Hochschule für Technik, Wirtschaft und Kultur Leipzig (FH)
Fachbereich Informatik, Mathematik und Naturwissenschaften

DIPLOMARBEIT

Konzeption einer einheitlichen Plattform für die Protokolldatenanalyse

eingereicht von: Thomas Steinbach
geboren am 07. April 1976
in Meerane

Betreuer: Prof. Dr. rer. nat. K. Hänßgen
Dipl.–Inf. S. Planert

Leipzig, 28. August 2002

KONZEPTION EINER EINHEITLICHEN PLATTFORM FÜR DIE PROTOKOLLDATENANALYSE

Anforderung

Als Geschäftsstelle Sachsen, in der T–Systems GEI GmbH, sind wir mit über 400 Mitarbeitern an 5 Standorten in Deutschland präsent. Unser Kerngeschäft liegt dabei in der Entwicklung und Einführung individueller Softwarelösungen.

Durch die dezentrale Organisation der Geschäftsstelle Sachsen ist eine aufwendige Serverinfrastruktur entstanden. Sie bildet das technische Rückgrat unserer Geschäftsaktivitäten.

Die auf den Servern anfallenden Protokolldaten werden nur teilweise und mit großem Aufwand ausgewertet.

Im Rahmen dieser Diplomarbeit ist ein Konzept zu entwerfen, welches unter Beachtung der rechtlichen Rahmenbedingungen und mit geringer Belastung der bestehenden Infrastruktur die anfallenden Protokolldaten für eine zentrale Auswertung bereitstellt. Insbesondere ist dabei die Sicherheit der Protokolldaten und deren Aufbewahrung nach einer Analyse zu betrachten. Ferner soll das System die Möglichkeit zur Anpassung an eine sich ändernde Infrastruktur besitzen und über Benachrichtigungsmöglichkeiten für Administratoren verfügen.

Nach dem Entwurf eines geeigneten Systems ist dieses prototypisch für den Standort Leipzig zu implementieren und in Betrieb zu nehmen. Die Grundlage für dieses System soll das Betriebssystem Linux bilden.

Betreuender Hochschullehrer:	Prof. Dr. rer. nat. K. Hänßgen
Fachbereich:	Informatik, Mathematik und Naturwissenschaften
Betreuer:	Dipl.–Inf. S. Planert
Beginn:	10. Juni 2002
Einzureichen bis:	10. September 2002

Kurzreferat

Bibliographische Bezeichnung

Konzeption einer einheitlichen Plattform für die Protokolldatenanalyse.
Thomas Steinbach, 2002 - 148 S., 20 Abb., 9 Tab., 45 Lit.,
Hochschule für Technik, Wirtschaft und Kultur Leipzig (FH),
Fachbereich Informatik, Mathematik und Naturwissenschaften, Diplomarbeit.
Stichworte: Protokolldaten, Logfiles, Protokollserver

Zusammenfassung

Die im täglichen Betrieb einer komplexen IT–Infrastruktur anfallenden Protokolldaten werfen zahlreiche Probleme auf. Zum einen existieren zahlreiche rechtliche Gegebenheiten, welche deren Erzeugung, Auswertung und Aufbewahrung reglementieren, und zum anderen gibt es verschiedene Anforderungen an eine Auswertung dieser Daten.

Das vorgestellte Konzept bietet einen Ansatz, diese Probleme zu lösen. Dabei werden sowohl die rechtlichen als auch die technischen Aspekte zur Bereitstellung der Protokolldaten für eine Auswertung betrachtet. Insbesondere werden dabei die Anonymisierung vor und die Aufbewahrung nach der Analyse der Daten erläutert.

Für die Realisierung wurden freie Open Source Softwareprodukte verwendet und mit deren Hilfe ein funktionsfähiges System zur Anonymisierung, Sicherung, Bereitstellung zur Analyse und zur Aufbewahrung der Protokolldaten entwickelt. Das System lässt sich in eine bestehende Infrastruktur integrieren.

Für Sylvia und Ruben

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Geleitworte

Die vorliegende Diplomarbeit wäre ohne die engagierte Mitarbeit vieler anderer nicht so entstanden. Daher möchte ich hier die Gelegenheit nutzen, all jenen meinen herzlichen Dank zu sagen. Diese Worte richten sich insbesondere an Prof. Dr. K. Hänßgen und S. Planert, welche die Arbeit betreut, unterstützt und sich für die Einhaltung der zeitlichen Vorgaben eingesetzt haben.
Bedanken möchte ich mich weiter bei allen, von denen ich bei der Erarbeitung der Konzeption mit Anregungen und Kritik unterstützt wurde. Besonders sind hierbei Michael Weiser, Steffen Wolf, Thomas Pönitzsch, Heiko Jehmlich und Uwe Wendt zu nennen.
Für die Korrektur und zahlreiche Anmerkungen zur Gestaltung bedanke ich mich bei Ulrike Schauer, Katja Maischatz, Monique Brögge und Herrn Gerd Kolitsch.
Nicht zuletzt möchte ich mich bei meiner Familie bedanken: Bei meiner Frau Sylvia, welche mir die zeitlichen Freiräume geschaffen hat und Rückhalt zum schreiben dieser Arbeit gab, und unserem Sohn Ruben, der mich immer wieder aus den verschiedensten Schreibgefechten befreite.

Leipzig, im August 2002 Thomas Steinbach

Kapitel 1

Einleitung

Die Erstellung von Protokolldaten ist für den reibungslosen Betrieb von IT-Systemen unerlässlich. Deshalb bieten heutige Systeme Funktionen zum Protokollieren.

Von Interesse sind diese Daten aus verschiedenen Gründen. Es lassen sich Nutzerprofile für Werbezwecke erstellen. Systemfehler und -einbrüche können frühzeitig erkannt werden. Die Erstellung von Statistiken zur Auslastung stellt ein wichtiges Instrument für die Erkennung notwendiger Erweiterungen und Wartung der Systeme dar.

Der Gesetzgeber hat diese Nutzung jedoch mit einer Vielzahl von Einschränkungen versehen. Dazu gehört insbesondere der Bereich des Datenschutzes bzw. des Schutzes der informationellen Selbstbestimmung (vgl. Kapitel 3). Weitere Probleme verursacht die Menge der anfallenden Informationen und deren Auswertung.

Das komplexe Thema der Protokolldaten wird oft unterschätzt, da sich in den Dateien eine Vielzahl an Informationen befindet, deren Kenntnis und Verwendung einen entscheidenden Vorsprung verschaffen kann.
Die vorliegende Arbeit beschreibt den Aufbau eines Systems, welches unter Berücksichtigung der oben genannten Einschränkungen eine Plattform bereitstellt, die eine Auswertung der Protokolldaten ermöglicht.

Zur Einführung in die Thematik der Protokolldaten werden in den Kapiteln 2 bis 5 diesbezügliche Grundlagen dargestellt. Im zweiten Teil, ab Kapitel 6, wird der Prototyp mit seinen speziellen Anforderungen und Voraussetzungen entwickelt

und ein Konzept sowie dessen Implementation vorgestellt. Für das Verständnis des Prototypen ist die Lektüre ab Kapitel 6 zu empfehlen, an den entsprechenden Stellen wird Bezug auf die Grundlagen genommen. Um jedoch einen umfassenden Einblick in das Thema zu bekommen, ist es zu empfehlen sich ab Teil I, Kapitel 2 zu informieren.

Im Kapitel 2 wird eine Möglichkeit der Einteilung von Protokolldateneinträgen nach Gruppen aufgezeigt. Im Anschluss daran werden einige relevante Dienste und deren Protokolldaten erläutert. Das Kapitel endet mit einer Beschreibung verschiedener Herangehensweisen zur Auswertung von Protokolldaten.

Mit den durch den Gesetzgeber festgelegten rechtlichen Rahmenbedingungen in Bezug auf Protokolldaten setzt sich Kapitel 3 auseinander. Betrachtet werden neben zahlreichen Gesetzen auch einige in diesem Zusammenhang wichtige Verordnungen sowie deren Interpretationen bzw. Interpretationsmöglichkeiten.

Durch die große Menge an sensiblen Informationen innerhalb der Protokolldaten sind diese besonderen Bedrohungen ausgesetzt. In Kapitel 4 werden einige „Angriffsszenarien“ beschrieben und den typischen Angreifertypen zugeordnet. Neben diesen Bedrohungsszenarien werden auch die Gruppen, aus denen diese Angreifer stammen, beschrieben.

Für die Sicherstellung der Integrität und Vertraulichkeit der Protokolldaten sind einige Kenntnisse zu den kryptographischen Grundlagen notwendig, welche in Kapitel 5 erläutert sind. In diesem Zusammenhang werden die Hashfunktionen MD5 und SHA1 sowie das asymmetrische Verschlüsselungsprotokoll mit öffentlichem Schlüssel nach RSA beschrieben.

Nach diesen Grundlagen werden in Kapitel 6 die Anforderungen, die an den Prototypen gestellt werden, beschrieben. Besonderes Augenmerk liegt dabei auf den Fragestellungen potenzieller Nutzer, welche vom System zu berücksichtigen sind.

Um den Erfordernissen der Entwicklung und des Einsatzes gerecht zu werden, sind einige Voraussetzungen zu erfüllen. Diese sind in Kapitel 7 zusammengefasst.

Das Konzept, welches in Kapitel 8 entwickelt wird, ist der Kern des Prototypen.

Hier werden die Vor- und Nachteile verschiedener Detailprobleme erläutert und die zu implementierenden Lösungen festgelegt. In einem Abschnitt zur Konfiguration werden die Anpassungsmöglichkeiten an spezielle Bedürfnisse der Infrastruktur dargestellt.

Die verschiedenen Komponenten des Prototypen, deren Arbeitsweise und Kommunikation untereinanderer werden in Kapitel 9 beschrieben.

In Kapitel 10 werden die neben der Software des Prototypen wichtigen vertraglichen Vorarbeiten für den Einsatz des Systems dargestellt. Durch diese Regelungen kann ein rechtlich einwandfreier Status für den Betrieb des Systems erreicht werden. Weiterhin wird die Leistungsfähigkeit des Systems durch Tests dokumentiert.

Die abschließende Betrachtung in Kapitel 11 gibt eine kurze Zusammenfassung zum Stand der Implementierung und den daraus resultierenden weiteren Aufgaben sowie mögliche Erweiterungen des Systems.

Im Anhang befinden sich die Protokolle der durchgeführten Tests und spezielle Informationen zur Konfiguration des Testsystems. Der zur vorliegenden Arbeit gehörende Datenträger umfaßt den Quelltext des gesamten Prototyps, eine elektronische Version dieser Arbeit, sämtliche Konfigurationsdateien und die Module, welche für den Einsatz des Systems notwendig sind.

Teil I

Theoretische Grundlagen

Kapitel 2

Inhalt, Erzeugung und Auswertung von Protokolldaten

2.1 Inhalt der Protokolldaten

Die Protokolldaten umfassen eine Vielzahl von Informationen, welche beim Betrieb eines Systems entstehen. Nahezu alle Dienste, die ein System bereitstellt, erstellen Meldungen in Protokolldateien.

Die Einträge in den Protokolldaten lassen sich in drei Gruppen einteilen. Diese Aufteilung wird durch Abbildung 2.1 verdeutlicht. Es gibt dienstspezifische Meldungen (Statusmeldungen), welche den Betrieb des Dienstes betreffen. Dazu gehören zum Beispiel das Starten und Beenden des Dienstes. Meldungen, welche Daten enthalten, die durch den Betrieb des Dienstes entstehen, werden als Ereignisse bezeichnet. Hierzu gehören Informationen über ausgelöste Anfragen, gesendete und empfangene Antworten und Verbindungsinformationen. Die dritte Gruppe umfasst alle beim Betrieb eines Dienstes auftretenden Unregelmäßigkeiten (Fehlermeldungen), insbesondere Fehler, welche durch den Dienst selbst oder dessen Benutzung ausgelöst wurden und Fehler, welche durch das Einwirken Dritter entstanden. Die letztgenannten werden als Angriffe bezeichnet. In Abhängigkeit von den Verfahren der Auswertung kann sich die Gruppenzugehörigkeit der Einträge verändern.

Die verschiedenen Arten von Meldungen können neben den Einträgen in Protokolldateien und der jeweiligen Auswirkung auf den Dienst auch Auswirkungen auf andere Dienste haben. Dabei können sich diese Dienste auf demselben System

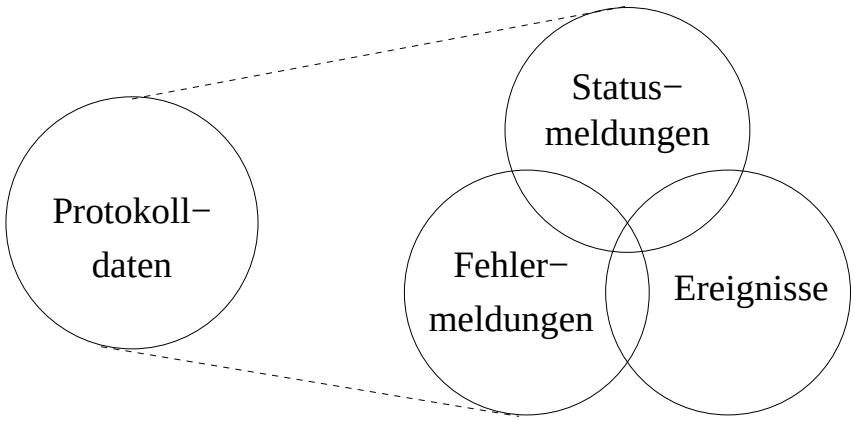

Abbildung 2.1: Aufteilung der Einträge in Protokolldateien in Gruppen

befinden oder auch auf entfernten Systemen, welche via Netzwerk miteinander verbunden sind. Meldungen, aus denen Angriffe oder Angriffversuche ersichtlich werden, können sich sowohl in den dienstspezifischen Meldungen als auch in den durch den Betrieb des Dienstes erzeugten Meldungen befinden.

Die Art der von den beeinflussten Diensten erzeugten Einträge kann sich dabei von der Art der Aktion des beeinflussenden Dienstes unterscheiden. So ist es möglich, dass Dienste auf Ereignissen in anderen Diensten mit Status– oder Fehlermeldungen reagieren. Ebenso können Fehler mit Statusmeldungen oder Ereignissen beantwortet werden.

2.1.1 Statusmeldungen

Bei Statusmeldungen handelt es sich um Einträge in den Protokolldaten, welche durch den Betrieb von Diensten entstehen. Diese enthalten Informationen zum Starten und Beenden des Dienstes, dem Laden und Entfernen von Modulen, dem Lesen von Konfigurationsdaten und ähnliches. Dabei können Aktionen, welche ein Dienst ausführt, zu Reaktionen anderer Dienste auf dem System führen. Ein Beispiel hierfür ist das Entfernen der Netzverbindung durch Beenden des Netzdienstes. Alle Dienste, welche die Netzverbindung benötigen, werden eine entsprechende Meldung erzeugen. Periodische Meldungen sind eine weitere Form von Statusmeldungen. Bei diesen Einträgen in die Protokolldateien wird zum Beispiel in regelmäßigen Abständen das ordnungsgemäße Funktionieren eines Dienstes gemeldet.

Eine Aktion eines Dienstes kann auch auf anderen Systemen Reaktionen auslösen. So bewirkt das Beenden des Domain Name Service (DNS) Dienstes, dass alle Anfragen zum Beispiel zur Auflösung eines symbolischen Namens für eine FTP–Verbindung scheitern. Der FTP–Dienst vermerkt eine entsprechende Meldung in seiner Protokolldatei. Durch die Abhängigkeiten, welche mitunter zwischen den Diensten auf verschiedenen Systemen in einem Netzwerk entstehen, kann es bei Reaktionen der Dienste verschiedene Ursachen geben. Eine eindeutige Zuordung von Reaktionen zu Ursachen kann nicht immer gefunden werden.

2.1.2 Ereignisse

Die in den Protokolldaten vermerkten Meldungen, welche durch Interaktionen eines Dienstes mit Benutzern bzw. mit anderen Diensten entstehen, werden als Ereignisse bezeichnet. Die Ereignisse sind somit durch den normalen Betrieb eines Dienstes anfallende Nutzdaten. Sie umfassen Verbindungs- und Benutzerinformationen, aber auch die Zeit und ggf. die Dauer von Anfragen und deren Beantwortung.

Wie bereits die Statusmeldungen können auch die Anfragen an einen Dienst Auswirkungen auf andere Dienste haben. Dabei hängt die Art des Eintrags in die Protokolldatei eines beeinflussten Dienstes vom Status dieses Dienstes und dem Aussehen der Anfrage ab. Wird ein *Proxy-cache* (vgl. Kapitel 2.2.2) zum Beispiel mit einer Anfrage nach einer bestimmten Internetseite beauftragt, so hat das Ereignis der Anfrage am *Proxy-Cache* ein Ereignis beim zuständigen *Webserver* (vgl. Kapitel 2.2.2) zur Folge.

2.1.3 Fehlermeldungen

Unter Fehlermeldungen werden alle Einträge in Protokolldateien zusammengefasst, welche eine Abweichung vom definierten Verhalten des Dienstes beinhalten. Somit sind Einträge aufgrund falsch formulierter Anfragen ebenso Fehlermeldungen wie durch Soft- oder Hardwareprobleme ausgelöste Meldungen.

Fehlermeldungen können Anhaltspunkte für Sicherheitslücken in Diensten oder dem gesamten System geben. Während Ereignisse und Statusmeldungen eher allgemeine Informationen beinhalten, welche z. B. für Statistiken von Interesse sind, finden sich unter den Fehlermeldungen die für den Systemverwalter wichtigen Hin-

weise auf Probleme. Diesen Meldungen kommt deshalb eine besondere Bedeutung zu, da häufig erst durch den Bezug zu einer Fehlermeldung die Relevanz von Einträgen unter den Ereignissen bzw. Statusmeldungen ersichtlich wird.

2.2 Erzeuger von Protokolldaten

Die verschiedenartigsten Dienste und Betriebssysteme erstellen Protokolldateien. Diese dienen der Nachvollziehbarkeit der Systemabläufe und Fehlerkontrolle. Häufig ist es möglich, den Umfang dieser Protokollierung festzulegen, d. h. es kann definiert werden, ob und welche Ablaufdaten ins Protokoll geschrieben oder ob nur bestimmte Ereignisse aufgenommen werden.

Nachfolgend wird ein Überblick über wichtige Dienste und deren Funktion gegeben. Aus diesen Daten lassen sich Bedrohungsszenarien und zu ergreifende Maßnahmen ableiten. Diese Szenarien werden im Kapitel 4 ab Seite 35 genauer betrachtet. Aufgrund der Vielzahl von Diensten erhebt diese Aufstellung keinen Anspruch auf Vollständigkeit.

2.2.1 Betriebssysteme

Viele Betriebssysteme stellen umfangreiche Funktionen zum Festhalten von Protokollinformationen zur Verfügung. Besonders Betriebssysteme für Server erlauben das Protokollieren von systemspezifischen Informationen.

Die Eigenschaften (Art, Anzahl, Aufbewahrungsfrist u. ä.) der Einträge in den Protokolldaten kann durch Konfiguration festgelegt werden. Dabei werden insbesondere Meldungen des Systemkerns protokolliert. Diese enthalten Informationen über das Starten und Beenden systemspezifischer Dienste, das Einbinden und Entfernen von Systemtreibern, Fehlermeldungen und andere den Ablauf des Systems betreffende Meldungen.

Eine Beschreibung der Protokollierungsmöglichkeiten eines Betriebssystems befindet sich in dessen Dokumentation. Das Dateiformat der Protokolldaten unterscheidet sich zwischen den Betriebssystemen. Einige Systeme legen Textdateien an, so die verschiedenen Varianten des Linux Systems. Andere erzeugen Dateien in einem speziellen Format, welche nur über spezielle Programme eingesehen werden

können; Beispiele hierfür sind die verschiedenen Windowsvarianten und Solaris.

2.2.2 Dienste

Neben den von Betriebssystemen erzeugten Protokolldaten gibt es zahlreiche Dienste, welche ebenfalls Informationen in Protokolldateien speichern. Diese Daten enthalten je nach Dienst verschiedene Informationen. Ähnlich den Betriebssystemen lässt sich bei verschiedenen Diensten auch die Detailtiefe der Meldungen einstellen.

Einige Beispiele für Dienste, die aus Sicht der Protokolldaten von Interesse sind, werden im Folgenden gegeben. Dabei erhebt die Aufstellung keinen Anspruch auf Vollständigkeit; die Dienste wurden sowohl mit Blick auf die Anforderung an die Implementierung und die Relevanz im Netzwerk ausgewählt.

Network File System (NFS)

Das NFS ist ein Protokoll für ein verteiltes Dateisystem. Es bietet umfangreiche Möglichkeiten zum Zugriff auf Dateien von und auf fernen Rechnern. Dabei werden zahlreiche Funktionen zur Verfügung gestellt, welche unter anderem die Zugriffsrechte sichern und das Zwischenspeichern überwachen.

Die aktuelle Protokollversion 4 wurde gegenüber früheren Versionen im Hinblick auf Sicherheits- und Geschwindigkeitsfragen verbessert. Ein Server bietet das NFS als Dienst an. Dies bedeutet, dass verschiedene Dateisysteme (z. B. Nutzerverzeichnisse, Laufwerke o. ä.) für die entfernte Nutzung auf Rechnern in einem Computernetzwerk freigegeben werden. Dabei ist es möglich, die Nutzung nur für bestimmte Hosts zuzulassen. Des Weiteren können nutzerbezogene Berechtigungen für die Freigaben definiert werden, so beispielsweise Nur-Leseberechtigung für Verzeichnisse mit Softwarepaketen. Die Verzeichnisse bleiben bis zur Trennung verbunden.

In den Protokolldaten des Servers finden sich Informationen darüber, welche Hosts auf Verzeichnisse zugreifen, d. h. diese verbunden haben. Weiterhin finden sich Informationen über fehlerhafte Zugriffe bzw. Zugriffsversuche durch nicht authorisierte Benutzer und Hosts. Neben diesen Informationen wird die Uhrzeit des Ereignisses protokolliert.

Auf der Seite des Hosts werden Informationen über den korrekten Verbindungsaufbau, mögliche Ausfälle in der Verbindung zum Server sowie Fehlermeldungen durch unauthorisierten Zugriff protokolliert.

Eine Spezifikation des NFS Protokolls findet sich im RFC3010 [Sea00].

Samba

Samba ist eine Implementation des SMB–Protokoll (Server Message Block–Protokoll). Dieses Protokoll definiert den gemeinsamen Zugriff von Systemen auf im Netzwerk verfügbare Resourcen. Von *Samba* wird des Weiteren sowohl die Koordination der Kommunikation, als auch die Bereitstellung derartiger Ressourcen unterstützt. In verschiedenen Betriebssystemen wird das SMB–Protokoll genutzt, um den gemeinsamen Zugriff auf Laufwerke und Drucker zu gestatten. Die Authentifizierung von Benutzern kann über dieses Protokoll realisiert werden.

Weitere Informationen zu *Samba* finden sich auf den Internetseiten des *Samba*–Projektes unter [Sam02]. Eine detailierte Beschreibung zur *Samba*–Installation unter UNIX–Systemen befindet sich unter [WS02].

Internet proxy-cache

Ein *Proxy* ist ein Dienst, welcher stellvertretend für andere Aktionen durchführt. Ein *Cache* speichert Daten für einen wiederholten Zugriff zwischen.

Mit anderen Worten, durch einen Proxy wird der Zugriff eines Programmes auf einen Dienst (z. B. World Wide Web) realisiert. Dabei greift die Anwendung des Nutzers nicht direkt auf den Dienst zu, sondern auf das Proxy-Programm. Ein *Cache* prüft, ob die Anfrage gegebenenfalls aus den zwischengepufferten Daten zu beantworten ist, oder ob die Anfrage an den Dienst weitergeleitet werden muss. Gepuffert werden Bilder, Tondaten, Texte, Programme und andere Dateien.

Ein bekannter Vertreter der *Proxy–Cache*–Programme ist der *Squid*–Cache[1]. Er speichert die Antwortdaten, welche aus den Nutzeranfragen resultieren, zwischen.

[1] Der Squid ist ein Internet–Cache mit Proxy–Funktionalität

Dadurch können sich wiederholende Anfragen schneller beantwortet werden. Dieses Verfahren spart sowohl Zeit als auch Kosten, da die Daten nicht erneut übertragen werden, d. h. keine externe Verbindung notwendig ist. Der *Squid* bietet neben der Unterstützung von verschiedenen Protokollen die Möglichkeit, Hierarchien von *Proxys* aufzubauen. Die Funktionalität wird jedoch durch die Notwendigkeit, alle Anfragen im HTTP (Hypertext Transfer Protokoll) zu formulieren, eingeschränkt. Die Nutzerprogramme müssen diese Funktionalität unterstützen, um über den *Squid* zu kommunizieren.

Detaillierte Informationen zum *Squid proxy–cache* befinden sich in der Internetpräsenz [SQU02].

In den Protokolldateien des *Squid* sind neben Statusinformationen vor allem Meldungen bzgl. Nutzeranfragen gespeichert, d. h. zum Beispiel welche Seiten im World Wide Web von wem angefragt wurden. Aus den Daten lassen sich Informationen über Art (Text, Bilder, Ton etc.) und Umfang der Anfragen ableiten.

Aufgrund seiner Verbreitung existieren zahlreiche Programme zur statistischen Aufbereitung der Protokolldaten des *Squid*. Zu einigen finden sich Verweise auf der oben genannten Internetseite.

Webserver

Ein Webserver ist ein Programm bzw. Dienst, welcher die Aufgabe hat, eingehende Anfragen in Datei- oder Programmnamen umzuwandeln. Die Dateien werden an den Anfragenden gesendet, die Programme werden ausgeführt und deren Ausgaben an den Anfragenden geschickt [LL99].

Hierfür werden vom Anwenderprogramm sogenannte Universal Resource Lokatoren (URL)[2] in eine Anfrage umgewandelt.

Ein URL besteht aus drei Teilen: dem *Protokoll*, welches den Kommunikationskanal zum *Server* definiert, dem *Server*, welcher der Adressat der Kommunikation ist, und dem Programm oder der Datei, welche als *Pfad* übermittelt werden. Abbildung 2.2 verdeutlicht diesen Aufbau.

[2] In der Literatur findet sich auch häufig die Bezeichnung *Uniform Resource Locator*

```
<Protokoll>://<Server>/[Pfad]
```

Abbildung 2.2: Aufbau eines Universal Resource Locator.

Das einfachste Beispiel für den Umgang mit Webservern ist das „Surfen“ durch das World Wide Web[3] (WWW). Dabei werden Anfragen als URL formuliert und abgeschickt. Der angesprochene *Server* versucht den *Pfad* aufzulösen und liefert die gewünschte Seite (Datei) zurück.

Ein verbreiteter Webserver ist der *Apache*. Der Name *Apache* steht für „A PAtCHy server“. Dieser Name ist durch seine Entstehung begründet, da der *Apache* aus vorhandenem Code und einigen Patches entwickelt wurde. Die Entwicklungsgrundlage war der *NCSA httpd 1.3*. Weitere Informationen zu diesem Webserver finden sich unter [NCS02].

Der *Apache* bietet sowohl eine große Leistungsfähigkeit [WEB02], als auch eine hohe Anzahl an Funktionen. Die wichtigste Funktion ist die Möglichkeit, verschiedene Module einzubinden. So ist es möglich, über ein Modul gängige Schreibfehler in den Pfadangaben der Anfragen zu korrigieren. Weiterhin gibt es zahlreiche Module zur Authentifizierung von Nutzern und zum sicheren Austausch von Informationen durch verschlüsselte Kommunikation.

In den Protokolldaten finden sich Informationen über das Starten und Beenden von Modulen, die gestellten Anfragen und aufgetretenen Fehler. Die Daten zu Anfragen enthalten neben der Zeit die Pfadangabe und ggf. dazugehörige Parameter. Weiterhin befindet sich auch die Internet–Adresse (IP–Adresse) des Dienstnutzers in den Einträgen. Weitere Informationen zum *Apache* Webserver finden sich unter [APA02]. Dort befinden sich auch Dokumentationen zu den Modulen bzw. Verweise auf entsprechende Entwickler.

[3] An dieser Stelle sei darauf verwiesen, dass sich der Begriff „Durch das Internet Surfen“ nur auf das World Wide Web bezieht. Das Internet bietet jedoch einer Vielzahl von Diensten (Email, FTP, News, etc.) von denen das WWW nur ein Vertreter ist.

Domain Name Service

Ziel des Domain Name Service (DNS) ist die Bereitstellung eines geeigneten Namensraumes, so dass die verwendeten Namen in verschiedenen Systemen, Netzwerken, Protokollen und administrativen Einheiten einsetzbar sind. Zur Erfüllung dieser Aufgabe werden symbolische Namen statt der durch die Protokolle bedingten Adressen eingesetzt. Der DNS übernimmt die Zuordnung eines symbolischen Namens zu einer Adresse.

Das übliche Aussehen eines solchen symbolischen Namen ist:

```
beispiel.intra.netzwerk.org
```

Dabei ist *beispiel* der Hostname, *intra* der Name der Subdomain, *netzwerk* der Name für die Second–Level Domain und *org* die Bezeichnung der Top–Level Domain.

Auf diese Art werden die maschinenlesbaren Adressen in eine für die Benutzer verständliche Form gebracht. Für Anfragen an Rechner ist nicht mehr deren Adresse notwendig, sondern nur der symbolische Name. Nahezu alle Dienste unterstützen den Domain Name Service, d. h. sie erzeugen bei einer Anfrage mit symbolischem Namen automatisch eine Anfrage an den zuständigen DNS–Server. Dieser liefert die zum symbolischen Namen gehörende Adresse zurück, welche der Dienst dann verwendet.

