) sind zu verarbeiten. Um es dem Gridbenutzer ohne Kenntnis dieser Einschränkung zu ermöglichen, das Portal direkt aufzurufen erfolgt beim Aufbau einer ungesicherten Verbindung eine automatische Weiterleitung auf die SSL-gesicherte GIDS-Portal-Seite.

Authentifizierung Wie bereits o.g. darf es nur authentifizierten Benutzer ermöglicht werden, Inhalte des GIDS-Portals abzurufen. In seiner ersten prototypischen Implementierung wird die Authentifizierung durch den Apache Webserver zusammen mit OpenSSL durchgeführt. Hierbei wird zunächst die Validität des Zertifikats verifiziert. Anschließend findet eine Prüfung des Distinguished Names (DN) statt, welcher zwingend in einem X.509-Zertifikat enthalten ist. Befindet sich der DN in einer zu Testzwecken erstellten Whitelist, so wird der Zugang zum Portal gewährt. Andernfalls wird der Verbindungsaufbau abgebrochen und der Zugriff auf das GIDS-Portal bleibt verwehrt.

Durch die Integration in die DFN-CERT Infrastruktur werden auch die hier bereits vorhandenen erweiterten Mechanismen zur Zertifikatsprüfung genutzt. Hierbei wird unter anderem geprüft, ob das Zertifikat zurückgerufen (revoked) wurde. Es findet also beim Verbindungsaufbau die Prüfung von Certificate Revocation Liste (CRLs) sowie eine entsprechende Abfrage per Online Certificate Status Protocol (OCSP) statt. Auf diese Weise können auch bereits widerrufene Zertifikate erkannt und abgewiesen werden.

Autorisierung Nach erfolgter Authentifizierung wird die GIDS-Portalseite angezeigt. Der oben bereits erwähnte, im Zertifikat enthaltene DN spielt auch bei der Autorisierung eine entscheidende Rolle. Zur Sicherstellung, dass Gridbenutzer nur Alarmmeldungen von Ressourcen einsehen können, auf denen sie auch die Berechtigung zum Berechnen von Gridjobs besitzen, wird der Zugang zu einer Datenbank des Grid Resource Registration Service (GRRS) benötigt. In zwei Schritten werden diese Ressourcen ermittelt:

1. Im ersten Schritt werden aus der Tabelle `Dgrid_DnVO` passend zum angemeldeten DN des Zertifikats alle VOs ermittelt, in denen der aktuell angemeldete Benutzer Mitglied ist.

2. Im zweiten Schritt werden aus der Tabelle `Dgrid_RessVO` der GRRS-Datenbank alle diejenigen Ressourcen ermittelt, die einer der VOs des Benutzers erlauben, ihre Ressourcen zu nutzen.

Als Ergebnis steht eine Liste aller Ressourcen bereit, die einem Benutzer durch die Mitgliedschaft in VOs zur Verfügung stehen. Anhand dieser Ressourcen-Liste werden in allen Ansichten des GIDS-Portals Alarmmeldungen gefiltert. Die Zuordnung einer Alarmmeldung zu einer Gridressource erfolgt durch ein Ressourcen-Feld innerhalb der GIDS-eigenen Datenbank der Alarmmeldungen (siehe 4.2.1).

Abbild der GRRS-Datenbank zur Autorisierung Die GRRS-Datenbank bildet eine grundlegende Informationsquelle über Benutzer, Dos und Ressourcen innerhalb des D-Grids. In seiner grundlegenden Form sind in den Tabellen der Datenbank keine Primär- und Fremdschlüssel vorhanden. Diese werden allerdings zwingend für die Nutzung des Django Frameworks benötigt. Hier werden stets Primärschlüssel in jeder Tabelle benötigt, um eindeutige Python-Objekte erzeugen zu können. Aus diesem Grund befindet sich auf dem Portal-System ein stets aktuelles Abbild der GRRS-Datenbank, welches um die entsprechenden Primärschlüssel erweitert wird. Diese Datenbank wird alle 24 Stunden durch einen automatischen Cronjob mit den aktuellen Datensätzen der originalen GRRS-Datenbank gefüllt. Für die Benutzung des GIDS-Portals entstehen dadurch keinerlei Probleme. Sollte ein neuer D-Grid Benutzer auf das GIDS-Portal zugreifen, kann es zu einer maximal 24-stündigen Verzögerung kommen. Ist ein Benutzer noch nicht im lokalen GRRS-Abbild vorhanden, so wird ihm zwar der Zugang zum GIDS-Portal gewährt. Durch die fehlende Zuordnung zu VOs und entsprechend zu den Ressourcen werden allerdings keine Alarmmeldungen angezeigt.

Filterung Innerhalb des GIDS-Portals kann in der tabellarischen Darstellung, wie in Abschnitt 4.3.1 beschrieben, eine zeitliche, ein Filterung nach Schwere und auch eine Filterung nach Ressourcen vorgenommen werden. Diese Filterung ist in Django realisiert. Die Abfrage der Daten beim Zugriff auf die GIDS-eigene Datenbank bleibt hierbei unverändert. Die Filterung erfolgt entsprechend der Einstellungen auf der Webseite dynamisch vor der Übergabe an das Webseiten-Template. Darüber hinaus findet vorab eine Filterung nach den getätigten Einstellungen im Management-Bereich statt. Hierbei werden beispielsweise für Mitglieder einer VO, die ein Ressourcenanbieter von seinen Alarmmeldungen ausgeschlossen hat, entsprechende Alarme vorgefiltert.

4.3.1.2 Installation

Die für den Betrieb notwendigen Softwarepakete werden in folgender Tabelle 4.1 zusammen mit ihren direkten Abhängigkeiten aufgelistet. Es können eventuelle weitere Abhängigkeiten bestehen, die unter normalen Umständen durch die folgende Paket-basierte Installation automatisch aufgelöst werden.

```
portal-server:# apt-get install <Paketname>
```

Tabelle 4.1: Benötigte Pakete und ihre direkten Abhängigkeiten

Paketname	**direkte Abhängigkeiten**
apache2	apache2.2-common
libapache2-mod-python	libpython2.6 libssl0.9.8 python python-central zlib1g
python	libbz2-1.0 libc6 libdb4.8 libexpat1 libncursesw5 libreadline6 libsqlite3-0 python2.6-minimal
openssl	libssl0.9.8
mysql-server-5.1	libdbi-perl libgcc1 libmysqlclient16 libstdc++6 lsb-base mysql-client-5.1 mysql-server-core-5.1 passwd perl psmisc
python-django	python python-support

SSL Zertifikat & Schlüssel für Apache2 Webserver Da eine SSL-gesicherte Verbindung zum GIDS-Portal aufgebaut werden muss, ist ein Hostzertifikat für den entsprechenden

Rechner erforderlich. Sowohl das Zertifikat, als auch der dazugehörige private Schlüssel müssen auf dem Server vorhanden sein. Wir gehen im Folgenden davon aus, dass die beiden Dateien `portal-zertifikat.pem` und `portal-key.pem` im Verzeichnis `/etc/apache2/sslcert/` vorhanden sind.

Portal-Quellcode extrahieren Um dem Webserver den Portal-Code bereit zu stellen, muss das Archiv `gidsportal.tar.gz` mit folgendem Befehl in das Web-Root-Verzeichnis extrahiert werden. Wir gehen im Folgenden davon aus, dass das Portal im Verzeichnis `/var/www/` liegen soll und sich das Archiv im selben Verzeichnis befindet.

```
         portal-server:# cd /var/www
portal-server:/var/www/# tar -xvf gidsportal.tar.gz
```

Anlegen des GIDS-eigenen GRRS-DB Abbilds Mit dem folgenden SQL-Skript wird eine leere Datenbank erzeugt, die mit Hilfe eines per Cronjob ausgeführten Shellskripts gefüllt wird. Das Skript zum Befüllen der Datenbank mit aktuellen Daten folgt in Abschnitt 4.3.1.3.

```
DROP TABLE IF EXISTS ‘dgrid_dn_vo‘;
CREATE TABLE ‘dgrid_dn_vo‘ (
  ‘id‘ MEDIUMINT NOT NULL AUTO_INCREMENT,
  ‘member_id‘ int(38) default NULL,
  ‘vorname‘ varchar(50) default NULL,
  ‘nachname‘ varchar(100) default NULL,
  ‘member_dn‘ varchar(200) default NULL,
  ‘member_status‘ varchar(15) default NULL,
  ‘vo_long‘ varchar(15) default NULL,
  ‘vo_short‘ varchar(2) default NULL,
  PRIMARY KEY(id)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;


DROP TABLE IF EXISTS ‘dgrid_ress_vo‘;
CREATE TABLE ‘dgrid_ress_vo‘ (
  ‘id‘ MEDIUMINT NOT NULL AUTO_INCREMENT,
  ‘drv_ress_key‘ int(38) default NULL,
  ‘drv_vo‘ varchar(2) default NULL,
  PRIMARY KEY(id)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

DROP TABLE IF EXISTS ‘dgrid_ressourcen‘;
CREATE TABLE ‘dgrid_ressourcen‘ (
  ‘id‘ MEDIUMINT NOT NULL AUTO_INCREMENT,
  ‘ress_key‘ int(38) default NULL,
  ‘dr_kind‘ varchar(20) default NULL,
  ‘dr_scunicore‘ varchar(150) default NULL,
  ‘dr_scunicore6‘ varchar(150) default NULL,
  ‘dr_scglobus‘ varchar(150) default NULL,
  ‘dr_scglite‘ varchar(150) default NULL,
  ‘dr_scogsa‘ varchar(150) default NULL,
  ‘dr_scdcache‘ varchar(150) default NULL,
  ‘dr_scinteract‘ varchar(150) default NULL,
  ‘dr_scadmin‘ varchar(150) default NULL,
  ‘dr_scglobus2‘ varchar(150) default NULL,
  ‘dr_scglobus5‘ varchar(150) default NULL,
```

```
  `dr_shorty` varchar(40) default NULL,
  `dr_gram_node` varchar(100) default NULL,
  `dr_glite_node` varchar(100) default NULL,
  `dr_interact_node` varchar(100) default NULL,
  `dr_gram_node2` varchar(100) default NULL,
  `dr_gram_node5`varchar(100) default NULL,
  `dr_soinvest` varchar(1) default NULL,
  `dr_unicore_vers` varchar(1) default NULL,
  `dr_admin_mail` varchar(60) default NULL,
  `dr_prov_short` varchar(10) default NULL,
  `dr_prov_long` varchar(100) default NULL,
  `dr_desc` varchar(60) default NULL,
  `dr_admin_dn` varchar(200) default NULL,
  PRIMARY KEY(id)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;
```

4.3.1.3 Konfiguration

In diesem Abschnitt wird die Konfiguration beschrieben, um das GIDS-Portal einzurichten und an alle benötigten Schnittstellen anzubinden.

Apache2-Konfiguration zur automatischen Weiterleitung auf die SSL-gesicherte Portal-Seite Die automatische Weiterleitung auf die SSL-gesicherte Portal-Seite erfolgt in der Konfigurationsdatei`default` des Apache2 Webservers unter `/etc/apache2/sites-available/default`. Hier wird der folgende Eintrag ergänzt. Durch diese Weiterleitungsregel werden alle HTTP-Verbindungen automatisch auf die SSL-gesicherte Seite umgeleitet.

```
  <Location />
     RewriteEngine On
     RewriteCond %{HTTPS} off
     RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}
  </Location>
```

Apache2-Konfiguration der Portal-Seite Die komplette Konfigurationsdatei befindet sich im Anhang B. Im folgenden wird nur auf die für den Betrieb wichtigen Parameter eingegangen. Die Einstellungen erfolgen in der Datei `/etc/apache2/sites-available/gidsportal`.

```
<VirtualHost portal.grid-ids.de:443>
  # SSL aktivieren
  SSLEngine On

  # SSL-Zertifikat des Webservers
  SSLCertificateFile  /etc/apache2/sslcert/portal-zertifikat.pem

  # Schluessel passend zum SSL-Zertifikat
  SSLCertificateKeyFile /etc/apache2/sslcert/portal-key.pem

  # Verzeichnis in dem Zertifikats-Liste von CAs hinterlegt ist
  SSLCACertificatePath  /etc/apache2/sslcert/ca-certs/

  # Client-Authentifizierung aktivieren und zur Pflicht machen
  SSLVerifyClient require
```

```
  SSLVerifyDepth 10
  SSLUserName SSL_CLIENT_S_DN

  # URL des Servers
  ServerName      <URL unter der der Server erreichbar ist>

  # Verzeichnis in dem der Portal Quellcode liegt
  DocumentRoot    /var/www/DjangoGIDS/htdocs
  [..]
</VirtualHost>
```

Abschließend muss die neu erstelle Site noch aktiviert werden und die Konfiguration durch den Apache2 Webserver neu geladen werden. Dies erfolgt mit Hilfe der folgenden zwei Befehle:

```
  portal-server:# a2ensite gidsportal
  portal-server:# /etc/init.d/apache2 reload
```

