Universität Karlsruhe (TH)
Fakultät für Informatik
Zentrum für Multimedia

LDAP-Server für Single-Sign-On

Studienarbeit
von

Christian Heck

betreut von
Prof. Dr. rer. nat. Peter Deussen
Dipl. phys. Angela Brauch

30. August 2004

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

Netzwerkstrukturen an den einzelnen Instituten der Fakultät sind genauso wie viele Netzwerke in Unternehmen über Jahre hinweg gewachsen. So verwundert es kaum, dass in machen Einrichtungen ganz unterschiedliche Hard- und Softwarelösungen zum Einsatz kommen. Oft wurden die einzelnen Netzwerkkomponenten auch von verschiedenen Administratoren implementiert und nach deren persönlichen Präferenzen konfiguriert.

Diese Situation zeigt sich auch beim Zentrum für Multimedia an der Fakultät für Informatik. Im Bereich der Server ist dies noch recht einfach, da hier konsequent auf Debian[1] Linux gesetzt wird. Im Clientbereich hingegen gibt es eine Vielzahl an lokalen und mobilen Geräten, die den Präferenzen des Benutzers entsprechend mit Linux, Microsoft oder Macintosh betrieben werden.

Windows benutzt sein eigenes Authentifikationssystem ebenso wie Macintosh. Die Linux Systeme sind mittels NIS[2] so verbunden, dass der Benutzer auf allen Servern das gleiche Passwort hat. Trotzdem gibt es für den Mitarbeiter immer noch zwei Passwörter. Jeder Benutzer muß auf den genutzten Systemen eingerichtet und gepflegt werden.

Für jeden neuen Mitarbeiter und Hilfswissenschaftler muß der Administrator auf den entsprechenden Servern diesen neuen Nutzer einrichten und ihm ein Passwort, Benutzername, Email Adresse und natürlich die entsprechenden Rechte geben. Da diese Daten oft auch auf mehreren Rechnern

[1] Das Debian-Projekt ist eine Gemeinschaft von Individuen, die in Gemeinschaftsarbeit ein freies Betriebssystem entwickeln. Die aktuelle Version kann unter http://www.debian.de kostenlos und absolut frei bezogen werden.

[2] Network Information System

benötigt wurden, gibt es so Unstimmigkeiten und Fehler. Insbesondere auf den Pool-Rechnern und den mobilen Computern die von mehreren Mitarbeitern genutzt werden, ergeben sich häufig Konflikte durch veraltete Daten.

Bedingt durch diese Vorgehensweise haben sich auch gravierende Sicherheitssmängel ergeben. Gerade wenn Mitarbeiter und Hilfswissenschaftler das Institut verließen, ihre Daten auf dem einzelnen Rechnern und Server aber eine Zeit lang noch unbemerkt eingetragen blieben, konnte praktisch kaum noch nachvollzogen werden, wer sich wie und wann auf den Systemen eingeloggt hatte.

Aufgrund dieser Umstände ist diese Studienarbeit entstanden. Der Wunsch, alle Nutzer zentral verwalten zu können und dem Administrator eine einfache und zuverlässige Organisation der individuellen Rechte zu ermöglichen, waren die Kernpunkte, die gelöst werden sollten. Dazu sollten auch Geräte und Dienste soweit möglich über diese zentrale Stelle administriert werden können. Da eine ständige Fluktuation der Personen in universitären Einrichtungen, bedingt durch die hohe Anzahl an Hilfswissenschaftlern vorliegt, sollte für den Systemadministrator eine zentrale Personenverwaltung inkl. aller Adressen, Telefonnummer und Email-Adressen geschaffen werden.

1.2 Grundidee

Zur zentralen Verwaltung von Nutzerdaten hat sich in den letzten Jahren der Verzeichnisdienst als sehr nützlich erwiesen. Ein Verzeichnisdienst ist eine spezielle Datenbank, die für ein häufiges „Lesen“ der Daten entwickelt wurde. Die Entwicklung eines Signle-Sign-On[3] Services ermöglicht es so einem Benutzer sich an einem beliebigen Rechner im Netzwerk einzuloggen und seine individuellen Daten zentral abzurufen. Das Abrufen von Daten soll am ZeMM[4] auch dezentral, von außerhalb des Universitätsnetzwerkes und plattformunabhängig ermöglicht werden.

Während es in der Vergangenheit eine Vielzahl von herstellerspezifischen Verzeichnisdiensten gab, setzt sich heute mehr und mehr das standardisierte Lightweight Directory Access Protocol kurz LDAP als Schlüsseltechnologie zur Abfrage firmenweiter Verzeichnisdienste durch.