Kann ein DNS–Server eine Anfrage nicht beantworten, so leitet er sie an den im übergeordneten DNS–Server weiter. Die DNS–Server bilden eine Baumstruktur, wodurch jeder Nameserver nur eine kleine Datenbank zu verwalten hat, üblicherweise nur die Domain, für die er direkt zuständig ist [Moc87].

In den Protokolldaten befinden sich Meldungen über angefragte Namen, weitergeleitete Anfragen und die Status– und Fehlermeldungen des Dienstes. Zu den weitergeleiteten Anfragen wird jeweils der Adressat der Weiterleitung vermerkt.

Dynamic Host Configuration Protocol (DHCP)

Das DHCP versendet Konfigurations–Parameter an Netzwerk Computer. Ein Beispiel für einen solchen Parameter stellt die IP–Adresse dar, welche durch das DHCP einen bestimmtem Rechner zugeordnet werden kann.

Das Konzept des DHCP beruht auf dem Client–Server Modell, d. h. der Server liefert die Konfigurationsdaten auf Anfrage an die Clienten. Dabei ist durch den Systemverwalter sicherzustellen, dass jeweils nur ein Server auf eine Anfrage durch einen Rechner im Netz reagiert. Des Weiteren muss gewährleistet sein, dass jede Adresse nur einmal vergeben wird.

Es gibt drei Möglichkeiten der Adressvergabe durch das DHCP. So kann eine automatische Vergabe stattfinden, durch die eine dauerhafte Adresse an einen Client vergeben wird. Weiterhin gibt es die dynamische Vergabe, die eine Adresse an einen Client nur für eine bestimmte Zeit vergibt[4]. Die dritte Art besteht darin, die Adressen vom Systemverwalter fest vorzugeben und das DHCP nur zum Ausliefern zu benutzten.

Der Vorteil der dynamischen Vergabe liegt in der Möglichkeit, Adressen, die nicht in Benutzung sind, neu zu vergeben. Dies ist besonders für die Benutzung sogenannter Adress–Pools interessant.

Neben der Adresse können weitere Parameter zur Konfiguration von Clienten verschickt werden. Dazu gehören zum Beispiel die Adresse des Nameservers, die Netzmaske und der Domainname. Ferner kann die Lebensdauer der Informationen mit angegeben werden. Die Clienten erzeugen nach Ablauf dieser Zeit eine Anfrage an den DHCP–Server [Dro97].

Ein Client, der eine Anfrage an den DHCP–Server stellt, hat meist noch keine Adresse. Eine Identifikation der Clienten erfolgt dann mittels der Hardware–Adresse (MAC). Die Hardware–Adresse ist eine netzwerkkartenspezifische 12–stellige Hexadezimalzahl. Diese wird von den Herstellern vergeben und ist weltweit eindeutig[5]. Diese MAC findet sich neben den angefragten bzw. versandten Informationen in den Protokolldaten. Es ist damit möglich, die Verwendung einer Adresse zeitlich zuzuordnen.

File Transfer Protocol (FTP)

Das Übertragen von Dateien in Computernetzen ist eine wichtige Funktionalität, und das File Transfer Protokoll stellt eine Möglichkeit dar, Daten zu übertragen.

[4] Diese Zeitspanne wird als TTL (Time to Live) oder auch Lease-Time bezeichnet.

[5] Dieser Mechanismus bringt jedoch keine Sicherheit, da die MAC softwareseitig veränderbar ist.

Die wichtigsten Ziele des FTP sind, den Austausch von Dateien zu fördern[6], verschiedene Dateisysteme und Besonderheiten entfernter Computer für die Nutzer transparent erscheinen zu lassen und Daten möglichst effizient zu übertragen. Das FTP kann von einem Benutzer direkt über ein Terminal gesteuert werden. Es ist jedoch für den Einsatz durch Anwendungen entworfen [PR85].

Die Aufgabe des FTP–Dienstes besteht darin, die Anfragen von Benutzern zu beantworten. Dazu gehören der Versand und das Empfangen von Dateien und das Navigieren im Dateisystem des FTP–Dienstes. Daten über die Anfragen und deren Beantwortung finden sich in den Protokolldaten.

Simple Mail Transfer Protokoll (SMTP)

Ziel des SMTP ist der zuverlässige und effiziente Transport von elektronischer Post. Das Protokoll ist so entworfen, dass es vom unterliegenden Transportprotokoll (vgl. OSI Schichtenmodell in [Tan90]) unabhängig ist.

Abbildung 2.3 veranschaulicht das Kommunikationsmodell des SMTP. Dabei sind Sender und Empfänger *delivery agents*, die Emails von Benutzerprogrammen entgegen nehmen bzw. diese dem Empfänger zustellen.

Eine kurze Einführung in SMTP findet sich in [SBGK94], die Spezifikation des Protokolls ist in [Pos82] nachzulesen. Große Verbreitung hat die Implementierung von SMTP namens *sendmail* [sen02].

Mit Blick auf die Protokolldaten sind insbesondere der Absender und der Empfänger einer Email von Interesse, ferner die Absende– und Zustellzeiten, aber auch Meldungen über die Nichterreichbarkeit von Systemen bzw. gescheiterte Zustellversuche. Die Statusinformationen des *delivery agents* sind für den reibungslosen Ablauf der Emailkommunikation für den Systemverwalter von Bedeutung.

Weitere Informationen zum Thema Email finden sich in der Spezifikation des Internet Message Access Protocol (IMAP) unter [Cri96] und des Post Office Protocol (POP) unter [MR96].

[6] Vor dem Hintergrund von Tauschbörsen und Urheberrechtsproblemen mag dieses Ziel seltsam anmuten. Als das File Transfer Protocol entwickelt wurde, geschah dies im wissenschaftlichem Umfeld und sollte den Austausch in diesem Bereich fördern.

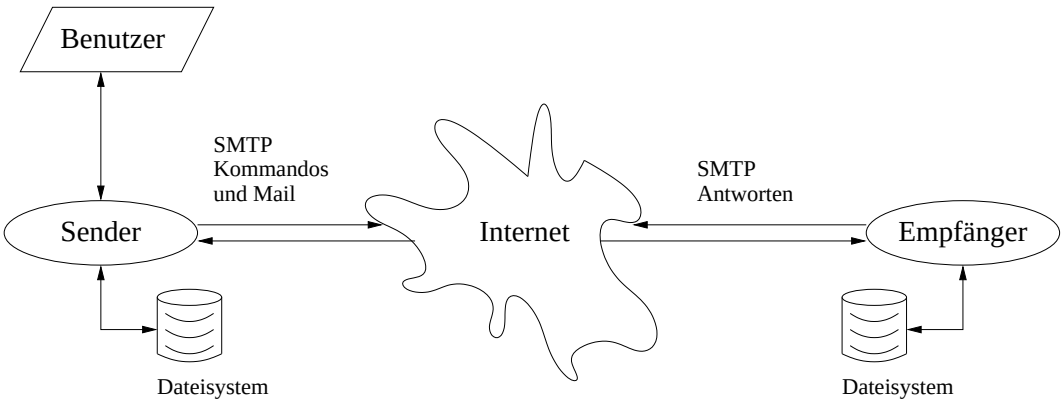

Abbildung 2.3: Darstellung des Kommunikationsmodells zum Versenden und Empfangen von Emails über SMTP, entnommen aus [SBGK94]

Lightweight Directory Access Protocol (LDAP)

Das LDAP ist ein Protokoll für den Zugriff auf ein sogenanntes *directory*. Bei einem *directory* handelt es sich um einen Verzeichnisdienst nach der *X.500* Spezifikation. Die *X.500* sind die CCITT[7] bzw. ISO–Empfehlungen für Verzeichnisdienste [SBGK94].

Abbildung 2.4 verdeutlicht den Aufbau eines solchen Verzeichnisses. Die Notation für diesen Eintrag lautet:

```
fb=fbimn, org=htwk, c=de
```

Der Vorteil des *X.500* besteht darin, dass es die verteilte Speicherung der Verzeichnisdaten erlaubt. Es können Hierarchien von Directory–Servern aufgebaut werden. Das bedeutet, dass jeder Server nur einen Teilbaum des Verzeichnisses speichern muss. Bekommt er eine Anfrage, welche er nicht beantworten kann, so kann er sie an einen anderen Server weiterleiten.

Um mit einem *X.500* Verzeichnis zu arbeiten, gibt es das Directory Access Protocoll (DAP). Es verfügt über drei Schnittstellen zum Verzeichnis, die *Leseschnittstelle*, die *Suchschnittstelle* und die *Abänderungsschnittstelle*. Das LDAP stellt eine Alternative zum DAP dar. Dabei wurde darauf geachtet, das LDAP möglichst „schlank“

[7] CCITT Comité Consultatif International Télégraphique et Téléphonique; internationales Standardisierungsgremium im Bereich der Telekommunikation. Das CCITT hat sich in ITU-TSS (International Telecommunications Union - Telecommunications Standardization Sector) umbenannt, die Abkrzung CCITT ist aber noch weit verbreitet.

zu halten. Dazu tragen besonders die Übertragung von Protokollelementen direkt auf der Transportebene und die BER–Kodierung[8] bei [YHK95].

Neben der Möglichkeit, die in einem Verzeichnis gespeicherten Daten zur Authentifizierung von Benutzern in einem Netzwerk zu verwenden, bietet das LDAP eine schnelle Zugriffsmöglichkeit auf die Verzeichniseinträge. Deshalb sind bei diesem Dienst auch insbesondere die Statusmeldungen in den Protokolldateien für den Systemverwalter von Interesse. Weiterhin sind Informationen bzgl. der Veränderung und des Hinzufügens von Einträgen von Interesse.

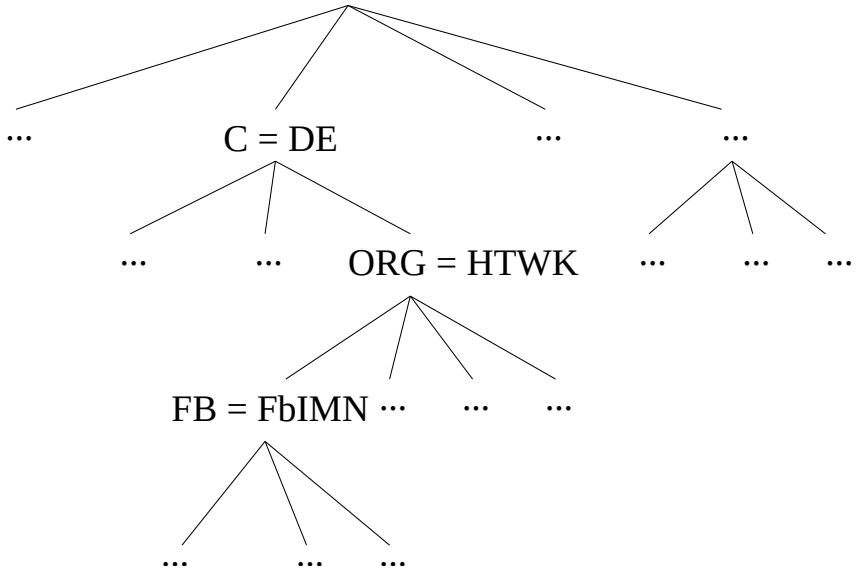

Abbildung 2.4: Exemplarischer Ausschnitt aus einem X.500 Directory Information Tree (DIT)

Secure Shell (SSH)

Telnet ist der erste Dienst, welcher im Internet implementiert wurde. Durch *Telnet* ist es möglich, ein Terminal an einem entfernten Rechner zu bedienen, so als wären der eigene Monitor und die eigene Tastatur an diesem Rechner angeschlossen. Da die Eingaben unverschlüsselt zwischen dem entfernten Rechner und dem Rechner des Benutzers übertragen werden, ist es für einen Angreifer möglich, Passwörter und Benutzerinformationen zu erhalten. Ferner ist es auch möglich, alle Eingaben

[8] Basic Encoding Rules – Nach ISO8825-1 (Eine weitergehende Beschreibung findet sich in [Cas02].

und Antworten zu lesen.
Um diese Sicherheitsprobleme des *Telnet* zu beheben, wurde zum Beispiel das Secure Shell Protokoll (*SSH*) entwickelt.

Durch den Einsatz von kryptographischen Verfahren wird es ermöglicht, sämtliche Daten verschlüsselt zu übertragen. Die *SSH* sichert dabei die Authentifizierung und die sichere Datenübertragung über unsichere Netze. Ziele des Einsatzes der *SSH* sind Sicherheitslücken im Internet Protokoll, bei der Leitwegerstellung und der Namensauflösung zu schließen.

Die *SSH* unterstützt das Weiterleiten von Ports, d. h. es werden andere Protokolle in eine *SSH*–Verbindung „verpackt“. Diese Protokolldaten werden als Nutzdaten der *SSH* übertragen und bleiben dadurch unverändert. Der Vorteil besteht in der sicheren Übertragung des „verpackten“ Protokolls. Des Weiteren bietet die SSH die Möglichkeit, die X11-Ausgaben eines Prozesses auf das eigene Terminal bzw. den eigenen X11 Desktop umzuleiten (tunneln) [Ylo95].

In den Protokolldaten der SSH finden sich, je nach eingestellter Detailtiefe, Informationen zur Erstellung neuer Schlüssel für die kryptographischen Algorithmen, Meldungen zum Austausch dieser Schlüssel bei der Erstellung einer neuen Verbindung und Informationen zu den verwendeten Benutzerkonten. Da eine Verbindung während der gesamten Nutzungszeit besteht, ist ein explizites Beenden der Verbindung notwendig. Dieses Ende wird ebenfalls protokolliert.

Nach dem Beenden einer Verbindung werden die für die Verbindung benutzten Verbindungs-Schlüssel ungültig und müssen bei einem erneuten Verbindungsaufbau wieder erzeugt werden.

2.3 Auswertung von Protokolldaten

Es gibt verschiedene Möglichkeiten, Systemprotokolle zu analysieren. Je nach Anforderung kann eine Auswertung z. B.

- wenn der Grund für einen Systemausfall gefunden werden soll,
- regelmäßig durch den Systemverwalter
- oder automatisiert stattfinden.

Das folgende Kapitel soll diese Möglichkeiten näher erörtern sowie Vor- und Nachteile der Ansätze aufzeigen. Die Betrachtung findet dabei sowohl unter Kosten- als auch Effizienz- und Sicherheitsaspekten statt [Ley96].

2.3.1 Sporadische Analyse

Die Systemprotokolldaten werden gesammelt und ggf. aufbewahrt. Tritt ein Problem auf, so werden diese Informationen verwendet, um die Ursache zu finden. Dazu ist es notwendig, dass ein Administrator die Protokollinformationen analysiert und aus vorhandenen Anomalien Rückschlüsse auf Systemfehler zieht.

Neben dem Zeitaufwand für die manuelle Analyse der Daten birgt diese Vorgehensweise zahlreiche Gefahren. Zur Sicherung der Integrität der Protokolldaten ist zusätzlicher Aufwand zu treiben, d. h. es muss sichergestellt sein, dass jegliche Veränderung an den Daten unmöglich ist bzw. alle Veränderungen gemeldet werden. Ein Angreifer kann sich die sporadische Analyse zu Nutze machen und unbemerkt Rechner für seine Zwecke einsetzen (z. B. für verteilte Angriffe, zur Verschleierung seiner Identität o. ä.). Weiterhin sind Maßnahmen notwendig, die Protokolldaten auch über einen Systemausfall hinaus verwenden zu können.

Im Normalbetrieb sind die Kosten für eine derartige Analyse sehr gering. Es werden lediglich die Ressourcen zum Speichern der Protokolldaten benötigt. Ferner gibt es keine Anforderung an personelle Ressourcen. Tritt jedoch ein Fehler auf, so entsteht ein Bedarf an personellen Ressourcen, d. h. ein Systemverwalter muss den Fehler analysieren. Zur Analyse ist jedoch jedesmal eine Einarbeitung notwendig, da diese Aufgaben vom normalen Tagesgeschäft abweichen. Während der Ursachensuche ist der Fehler weiterhin vorhanden, d. h. im Extremfall ein Totalausfall eines Systems. Während der Analyse steht der Administrator nicht für andere Aufgaben zur Verfügung.

Die Geschwindigkeit, mit der ein Fehler erkannt wird, ist sehr gering und es ist nicht gewährleistet, dass alle Fehler gefunden werden. Im normalen Betrieb können viele kleine Fehler auftreten, welche übersehen werden, obwohl diese unter Umständen parallel zum normalen Betrieb behoben werden könnten.

Dieser Ansatz ist aufgrund des hohen Sicherheitsrisikos sowohl in Bezug auf Kom-

promittierung von Rechnern als auch in Bezug auf den Ausfall wichtiger Bestandteile der Infrastruktur nicht zu empfehlen. Ferner birgt dieses Verfahren versteckte Kosten, zum einen durch die plötzlich benötigten personellen Ressourcen, zum anderen durch die Kosten durch Ausfälle, Reparatur und ähnliches.

2.3.2 Manuelle Auswertung

Die anfallenden Protokolldaten werden regelmäßig durch den Systemverwalter gelesen. Dabei stellt dieser auch kleinere Fehler fest, welche sofort behoben werden können. Die Dauer und die Genauigkeit der manuellen Auswertung wird durch die Menge der Protokolldaten bestimmt, d. h. je mehr Protokolldaten vorhanden sind, desto länger dauert deren Analyse. Soll die Zeit jedoch konstant gehalten werden, leidet die Genauigkeit, d. h. Anomalien, welche in den Daten versteckt sind (z. B. durch Systemeinbrüche) können nur noch bedingt gefunden werden.

Der Zeitaufwand für eine derartige Auswertung ist enorm. Damit sind ständig personelle Ressourcen für die Analyse der Protokolldaten gebunden, welche entsprechend Kosten verursachen.

Auch bei der manuellen Auswertung ist die Integrität der Daten ein Problem. Die Manipulation von Protokolldaten gehört zum Standardrepertoire eines Eindringlings [Sch98]. Durch die ständige Kontrolle können jedoch Probleme rechtzeitig erkannt und entsprechende Gegenmaßnahmen eingeleitet werden. Dadurch lassen sich Ausfallzeiten verringern.

Viele der Einträge in den Protokollen fallen beim normalen Betrieb eines Systems an, d. h. sie sind für die Analyse von geringerer Bedeutung. Derartige Einträge erschweren jedoch die manuelle Auswertung erheblich.
Dieses Verfahren bietet mehr Sicherheit als die spontane Auswertung, ist jedoch aufgrund des hohen personellen Aufwandes nur in kleinen Umgebungen sinnvoll. Eine Arbeitsteilung in größeren Umgebungen ist zwar denkbar (jeder Administrator überwacht einen Teil der Server), birgt jedoch Gefahren, da Szenarien, in denen eine Abhängigkeit der Protokolldaten vorliegt, nicht oder nur bedingt betrachtetet werden können. Auch gibt es einige Dienste, deren Protokolldaten aufgrund ihrer Menge nicht manuell ausgewertet werden können (z. B. *Squid* oder *Samba*).

2.3.3 Automatisierte Analyse

Die automatisierte Auswertung von Protokolldaten stellt eine kostengünstige und effiziente Alternative zur manuellen Analyse dar. Nach der Konfiguration des Analysesystems werden die Protokolldaten ausgewertet. Je nach Ereignis kann das System entsprechend darauf reagieren. Wichtige Ereignisse werden dem Administrator mitgeteilt. Dies kann je nach Konfiguration über verschiedene Kanäle ablaufen (z. B. Email, SMS, Pager).

Die automatisierte Analyse kann jedoch nicht alle Anomalien aufdecken bzw. alle denkbaren Fehler erkennen. Deshalb ist ein solches System so einzustellen, dass bei Protokollen aus dem normalem Betrieb nahezu keine Fehler gemeldet werden. Sobald jedoch eine Abweichung erkannt wird, müssen die entsprechenden Fehler gemeldet werden.

Der Vorteil eines automatisierten Systems ist unter anderem, dass das System nahezu in Echtzeit arbeiten kann, d. h. Fehler werden zeitnah entdeckt und gemeldet. Sogenannte *Intrusion Detection Systeme* (IDS) beruhen ebenfalls auf der automatisierten Analyse von Protokolldaten. Mit ihrer Hilfe ist die Erkennung von Angriffen möglich, Angaben über Systemfehler können durch sie jedoch nur bedingt verarbeitet werden.

Trotz dieser Vorteile ist auch hier der manuelle Eingriff von Administratoren notwendig. Diese müssen alle Fehler bearbeiten, welche das System entdeckt und bei jeder Umgebungsänderung Anpassungen am System vornehmen.

Eine große Gefahr birgt der Bereich automatischer Reaktionen auf entdeckte Fehler und Anomalien. Hier stehen zahlreiche sogenannte *Intrusion Response Systeme* (IRS) zur Verfügung. Dieser Bereich ist jedoch mit starken rechtlichen Problemen behaftet. So darf eine Reaktion nur in rechtlich einwandfreiem, d. h. mit Kenntnis aller Beweggründe und im der Situation angemessenem Umfang erfolgen. Dies ist allerdings nur in ganz bestimmten Fällen gegeben. Der Einsatz von IR Systemen ist nicht zu empfehlen. Eine ausführliche Studie zum Thema IDS und IRS findet sich in [KvH98]. Ferner werden die rechtlichen Rahmenbedingungen zu IRS im Kapitel 3.4 ab Seite 34 näher betrachtet.

Es gibt die Möglichkeit die verschiedenen Anforderungen an die automatisierte

Auswertung und Benachrichtigung der Systemverwalter durch sogenannte System–Management–Systeme zu realisieren. Diese Systeme bieten eine Vielzahl an Möglichkeiten zur Verwaltung und Überwachung von Systemen. Aufgrund ihres Leistungsspektrums ist es nur sinnvoll, diese Systeme einzusetzen, wenn durch sie weitere Aufgaben der Systemverwalter automatisiert oder vereinfacht werden. Mit den Komponenten eines solchen Systems steigen auch die Kosten sowohl für Anschaffung als auch für die Wartung der Systeme [Hä01].

Kapitel 3

Rechtliche Rahmenbedingungen

Der Gesetzgeber hat mit einer umfangreichen Gesetzessammlung eine Vielzahl von Fragestellungen in Bezug auf den Umgang mit Telekommunikations– und Computertechnik geregelt. Als Bestandteil des Gesetzes zur Regelung der Rahmenbedingungen für Informations- und Kommunikationsdienste (Informations- und Kommunikationsdienste-Gesetz IuKDG) sind insbesondere das Teledienstegesetz (TDG) und das Teledienstedatenschutzgesetz (TDDSG) zu nennen [IuK97]. Weitere Regelungen finden sich im Telekommunikationsgesetz (TKG) und im Bundesdatenschutzgesetz (BDSG) sowie in der Telekommunikations–Datenschutzverordnung (TDSV) und der Telekommunikations–Kundenschutzverordnung (TKV).

Die im Folgenden aufgezeigten rechtlichen Probleme und genannten Gesetze umfassen nur einen Teil der komplexen Materie rechtlicher Rahmenbedingungen im Umfeld von Protokolldatenerhebung, -speicherung und -analyse. Der Gesetzgeber ist derzeit mit der Überarbeitung des IuKDG beschäftigt, und es werden allerorten die gesetzlichen Regelungen in Bezug auf Datenschutz gelockert.

Die genannten Kommentare und Auslegungen zu den Gesetzestexten sind zum Teil sehr weit gefasst und bedürfen einer Prüfung für jeden konkreten Einzelfall.
Dessen ungeachtet soll dieses Kapitel den Blick für dieses Umfeld schärfen und auf einige wichtige Regelungen hinweisen.

Bereits im Grundgesetz für die Bundesrepublik Deutschland (GG) [GG01] wurden einige Regelungen zum Umgang mit nichtöffentlicher Kommunikation getroffen. Artikel 10 Absatz 1 des GG lautet:

> „Das Briefgeheimnis sowie das Post- und Fernmeldegeheimnis sind unverletzlich.“

Dadurch ist das Auswerten nichtöffentlicher Kommunikation grundsätzlich verboten. Mit einer Grundgesetzänderung vom 26.04.1968 wurde dieser Artikel um einen Absatz ergänzt (Artikel 10 Absatz 2), wodurch erstmals Eingriffe in das Fernmeldegeheimnis gestattet wurden [ELV01]. Diese Regelung findet auch im Umfeld der elektronischen Nachrichtenübermittlung Anwendung, d. h. sie ist nicht auf die „klassischen“ Medien beschränkt. Eine umfangreiche Interpretation dieses Artikels und der daraus erwachsenden Schwierigkeiten ist in [vMK00] nachzulesen.

Die folgenden Abschnitte sollen die für die Erfassung von Daten mittels Protokolldateien und deren Auswertung zu beachtenden gesetzlichen Regelungen beleuchten.

3.1 Regelungen zum Datenschutz

Die Grundlage der Regelungen zum Datenschutz ist das Bundesdatenschutzgesetz (BDSG). §1 Abs. 1 des BDSG lautet:

> „Zweck dieses Gesetzes ist es, den einzelnen davor zu schützen, daß er durch den Umgang mit seinen personenbezogenen Daten in seinem Persönlichkeitsrecht beeinträchtigt wird.“ [BDS00]

Neben dem Zweck regelt dieser Paragraph in den Abschnitten 2 bis 5 den Anwendungsbereich des Gesetzes.

Darauf aufbauend regelt das TDDSG den Datenschutz im Zusammenhang mit Diensten nach dem TDG bzw. Dienste nach dem Mediendienstestaatsvertrag (MDStV). Diese Regelung besteht im Wesentlichen aus einem generellen Verbot personenbezogene Daten zu erheben. Derartige Daten dürfen erhoben werden, sofern der Dienste–Nutzer dem zustimmt oder einer der Ausnahmetatbestände eintritt.

Die amtlichen Begründungen des TDG und TDDSG, entnommen aus [GM00], lauten:

Teledienstegesetz Gesetz über die Nutzung von Telediensten

> Rahmenbedingungen für das Angebot und die Nutzung von Telediensten durch Sicherstellung der Zugangsfreiheit sowie Schließung von Regelungslücken im Verbraucherschutz und Klarstellung von Verantwortlichkeiten der Diensteanbieter.

Teledienstedatenschutzgesetz Gesetz über den Datenschutz bei Telediensten

> Bereichsspezifische Regelungen zum Datenschutz bei Telediensten im Hinblick auf die erweiterten Risiken der Erhebung, Verarbeitung und Nutzung personenbezogener Daten.

Nach §9 BDSG haben öffentliche und nicht–öffentliche Stellen, die selbst oder im Auftrag personenbezogene Daten verarbeiten, die technischen und organisatorischen Maßnahmen zu treffen, um die Ausführung der Vorschriften des Bundesdatenschutzgesetzes und im Besonderen die in der Anlage des Gesetzes genannten Anforderungen zu gewährleisten. Erforderlich sind Maßnahmen nur, wenn ihr Aufwand in einem angemessenen Verhältnis zu dem angestrebten Schutzzweck steht [GS97]. Folgende Maßnahmen sind in der Anlage des BDSG [BDS01] mit Bezug auf den §9 formuliert:

> Maßnahmen „ ..., die je nach Art der zu schützenden personenbezogenen Daten oder Datenkategorien geeignet sind,
>
> 1. Unbefugten den Zutritt zu Datenverarbeitungsanlagen, mit denen personenbezogene Daten verarbeitet oder genutzt werden, zu verwehren (Zutrittskontrolle),
> 2. zu verhindern, dass Datenverarbeitungssysteme von Unbefugten genutzt werden können (Zugangskontrolle),
> 3. zu gewährleisten, dass die zur Benutzung eines Datenverarbeitungssystems Berechtigten ausschließlich auf die ihrer Zugriffsberechtigung unterliegenden Daten zugreifen können, und dass personenbezogene Daten bei der Verarbeitung, Nutzung und nach der Speicherung nicht unbefugt gelesen, kopiert, verändert oder entfernt werden können (Zugriffskontrolle),

4. zu gewährleisten, dass personenbezogene Daten bei der elektronischen Übertragung oder während ihres Transports oder ihrer Speicherung auf Datenträger nicht unbefugt gelesen, kopiert, verändert oder entfernt werden können, und dass überprüft und festgestellt werden kann, an welche Stellen eine Übermittlung personenbezogener Daten durch Einrichtung zur Datenübertragung vorgesehen ist (Weitergabekontrolle),

5. zu gewährleisten, dass nachträglich überprüft und festgestellt werden kann, ob und von wem personenbezogene Daten in Datenverarbeitungssysteme eingegeben, verändert oder entfernt worden sind (Eingabekontrolle),

6. zu gewährleisten, dass personenbezogene Daten, die im Auftrag verarbeitet werden, nur entsprechend den Weisungen des Auftraggebers verarbeitet werden können (Auftragskontrolle),

7. zu gewährleisten, dass personenbezogene Daten gegen zufällige Zerstörung oder Verlust geschützt sind (Verfügbarkeitskontrolle),

8. zu gewährleisten, dass zu unterschiedlichen Zwecken erhobene Daten getrennt verarbeitet werden können. “

Das TDDSG unterscheidet zwei Arten von Daten, zum einen *Bestandsdaten* und zum anderen *Nutzungs- und Abrechnungsdaten*. Bestandsdaten sind nach §5 TDDSG Daten, welche für die Begründung, die inhaltliche Ausgestaltung oder Änderung eines Vertragsverhältnisses erforderlich sind. Nutzungsdaten darf der Diensteanbieter nach §6 Abs. 1f erheben, verarbeiten und nutzen, jedoch nur, soweit dies für die Inanspruchnahme des Teledienstes notwendig ist und nur solange die Nutzung dauert. Nach der Nutzung sind die Daten sofort zu löschen. Nutzungsdaten sind z. B. welcher Dienst von welcher IP–Adresse gerade benutzt wird. Zum Schutz der Anwender wird der Begriff der Nutzung sehr eng ausgelegt.