Konfiguration der Django-Komponenten Damit Django auf die benötigten Datenbanken zugreifen kann, muss auch hier die Konfiguration an die Gegebenheiten angepasst werden. Im folgenden sind die zwei entsprechenden Anbindungen der Datenbanken gezeigt. Die komplette Django-Konfigurationsdatei des GIDS-Portals befindet sich ebenfalls im Anhang B.

```
DATABASES = {
  # Datenbank der Alarmmeldungen
    'gidsidmef': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'idmef',
        'USER': '<Benutzername>',
        'PASSWORD': '<Kennwort>',
        'HOST': '<DB-Host>',
        'PORT': '<DB-Port, Standard ist 3306>',
        },

    # Gespiegelte GRRS-Datenbank
    'grrs': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'grrs_mirror',
        'USER': '<Benutzername>',
        'PASSWORD': '<Kennwort>',
        'HOST': '<DB Host>',
        'PORT': '<DB-Port, Standard ist 3306>',
    },
}
```

Einrichtung des Cronjobs zur automatischen Erstellung des GRRS-Abbilds Um die Daten in der GIDS-eigenen GRRS-DB Kopie aktuell zu halten ist ein Cronjob einzurichten, welcher alle 24 Stunden das folgende Skript ausführt:

```
#!/bin/sh
#
# Dumps all necessary tables from GRRS-DB
```

```
# Imports data into local DB-mirror
#

# Settings for remote DB
REMOTE_DB_HOST="grrs-db.fz-juelich.de"
REMOTE_DB_USER="<Benutzername>"
REMOTE_DB_PASSWORD="<Kennwort>"
REMOTE_DB_DBNAME="dgrid"
REMOTE_DB_TABLES="dgrid_dn_vo dgrid_ress_vo dgrid_ressourcen"

# Settings for local DB
LOCAL_DB_NAME="grrs_mirror"
LOCAL_DB_USER="<Benutzername>"
LOCAL_DB_PASSWORD="<Kennwort>"

# Dump and import
mysqldump -h $REMOTE_DB_HOST -u $REMOTE_DB_USER\
  -p$REMOTE_DB_PASSWORD -C $REMOTE_DB_DBNAME\
  --lock-tables=false --complete-insert\
  --no-create-db --no-create-info\
  --tables $REMOTE_DB_TABLES | mysql -u $LOCAL_DB_USER\
  -p$LOCAL_DB_PASSWORD $LOCAL_DB_NAME
```

Kapitel 5

Datenschutz

Im Projekt GIDS werden sicherheitsrelevante Informationen innerhalb einer geschlossenen Nutzergruppe im D-Grid zum Zwecke einer verbesserten Angriffserkennung ausgetauscht. Zum Schutz dieser Daten wurde im Rahmen des GIDS-Projektes ein Datenschutzkonzept erarbeitet und auf das IDMEF Datenaustauschformat angewendet. Dabei spielen zwei Klassen von Daten eine wichtige Rolle:

Personenbezogene Daten sind durch das Bundesdatenschutzgesetz bezüglich deren Erhebung, Verwendung, Transport und Speicherung geschützt. Dabei wird zwischen den Daten unterschieden, die direkt auf eine Person hinweisen (bestimmt) oder mittels zusätzlichen Wissen auf eine Person schliessen lassen (bestimmbar).

Sicherheitskritische Daten geben Informationen preis, die von den GIDS-Partnern oder dem Betreiber als sicherheitskritisch eingestuft werden. Zum Beispiel sind das Informationen über die IDS-Sensorik, die von einem Angreifer zur Verschleierung von Angriffen missbraucht werden können.

Dieses Datenschutzkonzept betrifft im Wesentlichen die Weitergabe und Verarbeitung der IDS-Sensordaten. Jedoch betrifft es auch den Zugriff und die Speicherung der Daten im GIDS-Portal.

5.1 Umsetzung der Datenschutzrichtlinie

Zur Einhaltung des Datenschutzgesetzes gibt es zwei Alternativen, die entweder die Vermeidung personenbezogener Daten oder die Schaffung einer gesetzlichen Grundlage als Ziel haben. Die Vermeidung kann technisch durch die Anonymisierung der Daten realisiert werden. Dafür ist ausreichend, dass die Daten keinen Bezug zu genau einer Person zulassen. Als Nachteil der Anonymisierung ist die Korrelation der Daten nicht mehr möglich. Zwar kann dieser Nachteil durch Pseudonymisierung vermieden werden, jedoch ist streng genommen, der Bezug zu der entsprechenden Person immer noch möglich. Vorteil ist die einfache technische Realisierung. Die Alternative ist, eine gesetzliche Norm zur berechtigten Nutzung personenbezogener Daten zu schaffen. Dies kann beispielsweise durch eintsprechende Verträge zwischen den GIDS-Partnern und dem Betreiber realisiert werden, in denen dem Betreiber das Recht eingeräumt wird, die Daten zu verarbeiten (Auftragsdatenverarbeitung). Weiterhin kann argumentiert werden, dass die Verarbeitung dieser Daten zur Beseitigung von Störungen durch Angriffe notwendig ist.

Während der Projektlaufzeit fiel die Entscheidung für die Anonymisierung personenbezogener Daten. Der entscheidende Vorteil dieser Lösung ist, dass keine vertragliche Bindung notwendig ist. Die Erfahrung mit ähnlichen Diensten hat gezeigt, dass dies die Schwelle des Beitritts und die Akzeptanz deutlich erhöht. Deshalb wird diese Variante auch für den Start

des GIDS-Dienstes gewählt. Da die technischen Möglichkeiten die Verarbeitung der nicht-anonymisierten Daten vorhanden ist, kann auf diese Alternative später umgeschwenkt werden; zum Beispiel, wenn die Akzeptanz des GIDS-Dienstes sichergestellt ist.

5.1.1 Löschung innerhalb einer Seite auf Attributebene

In einem ersten Schritt sollen in der Testphase alle Attribute, die mit (d) markiert sind, komplett gelöscht werden. Somit ist maximaler Datenschutz garantiert und es kann ausgetestet werden, in wie weit sich noch sinnvolle Korrelationen durchführen lassen. Dieser Test soll bis Ende 2011 durchgeführt werden. Je nach Ergebnis dieses Testbetriebs muss dann neu entschieden werden, ob diese strikte Datenschutzauslegung beibehalten wird und die damit zu erreichende Erkennungsleistung akzeptabel ist oder ob man mit Hilfe von Pseudonymisierung (beispielsweise CryptoPAn) oder andersweitiger Anonymisierung zum gewünschten Ziel gelangt.

Die mit (ls) markierten Attribute werden in der Defaulteinstellung nicht verändert. Welche von diesen Attributen gelöscht werden, müssen die Ressourcenprovider selbst treffen. Im Programm, dass innerhalb des GIDS-Projektes entwickelt wurde, um die Alarme an alle anderen Sites schicken zu können, werden den Administratoren Konfigurationsdateien zur Verfügung gestellt, mit denen sie eine Löschung der Attribute einfach durchführen können.

Wichtig ist dabei anzumerken, dass die Sensorenbetreiber selber dafür verantwortlich sind, ob lokale Sicherheitsrichtlinien oder der Datenschutz verletzt sind. Somit können noch weitere, hier nicht vermerkte Felder kritisch in Bezug einer Weitergabe sein, auf der anderen Seite können auch hier markierte Felder im Kontext eines Ressorcenproviders völlig harmlos sein und problemlos weitergegeben werden. In der angesprochenen Konfigurationsdatei ist eine feingranulare Löschung möglich.

5.1.2 Datengrundlage/Filterung ganzer Alarmmeldungen

Gemäß dem Grundsatz, dass nicht gesammelte Daten diejenigen Daten sind, die nach Datenschutzaspekten am Besten geschützt sind, sollen im GIDS-Projekt nur Daten ausgetauscht werden, die im direkten Zusammenhang mit Grid-Ressourcen stehen. Es liegt somit im Verantwortungsbereich der Ressourcenprovider, alle nicht Gridressourcen betreffende Alarme auszufiltern. Weiterhin besteht die Möglichkeit, ganze Alarmmeldungen auszufiltern, falls diese der Zielsetzung von GIDS entgegensprechen oder es lokale Sicherheitsbedenken gegen eine Weitergabe bestehen.

5.1.3 Löschung externer Alarmmeldungen

Wird eine Alarmmeldung an andere Sites verschickt, so wird diese beim Absenden mit einem Gültigkeitsdatum versehen. Dieses entspricht der Gültigkeitsdauer der Alarmmeldung bei der lokalen Site, das heißt, einer vom Erstellungszeitpunkt aus gerechnet den lokalen Datenschutzrichtlinien entsprechende Anzahl von Tagen. Wird nichts vorgegeben, so wird diese Anzahl von Tagen den aktuellen Gerichtsentscheiden entsprechend auf sieben Tage gesetzt. Es ist jedoch im Ermessensspielraum eines jeden Ressorcenproviders, diese Frist gegebenenfalls zu erhöhen oder zu verringern. Die Gültigkeitsdauer von Alarmen externer Sites ist jedoch unabhängig von einer solchen Änderung.

Bei jeder Site wird ein Cronjob installiert, der die Einhaltung dieser Löschung übernimmt. Mit Teilnahme am GIDS-Netzwerk erklärt sich jede Site damit einverstanden, dass sie dafür verantwortlich sind, dass dieser Cronjob seinen Dienst täglich wenigstens einmal durchführen kann.

5.2 Anwendung auf den IDMEF Standard

Welche sicherheitsrelevanten Informationen in einer IDMEF-Nachricht versendet werden beziehungsweise welche Teile innerhalb einer Nachricht weitergegeben werden, ist Inhalt dieses Abschnitts. Dazu wird der Standard IDMEF kurz vorgestellt und an wichtigen Stellen, an den

es aus datenschutzrechtlicher Hinsicht Bedenken geben könnte oder die Daten zu sensibel sind, um sie weiterzugeben.

Im Folgenden sind zur Übersicht alle im IDMEF-Standard verwendeten Klassen aufgeführt. Dabei bedeutet ✔in der Spalte *D*, dass dieses IDMEF-Feld den Datenschutz erfüllt und ✔in der Spalte *LS*, dass lokale Sicherheitsrichtlinen nicht tangiert werden. Analog dazu bedeutet ein ✘in der Spalte *D*, dass dieses IDMEF-Feld den Datenschutz potentiell verletzen könnte und ✘in der Spalte *LS*, dass lokale Sicherheitsrichtlinen verletzt werden könnten.

Klasse	**Attribut oder Unterklasse**	**D**	**LS**
Action (5.3.13)	category	✔	✔
	Inhalt	✘	✘
AdditionalData (5.3.11)	meaning	✔	✔
	type	✔	✔
	Inhalt	✘	✘
Address (5.3.17)	address	✘	✘
	category	✔	✔
	ident	✔	✔
	netmask	✔	✘
	vlan-name	✔	✘
	vlan-num	✘	✘
Alert (5.3.1)	AdditionalData	✔	✔
	Analyzer	✔	✔
	Assessment	✔	✔
	AnalyzerTime	✔	✔
	Classification	✔	✔
	CreateTime	✔	✔
	DetectTime	✔	✔
	messageid	✔	✔
	Source	✔	✔
	Target	✔	✔

Klasse	Attribut oder Unterklasse	D	LS
Analyzer (5.3.6)	Analyzer	✔	✔
	analyzerid	✔	✔
	class	✔	✔
	manufacturer	✔	✘
	model	✔	✘
	name	✔	✘
	Node	✔	✔
	ostype	✔	✘
	osversion	✔	✘
	Process	✔	✔
	version	✔	✘
Assessment (5.3.10)	Action	✔	✔
	Confidence	✔	✔
	Impact	✔	✔
Checksum (5.3.28)	algorithm	✔	✔
	key	✔	✘
	value	✔	✔
Classification (5.3.7)	ident	✔	✔
	Reference	✔	✔
	text	✔	✔
Confidence (5.3.14)	rating	✔	✔
	Inhalt	✔	✔
CorrelationAlert (5.3.3)	alertident	✔	✔
	name	✔	✔
File (5.3.24)	access-time	✔	✔
	category	✔	✔
	Checksum	✔	✔
	create-time	✔	✔
	data-size	✔	✔
	disk-size	✔	✘
	FileAccess	✔	✔
	file-type	✔	✘
	fstype	✔	✘
	ident	✔	✔
	Inode	✔	✔
	Linkage	✔	✔
	modify-time	✔	✔
	name	✘	✘
	path	✘	✘
FileAccess (5.3.25)	Permission	✔	✘
	UserId	✔	✔
Heartbeat (5.3.5)	AdditionalData	✔	✔
	Analyzer	✔	✔
	AnalyzerTime	✔	✔
	CreateTime	✔	✔
	HeartbeatIntervall	✔	✔
	messageid	✔	✔