Die Auslegung zur Aufbewahrungsdauer von Nutzungsdaten und die Verpflichtung zur Datensparsamkeit lassen eine Speicherung dieser Daten von vornherein deplaziert erscheinen. Eine Auslegung, dass Nutzungsdaten nur im Speicher des jeweiligen dienstanbietenden Servers den Erlaubnistatbestand bereits ausfüllen, ist wahrscheinlich.

Handelt es sich bei den Informationen, die Gegenstand des Teledienstes sind, um personenbezogene Daten Dritter (z.B. Kommunikation mit einem personenbezogenen Auskunftssystem), so gilt hinsichtlich der Zuverlässigkeit der Übermittlung per Abruf und der weiteren Verarbeitung und Nutzung beim Empfänger das BDSG oder LDSG:

> „*Unberührt von den Bestimmungen des Entwurfs des Teledienstedatenschutzgesetzes bleibt im übrigen die Erhebung, Verarbeitung und Nutzung derjenigen personenbezogenen Daten, die von öffentlichen und nicht-öffentlichen Stellen bei der Nutzung von elektronischen Kommunikationsdiensten zur Kenntnis genommen und ggf. weiterverarbeitet werden. Für die Nutzung von Telediensten gelten insoweit die Vorschriften des Bundesdatenschutzgesetzes, der Landesdatenschutzgesetze oder sonstiger bereichsspezifischer Datenschutzvorschriften. Die Zulässigkeit der Auswertung und weiteren Verarbeitung von personenbezogenen Daten, die beispielsweise in Stellengesuchen oder Immobilienanzeigen im Internet enthalten sind, durch eine nicht–öffentliche Stelle ist danach regelmäßig auf der Grundlage der §§27ff BDSG zu beurteilen.* Mit anderen Worten: Die ggf. für das Produkt oder die Dienstleistung, die der Online-Nutzer in Anspruch nimmt, bislang geltenden Datenschutzvorschriften bleiben unberührt.“ (Hervorhebung im Orginal) [GM00]

Ein Teledienst wird angeboten wenn (vgl. §3 Nr.1 TDG und §2 Nr.1 TDDSG):

- eigene Teledienste zur Nutzung bereitgehalten werden (Content-Provider),
- fremde Teledienste zur Nutzung bereitgehalten werden (Host/Service-Provider) oder
- der Zugang zu fremden Telediensten vermittelt wird (Access-Provider)

Laut der hessischen Landesdatenschutzbeauftragten:

> „Da niemand sich selbst etwas anbieten kann, müssen folglich Anbieter und Nutzer verschiedene Personen oder Stellen sein. Die nicht

> wissenschaftlich Bediensteten, das wissenschaftliche Personal und die Studierenden sind jedoch Mitglieder der öffentlichen Körperschaft Hochschule. Als solche haben sie einen hochschulrechtlichen Anspruch, alle Einrichtungen der Hochschule im Rahmen der Benutzungsordnung zu nutzen. Ermöglicht die Hochschule einen Internet-Zugang, haben alle Mitglieder grundsätzlich ein Nutzungsrecht. Steht den Mitgliedern der Hochschule der Internet-Zugang nur zur dienstlichen und/oder hochschulrechtlich festgelegten Aufgabenerfüllung in Forschung, Lehre und Verwaltung zur Verfügung und ist die private Nutzung ausdrücklich ausgeschlossen, erfolgt die Nutzung nicht durch von der Hochschule verschiedene Personen, sondern durch Mitglieder der Hochschule. Es handelt sich demnach nicht um ein Anbieter–Nutzer–Verhältnis.“ [GM00]

Parallel dazu gilt dies auch für Arbeitnehmer, die sich unternehmenseigener Teledienste oder fremder Teledienste, bei denen der Arbeitgeber als Nutzer auftritt, zur Ausübung ihrer Tätigkeit für das Unternehmen bedienen. In diesem Sinne handelt es sich bei den Arbeitnehmern nicht um Nutzer nach dem Teledienstedatenschutzgesetz.

Ist die Nutzung dieser Dienste auch für private Zwecke gestattet, z. B. der Empfang und Versand privater Emails am Arbeitsplatz, liegt zwischen dem Mitarbeiter und dem Arbeitgeber ein Nutzungsverhältnis vor. Dies hat zur Folge, dass das TDDSG zur Anwendung kommt. Gleiches gilt, wenn ein konzernangehöriges Unternehmen den Zugang zu Diensten für private Zwecke öffnet. Strittig ist die Frage, welches Gesetz zur Anwendung kommt, wenn private Daten, welche durch Verschulden oder Nichtverschulden eines Arbeitnehmers entstehen, obwohl dies durch den Arbeitgeber explizit untersagt wurde.

Bei privater Nutzung greifen die Vorschriften des TDDSG und ansonsten generell die des BDSG. Soweit bei Nutzung des Teledienstes – gleichgültig ob für dienstliche oder private Zwecke – durch den Mitarbeiter anfallende leistungs- und verhaltensbezogene Daten von dem Unternehmen (Arbeitgeber) gespeichert werden, unterliegt die Nutzung des Teledienstes durch das Unternehmen der Mitbestimmung der Mitarbeitervertretung (§87 Abs. 1 Nr. 6 BetrVG, §75 Abs. 3 Nr. 17 BPersVG).

Bei den Mitarbeitern eines Unternehmes handelt es sich um eine „geschlossene Nutzergruppe“. Im „Bericht der Bundesregierung über die Erfahrungen und Entwicklungen bei den neuen Informations- und Kommunikationsdiensten im Zusammenhang mit der Umsetzung des Informations- und Kommunikationsdienste-Gesetzes (IuKDG)“ fordert der Deutsche Bundestag die Bundesregierung deshalb auf, bei der Prüfung des Gesetzes die Frage einzubeziehen, ob die beschlossenen Regelungen in internen Netzen und geschlossenen Nutzergruppen praktikabel sind. Ferner werden evtl. Anpassungen und Ergänzungen gefordert. Derzeit enthält das IuKDG keine Regelungen, um geschlossene Nutzergruppen vom Anwendungsbereich dieses Gesetzes auszuschließen. Dies ist zum einen durch die nahezu unmögliche definitorische Eingrenzung der Begriffe „geschlossene Nutzergruppe“ und „internes Netz“ zu begründen. Zum anderen sind Gründe in den technischen Möglichkeiten zu sehen [BTD99].

Des Weiteren ist eine Herausnahme der Regelungen zu „verbundenen Unternehmen“ aus dem Anwendungsbereich des TDG nicht im Sinne einer sachgerechten rechtlichen Eingrenzung. Bei den verbundenen Unternehmen stehen unterschiedliche Strukturen und eine oftmals sehr hohe Zahl von Beschäftigten sowie die vielfältigen Betätigungs- und Nutzungsmöglichkeiten in den Netzen, nach Auffassung der Bundesregierung, einer Zuordnung als geschlossene Nutzergruppe entgegen [BTD99].

Zulässig ist die Verarbeitung personenbezogener Daten ferner bei Einwilligung des Betroffenen. Dies gilt für alle im TDDSG geregelten Verarbeitungen, d. h. alle Verarbeitungs- und Nutzungsbeschränkungen können durch Einwilligung des Betroffenen aufgehoben werden. Diese Einwilligung kann nach §3 Abs. 6 jederzeit mit Wirkung für die Zukunft widerrufen werden.

3.2 Personenbezogene Daten in Protokolldaten

„Personenbezogene Daten sind Einzelangaben über persönliche oder sachliche Verhältnisse einer bestimmten oder bestimmbaren natürlichen Person.“ (§3 Abs. 1 BDSG)

In den Protokolldaten befinden sich verschiedene im Sinne von §3 Abs. 1 BDSG personenbezogene Angaben. Insbesondere lassen sich IP–Adressen bestimmten Personen zuordnen. Somit befinden sich in den Daten Informationen zum Nutzerverhalten. Folgende Angaben lassen sich direkt ablesen bzw. sind bestimmbar:

- Welcher Dienst wurde in Anspruch genommen?
- Wann wurde der Dienst genutzt?
- Welche Anfragen wurden gestellt?
- Sind Fehler aufgetreten?

Neben den IP–Adressen finden sich auch Namen von Benutzerkonten und Email–Adressen in den Daten. Nach dem im Kapitel 3.1 beschriebenen Regelungen nach BDSG, TDG und TDDSG sowie MDStV ist die Aufbewahrung dieser Daten untersagt.

In Tabelle 3.1 werden die in den Hamburger Datenschutzheften beschriebenen Ebenen des Datenschutzes gezeigt. Ferner werden die zuständigen Kontrollorgane für die Ebenen angegeben.

Laut [BTD99] wird vorerst keine Definition „geschlossener Nutzergruppen“ im IuKDG gegeben, d. h. solange eine private Nutzung von Diensten innerhalb eines Unternehmens nicht untersagt wird, befinden sich Arbeitgeber und Mitarbeiter in einem Nutzungsverhältnis im Sinne des TDDSG. Zur Nutzung der personenbezogenen Informationen in den Protokolldaten bedarf es daher der Zustimmung der Betroffenen (Mitarbeiter).

3.3 Vereinbarungen durch Nutzervertreter

Speziell das TDDSG erlaubt einige Lockerungen in Bezug auf Erhebung, Speicherung und Verarbeitung von Protokolldaten, sofern die Betroffenen ihr Einverständnis dazu geben.

Um die Rechte der Betroffenen zu wahren, gibt es in Institutionen und Unternehmen Nutzervertreter. Dies bedeutet, dass in Unternehmen die Interessen der

Ebene	Rechtsgrundlagen	Datenschutzaufsicht
Inhaltsebene[a]	„Offline–Recht“: BDSG, SGB, BGB, Arbeits– und Dienstrecht	In Abhängigkeit von der Rechtsgrundlage: Aufsichtsbehörden (z. T. als Daueraufsicht, z. T. noch anlass-bezogen), LfDs[b] und BfD[c] (Daueraufsicht)
Transportbehälterebene (Teledienste, Mediendienste)	„Online Recht“: Teledienstedatenschuzgesetz, Mediendienste–Staatsvertrag	Aufsichtsbehörden (Daueraufsicht)
Transportebene (Telekommunikation, z.B. ISDN, X.25)	Telekommunikationsrecht TKG, TDSV	Bundesbeauftragter für Datenschutz (Daueraufsicht)

Tabelle 3.1: Datenschutz bei Multimedia und Telekommunikation, entnommen aus Hamburger Datenschutzhefte aus [GM00]

[a] Die Inhaltsebene beinhaltet Bestelldienste, Bankdienste, Versicherungen, selbstständiges Teleworking, Antragsbearbeitung, Informationsabruf

[b] LfD - Landesbeauftragte für den Datenschutz

[c] BfD - Bundesbeauftragte für den Datenschutz

Mitarbeiter (Nutzer von Diensten) durch den Betriebsrat wahrgenommen werden.

Es werden Regelungen zum Dienstangebot des Unternehmens getroffen, welche als Richtlinie für die Systemverwalter gelten. Werden keine oder für einzelne Dienste keine Regelungen getroffen, so findet das IuKDG und speziell das TDDSG für alle bzw. die Dienste ohne Regelung Anwendung.

Nach §108 V Nr. 4 Betriebsverfassungsgesetz (BetrVG) ist eine Überwachung der Email bzw. Telekommunikation der Mitarbeiter durch den Arbeitgeber in begrenztem Umfang gestattet. Allgemeine Rechtsbestimmungen für die Benutzung von Email durch Mitarbeiter eines Unternehmens existieren derzeit noch nicht. Der Betriebsrat erarbeitet seine Regelungen auf der Grundlage des BetrVG. Laut §87 I Nr. 6 BetrVG ist dieses Mitbestimmungsrecht erzwingbar. Auch hier sind die rechtlichen Ansprüche des einzelnen Nutzers im Einzelfall zu prüfen [Sch00a].

3.4 Weitere relevante Bestimmungen

Neben den Bestimmungen zum Datenschutz zur Wahrung der Persönlichkeitsrechte des Einzelnen gibt es andere Bestimmungen, welche beim Umgang mit Protokolldaten und deren Verarbeitung zu beachten sind.

Tabelle 3.2 gibt eine Zuordnung von Diensten zu den wichtigen gesetzlichen Regelungen an. Diese Zuordnung ist nicht als starres Gebilde zu interpretieren. Es spielen stets die im Einzelfall auftretenden Gegebenheiten eine Rolle. Weiterhin sind in diesem Umfeld auch andere Gesetze zum Beispiel das BDSG anwendbar.

Dienst	Rundfunk	Mediendienst	Teledienst
WWW		X	X
Email			X
Foren (offen)		X	
Foren (geschlossen)		X	X
IRC		X	
Teleshopping	X	X	X
Video-on-demand	X	X	X

Tabelle 3.2: Mögliche Zuordnung verschiedener Internetdienste zu den Gesetzen. Quelle: [Jae00]

Das automatisierte Reagieren auf Unregelmäßigkeiten verursacht häufig Tatbestände, welche unter das Strafgesetz (StGB) fallen. Sofern der Einsatz eines solchen Systems eine Notwendigkeit darstellt, sollten die Reaktionen nur passiver Natur sein. Es sind die rechtlichen Konsequenzen zu prüfen, da in diesem Bereich neben Geld- auch Freiheitsstrafen drohen können.

Weitere Hinweise finden sich im Grundschutzhandbuch des Bundesministerium für Sicherheit in der Informationstechnologie (BSI) [ITG01].

Kapitel 4

Unerwünschte Manipulation an Protokolldaten durch den Einfluss Dritter

Protokolldaten stellen aufgrund der Fülle an Informationen über ein System, die umgebende Infrastruktur und das Verhalten der Nutzer des Systems ein lohnendes Ziel für Angreifer dar. Die Angreifer kommen dabei je nach Art und Standort des Systems aus verschiedenen Bereichen und lassen sich wie folgt einteilen [Sch00b]:

Hacker sind Angreifer von Systemen, welche durch die Angriffe und den daraus gewonnen Informationen die Sicherheit von Systemen erhöhen wollen. Meist publizieren sie ihre Ergebnisse, ferner entwickeln sie auch Soft– und Hardware.

Cracker stellen den Gegenpol zu Hackern dar, sie dringen in Systeme ein und zerstören Daten, manipulieren Dienste bzw. benutzen die kompromittierten Systeme zum Angriff auf andere Computer. Wie die Hacker verfügen sie über Kenntnisse von Betriebssystemen, Programmiersprachen und Implementierungsdetails von Diensten.

„Script kiddies“ benutzen im Internet verfügbare Programme und Skripte, um diese gegen Systeme einzusetzen. Obwohl sie nicht über das Hintergrundwissen der Cracker verfügen, stellen sie eine ähnliche Gefahr dar, da durch sie Hintertüren in Systeme geraten können und/oder Systeme und Daten unbrauchbar gemacht werden.

Mitarbeiter sind in einer besonderen Position im Hinblick auf Angriffe. Durch ihr

Wissen über die Infrastruktur eines Unternehmens sind Angriffe von innen möglich. Dieses Wissen kann jedoch per *social engineering*[1] auch an andere weitergegeben werden und vergrößert damit die Angriffschancen von außen.

Spione sind alle, die aus kommerziellen oder wirtschaftlichen Interessen in Systeme eindringen. Häufig versuchen diese Angreifer, unbemerkt zu bleiben. Ihr Eindringen dient der Informationsbeschaffung. Sie verfügen häufig zusätzlich zu den Kenntnissen der Hacker und Cracker noch über detaillierte Informationen zum Nutzerverhalten und der Infrastruktur und deren Ressourcen.

Neben den klassischen Angriffsszenarien für IT-Systeme, welche in [ITG01] und [KvH98] ausführlich beschrieben sind, gibt es spezielle Gefahrensituationen im Zusammenhang mit Protokolldaten. Die wichtigsten Arten der Manipulation dieser Daten werden nachfolgend beschrieben.

4.1 Entfernen von Einträgen

Das Entfernen von Einträgen in den Protokolldaten dient häufig dem Verbergen von Angriffen. Dabei werden von möglichst allen Diensten die entsprechenden Ereignismeldungen aus den Protokolldateien entfernt.

Diese Art der Manipulation ist aufwendig. Sie fordert vom Angreifer zahlreiche Informationen über das System, das Nutzerverhalten und evtl. sogar über Arbeitsabläufe der Systemverwalter. Ein Angriff auf ein System, welcher das Entfernen von Einträgen zum Verbergen des Angriffs beinhaltet, bedarf damit einer langen Vorbereitungszeit. Er wird dadurch in der Regel nur von Personen oder Gruppen mit einem ganz bestimmten, meist kommerziellen Interesse durchgeführt. Es existieren zahlreiche Programme, welche das Entfernen von Ereignismeldungen, z. B. die über die Verbindung des Angreifers mit dem angegriffenen System, automatisiert vornehmen. Derartige Manipulationen lassen sich jedoch durch sogenannte Intrusion Detection Systeme (IDS) erkennen.

Neben dem Verbergen der Kompromittierung eines Systems kann das Entfernen von Protokolldateien auch dem Verbergen von an andere Systeme gestellten Anfra-

[1] *social engineering* - Maßnahmen, welche direkt auf Personen angewendet werden, falls dies einen technischen Angriff erleichtern kann. Zu diesen Maßnahmen zählen zum Beispiel Bestechung, ausspionieren, Erpressung u. a.

gen dienen. Ein weiterer Aspekt ist das Löschen von Fehler– oder Statusmeldungen, welche dem Systemverwalter eine Fehlerdiagnose erschweren. Durch das Löschen von Protokollmeldungen eines Dienstes kann dessen Betrieb auch weitgehend vor dem Systemverwalter verborgen werden. Auf diese Art ist der Einbau von Hintertüren in Systeme denkbar.

Begründet durch das umfangreiche Angebot an Programmen, welche die Einträge in den Protokolldateien automatisiert entfernen, wird das Löschen von Ereignissen und Statusmeldungen zur Verschleierung von Angriffen von allen weiter oben definierten Gruppen durchgeführt. Dabei finden diese Programme häufig bei komplexeren Zielstellungen Anwendung.

4.2 Gezieltes Ändern

Im Gegensatz zum Entfernen von Protokolleinträgen werden beim Ändern keine Daten entfernt, sondern bestehende Einträge so verändert, dass sie sich als ein Verhalten des Dienstes ohne Manipulation von außen interpretieren lassen. Diese Verfahrensweise ist zur Verschleierung von Angriffen besser geeignet, da sie das Protokollieren einer Verbindung erlaubt, welche später zum Beispiel eine erhöhte Systembelastung erklärbar macht.

Um das Ändern von Einträgen möglichst effektiv zu gestalten und einen Angriff möglichst lange verbergen zu können, ist allerdings ähnlich dem Entfernen von Einträgen die Kenntnis einer Vielzahl von Parametern des Systems und dessen Umfeldes notwendig. Das erfolgreiche Verschleiern eines Angriffes durch gezieltes Ändern von Protokolldateien erfordert einen großen Aufwand in der Vorbereitung des Angriffes. Durch Intrusion Detection ist es ebenso möglich, die Änderungen bzw. die Änderungsversuche zu erfassen und damit den Angriff festzustellen.

Für das automatisierte Ändern von Protokolldateien ist neben der Kenntnis, welche Dienste eine Verbindung erfassen, auch das Umfeld des zu kompromittierenden Systems zu berücksichtigen. Dadurch müssen evtl. vorhandene Programme für jeden Angriff speziell vorbereitet und angepasst werden. Diese Arbeiten erhöhen die Kosten für einen derartigen Angriff enorm.

Neben dem Verbergen kann der Angreifer weitere oder auch andere Ziele verfol-

gen. So könnte er bestimmte Anfragen eines Nutzers verändern, um diesen als Verursacher von Angriffen darzustellen. Durch derartige Maßnahmen lässt sich die Glaubwürdigkeit eines Nutzers nachhaltig beeinflussen.

Des Weiteren kann das Ändern von Status– und Fehlermeldungen das Auffinden von Fehlern und Sicherheitslücken deutlich erschweren. Der Systemverwalter wird unter Umständen in eine falsche Richtung, d. h. der Behebung von gemeldeten, aber nicht existierenden Problemen, geführt. Dadurch kann ein Angreifer die notwendige Zeit für die Umsetzung seiner Ziele gewinnen. Ferner können solche Meldungen auch den Zweck verfolgen, dass der Systemverwalter bestimmte Dienste evtl. auch auf fernen Rechnern startet oder beendet. Die dadurch auftretenden Sicherheitsprobleme können dann vom Angreifer genutzt werden.

Da das Ändern von Protokolldateien umfangreiches Wissen voraussetzt, werden diese Angriffe nicht von „Script Kiddies“ durchgeführt. Das Ziel einer solchen Manipulation ist nicht die Zerstörung des Systems, sondern dient der Gewinnung von Informationen oder dem Unerkanntbleiben. Deshalb bedienen sich auch Cracker nur selten dieser Eingriffe in Systeme.

4.3 Überfluten der Protokolldateien

Die dritte Art der Manipulation von Protokolldaten ist das Überfluten. Dabei werden Systeme oder Dienste mit Anfragen überhäuft. Die Flut an entstehenden Protokolleinträgen ist für den Angreifer aus verschiedenen Gründen interessant.

Ein Ziel kann das Ausschalten bestimmter Dienste oder Systeme sein. Ein solcher Angriff wird als „Denial of Service“ (DoS) Attacke bezeichnet. Dabei werden so viele Anfragen erzeugt, dass die Netzanbindung, der Prozessor oder Peripherie–Geräte überlastet werden, welche die Ausführung eines oder mehrerer Dienste blockieren. Durch das Blockieren eines Dienstes kann zum Beispiel eine Überwachung anderer Systeme blockiert oder die Erzeugung von Alarmmeldungen und Reaktionen durch Intrusion Detection bzw. Response Systeme verzögert oder ganz unterbunden werden.

Durch die Anzahl der generierten Ereignisse ist es für einen Angreifer möglich, unentdeckt zu operieren, da sich die von ihm erzeugten Ereignisse nicht von

weitergehenden Aktionen trennen lassen. Die Folge einer Überflutung von Protokolldateien ist, dass die Protokolldateien schwer auswertbar werden. Weiterhin ist bei einem solchen Angriff nicht gesichert, dass alle Ereignisse und Meldungen erfasst werden. Somit sind die Ergebnisse einer Analyse derartiger Angriffe nicht zuverlässig.

Derartige Angriffe werden vorwiegend von „Script Kiddies“ ausgeführt. Es existieren Programme, welche für derartige Angriffe einsetzbar sind. Meist ist dabei jedoch das Infiltrieren mehrerer Computer notwendig, um durch eine verteilte Aktion eine möglichst große Wirkung zu erzielen.

Weitere potenzielle Angreifer können Beauftragte konkurrierender Unternehmen sein, welche sich durch einen Ausfall des Konkurrenten Marktanteile oder größere Umsätze erhoffen. Da mit dieser Angriffsart häufig auch eine Zerstörung des angegriffenen Systems einhergeht, wird sie auch von Crackern benutzt.

Kapitel 5

Kryptographische Grundlagen

Zur Sicherung der Protokolldaten vor unerwünschten Lesern und vor Manipulationen können verschiedene Verfahren der Kryptographie Anwendung finden. An dieser Stelle wird ein kurzer Überblick über die Funktionsweise und den Aufbau einiger wichtiger Hashfunktionen gegeben. Weiterhin wird ein Ansatz zum sicheren Übermitteln von Nachrichten beschrieben. Diese sind für den Integritätsschutz und die Sicherung der Vertraulichkeit von Protokolldaten notwendig.
Eine Vertiefung zu den hier angesprochenen Grundlagen findet sich in [Sch96]. Dort gibt es neben zahlreichen Beispielen auch Implementierungsvorschläge für verschiedene Algorithmen aus dem Bereich Kryptologie.

5.1 Funktionsweise von Hashfunktionen

Bevor Hashfunktionen definiert werden können, ist es zunächst notwendig, die sogenannten Einwegfunktionen zu erläutern, da diese die mathematische Grundlage der Hashfunktionen bilden.

Eine Einwegfunktion ist eine Funktion, deren Funktionswert sich für einen gegebenen Wert aus dem Definitionsbereich der Funktion einfach berechnen lässt. Die Umkehrung, d. h. die Berechung des Urbildes zu einem gegebenen Funktionswert ist hingegen nicht oder nur sehr schwer möglich:

$$\forall x \exists y | f(x) = y$$
$$\nexists g | g(f(x)) = x$$

Es gibt Abwandlungen dieser Funktionen. Für deren Beschreibung sei auf [Sch96] verwiesen.

Nachdem die Funktionsweise von Einwegfunktionen bekannt ist, liegt die Idee der Hashfunktionen[1] nahe:

Hashfunktion Eine Hashfunktion ist eine Funktion, die eine Eingabe variabler Länge in einen Ausgabe fester Länge umwandelt. Die Ausgabe wird als Hashwert bezeichnet.

Der Hashwert ist ein eindeutiger Wert (Schlüssel). Aufgrund der Beschaffenheit von Hashfunktionen werden für verschiedene Eingaben die gleichen Schlüssel erzeugt. Eine Hashfunktion ist so beschaffen, dass aus dem gegebenen Schlüssel keine gültige Eingabe erzeugt werden kann und die Kenntnis der Eingabe es nicht ermöglicht eine Eingabe zu erzeugen, welche den gleichen Schlüssel besitzt (Kollisionsfreiheit).

Ein Hashwert ist mit einem Fingerabdruck vergleichbar. Jeder Mensch hat einen Fingerabdruck. Dabei wird nicht ausgeschlossen, dass zwei Menschen mit dem gleichen Fingerabdruck existieren. Diese beiden Menschen lassen sich jedoch nicht mit vertretbarem Aufwand benennen.

5.2 Spezielle Hashfunktionen

5.2.1 Message Digest (MD5)

Message Digest Version 5 (MD5) stellt eine Weiterentwicklung der von Ron Rivest entworfenen MD4 Hashfunktion dar. Nach der Verarbeitung der Eingabe liegt ein Hashwert der Länge 128 Bit vor. Das Verfahren arbeitet die Eingabe in 512 Bit großen Blöcken ab, wobei jeder Block in 16 Teilblöcke a 32 Bit zerlegt wird. Die Ausgabe erfolgt in vier 32 Bit langen Blöcken, welche verkettet werden.

Abbildung 5.1 verdeutlicht den Ablauf des MD5 Algorithmus; die Verkettungsvariablen A, B, C und D werden mit Konstanten initialisiert. In den Runden wird jeweils 16mal eine andere Operation ausgeführt, wobei jede Operation drei der vier Verkettungsvariablen zur Berechnung einer nichtlinearen Funktion verwendet. Das Ergebnis wird der vierten Variablen hinzuaddiert.

[1] Häufig findet sich auch die Bezeichnung Einweg–Hashfunktion.

Die Kompressionsfunktion innerhalb des MD5 bietet eine Angriffsmöglichkeit zur Erzeugung von Kollisionen. Da dies nur in speziellen Konstellationen auftritt, ist das MD5 Hashverfahren dennoch weiterhin verbreitet.

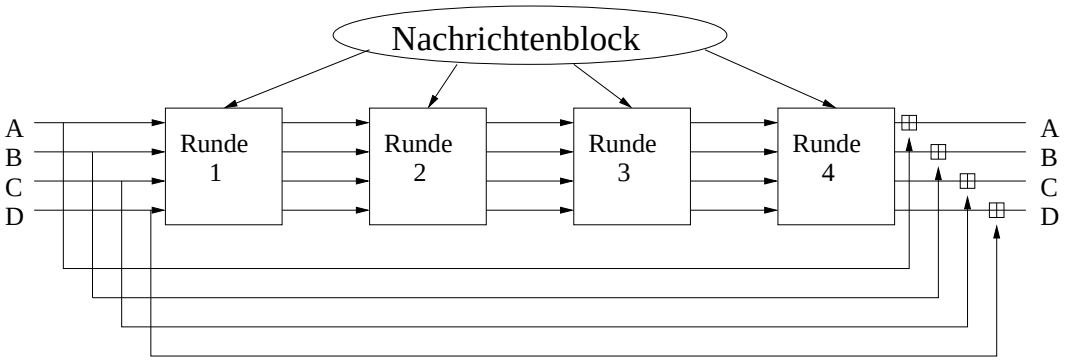

Abbildung 5.1: Ablauf der Hauptschleife der MD5 Hashfunktion (entnommen aus [Sch96])

5.2.2 Secure Hash Algorithmus (SHA)

Der Secure Hash Algorithmus (SHA) verläuft ähnlich dem MD5 Algorithmus, dabei liefert dieser eine 160 Bit lange Ausgabe. Dementsprechend gibt es fünf Verkettungsvariablen mit je 32 Bit. Es werden vier Runden mit je 20 Operationen durchgeführt. Dabei unterscheiden sich die Operationen durch die zugrunde liegenden Funktionen und Konstanten vom MD5 Algorithmus.

Abbildung 5.2 zeigt den Ablauf einer Operation, k_t ist dabei eine Konstante, w_t ein Teil des Eingabewortes. Die Ausgabe des Zeitpunktes $t-1$ wird als die Eingabe des Zeitpunktes t genutzt.

5.3 Asymmetrisches Verschlüsselungsverfahren mit öffentlichem Schlüssel

Diese Form der Verschlüsselung basiert auf Einwegfunktionen mit Hintertür, d. h. es ist mit Kenntnis einer „geheimen“ Information möglich, eine Umkehrung herbeizuführen [Sch96].

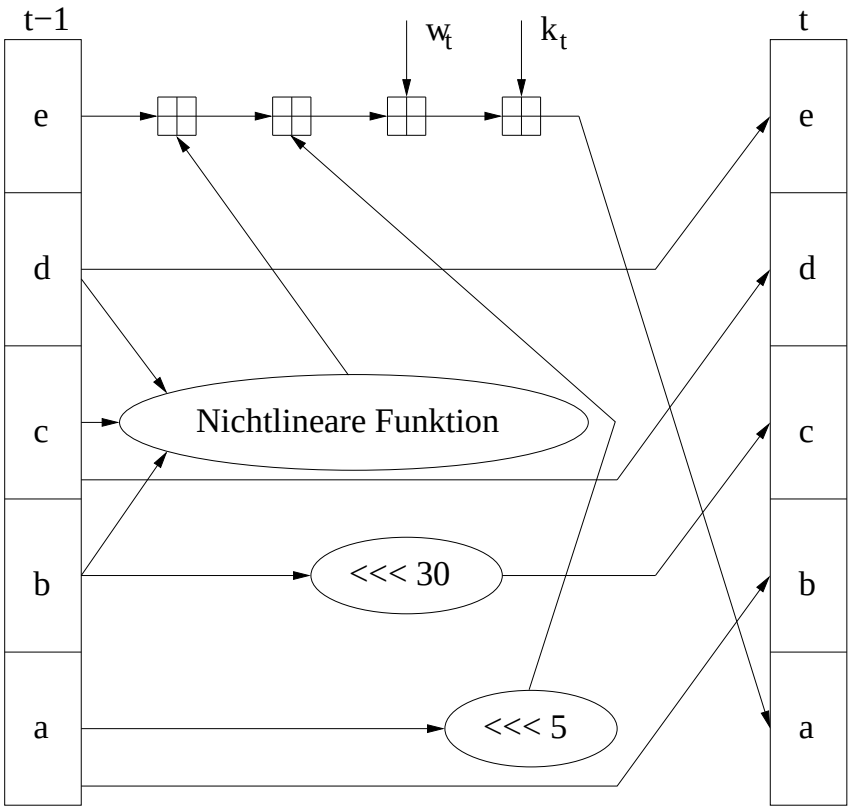

Abbildung 5.2: Ablauf einer Operation des SHA Algorithmus (nach [Sch96])

Das Verfahren[2] lässt sich am besten anhand eines Briefkastens erklären. Die Aufbewahrung von Post in einem Briefkasten kann als Verschlüsselung mit öffentlichem Schlüssel angesehen werden. Dabei ist der Briefschlitz der öffentliche Schlüssel, d. h. jeder kann eine Nachricht über diesen Weg verschlüsseln. Die Post ist im Briefkasten verschlüsselt. Mit Hilfe des Briefschlitzes ist es nicht möglich, an die Nachrichten im Briefkasten zu gelangen. Allein der private (geheime) Schlüssel ermöglicht das Entschlüsseln der Nachrichten. Beim Briefkasten ist der private Schlüssel der Schlüssel zum Briefkasten.

Die Kommunikation über Verschlüsselungsverfahren mit öffentlichem Schlüssel funktioniert dabei sehr einfach. Mit den in der Kryptologie zur Veranschaulichung verwendeten Kommunikationspartnern Alice und Bob würde das Übermitteln einer geheimen Nachricht von Alice an Bob wie folgt ablaufen:

1. Bob sendet Alice seinen öffentlichen Schlüssel. Der Schlüssel darf von allen

[2] Häufig auch als RSA-Verfahren bezeichnet

gesehen werden.

2. Alice verschlüsselt die Nachricht mit Bobs öffentlichem Schlüssel.

3. Alice sendet Bob die verschlüsselte Nachricht.

4. Bob entschlüsselt die Nachricht mit seinen privaten Schlüssel.

Sofern der zugrunde liegende Verschlüsselungsalgorithmus sicher ist und die Integrität des öffentlichen Schlüssels nicht verletzt ist, gibt es für einen Angreifer keine Möglichkeit, an die Nachricht zu gelangen.

Derartige Verfahren beruhen meist auf der Faktorisierung von großen Primzahlen, da diese Aufgabe noch nicht in polynomialer Zeit zu lösen ist. Mathematisch sieht der Ablauf so aus.

$$n = p \cdot q \qquad | \, p, q \in Primzahlen \tag{5.1}$$

$$d = e^{-1} \, mod \, ((p-1)(q-1)) \qquad | \, ggT(e, ((p-1)(q-1))) = 1 \tag{5.2}$$

$$c = m^e \, mod \, n \tag{5.3}$$

$$m = c^d \, mod \, n \tag{5.4}$$

n aus Gleichung 5.1 und e bilden den öffentlichen Schlüssel. Die zufälligen Primzahlen p und q müssen geheim bleiben und sollten nach der Berechnung von 5.2 (dem privaten Schlüssel) „vergessen“ (gelöscht) werden. Der Ablauf der Verschlüsselung wird durch Formel 5.3 dargestellt. Die Nachricht m wird mit dem öffentlichen Schlüssel e, n zum Chiffrat c verschlüsselt. Die Umkehrung findet sich in Gleichung 5.4. Hier wird das Chiffrat mit dem geheimen Schlüssel d und dem n aus dem öffentlichen Schlüssel entschlüsselt. Das Ergebnis ist die Nachricht m.

Teil II

Realisierung

Kapitel 6

Anforderungsanalyse zur Zentralisierung und Auswertung von Protokolldaten

Nach den Grundlagen zu Diensten, Protokolldatenauswertung, rechtlichen Rahmenbedingungen, Angriffsszenarien und Sicherheit, beschäftigen sich die nächsten Kapitel mit dem konkreten Entwurf eines Prototypen zur Zentralisierung und Vorbereitung der Auswertung von Protokolldaten. Dabei geht es in erster Linie um die Bereitstellung einer Schnittstelle für Standard-Software aus dem Bereich Protokolldatenanalyse. Es wird auf die Grundlagen und Algorithmen zur Mustererkennung, sowie die Betrachtung der Analyse der Daten verzichtet.

Es gibt unterschiedliche Anforderungen und Fragestellungen, welche durch den zu entwickelnden Prototypen erfüllt werden sollen. Diese werden in den folgenden Abschnitten betrachtet.

6.1 Zentralisierung

Mit steigender Größe und Verzweigung von Netzwerken, wachsen die Anforderungen an die Administratoren. Eine komplexe Netzstruktur verursacht einen großen Verwaltungsaufwand. Für eine flächendeckende Bereitstellung von Diensten werden Redundanzen im Netzwerk benötigt, welche langsame Verbindungen zwischen Nutzern und den dienstanbietenden Servern minimieren; es entstehen ähnlich strukturierte Netzwerkknoten in geographischer Nähe zu den Nutzern.

Den durch die Struktur bedingten Aufwand für die Systemverwalter bei der Auswertung der Protokolldateien gilt es zu verringern. Dazu ist es notwendig, die Protokolldateien für die Auswertung zentral bereitzustellen und zu bearbeiten.

Für die Bereitstellung gibt es zwei Möglichkeiten. Zum einen können die Protokolldateien vor Ort, d. h. auf den Systemen, auf denen sie entstehen, vorbereitet werden und im Anschluss daran an ein zentrales System versandt werden. Zum anderen besteht die Möglichkeit, die Protokolldateien unverändert zu sammeln und eine zentrale Vorbereitung für eine Auswertung durchzuführen.

Als Lösung wird die Variante mit nur einer zentralen Stelle angestrebt. Die Alternative einer Plattform pro Standort wurde nicht betrachtet, da sie die Vereinfachung der gewählten Lösung darstellt und für den späteren Einsatz einen Mehraufwand seitens der Systemverwalter bedeutet.

In Abbildung 6.1 wird die Menge der in einem Versuch anfallenden Daten dargestellt. Diese Messreihe kann aufgrund der Laufzeit als zuverlässige Kenngröße für die im Mittel anfallenden Daten gelten. Während der Tests gab es einige technische Schwierigkeiten, wodurch für ca. 30 Tage keine Messwerte geliefert wurden. Die Extrempunkte in der Messreihe sind durch ein siebentägiges Sammeln der Protokolldateien begründet. Dabei wird jede Woche eine neue Datei für die Protokolldaten eines Dienstes angelegt. Die Daten einer Woche werden innerhalb dieser Datei gesammelt. Nach vier Wochen wird jeweils die älteste Datei gelöscht.

Die Protokolldaten werden über die bestehende Infrastruktur transportiert. Dabei sollen die Nutzer dieser Struktur nicht durch das System beeinträchtigt werden. Die Aufgabe stellt hier die Anforderung, aktualisierte Protokolldaten einmal täglich bereitzustellen.

6.2 Fragestellungen zur Auswertung der Protokolldaten

Es gibt eine Vielzahl verschiedener Anforderungen, wofür Protokolldaten eingesetzt werden können. Daraus ergeben sich zahlreiche Fragestellungen, welche

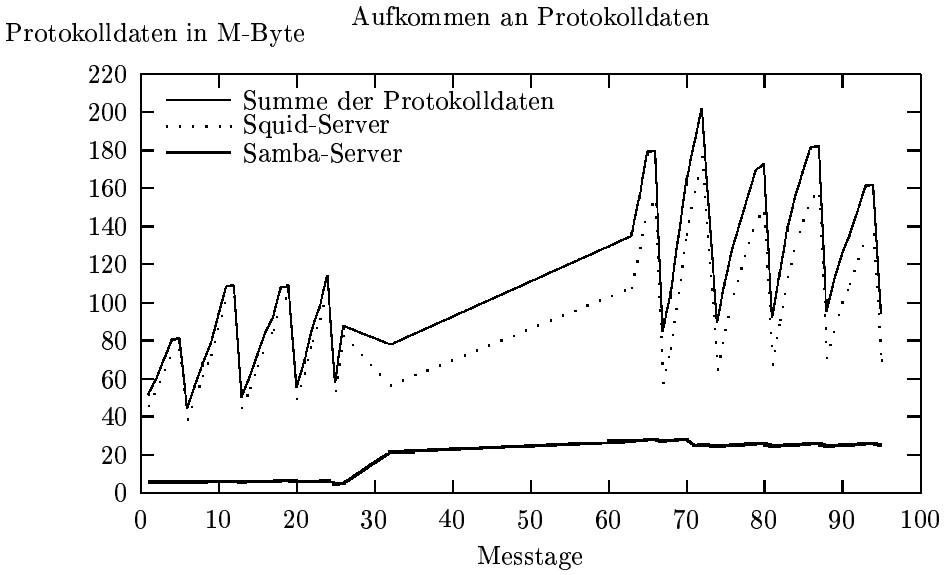

Abbildung 6.1: Durch Tests ermittelte Werte der Menge der pro Tag anfallenden Protokolldaten in Megabyte

in diesem Kontext geklärt werden müssen. Verschiedene Stellen innerhalb eines Unternehmens haben verschiedene Sichtweisen und Aufgaben, welche unter Zuhilfenahme von Informationen aus Protokolldaten gelöst werden können.

Die Administratoren haben die Aufgabe, die Funktionstüchtigkeit der Infrastruktur zu gewährleisten. Deshalb sind hier besonders Fragen wie *„Gibt es Unregelmäßigkeiten in einzelnen oder mehreren Systemen?“* oder *„Wie soll eine Benachrichtigung der Systemverwalter stattfinden?“* von Interesse. Der Betriebsrat sichert die Rechte der Mitarbeiter, wobei Vereinbarungen mit dem Arbeitgeber notwendig sind. Die hier zu beantwortende Frage ist *„Welche Veränderungen müssen an den Protokolldaten aus rechtlicher Sicht vorgenommen werden?“*, d. h. wie kann die informationelle Selbstbestimmung der Mitarbeiter gewährleistet werden.

Weiterhin sind Fragen zur Veränderung, Auswertung und Aufbewahrung zu klären:

- *Welche Änderungen sind für eine automatisierte/teilautomatisierte Auswertung notwendig?*
- *Welche Dateien sollen aufbewahrt werden?*
- *Wie lange sollen die Daten aufbewahrt werden?*

- *Welche rechtlichen Bedingungen gibt es für die Aufbewahrung?*
- *Welche sonstigen Regelungen sind zu beachten?*
- *Welche Auswertungssysteme können zum Einsatz kommen?*

Aufgrund rechtlicher Regelungen ist es u. U. notwendig, die Daten zu verändern. Insbesondere die Entfernung von personengebundenen Daten ist hierbei für die Einhaltung der rechtlichen Rahmenbedingungen (speziell dem Datenschutz) ein zu klärender Punkt.

6.3 Technische Anforderungen

Neben den Anforderungen durch verschiedene Stellen und Institutionen und der Anforderung an die Infrastruktur gibt es technische Anforderungen. Das System muss die Integrität, Vertraulichkeit und Sicherheit der Protokolldaten gewährleisten. Um dies sicherzustellen, ist ein Konzept mit Maßnahmen zur Datensicherheit notwendig. Die folgenden Punkte müssen dabei berücksichtigt werden:

- *Wie können Manipulationen erkannt werden?*
- *Wie ist der Zugriff durch unautorisierte Dritte zu verhindern?*
- *Welche Maßnahmen sind notwendig, um die Sicherheit auch bei der Aufbewahrung zu gewährleisten?*

Das System ist ebenso für die Reduktion der Protokolldaten, auf die jeweils notwendigen Dateien und Einträge zuständig. Durch die Konfiguration auf den Protokolldatenerzeugern können bereites analysierte Protokolldateien erneut Übertragen werden.

Kapitel 7

Voraussetzungen zur Realisierung

Nach der Analyse der Aufgabenstellungen stellen sich einige Anforderungen, welche für die prototypische Entwicklung einer Lösung notwendig sind. Diese sind zum einen an der bestehenden Infrastruktur vorzunehmen, zum anderen als Teil der Lösung zu betrachten.

Für die Auswertung von Protokolldaten ist das Synchronisieren der Zeit auf allen beteiligten Systemen notwendig, da eine Analyse von Ereignissen insbesondere auf dem Aufeinanderfolgen bestimmter Einträge in Protokolldateien basiert. Um diese Ereignisse entsprechend zuordnen zu können und Vergleiche mit anderen Systemen zu gewährleisten, ist diese Synchronizität erforderlich.

Das System, welches den Sammelpunkt der Protokolldaten darstellt, ist ausreichend zu dimensionieren. Die aus den Experimenten ersichtliche Menge an Protokolldaten ist hier zumindest ein guter Ausgangspunkt für die notwendige Speicherkapazität. Die benötigte Rechenleistung hängt vom Umfang der durchzuführenden Analysen ab. Ziel ist es, die Analyse des Vortages möglichst zügig zu beenden, um kurzfristig evtl. notwendige Maßnahmen ergreifen zu können.

Alle Erzeuger von Protokolldaten können mit dem Protokollsystem kommunizieren, d. h. insbesondere, dass sich diese nicht in getrennten Netzwerken befinden dürfen, bzw. dass der Protokollserver Verbindungen in verschiedene Netzwerke unterhält.

7.1 Schnittstelle zwischen den Plattformen

Die Protokolldaten werden von verschiedenen Betriebssystemen und verschiedenen Diensten erzeugt. Um eine Auswertung auf einem zentralen System stattfinden zu lassen, ist es notwendig, die in speziellen Dateiformaten angelegten Protokolldaten in ein für das jeweilige Auswertungssystem lesbares Format zu überführen.

Diese Konvertierung soll so erfolgen, dass sich auf dem zentralen Protokollserver nur die für die Auswertung notwendigen Daten befinden. Es ist für die Systeme, welche ihre Daten in einem speziellen Format ablegen, zu klären, wo welche Umwandlung stattfinden kann und welche Auswirkung diese Umwandlung ggf. auf die Protokolldaten hat.

Eine Möglichkeit zur Kommunikation der protokollierenden Systeme in Richtung Protokollserver muß gewährleistet sein. Ferner ist eine Absicherung dieser Kommunikation durch kryptographische Verfahren notwendig, da die Daten personenbezogene Informationen enthalten.

7.2 Dimensionierung und Hardwareanforderungen an den Protokollserver

Aus den der Abbildung 6.1 zu entnehmenden Zahlen ergibt sich die Anforderung an die temporäre Speicherkapazität des Systems. Da in den Tests nur die Protokolldaten von zwei Servern betrachtet wurden, sind diese Zahlen jeweils mit einem entsprechenden Faktor für die Gesamtzahl der Protokolldaten erzeugenden Systeme zu skalieren.

Bei der Anforderung an die Speicherkapazität ist ebenso die Archivierung der Daten zu berücksichtigen, welche in Kapitel 8.3 genauer betrachtet wird. Hierbei sind vor allem die Fristen und die Art der Speicherung von Interesse.

Ein weiterer Faktor für die Auswertung der Protokolldaten ist die Rechenleistung des Systems. Sowohl für die Transformation der Daten, als auch für deren Auswertung bestimmt dieser Faktor neben der Geschwindigkeit des Dateisystems die Dauer der jeweiligen Aktion. Je nach eingesetzter Auswertungssoftware und der Tiefe der Betrachtung der Daten kann die benötigte Rechenleistung stark schwan-

ken. Weitere Informationen hierzu finden sich in Kapitel 8.1.4.

Zur Kommunikation mit den Erzeugern der Protokolldaten sind die entsprechenden Verbindungen in die Netzwerke notwendig.

Weitere Anforderungen an die Hardware werden durch die jeweils angestrebten Detaillösungen, wie zum Beispiel ein Bandlaufwerk für die Archivierung oder ein Multiprozessorsystem für komplexe bzw. parallele Berechnungen gestellt. Diese Anforderungen werden an den entsprechenden Punkten in der Konzeption angesprochen.

Kapitel 8

Konzept zur Zentralisierung und Bereitstellung einer Infrastruktur für die Auswertung von Protokolldaten

Neben den technischen Anforderungen für ein System zur Zentralisierung von Protokolldaten sind bereits in der Konzeption und deren Vorbereitung die rechtlichen Gegebenheiten zu prüfen. Dabei stellen vor allem die in den Protokolldaten enthaltenen personenbezogenen Daten ein Problem dar.

Eine Lösung ist nur durch das Zusammenwirken von verschiedenen Faktoren möglich. Wie im Kapitel 3 bereits dargelegt, sind die meisten gesetzlichen Regelungen in diesem Zusammenhang mit einer Erlaubnismöglichkeit durch den Betroffenen ausgestattet. Neben dieser Möglichkeit muss das System so geschaffen sein, dass es auch ohne oder mit einer eingeschränkten Erlaubnis funktionsfähig ist.

Dieses Konzept wird neben der technischen Betrachtung ebenso die jeweils notwendigen Berechtigungen aufzeigen. Diese Berechtigungen können für Unternehmen gemeinsam mit dem Betriebsrat ausgehandelt werden und in einer Betriebsvereinbarung resultieren.

Der erste Schritt beim Aufbau des Systems ist die Betrachtung, woher die Daten kommen und wie diese Daten beschaffen sind. Im Anschluss daran ist die ggf. notwendige Vorverarbeitung der Daten festzustellen. Unter Vorverarbeitung werden hier alle Transformationen an den Protokolldaten betrachtet, welche sowohl zur Vorbereitung der Daten für die weitere Bearbeitung durch Analyseprogramme, als

auch dem Entfernen von personenbezogenen Daten und der Einhaltung der gesetzlichen Bestimmungen dienen. Weiterhin ist die Organisation der Protokolldateien auf dem Protokollserver mit Blick auf die Auswertung notwendig. Abschließend erfolgt die Betrachtung der Möglichkeiten zur Aufbewahrung und der langfristigen Sicherung der analysierten Daten.

8.1 Aufbau des Systems der Zentralisierung

Zum Aufbau des Systems ist es notwendig, dass die Protokolldaten auf den Protokollserver gelangen. Des Weiteren ist eine Vorverarbeitung notwendig. Abbildung 8.1 gibt einen schematischen Überblick über die notwendigen Komponenten dieses Systems und deren Kommunikation.

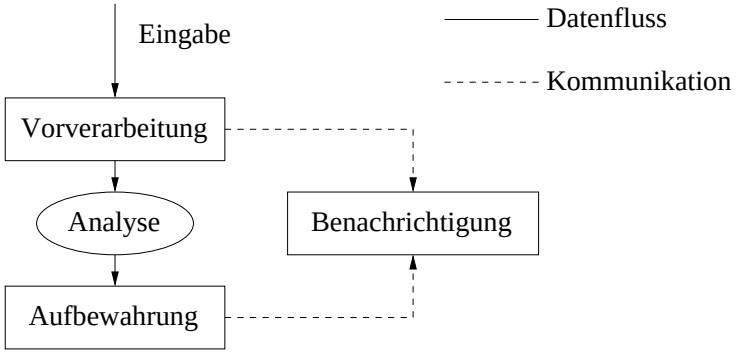

Abbildung 8.1: Die Grundkomponenten des Systems und deren Zusammenwirken in Bezug auf Datenverarbeitung und Kommunikation

8.1.1 Transfer der Protokolldaten auf den Protokollserver

Es gibt verschiedene Möglichkeiten, die Protokolldaten auf den Protokollserver zu transferieren. Zum einen kann der Protokollserver sich die Daten „abholen“. Auf diese Weise könnte eine optimale Ausnutzung der Kapazität des Protokollservers erfolgen, da die Daten sequentiell von den Erzeugern abgeholt würden und die Belastungskurve des Systems dadurch geglättet wird. Diese Variante hat jedoch den Nachteil, dass der Protokollserver die Dateien auch zu für die Erzeuger ungünstigen Zeiten abholen kann. Das kann zu Beeinträchtigungen der normalen Nutzung der Protokolldatenerzeuger führen. Zum anderen können die Erzeuger ihre Daten direkt zum Protokollserver übermitteln. Dies kann in Zeiten geringer Belastung der

Erzeuger erfolgen. Da dadurch weniger Beeinträchtigungen zu erwarten sind, wird diese Lösung favorisiert.

Für die Übertragung der Protokolldaten über ein vorhandenes Netzwerk gibt es ebenso verschiedene Möglichkeiten. Die Daten können mittels File Transfer Protokoll auf den Protokollserver gelangen. Weitere Varianten sind der Einsatz des Network File Systems oder des Samba[1]–Dienstes. Aufgrund der bereits zur Verfügung stehenden Infrastruktur und der Vor– und Nachteile der Dienste, welche in Kapitel 2.2.2 erläutert werden, wurde für die Tests die Möglichkeit per *scp*[2] Dateien zu übertragen genutzt.

8.1.2 Vorverarbeitung der Protokolldaten

Die Vorverarbeitung entfernt alle für die weitere Verarbeitung nicht benötigten Protokolldateien. Nach diesem Schritt sind nur noch relevante Protokolldateien vorhanden, diese sind jedoch nach derzeitiger Gesetzeslage aufgrund der enthaltenen personenbezogenen Daten teilweise illegal. Die Vorverarbeitung muß diese Daten anonymisieren bzw. pseudonymisieren, um eine weitere Verarbeitung zu ermöglichen.

Aus rechtlicher Sicht tritt bereits hierbei folgendes Problem auf: Nach TDG und TDDSG ist die Speicherung von personenbezogenen Daten nur zulässig, sofern diese für die Erbringung eines Dienstes notwendig sind und dann nur für die Dauer der Erbringung des Dienstes. Für den ordnungsgemäßen Betrieb von IT–Systemen und Diensten ist diese Regelung nicht zweckmäßig, da die Protokolldaten für den Betrieb dieser Systeme und der damit verbundenen Dienstebereitstellung notwendig sind.

An dieser Stelle muss gemeinsam mit den Vertretern der Betroffenen eine Erleichterung geschaffen werden, die es erlaubt, bis die Dienste dies selbst unterstützen, die Daten zumindest für die Dauer einer Pseudonymisierung bzw. Anonymisierung zu speichern.

[1] Es wurde von einem TCP/IP Netzwerk ausgegangen, in anderen Netzwerken stehen unter Umständen andere Möglichkeiten zur Übertragung vom Dateien zur Verfügung.

[2] *scp – Secure Copy* ist ein Programm, welches die Funktionalität der *ssh* verwendet um Dateien verschlüsselt zu übertragen.

Zu anonymisierende Einträge

Für die Anonymisierung bzw. Pseudonymisierung ist es notwendig zu wissen, welche Einträge in den Protokolldateien überhaupt personenbezogen sind.

Zum Beantworten dieser Frage ist es nötig, einige weitergehende Betrachtungen anzustellen. Unstrittig ist, dass Informationen über Benutzerkonten in Protokolldaten als personenbezogene Daten gelten, da eine eindeutige Verbindung zwischen einem Benutzerkonto und einer realen Person[3] (Betroffener) vorliegt.

Strittig ist die Frage, ob IP–Adressen als personenbezogene Daten angesehen werden müssen oder nicht. Um diesen Sachverhalt etwas näher beleuchten zu können, wurde im Kapitel 2.2.2 auf Seite 15 der Dienst DHCP näher betrachtet. Für die Vergabe von IP–Adressen mittels DHCP gab es zwei Möglichkeiten: Zum einen die Zuweisung einer eindeutigen IP–Adresse anhand der Hardwareadresse der Netzwerkkarte des anfragenden Systems und zum anderen die Zuweisung einer zufälligen IP–Adresse aus einem Adresspool. In der ersten Variante ist es dadurch leicht nachvollziebar, welcher Rechner einen Eintrag in die Protokolldaten erzeugt hat. Ist es möglich, die Rechner einzelnen Personen zuzuordnen, so gelten die erzeugten Protokolleinträge mit der IP–Adresse dieses Rechners als personenbezogen.

Weiter zu betrachten sind die DNS–Namen der Rechner (vgl. Kapitel 2.2.2 auf Seite 15). Gibt es die Möglichkeit, eine eindeutige Zuordnung zwischen Rechnernamen und Personen zu erhalten, so gelten diese Namen ebenfalls als personenbezogene Daten und müssen bearbeitet werden.

Bei speziellen Diensten gibt es unter Umständen weitere personenbezogene Informationen, zum Beispiel Telefonnummern.

Möglichkeiten zur Anonymisierung

Die oben genannten personenbezogenen Daten können auf verschiedene Weise anonymisiert werden. Zuerst ist es notwendig herauszufinden, welche Information verändert werden muss. Hierzu finden entsprechende Reguläre Ausdrücke[4] An-

[3] Gäste- und spezielle Gruppenkonten sind hierbei nicht Gegenstand der Betrachtung.

[4] Reguläre Ausdrücke sind eine generelle Notation zur Beschreibung von Textmustern [Fri00].

wendung. Die IP–Adressen lassen sich über den in den DHCP Konfigurationen festgelegten Bereichen abgleichen. Für das Auffinden von Benutzerkonten kann die Liste der Benutzerkonten aus dem LDAP Verzeichnis verwendet werden[5] (vgl. Kapitel 2.2.2 ab Seite 18).

Bei der Anonymisierung ist jedoch zum Teil nicht festzustellen, ob es sich um einen personenbezogenes Datum oder nicht handelt. Insbesondere die Benutzeridentifikationsnummern lassen sich nur schwer oder gar nicht automatisiert ersetzen.

Nachdem die Informationen gefunden sind, müssen diese geeignet bearbeitet werden. Mit Blick auf eine Auswertung durch Standardsoftware kommt das Löschen der personenbezogenen Daten nicht in Frage, d. h. die Daten müssen verändert werden.

IP–Adressen Für die Veränderung vom IP–Adressen gibt es zwei Möglichkeiten. Erstens kann die IP–Adresse durch den Hash–Wert der IP–Adresse ersetzt werden. Dieses Vorgehen hat den Vorteil, dass im Falle eines Angriffs oder Problemen anhand konkreter Verdachtsmomente die IP–Adresse des Verdächtigen gehasht werden kann. Stimmen die Schlüssel nicht überein, so ist der Verdächtige entlastet. Allerdings birgt dieser Ansatz auch Nachteile. So kann es als vertretbar angesehen werden, alle IP–Adressen[6] zu hashen und die Schlüssel zu vergleichen, da der Aufwand hierfür bei maximal 4.294.967.296 Berechnungen der Hashfunktion und ebensovielen Vergleichen läge. Diese Berechnungenen wären unter Einbeziehung aller IP–Adressen notwendig. Im Normalfall handelt es sich jedoch nur um Teilbereiche der IP–Adressen, welche betrachtet werden müssen. Die Zahl der Berechnungen lässt sich dadurch im Mittel leicht um den Faktor 512 bzw. 131072 reduzieren[7]. Ein weiterer Nachteil ist, dass Hashwerte in ihrem Aussehen den IP–Adressen nicht ähnlich sind, was die Verarbeitung durch Standardsoftware verhindert. Die Sicherheit lässt sich durch hinzunehmen zufälliger Stellen zur IP–Adresse und dem Hashen des entstehenden Wertes erhöhen [8].

[5] Im allgemeinen Fall steht diese Liste nicht zur Verfügung. Für die Umgebung in welcher die Lösung gesucht wird, ist ein jedoch solcher Server vorhanden.

[6] Für IP Version 4 (IPv4) ist dies zutreffend, mit der Version 6 (IPv6) wächst dieser Aufwand enorm an. Im vorliegenden Fall wurde nur von der Version 4 ausgegangen. Weitere Informationen zu IPv4 und IPv6 finden sich in [SB02].

[7] Wenn 8 bzw. 16 Bit der IP Adresse bereits bekannt sind.