Klasse	**Attribut oder Unterklasse**	**D**	**LS**
Impact (5.3.12)	completion	✔	✔
	severity	✔	✔
	type	✔	✔
	Inhalt	✘	✘
Inode (5.3.27)	change-time	✔	✔
	c-major-device	✔	✔
	c-minor-device	✔	✔
	major-device	✔	✔
	minor-device	✔	✔
	number	✔	✔
Linkage (5.3.26)	category	✔	✔
	File	✔	✔
	name	✘	✘
	path	✘	✘
Node (5.3.16)	Address	✔	✔
	category	✔	✔
	ident	✔	✔
	location	✔	✘
	name	✘	✘
OverflowAlert (5.3.4)	buffer	✘	✘
	program	✔	✔
	size	✔	✔
Process (5.3.20)	arg	✘	✘
	env	✘	✘
	ident	✔	✔
	name	✔	✘
	path	✘	✘
	pid	✔	✔
Reference (5.3.15)	meaning	✔	✔
	name	✔	✔
	orgin	✔	✔
	url	✔	✔
Service (5.3.21)	iana_protocol_name	✔	✘
	iana_protocol_number	✔	✘
	ident	✔	✔
	ip_version	✔	✘
	name	✔	✘
	port	✔	✘
	portlist	✔	✘
	protocol	✔	✘
SNMPService (5.3.23)	command	✘	✘
	contextEngineID	✔	✘
	contextName	✔	✘
	messageProcessingModel	✔	✘
	oid	✔	✘
	securityLevel	✔	✘
	securityModel	✔	✘
	securityName	✔	✘

Klasse	Attribut oder Unterklasse	D	LS
Source (5.3.8)	ident	✔	✔
	interface	✔	✘
	Node	✔	✔
	Process	✔	✔
	Service	✔	✔
	spoofed	✔	✔
	User	✔	✔
Target (5.3.9)	decoy	✔	✔
	File	✔	✔
	ident	✔	✔
	interface	✔	✘
	Node	✔	✔
	Process	✔	✔
	Service	✔	✔
	User	✔	✔
ToolAlert (5.3.2)	alertident	✔	✔
	command	✔	✔
	name	✔	✔
User (5.3.18)	category	✔	✔
	ident	✔	✔
	UserId	✔	✔
UserId (5.3.19)	ident	✔	✔
	name	✘	✘
	number	✘	✘
	tty	✔	✘
	type	✔	✔
WebService (5.3.22)	arg	✘	✘
	cgi	✔	✘
	http-method	✔	✔
	url	✔	✘

5.3 IDMEF

IDMEF (The Intrusion Detection Message Exchange Format) ist ein im RFC 4765 beschriebenes XML-Format aus dem Jahr 2007. Es besteht im Wesentlichen aus den beiden Klassen Alert und Heartbeat und deren Unterklassen.

5.3.1 Alert

Die Alert-Klasse enthält alle notwendigen Informationen, um einen Angriff oder Angriffsversuch vollständig beschreiben zu können. Die Mehrzahl aller Nachrichten, die im Projekt GIDS versendet werden, sind dieser Klasse zuzuordnen.

Unterklassen

Name	Erklärung	Anzahl
AdditionalData	siehe 5.3.11	0 bis ∞
Analyzer	siehe 5.3.6	1
Assessment	siehe 5.3.10	0 oder 1
AnalyzerTime	Die momentane Zeit des Analyzers.	0 oder 1
Classification	siehe 5.3.7	1
CreateTime	Ein Zeitstempel, der angibt, wann die Alarmmeldung erstellt wurde.	1
DetectTime	Ein Zeitstempel, der angibt, wann (zum ersten Mal) eine verdächtige Aktion bemerkt wurde. Das Erstellen der Meldung (CreateTime) kann auch später erfolgen.	0 oder 1
Source	siehe 5.3.8	0 bis ∞
Target	siehe 5.3.9	0 bis ∞

Spezialisierungen

Name	Erklärung	Anzahl
CorrelationAlert	siehe 5.3.3	0 oder 1
OverflowAlert	siehe 5.3.4	0 oder 1
ToolAlert	siehe 5.3.2	0 oder 1

Attribute

Name	Erklärung	Anzahl
messageid	Ein eindeutiger Identifier für die Alarmmeldung.	0 oder 1

5.3.2 ToolAlert

Wurde für den Angriff ein Tool verwendet und dieses Tool wurde durch die Sensorik erkannt, so können spezielle Informationen über dieses Programm in der Klasse ToolAlert übermittelt werden.

Unterklassen

Name	Erklärung	Anzahl
alertident	Eine Liste von Alarm-Identifiern (siehe Attribute der Alert-Klasse in 5.3.1). Da domänenübergreifend nicht garantiert werden kann, dass die Identifier eindeutig sind, kann die Angabe eines jeden Alarm-Identifiers durch die Angabe des Analyzer-Identifiers (siehe Attribute der Analyzer-Klasse in 5.3.6) ergänzt werden.	1 bis ∞
command	Das Kommando oder die Operation, die das Tool ausführen sollte, beispielsweise `BackOrifice ping`.	0 oder 1
name	Ein Bezeichner für den Alarm. Kann beispielsweise der Name des verwendeten Tools sein.	1

5.3.3 CorrelationAlert

Mit Hilfe der CorrelationAlert-Klasse können zusammengehörige Alarmmeldungen markiert werden.

Unterklassen

Name	Erklärung	Anzahl
alertident	Eine Liste von Alarm-Identifiern (siehe Attribute der Alert-Klasse in 5.3.1). Da domänenübergreifend nicht garantiert werden kann, dass die Identifier eindeutig sind, kann die Angabe eines jeden Alarm-Identifiers durch die Angabe des Analyzer-Identifiers (siehe Attribute der Analyzer-Klasse in 5.3.6) ergänzt werden.	1 bis ∞
name	Ein Bezeichner für den Korrelationsalarm, beispielsweise *massiver SSH-Scan*.	1

5.3.4 OverflowAlert

Speziell für Buffer Overflows existiert eine eigene Alarmklasse.

Unterklassen

Name	Erklärung	Anzahl
(ls)(d)buffer	Der Inhalt des Buffers, soweit er vom Sensor ermittelt werden konnte.	0 bis 1
program	Das Programm, das den Angriff auszuführen versucht. (Also nicht das angegriffene Programm)	1
size	Die Größe des Buffers	0 bis 1

Erläuterung

buffer. Der Buffers kann jeglichen Inhalt enthalten, da es hier stark vom Programm abhängt, was im Buffer steht. Es ist somit a priori der Inhalt dieses Feldes nicht abschätzbar.

5.3.5 Heartbeat

Jeder Sensor schickt in regelmäßigen Abständen Heartbeat-Meldungen an die zentralen Einheiten, um zu zeigen, dass er noch funktionsfähig ist. Zwar werden am Ende nicht alle Heartbeats aller beteiligten Sensoren an alle anderen Sites geschickt, innerhalb der GIDS-Infrastruktur müssen aber trotzdem Heartbeats verschickt werden, um zu zeigen, welche teilnehmenden Sites noch erreichbar sind.

Unterklassen

Name	Erklärung	Anzahl
AdditionalData	siehe 5.3.11	0 bis ∞
Analyzer	siehe 5.3.6	1
AnalyzerTime	Die momentane Zeit des Analyzers.	0 bis 1
CreateTime	Ein Zeitstempel, der angibt, wann die Heartbeatmeldung erstellt wurde.	1
Heartbeat-Intervall	Das Intervall, nach dem eine neue Heartbeatnachricht generiert wird.	0 bis 1

Attribute

Name	Erklärung	Anzahl
messageid	Ein eindeutiger Identifier für die Heartbeatmeldung.	0 bis 1

5.3.6 Analyzer

Die Informationen über den oder die Sensoren, die für die Erkennung und Weiterleitung der Alarm- oder Heartbeatmeldungen verantwortlich sind, werden in der Analyzer-Klasse zusammengefasst. Sind mehrere Sensoren an der Erkennung oder Weiterleitung beteiligt, werden diese in absteigender Reihenfolge rekursiv im XML-Baum repräsentiert, das heißt, das kleinste Kindelement hat als erstes eine verdächtige Aktion registriert und die Alarmmeldung generiert. Alle anderen Sensoren haben dann nur noch Informationen ergänzt, geändert oder die Meldung weitergeleitet.

Unterklassen

Name	Erklärung	Anzahl
Analyzer	siehe 5.3.6	0 bis 1
Node	siehe 5.3.16	0 bis 1
Process	siehe 5.3.20	0 bis 1

Attribute

Name	Erklärung	Anzahl
analyzerid	Ein eindeutiger Identifier für den Sensor. Zwar ist die Angabe einer analyzerid strenggenommen optional, sie wird jedoch verpflichtend, wenn in irgendeinem anderen Kontext ein Identifier gesetzt wird, zum Beispiel eine messageid der Heartbeat-Klasse.	0 bis 1
class	Der Typ des Sensors	0 bis 1
(ls)manufacturer	Der Hersteller des Sensors	0 bis 1
(ls)model	Die genaue Modellbeschreibung des Sensors	0 bis 1
(ls)name	Ein frei wählbarer Name, der die Identifizierung des Sensors für Menschen erleichtert.	0 bis 1
(ls)ostype	Der Name des Betriebssystems. Ist im Allgemeinen die Ausgabe des „uname s“-Kommandos.	0 bis 1
(ls)osversion	Die Version des Betriebssystems. Ist im Allgemeinen die Ausgabe des „uname r“-Kommandos.	0 bis 1
(ls)version	Die Versionsnummer	0 bis 1

Erläuterung

Sensoren gehören zu den kritischen Infrastrukturen und die gemachten Angaben könnten einem potentiellen Angreifer Interna preisgeben, die eventuell für Angriffe ausgenutzt werden könnten, vor allem wenn die Sensoren auf Systemen installiert sind, die nicht immer aktuell gehalten werden (können).

manufacturer, model & version. Ist bekannt, welche Sensoriken verwendet wird, kann ein Angreifer versuchen, unter dem Radar dieser Produkte zu bleiben.

name. Wird der Wert nicht durch einen neuen Namen ersetzt, ist in der Standardinstallation ein Rückschluss auf den verwendeten Sensor möglich.

ostype & osversion. Da jedes Betriebssystem eigene Sicherheitslücken hat, kann diese Angabe einem Angreifer helfen, die Schutzmaßnahmen zu umgehen.

5.3.7 Classification

Die Classification-Klasse gibt einem Alarm einen „Namen“. Sie hilft, Meldungen zu Alarmklassen zuzuordnen.

Unterklassen

Name	Erklärung	Anzahl
Reference	siehe 5.3.15	0 bis ∞

Attribute

Name	Erklärung	Anzahl
ident	Ein Identifier für diese Alarmklassifikation.	0 bis 1
text	Ein Text, der den Alarm klassifiziert, beispielsweise „Port-Scan“. Anmerkung: In diesem Attribut wird der Name der Klassifizierung von Alarmen angegeben. Dieser enthält keine personenbezogenen Informationen.	1

5.3.8 Source

Wurde von einem Sensor eine oder mehrere eindeutige Quellen von verdächtigen Aktionen identifiziert, so werden die darüber verfügbaren Informationen in der Source-Klasse abgelegt.

Unterklassen

Name	Erklärung	Anzahl
Node	siehe 5.3.16	0 bis 1
Process	siehe 5.3.20	0 bis 1
Service	siehe 5.3.21	0 bis 1
User	siehe 5.3.18	0 bis 1

Attribute

Name	Erklärung	Anzahl
ident	Ein eindeutiger Identifier für diese Instanz der Source-Klasse.	0 bis 1
(ls)interface	Sind mehrere Netzwerkinterfaces an dem angegriffenen Rechner vorhanden, kann hier das betroffene Interface genannt werden.	0 bis 1
spoofed	Wenn vom Sensor erkannt werden kann, dass die IP-Adresse des Angreifers gefälscht wurde, kann dies in diesem Attribut vermerkt werden.	0 bis 1

Erläuterung

interface. Die Angabe der intern genutzten Interfaces kann eventuell nicht erwünscht sein.

5.3.9 Target

Analog zur Source-Klasse stellt die Target-Klasse Informationen zu dem Ziel oder den Zielen einer verdächtigen Aktion bereit.

Unterklassen

Name	Erklärung	Anzahl
File	siehe 5.3.24	0 bis ∞
Node	siehe 5.3.16	0 bis 1
Process	siehe 5.3.20	0 bis 1
Service	siehe 5.3.21	0 bis 1
User	siehe 5.3.18	0 bis 1

Attribute

Name	Erklärung	Anzahl
decoy	Wenn vom Sensor erkannt werden kann, dass die IP-Adresse des Zielrechners gefälscht wurde, kann dies in diesem Attribut vermerkt werden.	0 bis 1
ident	Ein eindeutiger Identifier für diese Instanz der Target-Klasse.	0 bis 1
(ls)interface	Sind mehrere Netzwerkinterfaces an dem angegriffenen Rechner vorhanden, kann hier das betroffene Interface genannt werden.	0 bis 1

Erläuterung

interface. Die Angabe der intern genutzten Interfaces kann eventuell nicht erwünscht sein.