[8] Standardmäßig werden auf diese Art die Passwörter in UNIX Systemen gesichert, zum eingegebenen Passwort werden zum Beispiel 12 Bit zufällig hinzugenommen für das Prüfen des Passwortes

Zweitens kann das System eine zufällige Zuordungstabelle erstellen, welche eine eindeutige Abbildung von personenbezogenen IP–Adressen zu Adressen in der Zuordnungstabelle zulässt. Diese Tabelle wird entsprechend der Intervallzeit, welche durch das eingesetzte Auswertungssystem bestimmt wird, aktualisiert. Eine tägliche Erzeugung der Tabelle gewährleistet die Auswertbarkeit der Protokolldaten durch Standardsoftware, ferner lassen sich auch Bezüge zwischen den Daten herstellen. Ein Schwachpunkt ist, dass Unregelmäßigkeiten, welche über die Lebensdauer der Tabelle andauern, nicht festgestellt werden können. Die Tabelle ist nur für die Dauer der Anonymisierung aller Dateien vorhanden, d. h., sie wird nur im Speicher gehalten und für die Zeit der Transformation benutzt. Mit dem Ende der Transformationen wird die Zuordnungstabelle entfernt. Die Daten sind anonymisiert, besitzen jedoch noch alle Relationen zwischen den Informationen.

Da die durch die zweite Variante entstehenden Daten ohne weitere Bearbeitung von Standardsoftware benutzt werden können, wird diese für den Prototyp bevorzugt.

Benutzerkonten Das Verändern der Benutzerkonten funktioniert ähnlich. Anstelle der über das LDAP Verzeichnis gefundenen Benutzerkonten wird ein *Dummystring* in die Protokolldaten eingefügt. Neben dem auf diese Weise entfernten Namen des Benutzerkontos muss auch die Benutzeridentifikationsnummer (*uid*) in den Protokolldaten bearbeitet werden. Benutzerkontoinformationen, welche nicht im LDAP vorhanden sind, bleiben in den Protokolldaten erhalten.

Rechnernamen Neben den IP–Adressen treten auch Rechnernamen in den Protokolldaten auf. Diese werden durch den Domain Name Service, welcher in Kapitel 2.2.2 beschrieben wird, in IP–Adressen übersetzt. Durch diese Verbindung gelten für die Rechnernamen dieselben Auflagen wie für die IP–Adressen, d. h., die Rechnernamen müssen ebenso entfernt werden. Dies kann ebenfalls unter Verwendung einer Zuordnungstabelle geschehen. Um die Relationen der Einträge in den Protokolldaten nicht zu verändern, sollte die Zuordnung IP–Adresse — Rechnername erhalten bleiben. Eine Erweiterung der Zuordnungstabelle der IP–Adressen ist dafür das geeignete Mittel.

sind dann 4096 (2^{12}) Werte zu testen

8.1.3 Reihenfolge der Aktionen

Für das Übertragen und die Vorverarbeitung gibt es wiederum zwei Möglichkeiten: Einerseits kann die Vorverarbeitung auf jedem System erfolgen, und die bereits bearbeiteten Daten werden zum Protokollserver übertragen. Andererseits können die Protokolldaten auch nach ihrer Übertragung zum Protokollserver vorverarbeitet werden.

Mit Blick auf den zu erwartenden Aufwand für die Entwicklung von Vorverarbeitungsprogrammen sowohl für jeden Dienst als auch für jedes Betriebssystem erscheint auch hier der zentralisierte Ansatz sinnvoll. Ein weiterer Aspekt, der für die Vorverarbeitung nur auf dem Protokollserver spricht, ist der Installations– und Wartungsaufwand für die Vorverarbeitungsprogramme auf den Erzeugersystemen.

Da bei dem gewünschten Ansatz jedoch wiederum personenbezogene Daten verarbeitet werden, ist auch hierfür eine entsprechende Regelung notwendig. Ziel ist es, nur anonymisierte bzw. pseudonymisierte Daten für eine Auswertung zur Verfügung zu haben. Um dies zu realisieren, ist die zeitlich beschränkte Speicherung und eine gesicherte Übertragung der Daten notwendig.

8.1.4 Zeitliche Parameter des Systems

Für die Anforderungen an das System ist es ausreichend, die Protokolldaten einmal pro Tag zu empfangen. Dies kann automatisiert außerhalb der üblichen Belastungszeiten der Erzeugersysteme erfolgen. Nachdem die Daten eingetroffen sind, wird mit der Vorverarbeitung begonnen.

Je nach Leistungsfähigkeit des Systems und der Menge der zu verarbeitenden Protokolldaten ist es möglich, dieses Intervall zu verkürzen. Aufgrund der Beschaffenheit der Protokolldateien der einzelnen Dienste und mit Blick auf die mit der Auswertung von Protokolldaten verbundenen Ziele erscheint eine Verlängerung des Intervalls als nicht zweckmäßig.

Bei einer Verkürzung des Intervalls sind neben der Leistungsfähigkeit des Protokollservers insbesondere die Belastung und Beeinträchtigung der Infrastruktur zu berücksichtigen.

8.2 Sicherung der Vertraulichkeit und Integrität der Protokolldaten

Um einen ordnungsgemäßen Ablauf und korrekte Ergebnisse zu gewährleisten, ist die Sicherung der Integrität der Protokolldaten notwendig. Das System geht davon aus, dass die ihm übermittelten Daten nicht manipuliert sind.

Ziel der Vertraulichkeit ist es, durch geeignete Maßnahmen die Einsichtnahme in diese Daten durch Dritte zu unterbinden. Bei der Sicherung der Integrität geht es darum, Daten vor jedweder Veränderung zu schützen.

8.2.1 Vertraulichkeit

Nach dem Entfernen der personenbezogenen Daten sind in den Daten dennoch zahlreiche Informationen über die Infrastruktur, die Organisation von Diensten und das Nutzerverhalten im allgemeinen enthalten. Diese Daten sind sensibel und stellen einen großen Wert aus der Sicht eines potentiellen Angreifers (vgl. Kapitel 4) dar.

Um einen unerwünschten Einblick in die Protokolldaten zu vermeiden, sind die Nutzungsberechtigungen für das System so weit als möglich zu beschränken. Zugriff darf nur ein Systemverwalter und ggf. dessen Stellvertreter erhalten.

Da die Analyse nur auf nicht chiffrierten Dateien stattfinden kann, muss das System weitestgehend vor unautorisierten Zugriffen geschützt werden. Dazu gehört, dass nur die für das System notwendigen Dienste aktiviert sind und regelmäßig Aktualisierungen vorgenommen werden. Des Weiteren darf nur auf gesicherten Kanälen mit dem System kommuniziert werden. Eine Aufstellung der für das System notwendigen Dienste befindet sich im Anhang A.2.1.

Die Vertraulichkeit der Daten ist nicht nur für den Zeitraum der Analyse, sondern auch darüber hinaus zu gewährleisten. Auf die Sicherung der Vertraulichkeit bei der Aufbewahrung wird im Kapitel 8.3 eingegangen.

8.2.2 Integritätssicherung

Nachdem die Daten auf dem System eingetroffen sind, werden sie sofort für die Auswertung vorbereitet; die erhaltenen Originaldaten werden anschließend verworfen. Der letzte Schritt der Vorbereitung der Daten ist das Erzeugen eines eindeutigen Schlüssels für jede Datei. Hierzu werden Hashfunktionen genutzt. Diese werden im Kapitel 5 näher betrachtet.

Nach Abschluss der Vorverarbeitung liegen die zu bearbeitenden Dateien und zu jeder Datei ein eindeutiger Schlüssel vor. Zur Prüfung der Integrität der Dateien werden diese erneut gehasht und der erzeugte Schlüssel mit dem Schlüssel aus der Vorverarbeitung verglichen. Stimmen beide Schlüssel überein, so wurde an der Datei nichts verändert.

Es kann dadurch eine Manipulation an den Daten erkannt werden. Allerdings ist es auf diese Weise nicht möglich, die Veränderung zu verhindern oder wieder rückgängig zu machen. Um eine gezielte Veränderung unmöglich zu machen, wäre es notwendig, die Daten mit kryptographischen Mitteln zu verschlüsseln. Mit Blick auf die Analyse der Protokolldaten erscheint eine Verschlüsselung jedoch nicht zweckmäßig, da die Programme hierfür nicht ausgerüstet sind. Die Daten müssten vor der Analyse wieder entschlüsselt werden, was das Dilemma der gezielten Veränderung nur näher an die Analyse schieben würde.

Denkbar erscheint ebenso eine Mischform aus Hashfunktion und Verschlüsselung. Zu jeder Datei wird wie bereits beschrieben ein eindeutiger Schlüssel erzeugt. Die gleiche Datei wird unter Zuhilfenahme eines *Public–Key–Verfahren* (vgl. Kapitel 5.3) verschlüsselt. Im Ergebnis lägen dann die originale Datei, der eindeutige Schlüssel dieser Datei und das Chiffrat der Datei vor.

Es kommt ein *Public–Key–Verfahren* zum Einsatz, da bei diesem Verfahren nur der öffentliche Schlüssel auf dem System vorhanden ist. Dadurch gibt es keine Möglichkeit, unentdeckt Manipulationen an der chiffrierten Datei vorzunehmen. Der private Schlüssel befindet sich beim Systemverwalter.

Eine Übersicht über den Aufbau der Vorverarbeitungsstufe, welche die Integritätssicherung in Form Hashwerten und Chiffrat vornimmt, gibt Abbildung 8.2.

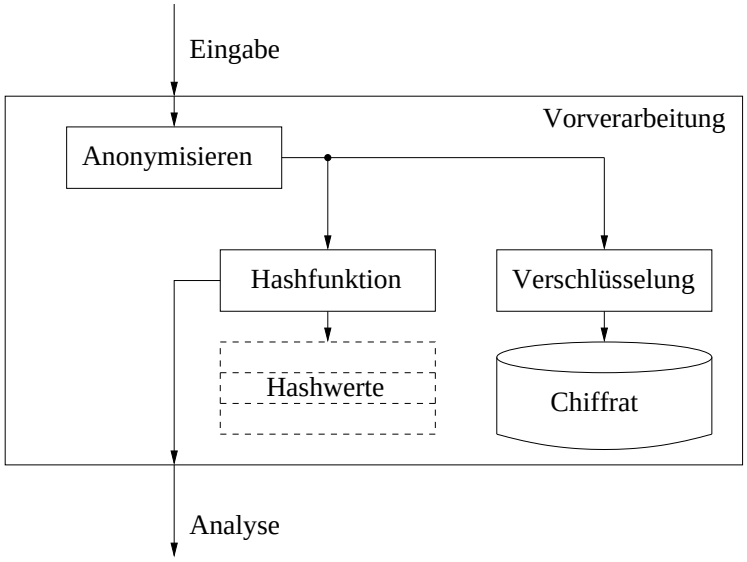

Abbildung 8.2: Übersicht über den Aufbau der Vorverarbeitungsstufe.

Nach dem Ablauf der Analyse wird der Schlüssel mit dem Schlüssel der analysierten Datei verglichen. Gibt es Abweichungen, so wird eine Benachrichtigung für den Systemverwalter erzeugt und alle Daten dieser Auswertung für eine Betrachtung aufbewahrt. Stimmen die Schlüssel überein, wird die Originaldatei für eine Archivierung aufbewahrt. Das Chiffrat wird entfernt, da keine Integritätsverletzung vorlag.

8.3 Aufbewahrung der Protokolldaten

Mit der Anonymisierung entfallen die rechtlichen Beschränkungen zur Aufbewahrung der Protokolldaten. Es bleiben jedoch einige Probleme in diesem Zusammenhang zu klären. Hier sind die Gewährleistung der Vertraulichkeit, der Integrität und nicht zuletzt die durch die Aufbewahrung anfallenden Kosten zu nennen. Ferner ist die Frage, welche Daten aufbewahrt werden sollen, zu klären.

Der einfachste Ansatz ist die Aufbewahrung im Round–Robin–Verfahren, wie sie auch von vielen Diensten angeboten wird. Hierbei werden die Daten in einer Art Ringpuffer gespeichert. Dabei ist dessen Größe durch das Aufbewahrungsintervall und das Analyseintervall festgelegt. Das Analyseintervall ist die Zeit zwischen zwei Analysevorgängen. Daten, die älter sind, werden aus dem Puffer entfernt. Dabei bleiben alle Daten auf der Festplatte. Durch dieses einfache System ist die

notwendige Festplattenkapazität nahezu konstant[9]. Für die Sicherung von Integrität und Vertraulichkeit ergeben sich jedoch bei diesem Verfahren starke Probleme. Um die Integrität zu gewährleisten, müssten die bereits erwähnten Chiffrate ebenfalls aufbewahrt werden, ferner ist die Ausbewahrung der eindeutigen Schlüssel bei dieser Form ebenfalls notwendig. Dabei muß die Integrität der Schlüssel gewährleistet werden. Für die Vertraulichkeit gibt es gegenüber der Analyse keine Veränderungen.

Bei diesem Ansatz gibt es auch keinerlei Schutz vor evtl. Katastrophen oder Hardwaredefekten. Das Prinzip der örtlich getrennten Aufbewahrung von Original und Sicherungsdaten ist verletzt. Aus diesen Gründen kommt dieser Ansatz nicht zum Einsatz.

Eine weitere Möglichkeit ist, den Protokolldatenserver mit einem Bandlaufwerk auszustatten und eine regelmäßige Sicherung auf Band vorzunehmen. Sofern die Bänder ordnungsgemäß aufbewahrt werden, wäre mit diesem Ansatz die Vertraulichkeit sofort gewährleistet. Da sich die Daten auf einem Band jedoch prinzipbedingt verändern lassen, verbessert sich die Integritätssicherung durch dieses Vorgehen nicht. Es müssten weiterhin die Chiffren gespeichert werden.

Neben den erhöhten Kosten durch das Bandlaufwerk bedeutet auch das regelmäßige Wechseln des Bandes einen zusätzlichen Aufwand. Im Gegensatz zum ersten Ansatz bietet dieser einige Vorteile. Die entstehenden Kosten verbieten diese Variante jedoch, sofern weitere mögliche Lösungen existieren.

Die Aufnahme der Daten in eine regelmäßige Sicherung innerhalb einer bestehenden Infrastruktur ist ein ähnlicher Ansatz. Hierbei entstehen jedoch keine zusätzlichen Kosten, da die Infrastruktur für Sicherungskopien bereits vorhanden ist. An dieser Stelle entstehen andere Probleme. Zum einen sind die Protokolldaten in einer solchen Lösung nicht von anderen Daten der Nutzer auf dem Sicherungsmedium isoliert. Ein Angreifer könnte dadurch zusätzliche Informationen über die Daten der Nutzer erhalten, so zum Beispiel deren Herkunft, Alter oder Erzeuger. Zum anderen ist unter Umständen die Umkehrung der Anonymisierung durch diese Vermischung möglich. Dieser Ansatz sollte daher erst nach eingehender Prüfung gewählt werden und findet für dieses System ebenfalls keine Anwendung.

[9] Ausgehend von einer im Mittel konstanten Größe der Protokolldaten.

Zur Sicherung der Integrität, welche an dieser Stelle die größte Schwierigkeit darstellt, ist die Nutzung eines Mediums, welches nur einmal beschrieben werden kann, sinnvoll. Für die in den Tests ermittelten Datenmengen ist die Nutzung von CD–ROM ausreichend, in größeren Infrastrukturen kann die Nutzung von DVD oder anderen Medien notwendig werden.

Die Aufbewahrung mittels eines nur lesbaren Mediums ist für die Umsetzung der Anforderungen die geeignete Lösung. Es werden die Daten vom System für eine Aufbewahrung vorbereitet. Ausgehend von einer wöchentlichen Sicherung werden alle Protokolldateien einer Woche gespeichert, für die Übertragung auf eine CD vorbereitet und je nach Konfiguration auf die CD übertragen. Die Vorbereitung besteht darin, bereits im System eine Image–Datei zu erstellen, welche das gesamte Filesystem und die zu sichernden Protokolldaten der zu erstellenden CD enthält.

Für diese Art der Sicherung ist es nicht unbedingt notwendig, dass das System selbst über einen CD–Brenner verfügt. Die Image–Datei kann zur Sicherung auch zu einem anderen System übertragen werden. Um die Integrität der Image–Datei zu gewährleisten, wird diese unmittelbar nach ihrer Erstellung mit einem eindeutigen Schlüssel versehen. Dieser Schlüssel verbleibt jedoch nicht im System, sondern wird dem Systemverwalter bekannt gemacht. Dies kann zum Beispiel über eine Email erfolgen.
Mit diesem Schlüssel ist es möglich, die Integrität direkt vor dem Übertragen der Image–Datei auf das Sicherungsmedium zu prüfen.

Für die Entscheidung, welche Daten in das Backup müssen, gibt es nur zwei Möglichkeiten: Entweder werden alle Daten gesichert oder nur die, bei denen die Analyse Unregelmäßigkeiten ergeben hat.

Wie bereits in Abbildung 2.1 gezeigt, existieren Überdeckungen in der Einteilung in den Protokolldateneinträgen. Es gibt Unterschiede in der Erkennung durch die Analyseprogramme und in der jeweiligen Konfiguration, d. h., es kann nicht ausgeschlossen werden, dass die Ursache einer Statusmeldung nicht schwerwiegende Fehler in anderen Systemen nach sich sieht. Weiterhin ergeben sich auch durch unterschiedliche Betrachtung und Analyseansätze verschiedene Ergebnisse. Um diese Möglichkeiten offen zu halten, werden alle verfügbaren Daten gesichert.
Für die Hardwareausstattung des Systems bedeutet dies, dass genügend Speicherka-

pazität für die zu erstellende Image–Datei und die zu sichernden Dateien vorhanden sein muss.

8.4 Konfigurationsparameter des Systems

Neben den festen Größen des Systems gibt es verschiedene Parameter, welche für die Nutzung speziell angepasst werden müssen. Dabei handelt es sich zum einen um Fragen des Kommunikationsflusses und zum anderen um Ablaufparameter innerhalb des Systems. Für die Installation und den Betrieb des Systems in einer bestehenden Infrastruktur sind diese Einstellungen notwendig.

Durch die Konfigurationsparameter wird der Ablauf und die Art der Aktionen der Vorverarbeitung eingestellt. Diese Parameter haben in der Grundinstallation jeweils eine sinnvolle Voreinstellung.

8.4.1 Auswahl des Hashverfahrens

Dieser Parameter ist zum Austausch des Hashverfahrens gedacht. Sollte die verwendete Hashfunktion kompromitiert werden, so stellt deren Einsatz keinen Integritätsschutz mehr dar, ferner kann die Vertraulichkeit an dieser Stelle aufgehoben sein. Da unter Umständen aus einem Schlüssel die Datei wieder hergestellt werden kann.

Ein weiterer Grund ist die Dauer eines Hashvorganges. Wird diese verringert, sinken dadurch auch die Anforderungen an das System und es können Kosten gespart werden.

Die Hashfunktionen werden als Modul in das System integriert, so dass mit vertretbarem Aufwand andere Ein–Weg–Funktionen integriert werden können.

8.4.2 Art der Bereitstellung für Analyse–Software

Diese Parameter regeln, wo die vorverarbeiteten Protokolldaten im Filesystem abgelegt werden und wie diese organisiert sein müssen, damit die Analyseprogramme mit ihnen arbeiten können. Insbesondere sind hier zu nennen:

- Verzeichnisstruktur (z. B. nach Server bzw. Diensten)

- Verzeichnisnamen
- Dateinamen
- spezifische Anforderungen der Analyseprogramme

8.4.3 Maßnahmen zur Anonymisierung und Pseudonymisierung

Parameter für IP–Adressen und Rechnernamen

Wie bereits weiter oben erwähnt, erstellt das System zur Anonymisierung eine Zuordnungstabelle für die IP–Adressen innerhalb der Infrastruktur. Durch diese Parameter kann eine Anpassung an den Analyse–Rhythmus erfolgen. Desweiteren kann der Modus der Zuordnungstabelle angepasst werden, d. h., es gibt die Möglichkeit die IP–Adressen einfach zu mischen. Auf diese Weise werden nur innerhalb der Infrastruktur verwendete Adressen einbezogen. Dann ist es möglich, einen Bereich, aus dem die Adressen entnommen werden sollen, vorzugeben. Dadurch stehen in den Dateien gänzlich andere IP–Adressen als in der Infrastruktur.

Dieser Parameter steuert gleichsam die Zuordnung der Rechnernamen. Dabei bedeuten gleiche IP–Adressen auch gleiche Rechnernamen in den Protokolldaten. Zur Vereinfachung kann der verwendete *Dummystring* für den Rechnernamen angegeben werden.

Neben den Informationen in den Protokolldateien gibt es Dienste, welche bereits in den Namen der Protokolldatei personenbezogene Daten speichern, d. h., mit den getroffenen Einstellungen werden auch die Protokolldateinamen berücksichtigt.

Eine Aufweichung dieser Anonymisierung bietet das System nicht an. Denkbar wäre dies in zwei Varianten, zum einen, dass die Zuordnungstabelle nur die tatsächlich in den Protokolldaten aufgetretenen IP–Adressen mischt, und zum anderen, dass die Zuordnungstabelle eine Eins–zu–Eins–Zuordnung vornimmt, wodurch keine Anonymisierung stattfinden würde.

Einstellungen für die Anonymisierung von Benutzerkonten

Weiterhin ist es möglich, die Ersetzung des Benutzerkontos vorzugeben. Auch hier sind verschiedene Modi möglich. So kann für jedes Konto ein eigener, zufälliger, aber für dieses Analyseintervall fester Wert verwendet werden. Alternativ kann für jedes Benutzerkonto der gleiche Wert in die zu anonymisierenden Protokolldaten einfließen.

In gleicher Weise wird dies für die Benutzeridentifikationsnummer durchgeführt. Es kann ebenfalls ein konstanter Wert angegeben werden, ansonsten wird jedem zufälligen Benutzernamen ein zufälliger Benutzeridentifikator zugeordnet.

8.4.4 Parameter zur Aufbewahrung alter Dateien

Wie bereits im Abschnitt 8.3 dargelegt, stellt die Aufbewahrung einige besondere Anforderungen an das System. Neben diesen Anforderungen gibt es einige veränderliche Größen im Zusammenhang mit der Vorbereitung der Sicherung. Diese Größen können mit der nachfolgend beschriebenen Funktionalität durch Parameter eingestellt werden.

Aufbewahrungsdauer

Die Aufbewahrungsdauer legt fest, nach welcher Zeit eine neue Image–Datei angelegt werden soll. Das Aufbewahrungsintervall ist hierbei die Zeit zwischen zwei Sicherungsvorgängen. Weiterhin kann eingestellt werden, was mit noch vorhandenen Image–Dateien passieren soll, d. h., sofern genügend Festplattenkapazität vorhanden ist, können die alten Image–Dateien weiter im System verbleiben.

Art der Aufbewahrung

Für die Aufbewahrung gibt es verschiedene Möglichkeiten. Zum einen kann nach der Analyse eine Image–Datei erstellt werden, welche nur die Protokolldaten für die aktuelle Analyse enthält. Diese Image–Dateien werden bis zum Erreichen des Sicherungsintervalls aufbewahrt. Zur Erstellung der Sicherung werden diese Image–Dateien dann zu einer Datei vereinigt.

Zum anderen ist die Aufbewahrung bis zur Sicherung in einem gepackten Archiv möglich. Dadurch kann Festplattenkapazität für die Dauer der Aufbewahrung

gespart werden. Dies ist allerdings keine Einsparung, da für die Erstellung der Image–Datei die Daten ungepackt vorliegen müssen.

Die einfachste Möglichkeit ist, die Protokolldaten ohne jegliche Veränderung für die Sicherung vorzuhalten. Dadurch ist ein einfacher Zugriff für die Sicherung möglich. Dabei kann es jedoch wiederum zu Problemen mit der Integrität kommen, da hier die Daten nicht wie in den obigen Varianten mittels Hashfunktion „gesichert“ werden können.

Anordnung der Dateien

Es besteht die Möglichkeit Image–Dateien zu vereinigen, wenn die Datei der letzten Sicherung noch vorhanden und somit noch nicht auf ein Sicherungsmedium übertragen ist, so wird zunächst geprüft, ob die zu sichernden Daten noch auf den Datenträger passen. Ist dies der Fall, so werden diese vereinigt. Zur Benachrichtigung erhält der Systemverwalter den Hash–Wert der alten Image–Datei und den der vereinigten. Sollte diese Wert nicht korrekt sein, so ist dieser Vorgang reversibel.

Weiterhin ist die Anordnung der Verzeichnisse und Dateinamen für die Sicherung getrennt einstellbar.

8.4.5 Art und Umfang der Benachrichtigung

Mit diesen Parametern kann der Systemverwalter die Art und den Umfang möglicher Benachrichtigungen festlegen. Dabei kann das System alle relevanten Informationen in einer Datei im System vorhalten. Ebenso können diese Informationen per Email an angegebene Emailadressen gesendet werden. Ein Verwenden beider Varianten ist ebenso möglich, dann enthält die Email zusätzlich den eindeutigen Schlüssel der auf dem System abgelegten Datei.

Der Umfang der Benachrichtigung kann nach Detailstufen bestimmt werden. Dabei lassen sich gezielt Informationen ein– bzw. ausschalten. Mögliche Stufen sind dabei:

Alle Informationen: Hier berichtet das System nach jedem Analyseintervall, welche Dateien mit welcher Grösse und welchen Hashwerten verarbeitet wurden. Ob Integritätsprobleme aufgetreten sind und wenn ja, wo. Ferner wird der Status des Systems und der Sicherungsdatei gemeldet.

Fehler und Sicherung: In Anlehnung an „Alle Informationen“ werden hier Integritätsprobleme und Fehlermeldungen übertragen. Ferner gibt es zu jedem Analyseintervall auch eine Meldung über den Status des Systems und der Sicherungsdatei.

Nur Fehler: Das System meldet Integritätsprobleme und Fehlermeldungen. Sonst wird nur eine Information über die Sicherungsdatei am Ende des Aufbewahrungsintervalls erzeugt.

Keine Meldung: Hier berichtet das System, je nach Einstellung der Parameter nur in eine Datei innerhalb des Systems. Es werden keine Informationen an den Systemverwalter übertragen. Insbesondere Integritätsverletzungen und Fehler im System können nicht oder nur bedingt nachvollzogen werden. Dieser Möglichkeit ist nicht zu empfehlen.

Das Aufkommen an Berichten richtet sich nach dem Analyse– bzw. Aufbewahrungsintervall. Der Umfang der Meldungen richtet sich nach obiger Einstellung und der Menge der anfallenden Dateien.

Kapitel 9

Implementierung einer Infrastruktur zur Zentralisierung und Auswertung von Protokolldaten

Die Implementierung umfasst einen Prototypen, welcher das vorgestellte Konzept realisiert. Besonderes Augenmerk liegt dabei auf der Anonymisierung und der Integritätssicherung der Protokolldaten.

Eine Anleitung zur Installation und Benutzung des Systems findet sich auf dem der Arbeit beiliegenden Datenträger siehe hierzu Anhang B.

9.1 Entwicklungsumgebung

Das System zur Verarbeitung und Bereitstellung der Protokolldaten wurde in Perl geschrieben. Eine Aufstellung und kurze Beschreibung der hierfür notwendigen Module befindet sich im Anhang A.2.1. Die Entwicklung erfolgte unter dem Betriebssystem *Linux* unter Verwendung der durch den *Vim–Editor*[1] zur Verfügung gestellten Möglichkeiten.

[1] Vi Improved, Clone des auf allen Unix Systemen vorhandenen *vi–Editors*. Der Vim stellt zahlreiche Funktionen zur Programmierung bereit. Weitere Informationen finden sich in [LR99].

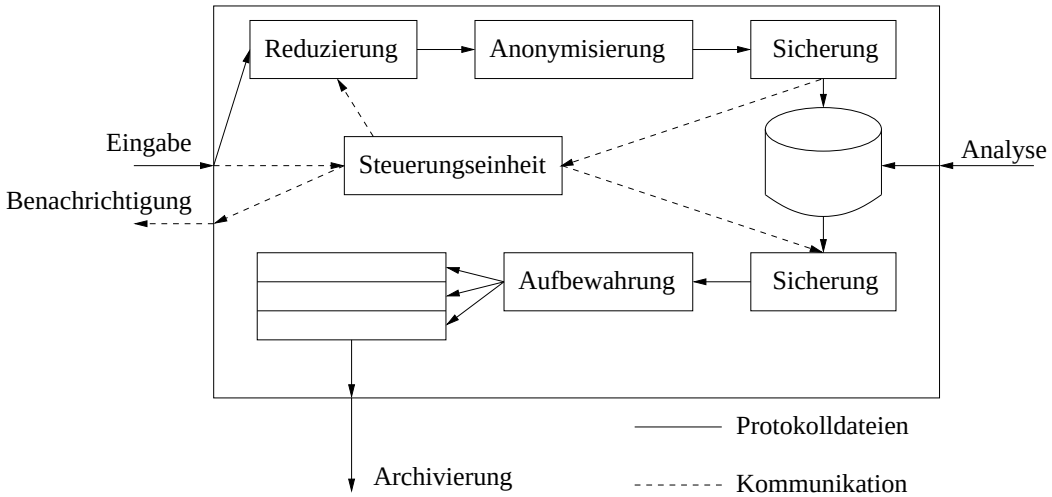

Abbildung 9.1: Schematischer Aufbau des Systems

9.2 Entwurf des Systems

Das System besteht aus einer Steuereinheit, welche die einzelnen Komponenten zur Reduzierung, Anonymisierung, Sicherung und Aufbewahrung steuert. In Abbildung 9.1 wird dieser Aufbau schematisch verdeutlicht. Die Dateien werden dem System übermittelt, anschliessend findet eine Reduzierung statt. Das Modul zur Reduzierung übergibt die verbleibenden Dateien an die Anonymisierung. Nach der Anonymisierung wird eine Integritätssicherung durchgeführt. Nach dieser Vorverarbeitung kann die Analyse stattfinden. Abschließend wird die Integrität der Dateien geprüft und die Vorbereitungen zur Aufbewahrung der Daten getroffen.