5.3.10 Assessment

Die Assessment-Klasse bietet den Sensoren die Möglichkeit, eine erste Einschätzung der verdächtigen Aktion zu machen. Weiterhin können die ergriffenen Gegenmaßnahmen protokolliert werden.

Unterklassen

Name	Erklärung	Anzahl
Action	siehe 5.3.13	0 bis ∞
Confidence	siehe 5.3.14	0 bis 1
Impact	siehe 5.3.12	0 bis 1

5.3.11 AdditionalData

Alle Daten, für die es keinen speziellen Platz in der IDMEF-Nachricht gibt, beispielsweise die Header-Daten eines IP-Paketes, können nach ihrem Datentyp sortiert in die AdditionalData mit hinzugefügt werden.

Attribute

Name	Erklärung	Anzahl
meaning	Eine kurze Beschreibung der in den AdditionalData versendeten Informationen.	0 bis 1
type	Ein Wert aus dem Wertbereich `boolean`/ `byte`/ `character`/ `date-time`/ `integer`/ `ntpstamp`/ `portlist`/ `real`/ `string`/ `byte-string`/ `xmltext`	0 bis 1

Inhalt

(ls)(d)Der Inhalt von AdditionalData kann dazu verwendet werden, um weitere Daten, die sonst im IDMEF-Standard keinen Platz gefunden haben, zu übermitteln oder um den IDMEF-Standard beliebig zu erweitern.

Erläuterung

Inhalt. Das Feld AdditionalData ist im IDMEF-Standarddazu gedacht, weitere Angaben, die für die Analyse eines Angriffs wichtig sind, mit angeben zu können oder den IDMEF-Standard an dieser Stelle zu erweitern. Was genau in den einzelnen Feldern gespeichert wird, kann man a priori aber nicht sagen.

5.3.12 Impact

Um eine Einschätzung der Auswirkungen einer verdächtigen Aktion geben zu können, ist die Impact-Klasse vorhanden.

Attribute

Name	Erklärung	Anzahl
completion	Ist ein Angriff erfolgreich gewesen? Kann der Sensor diese Frage beantworten, kann er sein Ergebnis in dieses Attribut schreiben.	0 bis 1
severity	Eine Einschätzung der Schwere der Auswirkung von Info bis hoch.	0 bis 1
type	Eine Kategorisierung der verdächtigen Aktion in verschiedene Typen ist mit Hilfe dieses Attributs möglich.	0 bis 1

Inhalt

(ls)(d) Textuelle Beschreibung der Auswirkung, soweit das vom Sensor unterstützt wird.

Erläuterung

Inhalt. Bei der Beschreibung der Auswirkung kann es unter Umständen passieren, dass personenbezogene Daten oder Daten, die unter dem Schutz der Sicherheitsrichtlinie stehen, weitergegeben werden. Beispielsweise könnte in der Meldung „Benutzerkennung XYZ kompromittiert“ stehen

5.3.13 Action

Wurde durch den Sensor irgendeine Aktion getriggert, beispielsweise eine Umkonfiguration der Firewall, so können diese Aktionen hier protokolliert werden.

Attribute

Name	Erklärung	Anzahl
category	Der Typ der durchgeführten Aktion.	1

Inhalt

(ls)(d)Erweiterte Beschreibung der Aktion.

Erläuterung

Inhalt. Der Inhalt kann personenbezogene Daten enthalten, wenn die Aktion auf einen speziellen Benutzer ausgerichtet ist, beispielsweise „blocked user XY“. Weiterhin kann es die lokale Sicherheitsrichtlinie verletzen, wenn der Name oder die Position der Firewall genannt wird.

5.3.14 Confidence

Je nachdem, aus welchen Regelsätzen eine Meldung generiert wurde, kann der Sensor eine Einschätzung geben, in wie weit er sich bei den gemachten Angaben „sicher“ ist und diese in der Confidence-Klasse ablegen. So ist es bei einer neuartigen heuristischen Meldung eher wahrscheinlich, dass ein gemeldeter Alarm sich am Ende als falsch erweist, als bei einer Signatur, die schon länger im Einsatz ist und nie Falschalarme generiert hat.

Attribute

Name	Erklärung	Anzahl
rating	Eine Einschätzung, die die eigene Aussagekraft in der Skala von „wenig“ zu „hoch“ oder in einem beliebigen vom Sensor gelieferten numerischen Wert bewertet.	1

Inhalt

Genauere Beschreibung der Einschätzung, beispielsweise ein genauer numerischer Wert zwischen 0.0 und 1.0.

5.3.15 Reference

Da Alarme von unterschiedlichen Herstellern von Sicherheitssystemen unterschiedlich genannt werden, ist eine einfache Korrelation nicht ohne weiteres gegeben. Daher bietet die Reference-Klasse Beschreibungen unterschiedlicher Hersteller zu verknüpfen und somit eine Möglichkeit bereit zu stellen, Korrelationen zu ermöglichen.

Unterklassen

Name	Erklärung	Anzahl
name	Der Name des Alarms. Stammt aus der in orgin genannten Quelle.	1
url	Ein Querverweis, auf dem weitere Informationen über den Alarm gefunden werden können.	1

Attribute

Name	Erklärung	Anzahl
meaning	Dieses Feld bietet die Möglichkeit, eine eigene Interpretation der Alarmmeldung zu geben.	0 bis 1
orgin	Der Quelle, aus der der Name des Alarms (name) stammt.	1

5.3.16 Node

Hosts und andere Netzwerkgeräte können durch die Node-Klasse identifiziert werden.

Unterklassen

Name	Erklärung	Anzahl
Address	siehe 5.3.17	0 bis ∞
(ls)location	Der Ort des Gerätes.	0 bis 1
(ls)(d)name	Der Name des Gerätes.	0 bis 1

Attribute

Name	Erklärung	Anzahl
category	Die Domäne, in der sich das Gerät befindet, beispielsweise „AFS“ oder „Windows NT“	0 bis 1
ident	Ein eindeutiger Identifier für diese Instanz der Node-Klasse.	0 bis 1

Erläuterung

location. Die genaue Angabe des Standortes eines Knotens kann aus sicherheitstechnischen Gesichtspunkten unerwünscht sein.

name. Der Namen kann personenbezogene Daten enthalten. Weiterhin kann diese Angabe aus sicherheitstechnischen Gesichtspunkten unerwünscht sein.

5.3.17 Address

Um einen Nutzer oder ein Gerät identifizieren zu können, gibt es in IDMEF die Möglichkeit, Adressinformationen in der Address-Klasse mit abzubilden. Diese reichen von IP-Adressen über MAC-Adressen bis hin zu E-Mail-Adressen.

Unterklassen

Name	Erklärung	Anzahl
(ls)(d)address	Die Adressinformation, beispielsweise die IP-Adresse.	1
(ls)netmask	Die Subnetzmaske der IP-Adresse.	0 bis 1

Attribute

Name	Erklärung	Anzahl
category	Die Kategorie, in der die Adressinformation (address) angegeben ist, beispielsweise IPv4-Adresse oder IPv6-Adresse.	0 bis 1
ident	Ein eindeutiger Identifier für die Instanz der Adress-Klasse.	0 bis 1
(ls)vlan-name	Der Name des VLANs, zu dem die Adresse gehört.	0 bis 1
(ls)vlan-num	Die Nummer des VLANs, zu dem die Adresse gehört.	0 bis 1

Erläuterung

Die Angabe einer Adresse ist sowohl für die Quelle und das Ziel eines Angriffs von entscheidender Bedeutung. Ohne die Angabe einer Netzwerkadresse ist beispielsweise ein DoS-Angriff, der von einer einzelnen Adresse ausgeht nicht zu erkennen. Auf der anderen Seite stellen das Datenschutzgesetz (BDSG) und die heutige Rechtsprechung eindeutig klar, dass beispielsweise IP-Adressen als personenbezogene Daten einem besonderen Schutz unterliegen und nicht ohne weiteres anlassunabhängig weitergegeben werden dürfen.

address. Die IP-Adresse, aber auch eine E-Mail-Adresse, ist klar ein personenbezogenes Datum und darf deshalb ohne besonderen Grund nicht weitergegeben werden. Ebenso kann es aus Sicht der lokalen Sicherheitsrichtlinie unerwünscht sein, beispielsweise interne Adressen nach außen zu geben.

netmask, vlan-name & vlan-num. Durch diese Angaben gibt man Interna preis, wie das Netz innen aufgebaut ist. Diese Angabe kann aus sicherheitstechnischen Gesichtspunkten unerwünscht sein.

5.3.18 User

In der User-Klasse kann ein Benutzer beschrieben werden. Sie dient jedoch hauptsächlich als Container für die UserId-Klasse.

Unterklassen

Name	Erklärung	Anzahl
UserId	siehe 5.3.19	1 bis ∞

Attribute

Name	Erklärung	Anzahl
category	Der Typ, der repräsentiert wird. Es kann sich dabei um einen Application-, ein Betriebssystembenutzer oder einen unbekannten Status handeln.	0 bis 1
ident	Ein eindeutiger Identifier für die Instanz der Benutzer-Klasse.	0 bis 1

5.3.19 UserId

In der UserId-Klasse können die Angaben zu einem Benutzer spezifiziert werden.

Unterklassen

Name	Erklärung	Anzahl
(ls)(d)name	Ein Benutzer- oder Gruppenname.	0 bis 1
(ls)(d)number	Eine Benutzer- oder Gruppennummer.	0 bis 1

Attribute

Name	Erklärung	Anzahl
ident	Ein eindeutiger Identifier für die Instanz der UserId-Klasse.	0 bis 1
(ls)tty	Das momentan vom Benutzer verwendete TTY.	0 bis 1
type	Der Typ der Userkennung. Wurde beispielsweise ein sudo-Befehl ausgeführt, ist es wichtig zu wissen, ob die angegebene Userkennung diejenige ist, die privilegiert ist oder nicht.	0 bis 1

Erläuterung

name. Zwar ist die Benutzerkennung „root“ datenschutzrechtlich nicht bedenklich, bei personalisierten Kennungen ist diese jedoch wenigstens innerhalb einer Domäne zumeist eineindeutig einer bestimmten Person zuzuordnen. Daher fällt diese Angabe wie auch die Angabe über die IP-Adresse unter das BDSG.

number. Innerhalb eines Systems ist eine Benutzernummer immer eindeutig einem Benutzer zuzuordnen. Somit ist die Angabe datenschutzrechtlich interessant.

tty. Diese Angabe erlaubt Rückschlüsse auf installierte Software und ist deshalb sicherheitstechnisch von Bedeutung.

5.3.20 Process

Für die Erkennung von Angriffen interessante Prozesse können in der Process-Klasse erläutert werden.

Unterklassen

Name	Erklärung	Anzahl
(ls)(d)arg	Wurden zum Ausführen des Programms Argumente mit übergeben, so werden diese hier in genau der Reihenfolge ihrer Angabe mit eingefügt.	0 bis ∞
(ls)(d)env	Benutzt, das Programm Umgebungsvariablen, so können diese hier mit angegeben werden.	0 bis ∞
(ls)name	Der Name des momentan ausgeführten Programms ohne Pfad- und Argumentangaben.	1
(ls)(d)path	Der vollständige Pfad zum Programm.	0 bis 1
pid	Die Prozess-ID des Programms.	0 bis 1

Attribute

Name	Erklärung	Anzahl
ident	Ein eindeutiger Identifier für die Instanz der Prozess-Klasse.	0 bis 1

Erläuterung

arg & env. In den übergebenen Argumenten ähnlich wie in den Umgebungsvariablen können sich sowohl bedenkliche Daten als auch datenschutzrechtlich relevante Daten befinden, beispielsweise bei den Argumenten des Befehls „ssh root@lrz.de“ ist eine Benutzerkennung mit angegeben worden.

name. Bei dem gemeldeten Prozess kann es sich um einen aus Sicht der Sicherheitsrichtlinie kritischen Prozess handeln, dessen Name eventuell nicht weitergegeben werden sollte.

path. Da bei der Angabe des Pfades unter Umständen der Benutzername auslesbar ist, wie beispielsweise in Home-Verzeichnissen, ist dieses Feld datenschutzrechtlich von Bedeutung.

5.3.21 Service

In der Service-Klasse können Netzwerk-Services auf Quell- oder Zielrechnern beschrieben werden.

Unterklassen

Name	Erklärung	Anzahl
(ls)name	Der Name des Services. Wenn möglich sollte der IANA-Name von bekannten Ports verwendet werden.	0 bis 1
(ls)port	Die verwendete Portnummer.	0 bis 1
(ls)portlist	Sind mehrere Ports in Verwendung, so kann man hier eine Liste aller Ports angeben.	0 bis 1
(ls)protocol	Ergänzende Angaben über das verwendete Protokoll können hier niedergeschrieben werden.	0 bis 1

Attribute

Name	Erklärung	Anzahl
(ls)iana_protocol_name	Der IANA-Protokollname.	0 bis 1
(ls)iana_protocol_number	Die IANA-Protokollnummer.	0 bis 1
ident	Ein eindeutiger Identifier für die Instanz der Service-Klasse.	0 bis 1
(ls)ip_version	Die IP-Version.	0 bis 1

Spezialisierung

Name	Erklärung	Anzahl
SNMPService	siehe 5.3.23	0 bis 1
WebService	siehe 5.3.22	0 bis 1

Erläuterung

iana_protocol_name, iana_protocol_number, ip_version, name, port, portlist & protocol. Diese Angaben geben viele Informationen weiter, wobei dies aus sicherheitstechnischen Überlegungen unter Umständen nicht erwünscht ist.

5.3.22 WebService

Als Spezialisierung der Service-Klasse bietet die WebService-Klasse die Möglichkeit ergänzende Angaben zu Web-Anwendungen anzugeben.

Unterklassen

Name	Erklärung	Anzahl
(ls)(d)arg	Die Argumente, die dem CGI-Skript mit übergeben wurden.	0 bis 1
(ls)cgi	Wurde ein CGI-Skript angefordert, so kann dieses hier benannt werden. Argumente, die dem CGI-Skript mit übergeben werden, werden aber in arg angegeben.	0 bis 1
http-method	Die http-Methode (POST oder GET).	0 bis 1
(ls)url	Die URL des Aufrufs.	1

Erläuterung

arg. In den übergebenen Argumenten können sich sowohl bedenkliche Daten als auch datenschutzrechtlich relevante Daten befinden, beispielsweise wenn bei den Argumenten eine Benutzerkennung mit angegeben wurde.

cgi & url. Aus Sicht der lokalen Sicherheitsrichtlinie ist die Weitergabe dieser Informationen eventuell zu beschränken, um Angriffsvektoren zu verkleinern.

5.3.23 SNMPService

Speziell für SNMP-Verkehr wurde die SNMPService-Klasse als eine eigene Spezialisierung der Service-Klasse mit eingeführt. Die anzugebenen Attribute beziehen sich zumeist auf RFC 3411.

Unterklassen

Name	Erklärung	Anzahl
(ls)(d)command	Das an den SNMP-Server verschickte Kommando.	0 bis 1
(ls)context-EngineID	Der Kontext-Engine-Identifier des Objektes.	0 bis 1
(ls)contextName	Der Kontextname des Objektes.	0 bis 1
(ls)message-Processing-Model	Die SNMP-Version.	0 bis 1
(ls)oid	Der Objekt-Identifier des Requests.	0 bis 1
(ls)securityLevel	Das Security-Level des Requestes.	0 bis 1
(ls)securityModel	Angabe, welches Security-Modell benutzt wurde.	0 bis 1
(ls)securityName	Der Security-Name des Objektes.	0 bis 1

Erläuterung

command, contextEngineID, contextName, messageProcessingModel, oid, securityLevel, securityModel & securityName. Diese Angaben geben viele Informationen weiter, wobei dies aus sicherheitstechnischen Überlegungen unter Umständen nicht erwünscht ist.

5.3.24 File

Alle Dateizugriffe, die im Zusammenhang mit einer verdächtigen Aktion stehen, können in der File-Klasse protokolliert werden.

Unterklassen

Name	Erklärung	Anzahl
access-time	Der Zeitpunkt des letzten Zugriffs auf die Datei.	0 bis 1
Checksum	siehe 5.3.28	0 bis ∞
create-time	Der Erstellungszeitpunkt der Datei.	0 bis 1
data-size	Die Größe der Datei in Bytes.	0 bis 1
(ls)disk-size	Die tatsächlich belegte Größe auf dem Datenträger in Bytes.	0 bis 1
FileAccess	siehe 5.3.25	0 bis ∞
Inode	siehe 5.3.27	0 bis 1
Linkage	siehe 5.3.26	0 bis ∞
modify-time	Der Zeitpunkt der letzten Änderung an der Datei.	0 bis 1
(ls)(d)name	Der Dateiname.	1
(ls)(d)path	Der Pfad zur Datei.	1

Attribute

Name	Erklärung	Anzahl
category	Der Kontext der Informationen, also ob die Informationen sich auf den Zeitpunkt vor der verdächtigen Aktion beziehen oder auf einen späteren Zeitpunkt.	1
(ls)file-type	Der MIME-Typ der Datei.	0 bis 1
(ls)fstype	Der Typ des Filesystems, beispielsweise FAT oder NTFS.	0 bis 1
ident	Ein eindeutiger Identifier für die Instanz der File-Klasse.	0 bis 1

Erläuterung

disk-size, file-type & fstype. Aus diesen Angaben kann man Rückschlüsse auf das verwendete System ziehen, die unter Umständen aus Sicht der lokalen Sicherheitsrichtlinie unerwünscht sind.

name. Bei der gemeldeten Datei kann es sich um einen aus Sicht der Sicherheitsrichtlinie kritischen Datei handeln, dessen Name eventuell nicht weitergegeben werden sollte oder der Dateiname erlaubt Rückschlüsse auf den Benutzer und ist daher aus datenschutzrechtlicher Sicht zu beanstanden.

path. Da bei der Angabe des Pfades unter Umständen der Benutzername auslesbar ist, wie beispielsweise in Home-Verzeichnissen, ist dieses Feld datenschutzrechtlich von Bedeutung.

5.3.25 FileAccess

Die FileAccess-Klasse repräsentiert die Zugriffsberechtigungen auf eine Datei.

Unterklassen

Name	Erklärung	Anzahl
(ls)Permission	Das Level der Zugriffsberechtigung von „kein Zugriff“ bis hin zu „Benutzer hat die volle Kontrolle über die Datei“.	1 bis ∞
UserId	siehe 5.3.19	1

Erläuterung

Permission. Die lokale Sicherheitsrichtlinie verbietet unter Umständen die Weitergabe der Angabe der genauen Berechtigungen. Somit soll ein möglicher Angriffsvektor verkleinert werden, indem offensichtliche Fehlkonfigurationen nicht öffentlich gemacht werden.

5.3.26 Linkage

In der Linkage-Klasse können Verknüpfungen im Dateisystem wie beispielsweise Hard-Links dargestellt werden.

Unterklassen

Name	Erklärung	Anzahl
File	siehe 5.3.24	1
(ls)(d)name	Der Dateiname.	1
(ls)(d)path	Der Pfad zur Datei.	1

Attribute

Name	Erklärung	Anzahl
category	Der Typ der Verknüpfung zwischen den Dateien, beispielsweise Hard-Links oder Shortcuts.	1

Erläuterung

name. Bei dem gemeldeten Prozess kann es sich um einen aus Sicht der Sicherheitsrichtlinie kritischen Prozess handeln, dessen Name eventuell nicht weitergegeben werden sollte.

path. Da bei der Angabe des Pfades unter Umständen der Benutzername auslesbar ist, wie beispielsweise in Home-Verzeichnissen, ist dieses Feld datenschutzrechtlich von Bedeutung.

5.3.27 Inode

Die Inode-Klasse wird verwendet, um zusätzliche Informationen, die in Inodes der Unix Dateisysteme enthalten sind, zu repräsentieren.

Unterklassen

Name	Erklärung	Anzahl
change-time	Der Zeitpunkt der letzten Änderung der Inode-Daten.	0 bis 1
c-major-device	Die Major-Device-Nummer der Datei, falls es sich um ein Character Special Device handelt.	0 bis 1
c-minor-device	Die Minor-Device-Nummer der Datei, falls es sich um ein Character Special Device handelt.	0 bis 1
major-device	Die Major-Device-Nummer des Geräts, auf dem die Datei sich befindet.	0 bis 1
minor-device	Die Minor-Device-Nummer des Geräts, auf dem die Datei sich befindet.	0 bis 1
number	Die Inode-Nummer.	0 bis 1

5.3.28 Checksum

Sind Checksummen von Dateien verfügbar und diese Information für die Analyse eines Vorfalls von Interesse, so können hier die benötigten Angaben gemacht werden.

Unterklassen

Name	Erklärung	Anzahl
(ls)key	Falls nötig, kann in diesem Attribut ein Schlüssel zur Berechnung der Checksum mit angegeben werden.	0 bis 1
value	Der berechnete Wert der Checksum.	1

Attribute

Name	Erklärung	Anzahl
algorithm	Der Krypto-Algorithmus, der zum Berechnen der Checksum verwendet wurde.	1

Erläuterung

key. Die Weitergabe eines kryptographischen Schlüssels, auch wenn er nur für die Berechnung von Checksummen verwendet wird, ist aus sicherheitstechnischen Überlegungen nicht anzuraten.

Kapitel 6

Zusammenfassung

Dieses Dokument präsentierte die prototypische Implementierung des Grid-IDS (GIDS) als Ergebnis des Arbeitspakets 6 des Projektes „Ein Grid-basiertes, föderiertes Intrusion Detection System zur Sicherung der D-Grid Infrastruktur“ (GIDS) (`http://www.grid-ids.de`). Das GIDS-Projekt ist ein Teilprojekt im Rahmen des D-Grid (`http://www.d-grid.de`) und wird vom Bundesministerium für Bildung und Forschung (BMBF, `http://www.bmbf.de`) gefördert. Ziel des Dokumentes ist die Zusammenfassung der Implementierung, Installation und Nutzung der prototypischen Komponenten des GIDS. Das Dokument untergliedert sich in verschiedene Teile, die sich aus der Struktur des Grid-IDS ergeben. Dies sind zusammengefasst:

GIDS-Bus. Der zentrale GIDS-Bus dient zur Realisierung der Interoperation der GIDS-Komponenten auf der Seite der Ressourcenprovider und des Betreibers. Die technische Realisierung erfolgt zusammengefasst auf einer Multicast-fähigen VPN-Infrastruktur und erfüllt die Sicherheitsanforderungen, die sich aus dem Projekt ergeben und in [4] beschrieben worden sind. Technisch erfolgt der Datentransport über verschlüsselte Kanäle, wobei die Partner entsprechend authentifiziert werden, um Missbrauch auszuschliessen. Des weiteren bietet die Bus-Infrastruktur Redundanz, die DoS Angriffe erschwert und die Ausfallsicherheit erhöht.

Komponenten auf der Seite der Ressourcenprovider. Dies sind sowohl die Sensork selbst, als auch die Komponenten zur Kommunikation über den GIDS-Bus. Um den Aufwand auf der Seite der Ressourcenprovider einfach zu halten und deren Autonomie zu wahren, wird auf der Prelude “Security Information and Event Management” (SIEM) Architektur aufgebaut. Diese umfasst eigene Host- und Netz-basierte IDS und unterstützt etablierte IDS wie beispielsweise Snort. Die Kommunikation erfolgt durch den Prelude-Manager über das Austauschformat IDMEF und unterstützt die Vertraulichkeit, Integrität und Authentizität der Daten beim Transport. Die Anbindung anderer, bestehender Komponenten wird durch die Bibliothek libprelude unterstützt. Der Export der Daten zum Betreiber und den Ressourcenprovidern erfolgt durch den GIDS-Agenten, der die Daten an den GIDS-Bus übergibt. Bei diesem Schritt können die Daten gemäß einer vom Ressourcenprovider angepassten Konfiguration gefiltert und anonymisiert werden.

Komponenten auf der Seite des Betreibers. Da der Betreiber selbst auch als Ressourcenprovider auftreten kann, können alle vorher beschriebenen Komponenten vom Betreiber eingesetzt werden. Zusätzlich stellt der Betreiber ein Portal zur Verfügung, das den GIDS-Benutzergruppen Zugriff auf die Angriffsdaten und globalen Grid-Statistiken ermöglicht. Unterstützt werden die Gruppen der Ressourcenprovider, virtuellen Organisationen (VO) und des Betreibers. Dabei wird der Zugriff auf die Daten gemäß des Datenschutzkonzeptes eingeschränkt. Um die Skalierbarkeit der Architektur zu verbessern, wurde das aufwändige Datenbankschema von Prelude durch ein eigenes, optimiertes ersetzt.

GIDS-Datenschutzkonzept. Innerhalb des GIDS werden sowohl sicherheitskritische Daten auf der Seite der Ressourcenprovider als auch Daten mit Personenbezug erhoben. Letz-

tere unterliegen dem Bundesdatenschutzgesetz (BDSG), das die Erhebung, Weitergabe und Nutzung dieser Daten strikt einschränkt. Des weiteren geben die Angriffsdaten der Ressourcenprovider kritische Informationen über aktuelle Sicherheitsvorfälle preis und lassen sich für gezielte Angriffe ausnutzen. Deshalb sind diese Daten streng vertraulich zu halten und ggf. vor dem Export an das GIDS zu filtern. Beides wurde auf konzeptioneller Ebene in dem GIDS-Datenschutzkonzept und auf technischer Ebene in dem GIDS-Agenten realisiert, die speziell an das zentrale Austauschformat IDMEF angepasst wurden. Zur Realisierung wurden alle Felder des Formates untersucht, in wie weit sicherheitskritische oder personenbezogene Daten vorhanden sein können. Auf dieser Basis wird eine Filterung oder Anonymisierung der Daten vorgenommen.