9.2.1 Steuereinheit

Die Steuereinheit läuft als Systemdienst und prüft ständig, ob dem System neue Daten übermittelt werden. Sind die Daten vollständig auf dem System, so wird durch die Steuereinheit die Reduzierung der Daten initiiert.

Alle Module melden ihren Status an die Steuereinheit. Nach Ablauf der Vorbereitung der Daten, welche aus Reduzieren, Anonymisieren und Sichern besteht, wartet die Steuerungseinheit einen definierbaren Zeitraum, bis der Vorgang der Sicherungsprüfung durch die Sicherungseinheit und die anschließende Aufbewahrung initiiert wird.

Token	Funktionsweise
*	Alle in Dateinamen zulässigen Zeichen mit Ausnahme des Punktes in beliebiger Anzahl
%	Ein nur aus Ziffern bestehender Bestandteil eines Dateinamen
!	Umkehrung der Anfrage, d. h. alle nicht auf das Muster passenden Ergebnisse
+	Das erste Element nach alphabetischer Sortierung der Ergebnismenge
∧	Der Begin einer Pfadangabe (Symbolisiert den linken Rand des Pfades, ist selbst kein Zeichen)

Tabelle 9.1: Anfragetoken zur Konfiguration des Reduzierers

Neben der Ablaufsteuerung ist die Steuereinheit auch für das Versenden von Benachrichtigungen und Informationen an den Systemverwalter zuständig. Beim Beenden des Systems werden die im Speicher vorgehaltenen Daten im Filesystem abgelegt. Über die Steuereinheit kann jederzeit der Ablauf und der Status des Systems kontrolliert werden.

9.2.2 Reduzierer

Neben den auszuwertenden Protokolldaten übermitteln die Protokolldatenerzeuger auch nicht benötigte Dateien, insbesondere Protokolldateien aus vergangenen Analysezeiträumen. Der Reduzierer entfernt die nicht notwendigen Dateien aus den Eingabedaten. Dazu zählen insbesondere Dateien, die in vergangenen Analysezeiträumen bereits bearbeitet wurden, jedoch bedingt durch die Konfiguration der Protokolldatenerzeuger erneut übertragen werden.

Für den Reduzierer existiert eine Konfigurationsdatei, in welcher Muster für die zu reduzierenden Daten eingegeben werden können. Es wurden die in Tabelle 9.1 dargestellten Token verwendet. Diese Token stellen Platzhalter für die von ihnen symbolisierten Zeichen dar, dadurch lassen sich Mengen definieren, für die ein bestimmtes Muster passt.

Nach Beendigung der Reduzierung wird eine Nachricht an die Steuereinheit gesen-

det. War das Entfernen der nicht benötigten Protokolldateien erfolgreich, so wird durch den Reduzierer die Anonymisierung gestartet.

9.2.3 Anonymisierung

Zunächst werden alle Dateien auf zu anonymisierende Daten geprüft. Dabei werden diese Informationen gesammelt. Aus den aus diesen Informationen erstellten Listen und den nach der Konfiguration festgelegten Parametern werden Regeln zur Transformation aufgestellt.

In einem zweiten Durchlauf werden die personenbezogenen Daten anhand der Regeln ersetzt. Im Ergebnis liegen nur noch anonymisierte Daten vor. Das Anonymisierungsmodul sendet seine Ergebnisse an das Steuermodul und startet die Sicherungseinheit.

9.2.4 Sicherungseinheit

Die Sicherungseinheit kommt sowohl in der Vorverarbeitung als auch bei der Aufbewahrung zum Einsatz. (Vgl. Schichten nach Abbildung 8.1). Nach dem Aufruf durch das Anonymisierungsmodul wird für jede Datei ein eindeutiger Schlüssel erzeugt. Dabei kommt derzeit der MD5 Algorithmus zum Einsatz (siehe Kapitel 5).

Nach Abschluss dieser Arbeiten werden die Hashwerte an das Steuerungsmodul übermittelt. Ferner gibt die Sicherungseinheit eine Statusmeldung an das Steuerungsmodul an. Nachdem diese erfolgt sind, beendet sich die Sicherungseinheit.

Auf den vorbereiteten Daten kann die Analyse stattfinden. Nach der Analyse, die per Aufruf von der Steuerungseinheit gestartet wird, wird mit den bekannten Hashwerten die Sicherungsfunktion erneut aufgerufen. Diese prüft, ob die Hashwerte zu den analysierten Dateien passen. Ist dies nicht der Fall, wird dies an die Steuereinheit gemeldet und sowohl die Protokolldateien, als auch deren Chiffrat separat im Filesystem abgelegt. Die vermeintlich fehlerhaften Dateien gehen nicht in die Aufbewahrung ein. Im positiven Fall werden die chiffrierten Daten gelöscht, die Steuereinheit bekommt einen Statusbericht, und das Aufbewahrungsmodul wird gestartet.

9.2.5 Aufbewahrung

Die analysierten Protokolldaten werden bis zum Erreichen des Aufbewahrungsintervalls entsprechend der in den Einstellungen festgelegten Parameter zwischengespeichert. Speziell bedeutet dies, dass von den Daten gepackte Archive angelegt werden. Deren Hashwerte werden an die Steuerungseinheit übermittelt.

Ist das Aufbewahrungsintervall erreicht, werden die Hashwerte von der Steuereinheit abgerufen und die Integrität der Archive geprüft. Gibt es hierbei Probleme, so findet eine Benachrichtigung des Systemverwalters statt. Die Aufbewahrung wird abgebrochen und die betroffenen Dateien gesondert im Filesystem abgelegt. Ist die Integrität gesichert, werden die Archive entpackt und einem CD–Filesystem[2] zugeführt. Zur entstandenen Image–Datei wird mittels Hashfunktion eine eindeutiger Schlüssel gebildet und diese zusammen mit dem Namen und Speicherort der Image–Datei über die Steuerungseinheit in eine Benachrichtigung geschrieben.
Der schematische Aufbau der Aufbewahrungsfunktion wird durch Abbildung 9.2 dargestellt. Es wurde auf die Darstellung der Kommunikation mit anderen Modulen verzichtet.

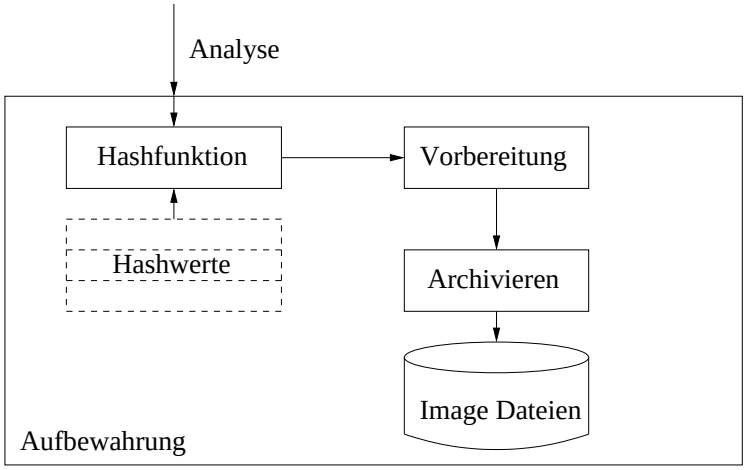

Abbildung 9.2: Die Funktionsweise des Aufbewahrungsmoduls

[2]Es wird der ISO9660 Standard mit der Joliet Erweiterung für lange Dateinamen verwendet.

9.3 Konfiguration des Protokollservers

Neben dem System zur Aufbereitung und Aufbewahrung der Protokolldaten ist auch der Server, auf welchem das System zum Einsatz kommt, zu betrachten.

Bei dem eingesetzten System handelt es sich um einen Standard PC mit dem Betriebssystem *Linux*; die Hardware-Konfiguration für die Testläufe ist der Tabelle 10.1 zu entnehmen.

Dienst	Beschreibung
sshd	Für gesicherten Zugang per Shell und die gesicherte Datenübertragung zum Server
ntp	Zur Synchronisierung der Serverzeit
sendmail	Zum Versenden der Benachrichtigungen

Tabelle 9.2: Für den Betrieb des Systems notwendige Dienste

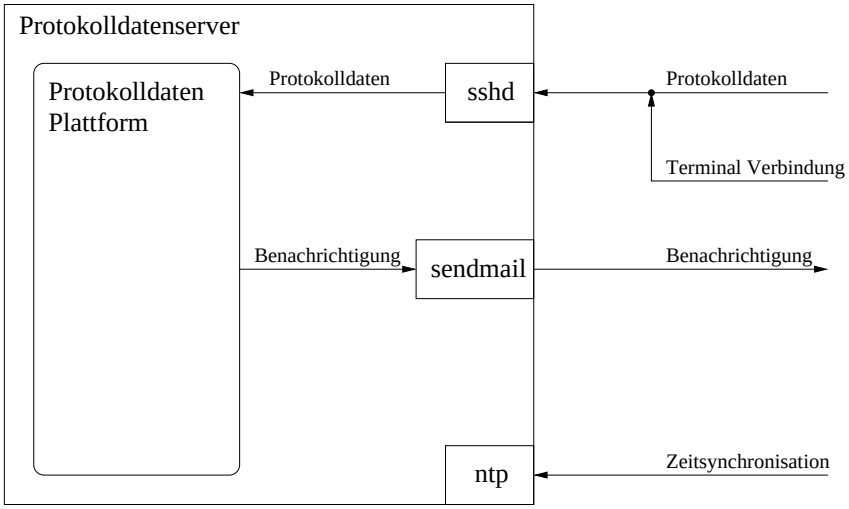

Abbildung 9.3: Kommunikation zwischen der Umgebung und dem Protokolldatenserver

Neben den Hardwareanforderungen an den Server werden verschiedene Dienste benötigt, welche in Tabelle 9.2 dargestellt sind. Die Konfigurationsdateien zu den Diensten befinden sich in Auszügen im Anhang A.2 bzw. vollständig auf dem beiligendem Datenträger (vgl. Anhang B). Weiterhin sind noch Dienste, welche für

den Betrieb notwendig sind, und Dienste, die die Netzanbindung realisieren, auf dem Protokolldatenserver aktiviert.

Die installierten Dienste ermöglichen die Kommunikation des Protokolldatenservers mit der Umgebung im Netzwerk. In Abbildung 9.3 ist diese veranschaulicht.

Kapitel 10

Auswertung

Nach dem Test der Implementierung an realen Daten und der Installation des Systems konnten nachfolgende Systemparameter ermittelt werden. Dabei handelt es sich sowohl um technische Parameter als auch um vertragliche Regelungsempfehlungen zur Einhaltung der rechtlichen Rahmenbedingungen. Das System ist in seiner derzeitigen Form einsatzfähig, allerdings sind einige Fragen vor einer Inbetriebnahme zu klären.

10.1 Rechtliche Regelungen

Für die Nutzung des Systems sind neben den technischen und infrastrukturellen Voraussetzungen auch die rechtlichen Regelungen zu beachten. Das System in seiner derzeitigen Form setzt Datenschutzaspekte unter der speziellen Berücksichtigung der Privatheit der Nutzer um. Das derzeit geltende Recht legt jedoch derartig strenge Maßstäbe an, dass vor dem Einsatz dieses Systems Regelungsbedarf mit den Benutzern besteht. Die notwendigen Regelungen sind in Unternehmen mit dem Betriebsrat durch eine Betriebsvereinbarung festlegbar. Im Hochschulumfeld ist dieses Problem etwas einfacher zu lösen, da zum einen kein Nutzungsverhältnis nach TDG/TDDSG vorliegt und aufgrund der Infrastruktur meist nur die Informationen über Benutzerkonten in den Protokolldaten als kritisch eingestuft werden müssen.

Folgende Regelungen sind deshalb mit den Nutzervertretern für den rechtlich einwandfreien Betrieb notwendig:

1. Erlaubnis zur zeitweisen Speicherung der Protokolldaten auf den

Komponente	Bezeichnung
Prozessor	Pentium II (Deschutes) 333 MHz
Speicher	128 MB
Festplatte	Maxtor 90432D2 4 GB
Netzwerk	100 MBit FastEthernet

Tabelle 10.1: Übersicht zur Hardwarekonfiguration des Protokolldatenservers

Erzeugersystemen auch über die Dauer der Dienstnutzung hinaus. Es findet keine Auswertung dieser Daten statt. Die Dauer der Speicherung richtet sich nach dem Analyseintervall von derzeit einem Tag, d. h. die Einträge der Daten sind höchstens 24 Stunden auf dem System gespeichert.

2. Einverständnis zur gesicherten Übertragung der Protokolldaten an den Protokolldatenserver mit einer sich sofort anschließenden Anonymisierung bzw. Pseudonymisierung der Protokolldaten. Die Orginaldateien werden nach ihrer Anonymisierung automatisch entfernt.

Neben diesen Regelungen gibt es die Möglichkeit, auch auf technischer Seite eine Erleichterung zu schaffen. So ist es denkbar, dass der DHCP–Dienst so konfiguriert wird, dass die Zuordnung Nutzer–IP–Adresse nicht mehr eindeutig ist. Das Problem bleibt jedoch bei Benutzerkonten weiterhin erhalten.

10.2 Technische Parameter

Alle Tests wurden mit den in Anhang A.2.2 aufgeführten Einstellungen der implementierten Softwaremodule durchgeführt. Die Wahl anderer Werte für die Parameter, speziell beim Anonymisierer (IP-Zuordnungstabelle bzw. Nutzerkennzeichen), liefert aufgrund der Struktur des Moduls ähnliche Werte in Bezug auf Laufzeit und Speicheranforderung. Die dem System zugrunde liegende Hardware ist in Tabelle 10.1 gelistet.

Die Daten, welche in Abbildung 6.1 gezeigt werden, bilden die Grundlage für die Tests des Systems. Die Ausschrift eines Systemlaufes wird in Abbildung 10.1 dargestellt. Hierbei meldet sich jedes Modul des Systems bei jeder Meldung mit seinem Namen. Die Steuerungseinheit dient dem Aufruf der Module. In diesem

Lauf waren 512 IP Adressen mit deren zugehörigen DNS Namen und 439 Nutzerkennzeichen in insgesamt 141 Protokolldateien zu suchen und zu anonymisieren.

In den Abbildungen 10.2 und 10.3 wird der Speicherbedarf des Anonymisierers in Abhängigkeit von der Zeit dargestellt. Dabei lassen sich die Operationen des Systems in diese Grafik einordnen. So ist die leichte Steigung am Beginn der Messung durch das Laden des Perl–Interpreter zu erklären. Im ersten Sprung werden die Daten zur Bearbeitung in den Speicher geladen. Die schmalen Bänder stellen die Vorverarbeitung und die Kompilation der regulären Ausdrücke dar. Im Anschluss findet die Anwendung auf die Daten statt, die in Abbildung 10.3 entsprechend der Anzahl der Einträge länger ist, als die in Abbildung 10.2. Bemerkenswert ist hierbei die Ähnlichkeit der beiden Grafiken. Weiterhin lässt sich ablesen, dass der maximale Speicherbedarf des Moduls in etwa dem doppelten der größten Protokolldatei entspricht, was sich auf die Auswertung der verwendeten regulären Ausdrücke zurückführen lässt. Die Spitze im Speicherbedarf am rechten Rand der Grafiken ist nicht durch das System bedingt und bedarf daher keiner Betrachtung.

Für eine Anpassung des Prototypen an die tatsächlichen Gegebenheiten ist auch die Größe der anfallenden Protokolldateien zu berücksichtigen. Um das System ausreichend robust gegen Überflutung zu machen und mit großen Protokolldateien gut umgehen zu können, empfiehlt sich die Zerlegung von großen Dateien in kleinere Blöcke.

10.2.1 Anonymisierung

Für den Aufbau der Zuordnungstabellen wird für jede IP–Adresse ein zugehöriger Domainname im DNS erfragt. Während der Tests war die dafür notwendige Zeit in Abhängigkeit von der Anzahl der DNS Anfragen konstant[1]. Ähnlich ist das Zeitverhalten für die Anfrage der Nutzerkennzeichen am LDAP–Server. Hier wurden für 439 Nutzerkennzeichen und deren interne Aufbereitung etwa 28 Sekunden, benötigt. In Abbildung 10.4 lässt sich erkennen, dass die Erstellung der Zuordnungstabelle mit obigen Angaben ein konstante Zeitdauer benötigt.

[1] Für 512 DNS–Anfragen und deren interne Aufbereitung benötigte das System etwa 10 Sekunden.

```
Steuerer: 1 Konfigurationsinformationen gefunden.
Steuerer: Starte Reduzierer.
Reduzierer: Bearbeite Verzeichnis /tmp/tmp/
Reduzierer: 6 Regeln gefunden
Reduzierer: 141 aus 260 Dateien aufbewahren.
Reduzierer: Lauf beendet.
Steuerer: Starte Anonymisierer.
Anonymisierer: Bearbeite Verzeichnis /tmp/tmp/
Anonymisierer: 6 Konfigurationsinformationen gefunden.
Anonymisierer: Aufbau der IP Zuordnungstabelle.
Anonymisierer: 511 IP-Adressen gefunden.
Anonymisierer: Benoetigte Zeit: 10 Sekunden.
Anonymisierer: Kontaktiere LDAP Server.
Anonymisierer: 439 Nutzerkennzeichen gefunden.
Anonymisierer: Benoetigte Zeit: 28 Sekunden.
Anonymisierer: Bearbeite Dateien.
Anonymisierer: Benoetigte Zeit: 509 Sekunden fuer alle Datei.
Anonymisierer: Benoetigte Gesamtzeit: 547 Sekunden.
Anonymisierer: Lauf beendet.
Steuerer: Starte Sicherer.
Sicherer: Bearbeite Verzeichnis /tmp/tmp/
Sicherer: 2 Konfigurationsinformationen gefunden.
Sicherer: Erzeuge Hashwerte.
Sicherer: Hashwerteerzeugung nach 4 Sekunden beendet.
Sicherer: Lauf beendet.
Steuerer: Starte Analyse.
Hier Startet die Analyse
Steuerer: Starte Sicherer.
Sicherer: Bearbeite Verzeichnis /tmp/tmp/
Sicherer: 2 Konfigurationsinformationen gefunden.
Sicherer: Erzeuge Hashwerte.
Sicherer: Hashwerteerzeugung nach 1 Sekunden beendet.
Sicherer: Lauf beendet.
Steuerer: Starte Aufbewahrer.
Aufbewahrer: Bearbeite Verzeichnis /tmp/tmp/
Aufbewahrer: 3 Konfigurationsinformationen gefunden.
Aufbewahrer: Erzeuge Isodatei /usr/local/log/archiv//20020806-16-21.iso.
Aufbewahrer: Erzeuge Nachricht.
Aufbewahrer: Erzeugung und Benachrichtigung nach 1 Sekunden beendet.
Aufbewahrer: Lauf beendet.
Steuerer: Lauf beendet. Gesamtzeit: 557 Sekunden.
```

Abbildung 10.1: Ausschrift des Systems bei einem Lauf über 17 MB Protokolldaten

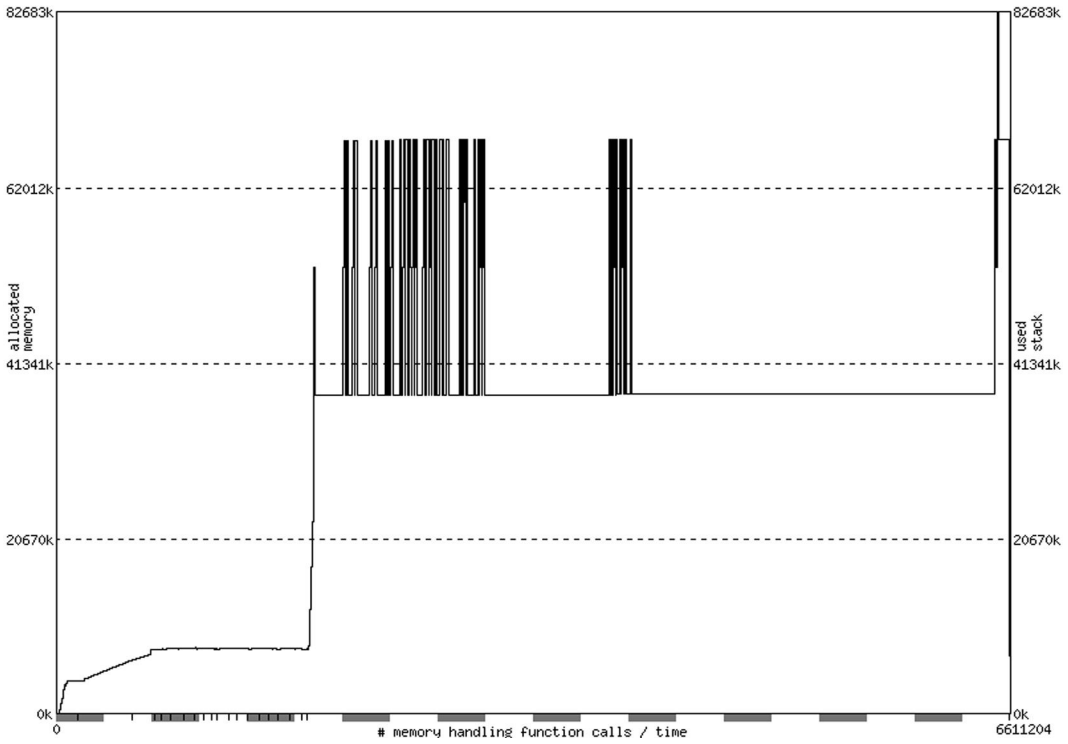

Abbildung 10.2: Speicherbedarf des Systems bei des Auswertung von insgesamt 17 MB Protokolldaten und 256 angegebenen IP Adressen

Die Notwendigkeit zur Reduzierung der Datenmenge wird durch die in Abbildung 10.1 gezeigte Ausschrift des Reduzierers untermauert. In den Tests wurden in Abhängigkeit vom Tag der Auswertung zwischen 30% und 60% der Protokolldaten als nicht für eine Auswertung notwendig erkannt und entfernt. Durch diese Maßnahme reduziert sich die Laufzeit für die Anonymisierung entsprechend.

Es besteht ein Zusammenhang zwischen der Größe der Protokolldatei, der Anzahl der zu anonymisierenden Einträge und die Laufzeit des Anonymisierungsmoduls. Wie Abbildung 10.5 verdeutlicht, ändert sich die Laufzeit entsprechend einer Veränderung beider Parameter. Es besteht auch ein Zusammenhang zwischen der Anzahl der zu anonymisierenden Einträge und der Größe der Protokolldateien. So wächst mit steigender Anzahl der Benutzer eines Netzwerkes auch die Anzahl der Einträge in den Protokolldateien. Zu erkennen ist, dass der Anstieg des Zeitbedarfs im Verhältnis zu Größe der Protokolldateien bei ca. der Hälfte des Speichers des Testsystems (64MB) einknickt. Protokolldaten von über 64 MB zwingen das System auf den Auslagerungsspeicher auszuweichen, welcher deutlich langsamer ist (vgl. Tabelle 10.3). Für Protokolldaten von mehr als 80 MB sind die Testwerte

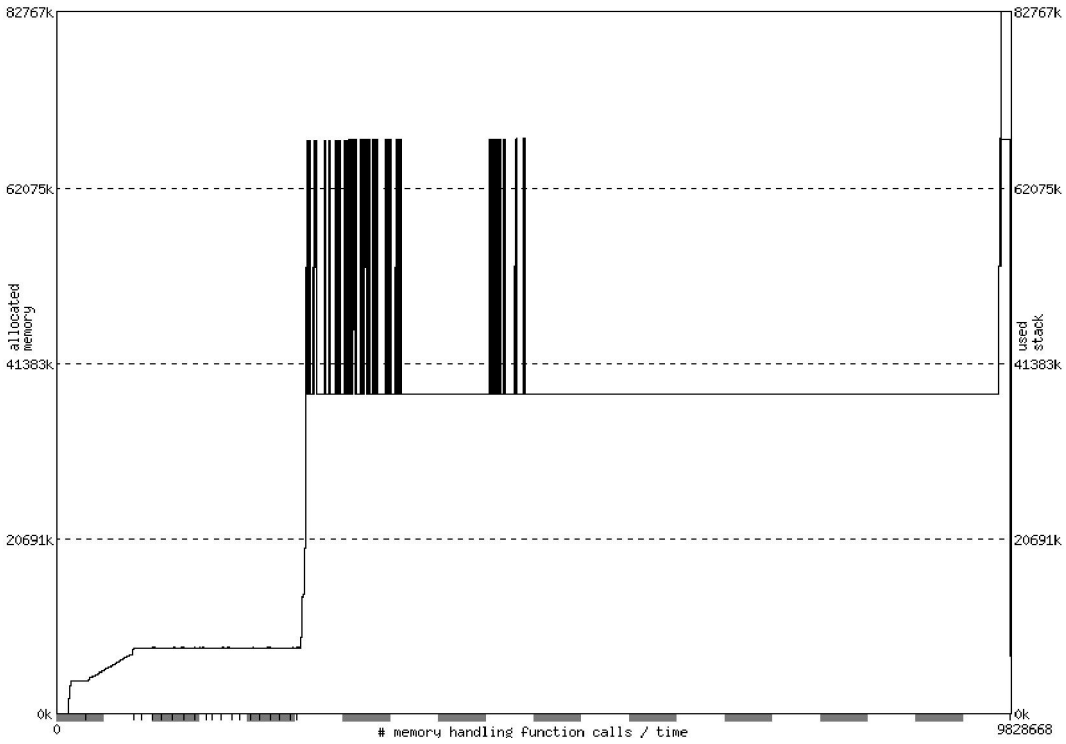

Abbildung 10.3: Speicherbedarf des Systems bei des Auswertung von insgesamt 17 MB Protokolldaten und 511 angegebenen IP Adressen

nicht mehr zu verwenden, da die Anonymisierung an dieser Stelle auf Grund von Speichermangel abbricht bzw. gar nicht startet. Vergleiche hierzu auch Abbildungen 10.6 und 10.7, in diesen sind jeweils die Anstiegsänderungen und auch der Speicherüberlauf zu erkennen.

Zur Feststellung der leistungsbegrenzenden Faktoren des Systems wurde eine Anpassung an eine andere Infrastruktur vorgenommen. Dort lief das System auf einem Rechner mit der in Tabelle 10.2 gezeigten Hardwarekonfiguration. Es wurde eine Protokolldatei mit 10 MB Größe verwendet. Die Anonymisierung dieser Datei benötigte mit 512 IP Adressen ca. 90 Sekunden. Daraus resultiert unter Einbeziehung der 439 Nutzerkennzeichen, welche nur auf dem Protokollserver vorhanden waren ergibt sich folgender Geschwindigkeitszuwachs z:

$$e_t = 2 \cdot 512E; \quad e_s = 2 \cdot 512E + 439E$$
$$g_t = 10MB; \quad g_s = 17MB$$
$$t_t = 90s; \quad t_s = 509s$$

Nach setzen der Konstanten erfolgt die Berechnung der Arbeitsgeschwindigkeit der

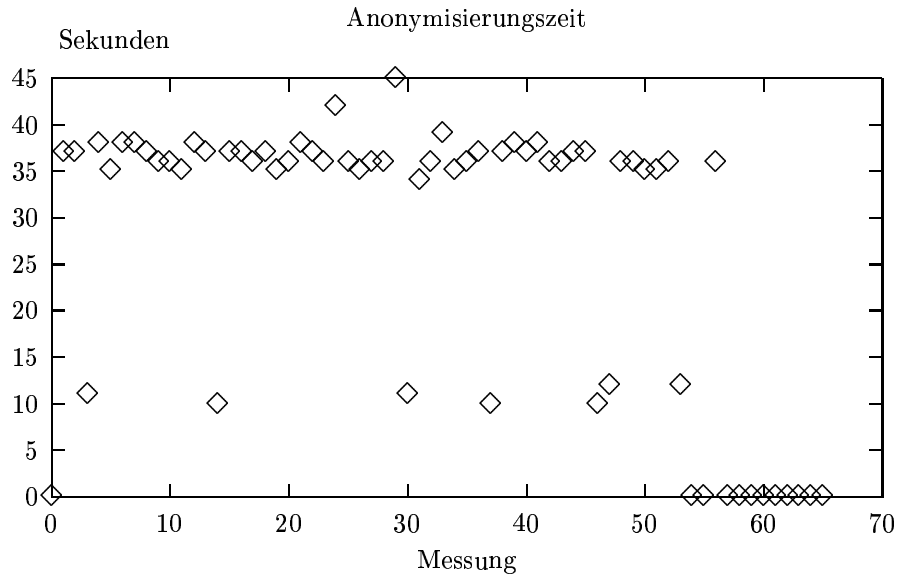

Abbildung 10.4: Zeitdifferenz zwischen der Anonymisierung mit und ohne Erstellung der Zuordnungstabelle

Systeme in Einträgen pro Sekunde:

$$s_t = \frac{e_t}{t_t}$$
$$s_t = 11,4$$
$$s_s = \frac{e_s}{t_s}$$
$$s_s = 2,9$$

Aus dem Verhältnis der Arbeitsgeschwindigkeiten ergibt sich der Geschwindigkeitszuwachs:

$$z = \frac{s_t}{s_s}$$
$$z = 3.9$$

Die Variablen e repräsentieren die Anzahl E der Einträge, welche zu anonymisieren sind. g ist die Menge der getesteten Protokolldaten in MegaByte (MB). t stellt die für den Lauf benötigte Zeit dar und s ist die Arbeitsgeschwindigkeit in Einträgen pro Sekunde. Die verwendeten Indizes bilden die Zuordnung für das jeweilige Testsystem t ist hierbei das Athlon Testsystem und s der Protokolldatenserver. Es wird zunächst berechnet, wieviele Einträge in der Sekunde durch das System bearbeitet

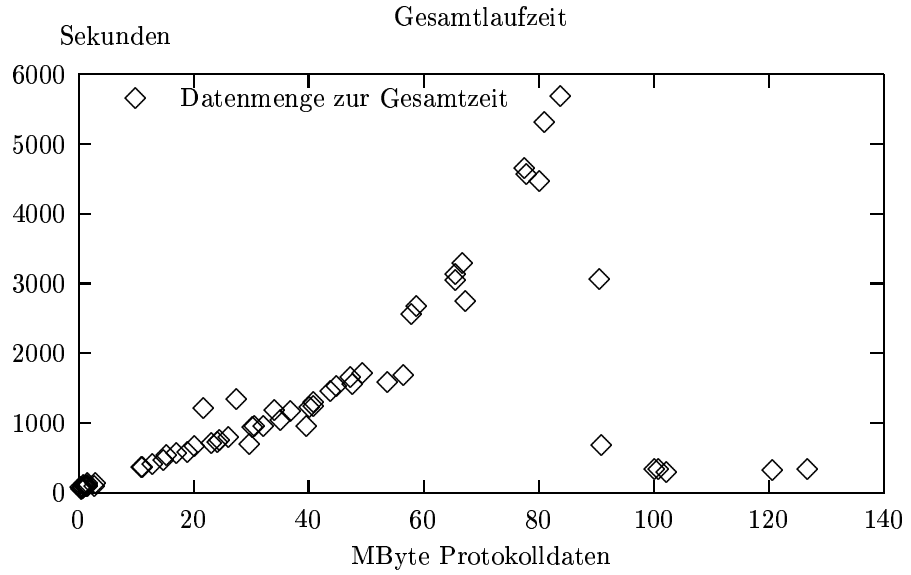

Abbildung 10.5: Zusammenhang zwischen der Laufzeit und der Größe der Protokolldateien.