Anhang A

Prelude-Manager Konfigurationsdatei

```
# Prelude Manager configuration file.
#
# <IMPORTANT>
#
# Sections are importants, and things won't work correctly if they are
# not un-commented. For example you need to uncomment [db] if you want
# the database plugin to be loaded.
#
# </IMPORTANT>


include = @LIBPRELUDE_CONFIG_PREFIX@/default/global.conf


# Address where the prelude-manager server is listening on.
# if value is unix, or unix:/path/to/unix/socket, an UNIX domain socket
# will be used.
#
# Multiple listen address are supported.
#
# listen = address:port
# listen = unix:/tmp/prelude-manager.socket
# listen = unix
#
listen = 127.0.0.1


# Sets the user/group ID as which prelude-manager will run.
# In order to use this option, prelude-manager must be run initially as
# root
#
# user = prelude
# group = prelude


# Number of second prelude-manager wait for an incoming client to
# successfully authenticate before droping the connection.
#
```

```
# connection-timeout = 10


#
# Scheduler settings for Prelude-Manager
#
# On systems with many concurrent sensors sending events to
# Prelude-Manager, Prelude-Manager might have an hard time keeping up
# with the demand for events reporting.
#
# The Prelude Manager scheduler allocate reporting time per sensor,
# allowing to define the maximum number of events processed for one
# sensor before processing others sensors events (in case a sensor is
# sending a continuous events burst, this prevent other sensors
# starvation).
#
# By default, for each sensor connected, a maximum of 100 events will
# be processed before processing others sensors events.
#
# Additionally, priority will be given to events depending on their
# priority. Assuming there is enough events of each priority, 50 high
# priority message will be processed, 30 medium, and 20 low (totalling
# the maximum of 100 described above).
#
# You might use the sched-priority option in order to change this
# setting:
#
# sched-priority = high:50 medium:30 low:20
#
#
# When the number of events waiting to be processed exceed the defined
# amount of reserved memory (default is 1 Megabyte), Prelude-Manager
# will start storing events on disk:
#
# sched-buffer-size = 1M


#
# TLS options (only available with GnuTLS 2.2.0 or higher):
# sets availables ciphers, key exchange methods, macs and compression
# methods.
#
# "NORMAL" option enables all "secure" ciphersuites, 256-bit ciphers
# included.
#
# "SECURE128" flag enables all "secure" ciphersuites with ciphers up to
# 128 bits.
#
# "SECURE256" flag enables all "secure" ciphersuites including the 256
# bit ciphers.
#
# "EXPORT" all the ciphersuites are enabled, including the low-security
# 40 bit ciphers.
#
# "NONE" nothing is enabled. This disables even protocols and
# compression methods.
```

```
#
# Note that much more settings might be enabled or disabled using this
# option: please see gnutls_priority_init(3) for more details.
#
# The default settings is "NORMAL".
# tls-options = NORMAL


#
# Number of bits of the prime used in the Diffie Hellman key exchange.
# Note that the value should be one of 768, 1024, 2048, 3072 or 4096.
# The default is 1024.
#
# dh-prime-length = 1024


# How often to regenerate the parameters used in the Diffie Hellman key
# exchange. These should be discarded and regenerated once a day, once
# a week or once a month. Depending on the security requirements.
#
# Generation is a CPU intensive operation. The value is in hours,
# 0 disables regeneration entirely. The default is 24 hours.
#
# dh-parameters-regenerate = 24


# If you want this Manager to retrieve message from another Manager
# (useful if the other Manager is located within a DMZ):
#
# child-managers = x.x.x.x
#
# This mean the messages should be gathered from x.x.x.x

#
# If you want a given reporting plugin to be protected against possible
# failure, use the failover option. Failover will prevent data sent to
# the report plugin to be lost in case this one fail.
#
# You might use this option multiple time for different plugins.
#
# failover = name_of_plugin


#
# Events normalization parameters
#
# Un-comment the following section in case you want to define any
# normalization parameters:
#
# [normalize]
#
# For each incoming events, Prelude-Manager will run a number of
# normalization routine: sanitize address, services information, etc.
#
# When the normalizer see an incoming IPv4 mapped IPv6 address, the
# default behavior is to map it back to raw IPv4. For example,
```

```
# ::ffff:192.168.0.1 will be mapped back to 192.168.0.1
#
# If you do not want IPv4 mapped IPv6 addresses, un-comment the
# following option:
#
# keep-ipv4-mapped-ipv6
#
# Alternatively, if you wish for any input IPv4 addresses to be
# converted to IPv6, un-comment the following option:
#
# ipv6-only


####################################
# Here start plugins configuration #
####################################

# [relaying]
#
# If you want the message caught by this manager to be relayed.
# You can use boolean AND and OR to make the rule.
#
# parent-managers = x.x.x.x || y.y.y.y && z.z.z.z
#
# This mean the emission should occur on x.x.x.x or, if it fail, on
# y.y.y.y and z.z.z.z (if one of the two host in the AND fail, the
# emission will be considered as failed involving saving the message
# locally).


# [db]

# The type of database: mysql, pgsql or sqlite3.
# type = mysql

# Only if you use sqlite3.
# file = /your/path/to/your/db/idmef-db.sql

# Host the database is listening on.
# host = localhost

# Port the database is listening on.
# port = 3306

# Name of the database.
# name = prelude

# Username to be used to connect the database.
# user = prelude

# Password used to connect the database.
# pass = xxxxxx



# [XmlMod]
```

```
#
# The Xmlmod plugin allow to report alert as IDMEF XML in a file,
# or to dump theses alert to stderr.
#
# The default behavior is to write output to stderr.
#
# Tell Xmlmod to disable output file buffering.
# This will prevent XML alerts to be truncated and thus make real-time
# parsing easier:
#
# disable-buffering
#
#
# Tell Xmlmod to check generated XML against IDMEF DTD:
# validate
#
# Tell Xmlmod to produce a pretty, human readable xml output:
# format
#
# logfile = stderr
# logfile = /var/log/prelude-xml.log



# [Debug]
#
# The Debug plugin allow to report alert as text in a file,
# or to dump theses alert to stderr.
#
# The default behavior is to write output to stderr.
#
# logfile = stderr
# logfile = /var/log/prelude.log
#
# You can specify the name of the IDMEF object to print (you might
# select multiple objects). If no object are provided, 'Debug' will
# print out the entire message.
#
# object = alert.classification.text, alert.source(0).node.address(0).address


# [TextMod]
#
# The Debug plugin allow to report alert as text in a file,
# or to dump theses alert to stderr.
#
# The default behavior is to write output to stderr.
#
# logfile = stderr
# logfile = /var/log/prelude.log


#[smtp]
#
# Sender to use for the mail message.
# sender = prelude@myhostname.
```

```
#
# Who the mail should be sent to.
# recipients = recipient1@hostname, recipient2@hostname
#
# SMTP server to use for sending mail
# smtp-server = localhost
#
# By default, the SMTP plugin send mail containing the whole IDMEF
# event. If you wish to send a subset of the information, you may
# customize the content of the generated mail through several options:
#
# You can define a specific subject to use with mail notification.
# The subject can include information from the event using IDMEF path.
# subject = Alert: $alert.classification.text
#
# You can define a specific message body to use for mail notification.
# As with the "subject" option, the template can include information
# from the event using IDMEF path.
#
# (Template example available in @DOCDIR@/smtp/template.example)
# template = /path/to/my/template
#
# You can provide your database settings here, so that the SMTP plugin
# retrieve alert linked to received CorrelationAlert from the database.
#
# dbtype = mysql
# dbname = prelude
# dbuser = prelude
# dbpass = passwd
# dbhost = localhost
# Other database options available include dbport, and dbfile (for
# sqlite3 database).
#
# If you have specified your database settings above, you can also
# use the correlated-alert-template option, which is like the "template"
# option but is specific to Correlated Alerts retrieved from database.
#
# (Template example available in @DOCDIR@/smtp/template.example)
# correlated-alert-template = /path/to/my/template




####################################
# Filtering plugins configuration  #
####################################

# The idmef-criteria filtering plugin allow you to filter events based
# on specific IDMEF-Criteria.
#
# [idmef-criteria]
# rule = alert.classification.text == 'User login successful'
# hook = relaying[default]
#
# Will forward any events that match the defined criteria to the
# default instance of the relaying reporting plugin. The rule argument
```

```
# might also be a filename containing the rules. Example:
#
# rule = /path/to/rule.file


# The thresholding filtering plugin allow you to suppress events based
# on their value.
#
# [thresholding]
# path = alert.classification.text, alert.source.node.address.address
# limit = 3600
# count = 1
# hook = relaying[default]
#
# Will forward one event with the unique alert.classification.text,
# alert.source.node.address.address value combination to the 'default'
# instance of the 'relaying' reporting plugin. Further events with the
# same value will be suppressed for 3600 seconds.
#
#
# [thresholding]
# path = alert.classification.text, alert.source.node.address.address
# threshold = 3600
# count = 10
# hook = relaying[default]
#
# Will forward every tenth event per 3600 seconds with the unique
# alert.classification.text, alert.source.node.address.address value
# combination to the 'default' instance of the 'relaying' reporting
# plugin.
#
# Note that limit and threshold might be combined, allowing to setup a
# limit as soon as the first threshold is reached.



####################################
# Prelude generic configuration    #
####################################

# [prelude]
#
# This is the global prelude section, where you can define Prelude
# related options. Option of matter for Prelude-Manager, are, most
# specifically, in the context of relaying, the connection options:
#
# The following settings instruct the operating system when to consider
# a connection dead in case sent data is left unacknowledged.
#
# Theses option are operating system specific, and might not work on
# certain platform. In case you modify these settings on an unsupported
# system, a warning message will be issued when the agent starts.
#
# Under Linux, the default system wide configuration is:
# tcp-keepalive-time   = 7200
# tcp-keepalive-probes = 9
```

```
# tcp-keepalive-intvl  = 75
#
# tcp-keepalive-time represents the number of seconds the connection
# needs to be idle before TCP begins sending out keep-alive probes.
#
# tcp-keepalive-probes represent the number of not acknowledged probes
# to send before considering the connection dead.
#
# tcp-keepalive-intvl represents the interval between subsequent
# keepalive probes.
#
# The average time to notice a dead connection can be calculated using:
# tcp-keepalive-time + (tcp-keepalive-probes * tcp-keepalive-intvl)
#
# Here is an example configuration:
# tcp-keepalive-time   = 60
# tcp-keepalive-probes = 3
# tcp-keepalive-intvl  = 10
#
# Using the above settings, a dead connection will be detected within
# 90 seconds.
```

Anhang B

Portal Konfigurationsdateien

B.1 Apache2 Konfiguration

```
<VirtualHost portal.grid-ids.de:443>
SSLEngine On

SSLCertificateFile    /etc/apache2/sslcert/portal-zertifikat.pem
SSLCertificateKeyFile /etc/apache2/sslcert/portal-key.pem
SSLCACertificatePath /etc/apache2/sslcert/ca-certs/


    SSLVerifyClient require
    SSLVerifyDepth 10
    SSLUserName SSL_CLIENT_S_DN

ServerName      portal.grid-ids.de
DocumentRoot    /var/www/DjangoGIDS/htdocs

ErrorDocument 500 /500.html
ErrorDocument 404 /404.html

<Directory "/var/www/DjangoGIDS/htdocs">
AllowOverride   all
Order           deny,allow
        Allow           from all
</Directory>

<Directory />
Order allow,deny
Deny from all
AllowOverride none
</Directory>

<Location /gidsportal >
     SetHandler python-program
        PythonInterpreter demonstrator-gidsportal
        PythonHandler django.core.handlers.modpython
        SetEnv DJANGO_SETTINGS_MODULE gidsportal.settings
        PythonOption django.root /gidsportal
        PythonDebug On
        PythonPath "['/var/www/DjangoGIDS/gidsportal/src',
        '/var/www/DjangoGIDS/gidsportal/src/gidsportal/portal/prelude',
```

```
        '/usr/lib/python2.5',
        '/usr/lib/python2.5/site-packages/mod_python'] + sys.path"
</Location>

Alias /gidsportal/media/ /var/www/DjangoGIDS/gidsportal/media/
    <Directory /var/www/DjangoGIDS/gidsportal/media>
     AllowOverride None
        Order         allow,deny
        Allow         from all
    </Directory>
<Location /gidsportal/media>
     SetHandler None
    </Location>

Alias /gidsportal/adminmedia/
  /var/www/DjangoGIDS/gidsportal/lib/django/contrib/admin/media/
    <Directory /var/www/DjangoGIDS/gidsportal/lib/django/contrib/admin/media>
     AllowOverride None
        Order         allow,deny
        Allow         from all
    </Directory>
<Location /gidsportal/adminmedia>
     SetHandler None
    </Location>
</VirtualHost>
```

B.2 Django Konfiguration

```
# Django settings for DjangoGIDS project.

DEBUG = False
#TEMPLATE_DEBUG = DEBUG

ADMINS = (
    # ('<Your Name>', '<your_email@domain.com>'),
)

MANAGERS = ADMINS

DATABASES = {
    'gidsidmef': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'idmef',
        'USER': '**************',
        'PASSWORD': '**************',
        'HOST': '**************',
        'PORT': '3306',
        },

    'grrs': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'grrs_mirror',
        'USER': '**************',
        'PASSWORD': '**************',
        'HOST': '**************',
```

```
        'PORT': '3306',
    },
}

# Language code for this installation. All choices can be found here:
# http://www.i18nguy.com/unicode/language-identifiers.html
LANGUAGE_CODE = 'de-de'
DECIMAL_SEPARATOR = '.'

SITE_ID = 1

# If you set this to False, Django will make some optimizations so as not
# to load the internationalization machinery.
USE_I18N = True

# If you set this to False, Django will not format dates, numbers and
# calendars according to the current locale
USE_L10N = True

# Absolute path to the directory that holds media.
# Example: "/home/media/media.lawrence.com/"
MEDIA_ROOT = '/var/www/DjangoGIDS/gidsportal/media/'

# URL that handles the media served from MEDIA_ROOT. Make sure to use a
# trailing slash if there is a path component (optional in other cases).
# Examples: "http://media.lawrence.com", "http://example.com/media/"
MEDIA_URL = '/gidsportal/media'

#STATICFILES_URL = '/media/'

# URL prefix for admin media -- CSS, JavaScript and images.
# Make sure to use a trailing slash.
# Examples: "http://foo.com/media/", "/media/".
ADMIN_MEDIA_PREFIX = '/gidsportal/adminmedia/'

# Make this unique, and don't share it with anybody.
SECRET_KEY = '<Secrets should be secret>'

# List of callables that know how to import templates
# from various sources.
TEMPLATE_LOADERS = (
    'django.template.loaders.filesystem.Loader',
    'django.template.loaders.app_directories.Loader',
#     'django.template.loaders.eggs.Loader',
)

MIDDLEWARE_CLASSES = (
    'django.middleware.common.CommonMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
'django.middleware.csrf.CsrfResponseMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
)

ROOT_URLCONF = 'gidsportal.urls'
```

```
TEMPLATE_CONTEXT_PROCESSORS = (
# 'django.core.context_processors.request',
    'django.core.context_processors.auth',
#    'django.core.context_processors.debug',
    'django.core.context_processors.i18n',
    'django.core.context_processors.media'
#    'django.contrib.staticfiles.context_processors.staticfiles',
)

TEMPLATE_DIRS = (
    # Put strings here, like "/home/html/django_templates"
    # or "C:/www/django/templates".
    # Always use forward slashes, even on Windows.
    # Don't forget to use absolute paths, not relative paths.
    "/var/www/DjangoGIDS/templates",
"/var/www/DjangoGIDS/src/gidsportal/portal/prelude/templates"
)

INSTALLED_APPS = (
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    # 'django.contrib.sites',
    'django.contrib.messages',
    # Uncomment the next line to enable the admin:
    'django.contrib.admin',
    # Uncomment the next line to enable admin documentation:
    # 'django.contrib.admindocs',
'gidsportal.portal',
'gidsportal.grrs',
'gidsportal.gidsidmef'
)


DATABASE_ROUTERS = ['gidsportal.dbRouter.GIDSDBRouter']
```

Anhang C

Prelude-LML Konfigurationsdatei

```
############################################
# Configuration for the Prelude LML Sensor   #
############################################

include = @LIBPRELUDE_CONFIG_PREFIX@/default/idmef-client.conf


# Address where the Prelude Manager Server is listening on.
# if value is "127.0.0.1", the connection will occur throught
# an UNIX socket.
#
# This entry is disabled. The default is to use the entry
# located in the Prelude system wide clients.conf. You may
# overwrite the default address for this sensor by uncommenting
# this entry.
#
# [prelude]
# server-addr = 127.0.0.1


# FILES TO MONITOR
#
# You should define the log message prefix-regex and time-format within
# a [format] section. If not specified, the default syslog format will
# be used.
#
# The prefix-regex should contain PCRE named subpatterns to pick out the
# information available in your syslog's prefix.
#
# The available field names are:
#   - hostname
#   - process
#   - pid
#   - timestamp
#
# Please see pcrepattern(3) manpage for help writing the prefix-regex
# In order to set the time-format, please have a look at the strptime(3)
# manpage.
```

```
#
# Example configuration for syslog output:
#
# Each [format] section might have several file entry.
# Each [format] section might have several udp-server entry.
#
# If a file or udp-server entry might is listed accross differents
# formats, then the first matching format for a given log entry will be
# used.
#
# Additionally, you can specify a pattern in a file entry. LML will then
# searches for all the pathnames matching pattern according to the rules
# used by the shell (see glob(7)).
#
# Example: file = /var/log/*/*.log
#


# CHARACTER ENCODING
#
# For each files added to a format, a character encoding can be specified
# using the 'charset' option. Example:
#
# [format=MyFormat]
# charset = ISO-8859-1
# file = /var/log/log1
# file = /var/log/log2
# charset = UTF-8
# file = /var/log/log3
# file = /var/log/*.log
# udp-server = 0.0.0.0
#
# This will set the character set for 'log1' and 'log2' to ISO-8859-1, and
# to UTF-8 for 'log3', any files that match '/var/log/*.log', and any log
# entry read from the '0.0.0.0' integrated UDP server.
#
# Note that if no character encoding is specified, the system will attempt
# to automatically detect the encoding used. If the detection fail, then
# system wide default (retrieved from locale LC_CTYPE) will be used.
#

# ALTERING GENERATED IDMEF Events
#
# Within each format, you might use the 'idmef-alter' option to modify
# generated events:
#
# Example: idmef-alter = alert.analyzer(-1).node.location = MyLocation;
#
# Note that 'idmef-alter' will never overwrite an IDMEF path that is
# already set. Use 'idmef-alter-force' if this is what you intend to do.
#


[format=syslog]
time-format = "%b %d %H:%M:%S"
prefix-regex = "^(?P<timestamp>.{15}) (?P<hostname>\S+) (?:(?P<process>\S+?)
```

```
               (?:\[(?P<pid>[0-9]+)\])?: )?"
file = /var/log/messages
# udp-server = 0.0.0.0


#
# Sample configuration for metalog:
#
[format=metalog]
prefix-regex = "^(?P<timestamp>.{15}) \[(?P<process>\S+)\] "
time-format = "%b %d %H:%M:%S"
file = /var/log/everything/current
# udp-server = 0.0.0.0


#
# Sample configuration for apache:
#
[format=apache]
time-format = "%d/%b/%Y:%H:%M:%S"
prefix-regex = "(?P<hostname>\S+) \S+ \S+ \[(?P<timestamp>.{20}) [+-].{4}\] "
file = /var/log/httpd/access_log
file = /var/log/apache2/access_log

[format=apache-error]
time-format = "%a %b %d %H:%M:%S %Y"
prefix-regex = "^\[(?P<timestamp>.{24})\] \S+ (\[client (?P<hostname>\S+)\] )?"
file = /var/log/httpd/error_log
file = /var/log/apache2/error_log



#
# Sample configuration for asterisk:
#
#[format=asterisk]
#time-format = "%b %d %H:%M:%S"
#prefix-regex = "^(?P<timestamp>.{15}) (?P<hostname>\S+) (?:(?P<process>\S+?)
                (?:\[(?P<pid>[0-9]+)\])? (\S*): )?"
#file = /var/log/asterisk/messages


#
# Specifies the maximum difference, in seconds and/or size, between
# the interval of two logfiles' rotation. If this difference is reached,
# a high severity alert will be emited. The K (kbytes) or M (mbytes)
# suffix might be used for size definition.
#
#max-rotation-size-offset = 1024
#max-rotation-time-offset = 300


#
# Maximum number of warning a given source should emit in case it can
# not parse log entry with the provided prefix_regex and time_format.
#
```

```
# -1 == unlimited number of warning
#  0 == no warning at all
#  X == print at most X warnings.
#
# warning-limit = -1


####################################
# Here start plugins configuration #
####################################

[Pcre]

ruleset=@configdir@/ruleset/pcre.rules


# [Debug]
#
# This plugin issue an alert for each packet.
# Carefull to the loging activity it generate.
#
# Triger Report to the console.
# stderr
```

Anhang D

Prewikka Konfigurationsdatei

D.1 Prewikka Konfiguration

```
[general]
# Number of heartbeat to analyze in the heartbeat analysis view.
#heartbeat_count: 30

# If the offset between two heartbeat is off by more than the specified
# offset (in seconds), the analyzer will be represented as offline.
#heartbeat_error_margin: 3

# This setting tell Prewikka to not show the full exception when
# an error occur:
#disable_error_traceback

# Open external (references, IP lookup, and port lookup) links
# in a new windows.
external_link_new_window

# When a defined number of classification, source, or target exceed
# the default value (10), an expension link will be provided to lookup
# the remaining entry.
#
#max_aggregated_source: 10
#max_aggregated_target: 10
#max_aggregated_classification: 10

# Asynchronous DNS resolution (require twisted.names and twisted.internet)
#
# While rendering view containing address scheduled for asynchronous
# DNS resolution, it is possible that the rendering terminate too fast
# for all DNS requests to complete.
#
# The dns_max_delay setting determine Prewikka behavior:
# - [-1] No DNS resolution is performed.
# - [0] Do not wait, immediatly send results to the client.
# - [x] Wait at most x seconds, then send results to the client.
#
# dns_max_delay: 0


# Default locale to use (default is English):
```

```
# default_locale: fr

# Default encoding to use (default is UTF8):
# encoding: utf8


[interface]
software: Prewikka
place: company ltd.
title: Prelude console

[host_commands]
#
# You can use the $host variable that will be substituted with
# the source/target host value.
#
#MyCommand: /path/to/command <parameters>
#Command Title: /usr/bin/test -x $host -a

[idmef_database]
#
# if your database is a sqlite file, please use:
#
# type: sqlite3
# file: /path/to/your/sqlite_database
#
type: mysql
host: localhost
user: prelude
pass: prelude
name: prelude

[database]
type: mysql
host: localhost
user: prelude
pass: prelude
name: prewikka

# Standard login / password authentication:
[auth loginpassword]
expiration: 60
# If there is no user with administrative right defined in the database,
# the initial user will be created according to these settings:
initial_admin_user: admin
initial_admin_pass: admin


# Rely on webserver for user authentication:
#
# User that authenticate for the first time won't have any permission.
# If the "default_admin_user" option is provided, the specified user will
# be granted ALL access, allowing to edit other users permissions.
#
# [auth cgi]
# default_admin_user: myuser
```

```
# Disable Prewikka authentication:
# [auth anonymous]


# Logging configuration:
# - You can activate several log section.
# - Log level might be set to all/debug, info, warning, error, critical.
#   If unspecified, the default level is "warning".

# [log stderr]
# level: info

# [log file]
# level: debug
# file: /tmp/prewikka.log

# [log syslog]
# level: info

# [log nteventlog]
# level: info

# [log smtp]
# level: warning
# host: mail.domain.com
# from: user@address
# to: recipient1@address, recipient2@address, recipientN@address
# subject: Subject to use
```

D.2 Apache2

```
NameVirtualHost *:80
NameVirtualHost *:443

<VirtualHost *:80>
        RewriteEngine On
        RewriteCond %{HTTPS} !=on
        RewriteRule ^/(.*) https://%{SERVER_NAME}%{REQUEST_URI} [R]
</VirtualHost>

<VirtualHost *:443>
    SSLEngine On
    SSLCertificateFile /etc/apache2/ssl/apache.pem
 ServerName my.server.org
 Setenv PREWIKKA_CONFIG "/usr/local/etc/prewikka/prewikka.conf"

<Location "/">
        Options ExecCGI

        <IfModule mod_mime.c>
                AddHandler cgi-script .cgi
        </IfModule>
```

```
        Order allow,deny
        Allow from all
</Location>

Alias /prewikka/ /usr/local/share/prewikka/htdocs/
ScriptAlias / /usr/local/share/prewikka/cgi-bin/prewikka.cgi

</VirtualHost>
```

Anhang E

Prelude-Import Konfigurationsdatei

E.1 Konfigurationsdatei

```
#######
#######
### Config-File für die GIDS-Import-Routine
#######
#######

#########
# Spezielle Einstellungen für die Verbindung zum GIDS-Bus
#########
[Emcast]
emcast = true        ; Schaltet Emcast-Unterstützung ein.
                     ; Die Alarme werden dann an den GIDS-Bus weitergeleitet.

url = 224.1.2.3:1234 ; Die Broadcastadresse des GIDS-Bus

buffer = 16384       ; Die Größe des Buffers zum Empfang der Daten

loopback = false     ; Wenn Nachrichten, die an den GIDS-Bus versendet
                     ; werden sollen, auch an die absendende Site
                     ; geschickt werden soll, muss hier "true"
                     ; eingetragen werden. ACHTUNG: Diese Einstellung
                     ; kann bei unsachgemäßer Verwendung eine Schleife
                     ; und damit eine Überlastung produzieren!!!
```