Komponente	Bezeichnung
Prozessor	Athlon 1 GHz
Speicher	256 MB
Festplatte	Quantum Fireball TM1280A 1,2 GB
Netzwerk	10 MBit Ethernet

Tabelle 10.2: Übersicht zur Hardwarekonfiguration des Testservers zur Bestimmung des leistungsbegrenzenden Faktoren.

werden. Das Verhältnis der beiden Geschwindikeiten ergibt den Geschwindigkeitszuwachs.

Anhand des berechneten Faktors und den anderen Kenngrößen der Testsysteme ist die Leistungssteigerung nicht allein durch die Geschwindigkeitssteigerung des Prozessors zu erklären, wobei dieser mit Blick auf die anderen Kenngrößen des Systems das größte Gewicht zuzuordnen ist. Weitere Faktoren sind der Speicherdurchsatz und die Schreib-/Lese–Geschwindigkeit der Festplatte. Tabelle 10.3 ermöglicht einen Vergleich zwischen den Leistungsdaten der Systeme in Bezug auf Datentransferraten. Als weitere Stör- und Einflussgrößen sind die sonstigen verwendeten Pheripheriegeräte sowie die auf den Testsystemen laufenden Dienste

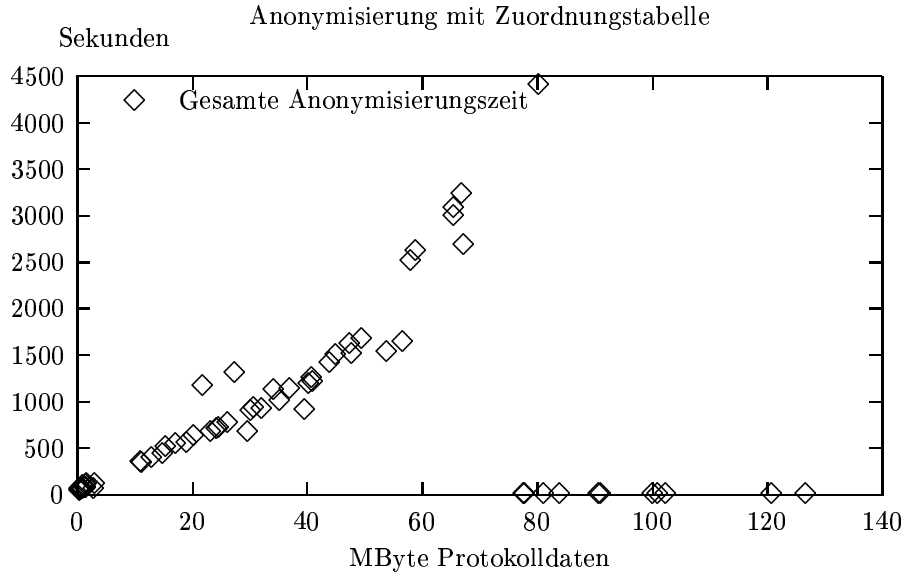

Abbildung 10.6: Verhalten der Gesamtdauer der Anonymisierung mit Aufbau der Zuordnungstabelle zur Größe der Protokolldateien.

	Testserver	Protokolldatenserver
Festplatte	6.6 MB/sek	5.5 MB/sek
Hauptspeicher	160.0 MB/sek	71.5 MB/sek

Tabelle 10.3: Vergleich des Datendurchsatzes von Festplatte und Hauptspeicher zwischen den beiden Testsystemen

zu nennen.

Die Anzahl der Dateien hat keinen Einfluss auf die Gesamtdauer eines Systemdurchlaufes. In den Tests ergaben sich die in Abbildung 10.8 gezeigten Werte, welche diese Vermutung untermauert.

10.2.2 Sicherung und Aufbewahrung

Die Sicherung ist derzeit so implementiert, dass von jeder Protokolldatei der Hashwert in einer Datei abgelegt wird und anschließend der Hashwert dieser Datei an den Systemverwalter übermittelt wird. Nach der Analyse wird dieser Vorgang erneut ausgeführt, was dem Systemverwalter die Kontrolle ermöglicht.

Für die Sicherung der Dateien vor Manipulationen kann das angestrebte Public–

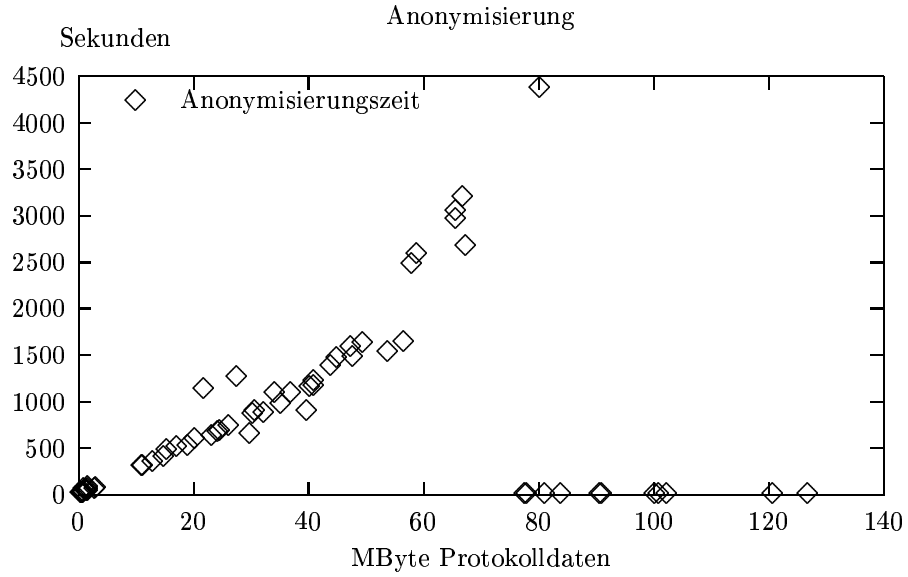

Abbildung 10.7: Verhalten der Dauer der Anonymisierung ohne Aufbau der Zuordnungstabelle zur Größe der Protokolldateien.

Key–Verfahren nur in Kombination mit einem symmetrischen Verschlüsselungsverfahren verwendet werden. Aufgrund der Verschlüsselungsgeschwindigkeit und der Menge der Protokolldaten ist der Weg über ein hybrides Verfahren[2] notwendig.

Der Verbleib der verschlüsselten Dateien auf dem System stellt ein Problem dar. Dort sind die Dateien zwar vor gezielter Manipulation geschützt; ein Angreifer kann diese Dateien jedoch einfach ersetzen. Des Weiteren ist die Absicherung durch organisatorische Maßnahmen zu prüfen. Auf die Implementation der Vertraulichkeit des Dateien wurde deshalb im Prototyp verzichtet.

Ein zu betrachtender Aspekt für weiter gehende Entwicklungen ist die Absicherung der Analyse, welche jedoch nicht vom System vorgenommen werden kann.

Eng verbunden mit der Sicherheit der Protokolldateien sind die Fragen zu deren Aufbewahrung. Hier wird derzeit nach einem zweiten Sicherungslauf von den Protokolldateien über den Aufruf des externen Programmes *mkisofs* ein CD Image erstellt. Der Systemverwalter wird über den Verbleib dieser Datei unterrichtet.

[2]Unter einem hybriden Verfahren in bezug auf Verschlüsselung versteht man die Anwendung eines symmetrischen Verschlüsselungsverfahrens, wobei die Schlüssel über ein asymmetrisches Verfahren ausgetauscht werden.

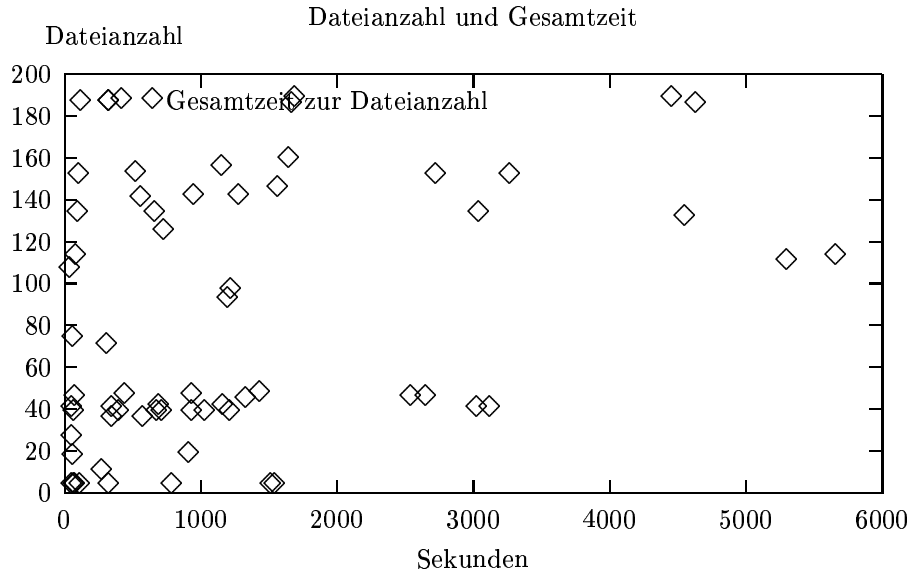

Abbildung 10.8: Vergleich zwischen der Anzahl der Protokolldateien und der Gesamtlaufzeit des Systems über diese Dateien

10.3 Weitere Betrachtungen

Neben den anonymisierten Protokolldateien, bei denen es sich ausschließlich um Textdateien handelt, sind auch die Protokolldateien in anderen Dateiformaten zu betrachten. Derzeit wird dies vom Prototypen nicht unterstützt. Hier kann ggf. ein enormer Aufwand entstehen, da für jeden Dienst ein Konverter notwendig ist, welcher die Dateien zuerst für die Anonymisierung bearbeitbar macht und anschließend eine Rückkonvertierung in das Ursprungsformat vornimmt.

Die Tests ergaben, dass im wesentlichen die in Tabelle 10.4 aufgeführten Dienste für eine Anonymisierung in Frage kommen. Den Hauptanteil stellen hierbei der *squid* und der *Samba* Dienst.

Protokolldatei	Beschreibung
boot	Systemstartinformationen
cron	Terminplaner für automatisierten Ablauf
ksyms	Meldungen des Systemkernes
maillog	Auflistung von Informationen über von Sendmail bearbeiteter Emails
messages	Systemmeldungen
samba	Verteilte Ressourcennutzung
secure	Meldungen des sshd
spooler	Druckerdienste
squid	proxy–cache
wtmp	Informationen zur Benutzung des Systems

Tabelle 10.4: Auf den Testservern installierte Dienste

Kapitel 11

Zusammenfassung

11.1 Umsetzung des Systems

Es wurde ein funktionsfähiger Prototyp entwickelt. Dieser nimmt die Daten von den Protokolldatenerzeugern entgegen und führt die notwendigen Transformationen und Veränderungen für eine Anonymisierung durch. Anschließend wird eine Integritätssicherung der Dateien vorgenommen und diese nach einer Analyse und erneuter Integritätsprüfung einer Aufbewahrung als CD–Image zugeführt. Nach der rechtlichen Absicherung des Systems durch eine Benutzererlaubnis stellt das System eine zentralisierte Plattform für die Protokolldatenauswertung bereit.

Die eingehenden Protokolldaten werden aufbereitet. Dabei wird darauf geachtet, dass alle personenbezogenen Informationen entfernt werden und der semantische Zusammenhang dennoch erhalten bleibt. An den Daten ändert sich aus Sicht der Analysesoftware nichts. Für ein System, was ggf. eine automatische Reaktion auf Grundlage der Protokolldaten auslöst, sind diese Daten jedoch nur in eingeschränktem Umfang nutzbar. So ist es beispielsweise nicht möglich, aufgrund der Einträge Nutzer oder Rechner zu sperren.

Von den Dateien werden Hashwerte erzeugt, welche der Integritätssicherung dienen und dem Systemverwalter mitgeteilt werden. Nach der Sicherung kann die Analyse stattfinden. Abschließend wird von den Daten ein CD-Image angelegt, welches im Filesystem abgelegt wird.

11.2 Mögliche und notwendige Erweiterungen des Prototyps

Durch den Fortschritt der Rechenleistung und die Arbeit vieler Wissenschaftler an alternativen Konzepten für Rechnerarchitekturen ist es wahrscheinlich, dass innerhalb kurzer Zeit einige Verschlüsselungsverfahren als nicht mehr sicher einzustufen sind. Deshalb benötigt das System eine parametrische Wahl des Verschlüsselungsverfahrens, um zur Sicherung der Protokolldaten das Verschlüsselungsverfahren austauschen zu können.

Zum Entfernen von personenbezogener Kommunikation aus binären Protokolldateien, zum Beispiel Windows–Protokolldaten, sind geeignete Konvertierungen zu finden. Hierbei stellt die Auswertbarkeit durch Analysesoftware ein Problem dar, da für diese Systeme die Daten erneut in das binäre Format übertragen werden müssen.

Für die Abschätzung der anfallenden Protokolldaten sind weitere Kennwerte der Infrastruktur zu betrachten. Für die Tests wurden nur zwei exemplarische Protokolldatenerzeuger verwendet. Dieses gibt zwar einen guten Richtwert für eine Vordimensionierung eines Servers muss aber für den konkreten Einsatz detaillierter analysiert werden.

Für die Sicherung der Integrität auf lange Sicht wäre der Ansatz eines „Ewigen Logfiles“ [Don02] sehr nützlich. Dabei geht es darum, Informationen in eine Protokolldatei zu schreiben und diese regelmäßig zu verschlüsseln. Der Hashwert der verschlüsselten Datei wird über ein anderes Medium unabänderlich bekannt gemacht. Hierzu kann zum Beispiel eine Zeitung, in welcher der Hashwert veröffentlicht wird, Verwendung finden. Dadurch ist diese Information allgemein bekannt und kann von einem Angreifer nicht mehr verändert werden.

Eine weitere interessante Erweiterung wäre ein Modul für die Analyse, welches mit der Steuereinheit kommuniziert. Alternativ könnte auch Standardsoftware für eine durch das System vorgenommene Steuerung angepasst werden.

11.3 Vergleich mit anderen Systemen

Derzeit gibt es nur einige wenige Managementsysteme, welche die Funktionalität zur Zentralisierung und Analyse bereitstellen, da diese jedoch die rechtlichen Aspekte nicht berücksichtigen, werden diese nicht eingehender betrachtet. Aufgrund der hohen Kosten und der Probleme im Einsatz in heterogenen Umgebungen sind diese für die Anforderungen nur bedingt geeignet.

Für die Analyse gibt es zahlreiche Programme, welche die Protokolldaten eines oder mehrerer Dienste auswerten. Zur Zentralisierung bieten einige Dienste, wie zum Beispiel der *syslogd*, das Speichern der Protokolldateneinträge auf entfernten Systemen an. Zur Anonymisierung und Pseudonymisierung gibt es derzeit keine Softwaresysteme, welche dies automatisiert durchführen. Für die Bereitstellung von Protokolldaten können auch andere als der implementierte Mechanismus zum Einsatz kommen.

11.4 Fazit

Auf dem Gebiet der Protokolldaten, insbesondere deren Organisation und den rechtlichen Gegebenheiten, bestehen aus technischer Sicht noch viele Aufgaben. So wären Softwaresysteme, welche unter Beachtung der rechtlichen Gegebenheiten selbstständig nur legale Protokolldaten erzeugen, wünschenswert. Der enorme Aufwand für die rechliche Absicherung dieser Daten und die damit verbundenen Kosten stehen nicht im Verhältnis zum dadurch erzielten Nutzen. Desweiteren herrscht ein breites Interesse an personenbezogenen Informationen. Nicht zuletzt deshalb bedarf das europäische und das deutsche Datenschutz– und Medienrecht einer Anpassung an die technischen Gegebenheiten. Die Vielzahl der Regelungen und Gesetze muss verringert und die diese transparenter werden [RPG01]. Im Vordergrund der Betrachtung steht stets das Recht des Einzelnen. Dabei ist jedoch darauf zu achten, dass die technische Realisierung mit vertretbarem Aufwand möglich bleibt. Gleichwohl darf dieses Recht nicht mit der Begründung der technischen Realisierbarkeit untergraben werden [Orw01].

Teil III

Anhang

Anhang A

Testprotokolle, Konfiguration und Quelltext

A.1 Testprotokolle des Systems

A.1.1 Test des Datenaufkommens

Für eine Abschätzung der Hard- und Softwareanforderung ist die Kenntnis des Datenaufkommens notwendig. Die folgende Tabelle stellt diese Informationen für die Testumgebung bereit. Die unter *Squidserver*, *Sambaserver* und *Gesamt* angegebenen Werte repräsentieren Datenmengen in KByte.

Nr	Datum	Squidserver	Sambaserver	Gesamt
1	20020403	45912	5766	51678
2	20020404	53916	5767	59683
3	20020405	64550	5823	70373
4	20020406	74549	5832	80381
5	20020407	75208	5847	81055
6	20020408	38704	5784	44488
7	20020409	50547	5848	56395
8	20020410	61725	5754	67479
9	20020411	72713	6002	78715
11	20020413	102757	6038	108795
12	20020414	102923	6053	108976
13	20020415	44778	5782	50560
14	20020416	55234	6023	61257
15	20020417	67066	5905	72971
16	20020418	77978	6139	84117

17	20020419	85803	6211	92014
18	20020420	101736	6231	107967
19	20020421	102658	6246	108904
20	20020422	49505	5887	55392
21	20020423	62863	6097	68960
22	20020424	80017	6146	86163
23	20020425	91603	6203	97806
24	20020426	108023	6245	114268
25	20020429	53671	4688	58359
26	20020430	82715	4928	87643
32	20020506	56633	21558	78191
63	20020606	108139	27092	135231
64	20020607	128997	27565	156562
65	20020608	151616	27768	179384
66	20020609	151776	27823	179599
67	20020610	57758	27197	84955
68	20020611	76182	27579	103761
69	20020612	105436	27942	133378
70	20020613	135818	28064	163882
71	20020614	158392	25029	183421
72	20020615	176480	25338	201818
74	20020617	64788	24626	89414
75	20020618	85825	24939	110764
76	20020619	104702	25245	129947
79	20020622	144343	25922	170265
80	20020623	146975	25985	172960
81	20020624	67513	24602	92115
82	20020625	90745	24848	115593
83	20020626	113707	25130	138837
84	20020627	130452	25332	155784
85	20020628	143203	25571	168774
86	20020629	155880	25933	181813
87	20020630	156013	25992	182005
88	20020701	70921	24601	95522
89	20020702	86982	24871	111853
90	20020703	100238	25105	125343
91	20020704	110116	25353	135469
92	20020705	123119	25672	148791

```
93          20020706    135644      25952       161596
94          20020707    135835      25984       161819
95          20020708    69576       24737       94313
```

A.1.2 Anfallende Dateien

Die Konfiguration der Muster für den Reduzierer erfordert eine Detailierte Kenntnis der Dateien. Hier eine Aufstellung der Dateinamen einer typischen Übertragung in der Testumgebung.

```
./sqsv/boot.log
./sqsv/boot.log.1
./sqsv/boot.log.2
./sqsv/boot.log.3
./sqsv/boot.log.4
./sqsv/cron
./sqsv/cron.1
./sqsv/cron.2
./sqsv/cron.3
./sqsv/cron.4
./sqsv/dmesg
./sqsv/fax
./sqsv/ksyms.0
./sqsv/ksyms.1
./sqsv/ksyms.2
./sqsv/ksyms.3
./sqsv/lastlog
./sqsv/maillog
./sqsv/maillog.1
./sqsv/maillog.2
./sqsv/maillog.3
./sqsv/maillog.4
./sqsv/messages
./sqsv/messages.1
./sqsv/messages.2
./sqsv/messages.3
./sqsv/messages.4
./sqsv/rpmpkgs
./sqsv/rpmpkgs.1
./sqsv/rpmpkgs.2
./sqsv/rpmpkgs.3
./sqsv/rpmpkgs.4
./sqsv/secure
./sqsv/secure.1
./sqsv/secure.2
./sqsv/secure.3
./sqsv/secure.4
./sqsv/squid/access.log
./sqsv/squid/access.log.1.gz
./sqsv/squid/access.log.2.gz
./sqsv/squid/cache.log
./sqsv/squid/cache.log.1.gz
./sqsv/squid/cache.log.2.gz
./sqsv/wtmp
./sqsv/wtmp.1
./sqsv/XFree86.0.log
./sasv/samba/log.
./sasv/samba/log.acnb008
./sasv/samba/log.acnb012
./sasv/samba/log.acpc023
./sasv/samba/log.b7pc58
./sasv/samba/log.bnb097
./sasv/samba/log.bscw-test
./sasv/samba/log.c5pc192
./sasv/samba/log.c5pc217
./sasv/samba/log.c5pc220
./sasv/samba/log.lpznb042
./sasv/samba/log.lpznb046
./sasv/samba/log.lpznb048
./sasv/samba/log.lpznb049
./sasv/samba/log.lpznb051
./sasv/samba/log.lpznb052
./sasv/samba/log.lpznb067
./sasv/samba/log.lpznb068
./sasv/samba/log.lpznb069
./sasv/samba/log.lpznb069.old
```

```
./sasv/samba/log.lpznb095          ./sasv/samba/log.lpzpc309
./sasv/samba/log.lpznb097          ./sasv/samba/log.lpzpc310
./sasv/samba/log.lpznb097.old      ./sasv/samba/log.lpzpc311
./sasv/samba/log.lpznb098          ./sasv/samba/log.lpzsamba
./sasv/samba/log.lpznb099          ./sasv/samba/log.lschul05
./sasv/samba/log.lpznb100          ./sasv/samba/log.lschul13
./sasv/samba/log.lpznb100.old      ./sasv/samba/log.lschul13.old
./sasv/samba/log.lpzpc201          ./sasv/samba/log.lteach01
./sasv/samba/log.lpzpc202          ./sasv/samba/log.nmbd
./sasv/samba/log.lpzpc203          ./sasv/samba/log.nmbd.1
./sasv/samba/log.lpzpc204          ./sasv/samba/log.nmbd.2
./sasv/samba/log.lpzpc204.old      ./sasv/samba/log.nmbd.3
./sasv/samba/log.lpzpc205          ./sasv/samba/log.smbd
./sasv/samba/log.lpzpc206          ./sasv/samba/log.smbd.1
./sasv/samba/log.lpzpc212          ./sasv/samba/log.smbd.2
./sasv/samba/log.lpzpc212.old      ./sasv/samba/log.smbd.3
./sasv/samba/log.lpzpc215          ./sasv/samba/log.smbmount
./sasv/samba/log.lpzpc217          ./sasv/samba/log.smbmount.old
./sasv/samba/log.lpzpc248          ./sasv/samba/log.snagate1
./sasv/samba/log.lpzpc249          ./sasv/samba/log.t6100
./sasv/samba/log.lpzpc259          ./sasv/secure
./sasv/samba/log.lpzpc259.old      ./sasv/secure.1
./sasv/samba/log.lpzpc261          ./sasv/secure.2
./sasv/samba/log.lpzpc277          ./sasv/secure.3
./sasv/samba/log.lpzpc278          ./sasv/secure.4
./sasv/samba/log.lpzpc279          ./sasv/spooler
./sasv/samba/log.lpzpc279.old      ./sasv/spooler.1
./sasv/samba/log.lpzpc280          ./sasv/spooler.2
./sasv/samba/log.lpzpc281          ./sasv/spooler.3
./sasv/samba/log.lpzpc282          ./sasv/spooler.4
./sasv/samba/log.lpzpc282.old      ./sasv/wtmp
./sasv/samba/log.lpzpc283          ./sasv/wtmp.1
./sasv/samba/log.lpzpc285          ./sasv/xferlog
./sasv/samba/log.lpzpc294          ./sasv/xferlog.1
./sasv/samba/log.lpzpc295          ./sasv/xferlog.2
./sasv/samba/log.lpzpc299          ./sasv/xferlog.3
./sasv/samba/log.lpzpc300          ./sasv/xferlog.4
./sasv/samba/log.lpzpc300.old
```

A.1.3 Test des Komplettsystems

Nachfolgend das Testprotokoll von zwei exemplarischen Systemläufen aus einem Langzeittest mit 95 Datensätze. Diese Daten bilden die Grundlage für die in Kapitel 10 gezeigten

Abbildungen 10.4, 10.5, 10.6, 10.7 und 10.8.

```
/usr/local/log/archiv/a01072002.tgz
Steuerer: 1 Konfigurationsinformationen gefunden.
Steuerer: Starte Reduzierer.
Reduzierer: Bearbeite Verzeichnis /tmp/tmp/
Reduzierer: 6 Regeln gefunden
Reduzierer: 134 aus 153 Dateien aufbewahren.
Reduzierer: Lauf beendet.
Steuerer: Starte Anonymisierer.
Anonymisierer: Bearbeite Verzeichnis /tmp/tmp/
Anonymisierer: 6 Konfigurationsinformationen gefunden.
Anonymisierer: Aufbau der IP Zuordnungstabelle.
Anonymisierer: 511 IP-Adressen gefunden.
Anonymisierer: Benoetigte Zeit: 9 Sekunden.
Anonymisierer: Kontaktiere LDAP Server.
Anonymisierer: 439 Nutzerkennzeichen gefunden.
Anonymisierer: Benoetigte Zeit: 29 Sekunden.
Anonymisierer: Bearbeite Dateien.
Anonymisierer: Benoetigte Zeit: 52 Sekunden fuer alle Datei.
Anonymisierer: Benoetigte Gesamtzeit: 90 Sekunden.
Anonymisierer: Lauf beendet.
Steuerer: Starte Sicherer.
Sicherer: Bearbeite Verzeichnis /tmp/tmp/
Sicherer: 2 Konfigurationsinformationen gefunden.
Sicherer: Erzeuge Hashwerte.
Sicherer: Hashwerteerzeugung nach 0 Sekunden beendet.
Sicherer: Lauf beendet.
Steuerer: Starte Analyse.
Hier Startet die Analyse
Steuerer: Starte Sicherer.
Sicherer: Bearbeite Verzeichnis /tmp/tmp/
Sicherer: 2 Konfigurationsinformationen gefunden.
Sicherer: Erzeuge Hashwerte.
Sicherer: Hashwerteerzeugung nach 1 Sekunden beendet.
Sicherer: Lauf beendet.
Steuerer: Starte Aufbewahrer.
Aufbewahrer: Bearbeite Verzeichnis /tmp/tmp/
Aufbewahrer: 3 Konfigurationsinformationen gefunden.
```

```
Aufbewahrer: Erzeuge Isodatei /usr/local/log/archiv/
  20020806-16-09.iso.
Aufbewahrer: Erzeuge Nachricht.
Aufbewahrer: Erzeugung und Benachrichtigung nach 0
  Sekunden beendet.
Aufbewahrer: Lauf beendet.
Steuerer: Lauf beendet. Gesamtzeit: 97 Sekunden.
-rw-rw-r--    1 loguser  loguser    917504 Aug  6 18:09
  /usr/local/log/archiv/20020806-16-09.iso

---
Steuerer: 1 Konfigurationsinformationen gefunden.
Steuerer: Starte Reduzierer.
Reduzierer: Bearbeite Verzeichnis /tmp/tmp/
Reduzierer: 6 Regeln gefunden
Reduzierer: 142 aus 262 Dateien aufbewahren.
Reduzierer: Lauf beendet.
Steuerer: Starte Anonymisierer.
Anonymisierer: Bearbeite Verzeichnis /tmp/tmp/
Anonymisierer: 6 Konfigurationsinformationen gefunden.
Anonymisierer: Aufbau der IP Zuordnungstabelle.
Anonymisierer: 511 IP-Adressen gefunden.
Anonymisierer: Benoetigte Zeit: 9 Sekunden.
Anonymisierer: Kontaktiere LDAP Server.
Anonymisierer: 439 Nutzerkennzeichen gefunden.
Anonymisierer: Benoetigte Zeit: 27 Sekunden.
Anonymisierer: Bearbeite Dateien.
Anonymisierer: Benoetigte Zeit: 896 Sekunden fuer alle Datei.
Anonymisierer: Benoetigte Gesamtzeit: 932 Sekunden.
Anonymisierer: Lauf beendet.
Steuerer: Starte Sicherer.
Sicherer: Bearbeite Verzeichnis /tmp/tmp/
Sicherer: 2 Konfigurationsinformationen gefunden.
Sicherer: Erzeuge Hashwerte.
Sicherer: Hashwerteerzeugung nach 3 Sekunden beendet.
Sicherer: Lauf beendet.
Steuerer: Starte Analyse.
Hier Startet die Analyse
```

```
Steuerer: Starte Sicherer.
Sicherer: Bearbeite Verzeichnis /tmp/tmp/
Sicherer: 2 Konfigurationsinformationen gefunden.
Sicherer: Erzeuge Hashwerte.
Sicherer: Hashwerteerzeugung nach 2 Sekunden beendet.
Sicherer: Lauf beendet.
Steuerer: Starte Aufbewahrer.
Aufbewahrer: Bearbeite Verzeichnis /tmp/tmp/
Aufbewahrer: 3 Konfigurationsinformationen gefunden.
Aufbewahrer: Erzeuge Isodatei /usr/local/log/archiv/
  20020806-16-45.iso.
Aufbewahrer: Erzeuge Nachricht.
Aufbewahrer: Erzeugung und Benachrichtigung nach 4 Sekunden beendet.
Aufbewahrer: Lauf beendet.
Steuerer: Lauf beendet. Gesamtzeit: 949 Sekunden.
-rw-rw-r--    1 loguser  loguser  30703616 Aug  6 18:45
  /usr/local/log/archiv/20020806-16-45.iso
```

A.1.4 Speichertest des Anonymisers

Die in Abbildung 10.2 und 10.3 dargestellten Zusammenhänge wurden durch den Einsatz des Programmes *memusage* herausgefunden. Die Testläufe sind nachfolgend aufgezeigt.

```
Anonymisierer:

17 MB Logdateien (191 Files)
256 IP Adressen

[loguser@lpzpc298 bin]$ memusage -p 256ip17Manon.png
  -t -T ./anon.pl /tmp/tmp/
Anonymisierer: Bearbeite Verzeichnis /tmp/tmp/
Anonymisierer: 5 Konfigurationsinformationen gefunden.
Anonymisierer: Aufbau der IP Zuordnungstabelle.
Anonymisierer: 255 IP-Adressen gefunden.
Anonymisierer: Benoetigte Zeit: 7 Sekunden.
Anonymisierer: Kontaktiere LDAP Server.
Anonymisierer: 439 Nutzerkennzeichen gefunden.
Anonymisierer: Benoetigte Zeit: 29 Sekunden.
Anonymisierer: Bearbeite Dateien.
```

```
Anonymisierer: Benoetigte Zeit: 364 Sekunden fuer alle
  Datei.
Anonymisierer: Benoetigte Gesamtzeit: 400 Sekunden.
Anonymisierer: Lauf beendet.

Memory usage summary: heap total: 16255862600,
  heap peak: 84666855, stack peak: 25548
         total calls   total memory   failed calls
 malloc|    2917600     1385738270              0
realloc|     846849    14870085514              0
  (in place: 182957, dec: 4409)
 calloc|         30          38816              0
   free|    2816440     1468719019


---
17 MB Logdateien (191 Files)
512 IP Adressen

[loguser@lpzpc298 bin]$ memusage -p 512ip17Manon.png
  -t -T ./anon.pl /tmp/tmp/
Anonymisierer: Bearbeite Verzeichnis /tmp/tmp/
Anonymisierer: 6 Konfigurationsinformationen gefunden.
Anonymisierer: Aufbau der IP Zuordnungstabelle.
Anonymisierer: 511 IP-Adressen gefunden.
Anonymisierer: Benoetigte Zeit: 11 Sekunden.
Anonymisierer: Kontaktiere LDAP Server.
Anonymisierer: 439 Nutzerkennzeichen gefunden.
Anonymisierer: Benoetigte Zeit: 28 Sekunden.
Anonymisierer: Bearbeite Dateien.
Anonymisierer: Benoetigte Zeit: 558 Sekunden fuer alle
  Datei.
Anonymisierer: Benoetigte Gesamtzeit: 597 Sekunden.
Anonymisierer: Lauf beendet.

Memory usage summary: heap total: 19266737124,
  heap peak: 84752378, stack peak: 30472
         total calls   total memory   failed calls
```

```
 malloc|    4295692     1517833853             0
realloc|    1293319    17748864455             0
  (in place: 384134, dec: 4409)
 calloc|         30          38816             0
   free|    4192930     1639963978
```

A.2 Konfiguration

A.2.1 Software

Auf dem Protokolldatenserver wurde eine Installation des Betriebssystems *RedHat Linux 7.2* vorgenommen. Als Kernel kam der von diesem System mitgelieferte Linuxkernel 2.4.7 zum Einsatz. Es werden neben den Standardpaketen zusätzlich die Dienste *sendmail*, *ntp*, *sshd*, *openldap* und die in Tabelle A.1 gelisteten Perlmodule benötigt.
Neben diesen Modulen ist die Konfiguration des Mail Transport Agenten *sendmail* notwendig, die verwendete Konfiguration ist hier in der M4 Macrosprache aufgeführt. Die Konfiguration wird für die Kommunikation des Systems per Email benötigt.

```
divert(-1)
dnl This is the macro config file used to generate
dnl the /etc/sendmail.cf
dnl file. If you modify thei file you will have to
dnl regenerate the
dnl /etc/sendmail.cf by running this macro config
dnl through the m4 preprocessor:
dnl
dnl        m4 /etc/sendmail.mc > /etc/sendmail.cf
dnl
dnl You will need to have the sendmail-cf package installed
dnl for this to work.
include(‘/usr/lib/sendmail-cf/m4/cf.m4’)dnl
define(‘confDEF_USER_ID’,‘‘8:12’’)dnl
VERSIONID(‘sendmail setup for mail hub lpzpc205’)dnl
OSTYPE(‘linux’)dnl
undefine(‘UUCP_RELAY’)dnl
undefine(‘BITNET_RELAY’)dnl
define(‘confAUTO_REBUILD’)dnl
define(‘confTO_CONNECT’, ‘1m’)dnl
define(‘confTRY_NULL_MX_LIST’,true)dnl
```

```
define(‘confDONT_PROBE_INTERFACES’,true)dnl
define(‘confDONT_EXPAND_CNAMES’,true)dnl
FEATURE(‘smrsh’,‘/usr/sbin/smrsh’)dnl
dnl FEATURE(‘redirect’)dnl
FEATURE(‘nocanonify’)dnl
FEATURE(‘always_add_domain’)dnl
FEATURE(‘relay_entire_domain’)dnl
dnl FEATURE(‘use_cw_file’)dnl
dnl
FEATURE(‘local_procmail’)dnl
define(‘PROCMAIL_MAILER_PATH’,‘/usr/bin/procmail’)dnl
define(‘ALIAS_FILE’, ‘/etc/aliases’)dnl
dnl
MAILER(smtp)dnl
MAILER(procmail)dnl
define(‘SMART_HOST’,‘smtp:[mail1.c1.dsh.de]’)dnl
dnl
MASQUERADE_AS(t-systems.com)dnl
MASQUERADE_DOMAIN(c1.dsh.de)dnl
FEATURE(‘masquerade_envelope’)dnl
FEATURE(‘allmasquerade’)dnl
FEATURE(‘masquerade_entire_domain’)dnl
EXPOSED_USER(‘root’)dnl
dnl
dnl FEATURE(‘mailertable’)dnl
FEATURE(‘virtusertable’,‘hash -o /etc/mail/virtusertable’)dnl
FEATURE(‘access_db’)dnl
FEATURE(‘blacklist_recipients’)dnl

dnl We strongly recommend to comment this one out if you want
dnl to protect yourself from spam. However, the laptop and
dnl users on computers that do not hav 24x7 DNS do need this.
FEATURE(‘accept_unresolvable_domains’)dnl
FEATURE(‘accept_unqualified_senders’)dnl
dnl FEATURE(‘relay_based_on_MX’)

LOCAL_CONFIG
Cw localhost lpzpc298.c1.dsh.de
```

```
dnl eof
```

A.2.2 Testkonfigurationen des Systems

Nachfolgend sind die in den Tests für die Systemkomponenten des Prototyps verwendeten Konfigurationen aufgeführt.

Reduzierer

```
# Muster f"ur bestimmte Dienste festlegen
# Legende:
#
#   * Buchstand oder Ziffen, Binde-- oder Unterstriche in
#     beliebiger Anzahl
#   % Ziffer oder Zahl
#   ! alle ausser auf dieses Muster passende
#   + Das erste Ergebnis nach alphapetischer Sortierung
#
#   Prefixe koennen Verzeichnisnamen sein. es duerfen auch
#   Namensbestandteile in den Mustern vorkommen.
#
#   ^ Zeilenanfang
#
# Squid
squid/*.*

# Samba
samba/!*.*.*

# ksyms
+ksyms.%


# Weitere Dateien
boot.log
XFree86.0.log

# Default
```

```
^/*/*/*/*
```

Anonymisierer

```
# IP - Adressen / Hostnamen
#
# Modus
#   ipmod = MIX
#         die selben IP und Hostnamen werden
#         in anderer Reihenfolge verwendet
#
#   ipmod = 123.123.123.123-123.123.123.124
#         ein IP Bereich wird zur Abbildung verwendet
#         (Derzeit nicht Implementiert, wirkt wie MIX)
#
#   ipmod = 127.0.0.1
#         Alle IP-Adressen durch diese Ersetzen
#
ipmod = MIX

# Zu anonymisierenden IP-Adress Bereiche
# Die Variable anoip kann beliebig oft auftreten
#
#   anoip = 172.20.16.*
#   anoip = 172.20.17.10-172.20.17.250
#   anoip = 172.20.7.1
anoip = 172.20.16.*
anoip = 172.20.17.*
#anoip = 172.20.16.10-172.20.16.100
#anoip = 172.20.16.5

# Nutzerkennzeichen
#
# Angabe des LDAP Server
#   ldaps = ldap.c1.dsh.de
ldaps = ldapmaster.c1.dsh.de

# Angabe des Suchpfades
#   lsuch =
```

```
lsuch = dc=c1,dc=dsh,dc=de
# Derzeit nicht Ausgewertet.

# Modus
#   nkmod = MIX
#         die selben Nutzerkennzeichen und NID
#         in anderer Reihenfolge werden verwendet
#
#   nkmod = NUM
#         die Nutzerkennzeichen werden durch ein
#         nummeriertes Nutzerkennzeichen ersetzt
#
#   nkmod = <wort>
#         alle Nutzerkennzeichen werden durch wort
#         ersetzt
nkmod = NUM
```

Sicherer

```
# Sicherer
#
# Benachrichtigungsmail
#
#   mail=
mail=Thomas.Steinbach@t-systems.com

#
# Standardbetreff
#
#  topic=
#
#topic=Sicherungsmail

#
```

```
# Datei der Hashwerte
#
# hash=
#
hash=/home/loguser/bin/hashes.txt
```

Aufbewahrer

```
# Aufbewahrer
#
# Benachrichtigungsmail
#
#   mail=
mail=Thomas.Steinbach@t-systems.com

# Image Kommando
#
# image=
image=mkisofs -R -L -quiet -o

#
# Verzeichnis der IsoImagedateien
#
# iso=
#
iso=/usr/local/log/archiv/
```

Steuerer

```
#
# Steuerer
#
```

```
# Kommando zum Start einer Analyse
#
# analyse=
analyse=echo "Hier Startet die Analyse"
```

A.3 Quelltexte

A.3.1 Testskripte

Datensammlung

Dieses Skript dient dem Hashen und Archivieren der von den Servern übermittelten Protokolldaten. Insbesondere für die Testläufe war das vorhalten der Protokolldaten über einen bestimmten Zeitraum notwendig.

```
#!/bin/bash

cd ~loguser/
datum=‘date +"%d%m%Y"‘

mkdir /tmp/archiv_$datum
cp -a logfiles/* /tmp/archiv_$datum/
rm -rf logfiles/*/*
bin/reduzierer.pl /tmp/archiv_$datum
tar -czf archiv/a$datum.tgz /tmp/archiv_$datum
touch archiv/hash.txt
md5sum archiv/a$datum.tgz >>archiv/hash.txt
touch hashs/‘md5sum archiv/hash.txt|awk ’{print $1;}’‘
rm -rf /tmp/archiv_$datum
cd -
```

Grafiken

Zur Ermittlung der Zusammenhänge in Bezug auf Datenmengen, Gesamtzeit und Anonymisierungszeit und deren Grafischer Darstellung wurden folgende Skripte in Verbindung mit dem Programm *gnuplot* verwendet.

Datenmenge zur Gesamtzeit

```
set terminal latex
set key left Left reverse
set out "../abb/datenmenge_gesamtzeit.tex"
set title "Gesamtlaufzeit"
set xlabel "Byte Protokolldaten"
set ylabel "Sekunden"
plot "laufzeitvergleich.txt" u 1:5 t "Datenmenge zur Gesamtzeit" w points
```

Gesamte Anonymisierungszeit

```
set terminal latex
set key left Left reverse
set out "../abb/datenmenge_anon.tex"
set title "Anonymisierung"
set xlabel "Byte Protokolldaten"
set ylabel "Sekunden"
plot "laufzeitvergleich.txt" u 1:3 t "Anonymisierungszeit" w points
```

A.3.2 System

Reduzierer

Das Regelerstellungssystem des Reduzierers – Die Regeln aus der Konfigurationsdatei werden in Perleigene Reguläre Ausdrücke transformiert.

```
open(FILE,$KONFDAT) or die "Konfigurationsdatei nicht gefunden.\n";
while(<FILE>) {
  if (!m/(#)|(^\s*$)/) {
    $a=$_;
    $a=~ s/!(.*)$/\(\?\!$1\)/g;
    $a=~ s/(\+)/#/g;
    $a=~ s/(\*)/\[a-zA-Z0-9_-\]\*/g;
    $a=~ s/(\%)/\\d+/g;
    $a=~ s/(\.)/\\\./g;
    $a=~ s/(\/)/\\\//g;
    $a=~ s/\s*$//g;
    if (!($a=~m/!/)) {
      $a .= "\$";
```

```
    }
    if ($a=~m/#/) {
      $a=~ s/\#//g;
      push @MRULES,"(".$a.")" ;
      $RULENUM++;
    } else {
      push @RULES,"(".$a.")" ;
      $RULENUM++;
    }
  }
}
close(FILE);
$RULES=join('|',@RULES);
$MRULES=join('%',@MRULES);

printf("Reduzierer: %d Regeln gefunden\n",$RULENUM);
```

Anonymisierer

Der Aufbau der Zuordnungstabelle.

```
printf("Anonymisierer: Aufbau der IP Zuordnungstabelle.\n");

foreach $a (@ANOIP) {
  $_=$a;
  if (m/(\d+)\.(\d+)\.(\d+)\.(\d+)\s*-
      \s*(\d+)\.(\d+)\.(\d+)\.(\d+)/) {
    $FROM=$_;
    $FROM=~s/(\d+)\.(\d+)\.(\d+)\.(\d+)\s*-
          \s*(\d+)\.(\d+)\.(\d+)\.(\d+)/$1\.$2\.$3\.$4/;
    $_=$FROM;
    @SRC=split(/\./);
    $_=$a;
    $TO=$_;
    $TO=~s/(\d+)\.(\d+)\.(\d+)\.(\d+)\s*-
        \s*(\d+)\.(\d+)\.(\d+)\.(\d+)/$5\.$6\.$7\.$8/;
    $_=$TO;
    @DEST=split(/\./);
    $i=0; $k=5;
    while ($k==5) {
```

```
    if ($DEST[$i] > $SRC[$i]) {
      $k=0;
    }
    if ($DEST[$i] < $SRC[$i]) {
      @a=@DEST;
      @DEST=@SRC;
      @SRC=@a;
$k=0;
    }
    $i++;
  }
  $i=0; $k=5;
  while ($i<4 && $k==5) {
    if ($SRC[$i] ne $DEST[$i]) {
$k=$i;
    }
    $i++;
  }

  push @IPTAB,join(".",@SRC);
  @l=@SRC;
  while(join(’’,@DEST)!=join(’’,@l)) {
    if ($l[3]<255) {
      $l[3]=$l[3]+1;
    } else {
      if ($k < 3) {
        $l[3]=0;
        if ($l[2]<255) {
          $l[2]=$l[2]+1;
        } else {
          if ($k < 2) {
            $l[2]=0;
            if ($l[1]<255) {
              $l[1]=$l[1]+1;
            } else {
        if ($k == 0) {
                $l[1]=0;
                $l[0]=$l[0]+1;
```

```
              }
            }
          }
        }
      }
    }
    push @IPTAB,join(".",@l);
  }
}
if (m/(\d+)\.(\d+)\.(\d+)\.(\d+)\s*$/) {
  $FROM=$_;
  $FROM=~s/(\d+)\.(\d+)\.(\d+)\.(\d+)
        \s*$/$1\.$2\.$3\.$4/;
  push @IPTAB,join(".",$FROM);
}
if (m/\*/) {
  @SRC=split(/\./);
  $FROM=join('.',@SRC);
  $t=0;$m=0;
  for ($i=0; $i<4; $i++) {
    if ($SRC[$i] eq "*") {
      $t++;
    }
  }
  $tiefe=256**($t);
  for ($i=0; $i<$tiefe; $i++) {
    $OUT="";
    for ($j=0; $j<($t); $j++) {
      $IP[$j]=($i / 256**($t-$j-1)) % (256**($t-$j));
    }
    if ($SRC[0] ne "*") {
      $OUT.=$SRC[0];
    } else {
      $OUT.= shift @IP;
    }
    $OUT.=".";
    if ($SRC[1] ne "*") {
      $OUT.=$SRC[1];
```

```
      } else {
        $OUT.= shift @IP;
      }
      $OUT.=".";
      if ($SRC[2] ne "*") {
        $OUT.=$SRC[2];
      } else {
        $OUT.= shift @IP;
      }
      $OUT.=".";
      if ($SRC[3] ne "*") {
        $OUT.=$SRC[3];
      } else {
        $OUT.= shift @IP;
      }
      push @IPTAB,$OUT;
    }
  }
}

@IPTMP=sort @IPTAB;
$t="";
foreach $a (@IPTMP) {
  if ($a ne $t) {
    $t=$a;
    $_=$a;
    $a=~s/\s*//g;
    push @IPT,$a;
    $n=nslookup(host => $a, type =>"A");
    $_=$n;
    if (m/\S/) {
      @K=split(/\./);
      push @DNS,quotemeta $K[0];
    }  else {
      push @DNS,"K\@\@\@\@\@K";
    }
  }
}
```

```
$_=$IPMOD;
if (m/MIX|(\d+)\.(\d+)\.(\d+)\.(\d+)\s*-
     \s*(\d+)\.(\d+)\.(\d+)\.(\d+)/) {
  $z=int(rand ($#IPT));
  $g=$#IPT-$z;
  for ($a=0; $a<=$#IPT; $a++) {
    if ($a<$z) {
      $NIPT[$a+$g+1]=$IPT[$a];
      $NDNS[$a+$g+1]=$DNS[$a];
    } else {
      $NIPT[$a-$z]=$IPT[$a];
      $NDNS[$a-$z]=$DNS[$a];
    }
  }
}

for ($i=0; $i<=$#NIPT; $i++) {
  $NIPT[$i]=substr($NIPT[$i],0,1)."\@".
            substr($NIPT[$i],1,length($NIPT[$i])-1);
  $NDNS[$i]=substr($NDNS[$i],0,1)."\@".
            substr($NDNS[$i],1,length($NDNS[$i])-1);
}

printf("Anonymisierer: %d IP-Adressen gefunden.\n",$#IPT);
$TIME2=time();
printf("Anonymisierer: Benoetigte Zeit: %d Sekunden.\n",
       $TIME2-$TIME1);
```

Modul	Beschreibung
File-List-0.2.1	Modul zum lesen von Dateisysteminformation (vergleichbar zum Unix–Befehl *find*)
File-Remove0.20	Modul zum entfernen von Dateien aus dem Dateisystem (vergleichbar mit dem Befehl *rm*)
perl-ldap-0.26	Zugriffsmodul für die Kommunikation mit einem LDAP–Server
Net-Nslookup-1.07	Modul zu Auflösung von IP–Adressen in symbolische Namen (DNS) (vergleichbar mit dem Befehl *nslookup*)
Net-DNS-0.24	Wir von *Net-Nslookup-1.07* benötigt und implementiert nahezu alle für das DNS notwendigen Funktionen.
Digest-MD5-2.20	Funktionspaket zur Erstellung von Hashwerten mit dem MD5 Algorithmus
MIME-Base64-2.12	Wird von verschiedenen Paketen benötigt, hauptsächlich zum kodieren von Strings
Digest-HMAC-1.01	Erlaubt das hinzufügen von Privaten Schlüsseln für Hashwerte.
Digest-SHA1-2.01	Implementiert den Secure Hash Algorithmus
Convert-ASN1-0.15	Modul zur Kodierung der LDAP Anfragen, wird vom Modul *perl-ldap-0.26* benötigt.
Crypt-Rijndael-0.05	Implementiert den Advanced Encryption Standard (AES).
MailTools-1.47	Modul zum Verarbeitung von Emails (so zum Beispiel: senden, lesen, zustellen oder kodieren)
Filesys-DiskFree-0.06	Vergleichbar mit dem Befehl *df*, erlaubt es die Auslastung eines Filesystems zu prüfen.
Filesys-DiskSpace-0.05	Vergleichbar mit dem Befehl *du*, erlaubt es den Speicherbedarf einzelner Verzeichnisse und Dateien zu prüfen.
Class-Loader-2.02	Modul zur Unterstützung anderer Pakete, erlaubt das automatisierte Laden von weiteren Modulen.

Tabelle A.1: Aufstellung der für das System installierten Perlmodule und deren Beschreibung

Anhang B

Inhalt der CD

Zur Orientierung auf dem zur Arbeit gehörenden Datenträger gibt dieses Kapitel einen Überblick über die gespeicherten Daten.

`/conf/dienste`
Konfigurationsdateien der auf dem Protokolldatenserver installierten Dienste

`/conf/sys`
Konfigurationsdateien des Prototypen

`/doc`
Die vorliegende Arbeit in verschiedenen Dateiformaten

`/pakete/perl`
Die für das System notwendigen Perlmodule

`/pakete/rpms`
Paket der Dienste die für den Betrieb notwendig sind

`/src/bin`
Der Quelltext des Prototypen

`/src/skripte`
Für die Test erstellte Skripte

`/messwerte`
Durch Test entstandenen Messungen und Ergebnisse

Literaturverzeichnis

[APA02] Apache HTTPD Project - The Apache HTTP Server Project. http://httpd.apache.org, April 2002.

[BDS00] *Bundesdatenschutzgesetz (BDSG)*, pages 171–201. In [com00], 2000.

[BDS01] Bundesdatenschutz Gesetz (BDSG). pages 2954, 2955, 2001.

[BTD99] Bericht der Bundesregierung über die Erfahrungen und Entwicklungen bei den neuen Informations- und Kommunikationsdiensten im Zusammenhang mit der Umsetzung des Informations- und Kommunikationsdienste–Gesetzes (IuKDG). page 14/1191, 1999.

[Cas02] B. Cassel. Basic encoding rules (ber). World Wide Web: http://renoir.vill.edu/~cassel/netbook/ber/node1.html, Juli 2002.

[com00] *Computerrecht e-commerce*. dtv, München, 2000.

[Cri96] M. Crispin. *Internet Message Access Protocol - Version 4rev1*. Network Working Group, Dezember 1996.

[Don02] L. Donnerhacke. Ewiges logfile. World Wide Web: http://www.iks-jena.de/mitarb/lutz/logfile/, Juli 2002.

[Dro97] R. Droms. *Dynamic Host Configuration Protocol*. Network Working Group, März 1997.

[ELV01] F. Eckert, R. Leiteritz und J. Voß. *Telekommunikationsüberwachung – Gegenwertige Grundlagen der Überwachung in der BRD*. TU Berlin, 2001.

[Fri00] J. E. F. Friedl. *Reguläre Ausdrücke*. O´Reilly Verlag, Köln, Januar 2000.

[GG01] Grundgesetz für die Bundesrepublik Deutschland. World Wide Web: http://bundesrecht.juris.de/bundesrecht/gg/, 2001.

[GM00] P. Gola und T. Müthlein. *TDG/TDDSG Teledienste/Teledienstedatenschutzgesetz - Kommentierungen für die Praxis*. Datakontext, Frechen, 2000.

[GS97] P. Gola und R. Schomerus. *Bundesdatenschutzgesetz (BDSG)*. Beck, München, 1997.

[Hä01] K. Hänßgen. Vorlesungsmaterial System– und Netzwerk–Managementsysteme, 2001.

[ITG01] IT Grundschutzhandbuch. World Wide Web: http://www.bsi.bund.de/-gshb/, Juli 2001.

[IuK97] Gesetz zur Regelung der Rahmendebingungen für Informations- und Kommunikationsgesetze (Informations- und Kommunikationsdienste–Gesetz — IuKDG). pages 1870–1880, 1997.

[Jae00] S. Jaeger. Datenschutz – Verbotene Protokolle. pages 6–12, Oktober/November 2000.

[KvH98] S. Karsch und J. von Helden. *Grundlagen, Forderungen und Marktübersicht für Intrusion Detection Systeme (IDS) und Intrusion Response Systeme (IRS)*. debis IT Security Services, 1998.

[Ley96] W. Ley. Analyse von Log–Informationen. World Wide Web: http://-www.cert.dfn.de/pre99papers/logdaten, Juni 1996.

[LL99] B. Laurie und P. Laurie. *Apache – Das umfassende Referenzwerk*. O´REILLY, Köln, 1999.

[LR99] L. Lamb und A. Robbins. *Textbearbeitung mit dem vi- Editor.* O´Reilly Verlag, Bonn, Juli 1999.

[Moc87] P. Mockapetris. *Domain Names - Implementation and specification*. Network Working Group, November 1987.

[MR96] J. Myers und M. Rose. *Post Office Protocol - Version 3*. Network Working Group, Mai 1996.

[NCS02] The NCSA HTTPd Home Page. World Wide Web: http://hoohoo.ncsa.uiuc.edu/, Mai 2002.

[Orw01] G. Orwell. *1984*. Ullstein Verlag, München, 2001.

[Pos82] J. B. Postel. *Simple Mail Transfer Protocol*. Information Sciences Institute University of Southern California, August 1982.

[PR85] J. Postel und J. Reynolds. *File Transfer Protocol (FTP)*. Network Working Group, Oktober 1985.

[RPG01] A. Roßnagel, A. Pfitzmann und H. Garstka. Modernisierung des Datenschutzrechts, September 2001.

[Sam02] SAMBA Web Pages. World Wide Web: http://www.samba.org, Juli 2002.

[SB02] T. Steinbach und F. Burkhardt. IPv4 versus IPv6, Januar 2002.

[SBGK94] M. Scheller, K.-P. Boden, A. Geenen und J. Kampermann. *Internet: Werkzeuge und Dienste*. Springer–Verlag, Berlin, 1994.

[Sch96] B. Schneier. *Angewandte Kryptgraphie*. Addison–Wesley, Bonn, 1996.

[Sch98] A. Schreiber. EDV–Paranoia unter Linux. World Wide Web: http://-www.clug.in–chemnitz.de/vortraege/paranoia/, November 1998.

[Sch00a] G. Schaub, editor. *Arbeitsrechts–Handbuch*. Verlag C. H. Beck, München, 2000.

[Sch00b] S. Schissler. Konzeption eines Intrusion–Detection–Systems zur Überwachung von komplexen Serversystemen. Master's thesis, Hochschule für Technik, Wirtschaft und Kultur Leipzig, Dezember 2000.

[Sea00] S. Shepler, B. Callaghan et. al. *NFS version 4 Protocol*. Network Working Group, Dezember 2000.

[sen02] Sendmail Home Page. World Wide Web: http://www.sendmail.org, Mai 2002.

[SQU02] Squid Web Proxy Cache. World Wide Web: http://www.squid-cache.org, April 2002.

[Tan90] A. S. Tanenbaum. *Computer–Netzwerke*. Wolfram's Fachverlag, 1990.

[vMK00] I. v. Münch und P. Kunig, editors. *Grundgesetz–Kommentar Band 1*. C. H. Beck'she Verlagsbuchhandlung, München, 2000.

[WEB02] Web Server Compare: The definitive guide to HTTP server specs. World Wide Web: http://webcompare.internet.com/, Mai 2002.

[WS02] D. Wood und K.-D. Schumacher. Linux Samba HOWTO. World Wide Web: http://www.tu-harburg.de/~semb2204/dlhp/HOWTO/DE-Samba-HOWTO-1.html, Juli 2002.

[YHK95] W. Yeong, T. Howes und S. Kille. *Lightweight Directory Access Protocol*. Network Working Group, März 1995.

[Ylo95] T. Ylonen. *Internet-Draft, The SSH (Secure Shell) Remote Login Protocol*. Network Working Group, November 1995.

Ehrenwörtliche Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig angefertigt habe und nur die im Literaturverzeichnis aufgeführten Quellen verwendet habe.

Leipzig, am 28. August 2002 . . . . . .