E.2 Standardwerte

```
#define ANALYZER_NAME "gids-import"
#define ANALYZER_CLASS "Collector"
#define ANALYZER_MODEL "Prelude Import"
#define ANALYZER_MANUFACTURER "http://www.grid-ids.de"
#define ANALYZER_VERSION "0.9"
#define CONF_NAME "gids-import.conf"
#define EMCAST_URL "224.1.2.3:1234"
#define BUFLEN "10485760" /* Default Buffer Länge */
```

Anhang F

Prelude-Export Konfigurationsdatei

```
#######
#######
### Config-File für die GIDS-Export-Routine
#######
#######

[Anonymisieren]
Action.Inhalt = true                        ; D / LS
AdditionalData.Inhalt = true                ; D / LS
Address.address = true                      ; D / LS
Address.netmask = false                     ; LS
Address.vlan-name = false                   ; LS
Address.vlan-num = true                     ; D / LS
Analyzer.name = false                       ; LS
Analyzer.manufacturer = false               ; LS
Analyzer.model = false                      ; LS
Analyzer.version = false                    ; LS
Analyzer.ostype = false                     ; LS
Analyzer.osversion = false                  ; LS
Checksum.key = false                        ; LS
File.name = true                            ; D / LS
File.path = true                            ; D / LS
File.disk-size = false                      ; LS
File.fstype = false                         ; LS
File.file-type = false                      ; LS
FileAccess.Permission = false               ; LS
Impact.Inhalt = true                        ; D / LS
Linkage.name = true                         ; D / LS
Linkage.path = true                         ; D / LS
Node.location = false                       ; LS
Node.name = true                            ; D / LS
OverflowAlert.buffer = true                 ; D / LS
Process.name = false                        ; LS
Process.path = true                         ; D / LS
Process.arg = true                          ; D / LS
Process.env = true                          ; D / LS
Service.name = false                        ; LS
Service.port = false                        ; LS
```

```
Service.portlist = false                    ; LS
Service.protocol = false                    ; LS
Service.ip_version = false                  ; LS
Service.iana_protocol_number = false        ; LS
Service.iana_protocol_name = false          ; LS
SNMPService.oid = false                     ; LS
SNMPService.messageProcessingModel = false ; LS
SNMPService.securityModel = false           ; LS
SNMPService.securityName = false            ; LS
SNMPService.securityLevel = false           ; LS
SNMPService.contextName = false             ; LS
SNMPService.contextEngineID = false         ; LS
SNMPService.command = true                  ; D / LS
Source.interface = false                    ; LS
Target.interface = false                    ; LS
UserId.name = true                          ; D / LS
UserId.number = true                        ; D / LS
UserId.tty = false                          ; LS
WebService.arg = true                       ; D / LS
WebService.cgi = false                      ; LS
WebService.url = false                      ; LS

[SiteSpezifika]
gidsRessource = lrz ; Die gleiche Angabe wie in der GRRS-Datenbank
validUntil = 7 ; Anzahl der Tage, die die Meldung bei anderen Sites
               ; gespeichert werden kann.

[Prelude}
heartbeat = 600 ; Anzahl der Sekunden,
                ; nach denen ein Heartbeat geschickt werden soll.
timeout = 1 ; Schlafe eine vorgegebene Anzahl von Millisekunden
            ; zwischen der Abfrage,
            ; ob neue Meldungen im Nachrichtenpool vorhanden sind.
manager = 127.0.0.1 ; Die Adresse des Prelude-Managers,
                    ; von dem man die Nachrichten empfängt.
importName = gids-import ; Name des Sensors, der für den
                         ; Importvorgang aus dem GIDS-Bus zuständig ist.


#########
# Spezielle Einstellungen für die Verbindung zum GIDS-Bus
#########
[Emcast]
emcast = true ; Schaltet Emcast-Unterstützung ein.
              ; Die Alarme werden dann an den GIDS-Bus weitergeleitet.
url = 224.1.2.3:1234 ; Die Broadcastadresse des GIDS-Bus
loopback = true ; Wenn Nachrichten, die an den GIDS-Bus
                ; versendet werden sollen, auch an die absendende
                ; Site geschickt werden soll, muss hier
                ; "true" eingetragen werden. ACHTUNG: Diese
                ; Einstellung kann bei unsachgemäßer Verwendung
                ; eine Schleife und damit eine Überlastung produzieren!!!
```

Anhang G

Prelude-Correlator Konfigurationsdatei

```
# This is a template configuration file for prelude-correlator
#
[BruteForcePlugin]
disable = true
#

#[GIDSPlugin]
#disable = false

# Disable BusinessHour correlation by default since it is very verbose
[BusinessHourPlugin]
disable = true

[GIDSBrutePlugin]
disable = false
#debug = 3
name = "DFN-CERT Correlator"
id = "001:002:dfn-cert.de"

[GIDSPlugin]
disable = false
#debug = 3
name = "DFN-CERT Correlator"
id = "002:003:dfn-cert.de"

[GIDSHighPlugin]
disable = false
#debug = 3
name = "DFN-CERT Correlator"
id = "007:007:dfn-cert.de"

[MyPlugin]
disable = true

#
[OpenSSHAuthPlugin]
disable = true
#
```

```
[EventScanPlugin]
disable = true
#
[EventStormPlugin]
disable = true
#
[EventSweepPlugin]
disable = true
#
[WormPlugin]
disable = true
repeat-target = 5
#
[DshieldPlugin]
disable = true
#
# How often the Dshield database should be reloaded (download + reload)
# (default: once a week). 0 to disable reloading.
# reload  = 604800
#
# The server address where the Dshield database is loaded from:
# server  = www.dshield.org
#
# URI used to retrive the dshield database:
# uri     = /ipsascii.html?limit=10000
#
# Define the maximum allowed time for downloading the database
# (only work with Python >= 2.6, default is 10 seconds)
# timeout = 10

# This plugin will report CorrelationAlert for events / sets of events
# that appear to have passed through a firewall known to protect the
# target machine.
#
# If no firewall ever emit block concerning a given host, then this host
# is considered un-protected, and there is no point in reporting
# CorrelationAlert.
#
# The 'flush-protected-hosts' variable allow you to define how much
# time a given target hosts should be considered as protected when a
# firewall drop is noticed for this machine.
#
# The plugin is disabled by default since it tend to be very verbose

[FirewallPlugin]
disable = True
flush-protected-hosts = 3600


##
# Logging configuration might also be defined in this file:
# http://docs.python.org/library/logging.html
```

Anhang H

Beispielhafte Regelsätze für Snort

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 2222
  (msg:"ET SCAN Potential GSISSH Scan"; flags:S,12;
   threshold: type threshold, track by_src, count 5, seconds 120;
   reference:url,en.wikipedia.org/wiki/Brute_force_attack;
   reference:url,doc.emergingthreats.net/2001219;
   classtype:attempted-recon; sid:92001219; rev:18;)
alert tcp $HOME_NET any -> $EXTERNAL_NET 2222
  (msg:"ET SCAN Potential GSISSH Scan OUTBOUND"; flags:S,12;
   threshold: type threshold, track by_src, count 5, seconds 120;
   reference:url,en.wikipedia.org/wiki/Brute_force_attack;
   reference:url,doc.emergingthreats.net/2003068;
   classtype:attempted-recon; sid:92003068; rev:6;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 2222
  (msg:"ET SCAN LibSSH Based GSISSH Connection -
   Often used as a BruteForce Tool"; flow:established,to_server;
   content:"GSISSH-"; content:"libssh"; within:20;
   threshold: type limit, track by_src, count 1, seconds 30;
   reference:url,doc.emergingthreats.net/2006435;
   classtype:misc-activity; sid:92006435; rev:6;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 2222
  (msg:"ET SCAN LibSSH Based Frequent GSISSH Connections
   Likely BruteForce Attack!"; flow:established,to_server;
   content:"GSISSH-"; content:"libssh"; within:20;
   threshold: type both, count 5, seconds 30, track by_src;
   reference:url,doc.emergingthreats.net/2006546;
   classtype:attempted-admin; sid:92006546; rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 2222
  (msg:"GPL SCAN GSISSH Version map attempt";
  flow:to_server,established; content:"Version_Mapper"; nocase;
  classtype:network-scan; sid:92101638; rev:7;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 2222
  (msg:"GPL SCAN gsissh-research-scanner"; flow:to_server,established;
  content:"|00 00 00|‘|00 00 00 00 00 00 00 00 01 00 00 00|";
  classtype:attempted-recon; sid:9617; rev:5;)
```

Anhang I

Cronjob zum Löschen von Meldungen nach dem Datenschutzkonzept

```
#!/bin/sh

###
# Nötige Pfad-Variable, um das Programm "preludedb-admin" zu finden.
###
PATH="/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin"

###
# Nötige Variablen, um die Datenbank anzusprechen.
###

DB_TYPE="mysql"     # Typ der Datenbank. Kann [mysql|pgsql|sqlite] sein.
DB_HOST="localhost" # IP-Adresse des Datenbankservers oder localhost.
DB_PORT="3306"      # Port, an dem der Datenbankserver lauscht.
DB_NAME="prelude"   # Name der Datenbank.
DB_USER="prelude"   # Benutzername, der benutzt wird,
                    # um auf die Datenbank zuzugreifen.
DB_PASS="xxx"       # Das zum obigen Benutzername passende Passwort.

###
# Datenschutzrichtline.
###

KEEP_INTERVAL_ALERT="7 days"
                    # Angabe, nach wie lange spätestens ein Alarm
                    # gelöscht werden soll.
                    # Kann auf [1 day|[2-..] days|1 month|[2-..]months|..]
                    # gesetzt werden.
KEEP_INTERVAL_HEARTBEAT="1 day"
                    # Angabe, nach wie lange spätestens ein Heartbeat
                    # gelöscht werden soll.
                    # Kann auf [1 day|[2-..] days|1 month|[2-..]months|..]
                    # gesetzt werden.
```

```
################################
###
# Never change across this line!
###
################################

###
# Umwandeln der in der Datenschutzrichtlinie gemachten Angaben in Strings,
# die von Prelude interpretiert werden können.
###

DATE_ALERT=$(date -d "now - $KEEP_INTERVAL_ALERT" +%Y-%m-%d)
DATE_HEARTBEAT=$(date -d "now - $KEEP_INTERVAL_HEARTBEAT" +%Y-%m-%d)
DATE_EXTERN=$(date -d "now" +%Y-%m-%d)

###
# Löschen der Einträge aus der Datenbank.
###

echo "Lösche Heartbeat-Nachrichten..." 1>&2
preludedb-admin delete heartbeat
  "type=$DB_TYPE host=$DB_HOST port=$DB_PORT name=$DB_NAME
  user=$DB_USER pass=$DB_PASS"
  --criteria "heartbeat.create_time <= $DATE_HEARTBEAT"
echo "" 1>&2
echo "Lösche Alarmmeldungen nach der lokalen Datenschutzrichtlinie..." 1>&2
preludedb-admin delete alert
  "type=$DB_TYPE host=$DB_HOST port=$DB_PORT name=$DB_NAME
  user=$DB_USER pass=$DB_PASS"
  --criteria "alert.create_time <= $DATE_ALERT"
echo "" 1>&2
echo "Lösche Alarmmeldungen nach der externen Datenschutzrichtlinie..." 1>&2
preludedb-admin delete alert
  "type=$DB_TYPE host=$DB_HOST port=$DB_PORT name=$DB_NAME
  user=$DB_USER pass=$DB_PASS"
  --criteria "alert.additional_data.meaning = 'gids_validUntil'
  && alert.additional_data.data <= $DATE_EXTERN"
echo "" 1>&2
echo "That's all Folks!" 1>&2
```

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] Wolfgang Hommel, Nils gentschen Felde, Felix von Eye, Jan Kohlrausch, and Christian Szongott. Architekturkonzept fur ein Grid-basiertes IDS. Meilensteinbericht, D-Grid, October 2010.

[2] Wolfgang Hommel, Nils gentschen Felde, Felix von Eye, Jan Kohlrausch, and Christian Szongott. Datenschutzmodell für ein Grid-basiertes IDS. Meilensteinbericht, D-Grid, July 2010.

[3] Wolfgang Hommel, Nils gentschen Felde, Felix von Eye, Jan Kohlrausch, and Christian Szongott. Grobskizze einer Architektur. Meilensteinbericht, D-Grid, April 2010.

[4] Helmut Reiser, Nils gentschen Felde, Felix von Eye, Jan Kohlrausch, and Christian Szongott. Anforderungs- und Kritierienkatalog (MS 6). Meilensteinbericht, D-Grid, January 2010.