Der erste Schritt bestand darin, die geeigneten Produkte für Verzeichnisdienste auf dem Markt zu vergleichen. Da die kommerziellen Produkte zur Bewältigung dieser Aufgabe im Bereich von mehreren tausend Euro lagen, war eine Lösung aus dem Open Source Bereich nahe liegend. Mit

[3] einmaliges Einloggen in das Netzwerk für alle Dienste und Daten
[4] Zentrum für Multimedia der Fakultät für Informatik

OpenLDAP[3] und Linux konnte für den Administrator eine Lizenzkostenneutrale Lösung gefunden werden. Sie ist skalierbar und ermöglicht bei Bedarf das Replizieren der Daten auf mehreren Servern zur Gewährung einer ausfallsicheren Datenhaltung.

1.3 Aufbau der Studienarbeit

Ziel dieser Studienarbeit ist es einen Teil der am Institut genutzten Dienste zentral über den neu installierten LDAP Server zu verwalten. Zunächst wird im Kapitel zwei auf die Besonderheiten von OpenLDAP eingegangen und ein kurze Erklärung der einzelnen Versionen abgegeben.

Die Struktur des ZeMM muss auf die Datenstruktur des LDAP Servers abgebildet und die richtige Konfiguration des Servers sichergestellt werden. Kapitel drei schildert die hierzu notwendigen Schritte. Es wird insbesondere auch auf die Auswahl eines geeigneten Editors zur Pflege des Verzeichnisdienstes eingegangen. Außerdem befasst sich Kapitel drei mit den einzelnen Modulen, die für einen reibungslosen Einsatz des OpenLDAP Servers geladen werden müssen und beschreibt die Integration der bestehenden Daten mittels der Software von Padl LLC.[13]

Kapitel vier zeigt in ein paar Beispielen, wie die Dienste für das ZeMM konkret umgesetzt wurden. Ganz besonders wichtig ist hier die Umsetzung von PAM (Plugable Authetication Modul) und NSS (Name Switch Service). Der Zugriff auf diese Daten wurde beim Login per SSH auf drei verschiedenen Servern realisiert, die ihre Daten bei Bedarf vom LDAP-Server beziehen. Ebenso wurde die zentrale Verwaltung sämtlicher Mitarbeiter mit ihren Email-Adressen gelöst. Auf dem institutseigenen Webserver wurde exemplarisch der Zugriff per FTP für eine Benutzergruppe ermöglicht.

In Kapitel fünf werden die großen Vorteile einer solchen Implementierung aufgezeigt. Werden aber auch die Hindernisse und Risiken einer solchen Umsetzung diskutiert werden. Gerade in stark heterogenen Umgebungen, wie sie an diesem Institut vorlagen, ist es nur schwer möglich die gewachsenen Strukturen komplett abzubilden.

Da der Einsatz der Tools von PADL LLC.[5] uns bei der Umsetzung nicht immer ganz einfach erschienen, soll im Anhang C ein wenig Hilfestellung gegeben werden. Diese bezieht sich insbesondere auf die Komandozeilen-Tools und deren Syntax.

[5]Die Seite von PDAL finden Sie unter: http://www.padl.com

Im Anhang D wird nochmal etwas genauer auf den Umgang mit dem JXplorer[17] Editor eingegangen. Da diese Software komplett in Java geschrieben ist, lässt sie sich auf allen gängigen Betriebssystemen nutzen. Es werden aber auch Alternativen genannt, so dass es dem Leser überlassen bleibt sich die passende Software zur Pflege der Daten auszuwählen. Da die Pflege der Daten oft auch dezentral über das Internet erfolgen soll, bieten einige Projekte um eine Browser-basierte Lösung an. Dazu werden im Anhang auch Referenzen zu LDAP in Verbindung mit PHP genannt.

Kapitel 2

Eine Einführung in LDAP

Ein Verzeichnisdienst stellt eine zentralisierte Datenbank für Informationen über beispielsweise Benutzer, Drucker oder Dienste dar. Hierbei sollen unter anderem Namen, Telefonnummern, Email, aber auch Fotos, Public Keys und Passwörter gespeichert werden, um im gesamten Intranet den Anwendungen und Mitarbeitern zur Verfügung zu stellen. Ein Verzeichnis beinhaltet oft beschreibende, attribut-basierte Informationen und bietet umfangreiche Möglichkeiten diese Informationen zu filtern.

Verzeichnisdienste sind im Vergleich zu den klassischen Datenbanken nicht darauf ausgelegt komplizierte Transaktionen durchzuführen und auch nicht inkonsistente Zustände rückgängig zu machen. Vielmehr wurden sie entwickelt, um häufige Anfragen und Suchen schnell beantworten zu können. Dabei bieten sie die Möglichkeit Informationen in einer baumartigen Struktur zu speichern, die streng hierarchisch organisiert ist. Einzelne Teilbäume können einfach repliziert werden und auf verteilten Servern ihre Dienste anbieten. Dies steigert nicht nur die Verfügbarkeit der Daten, sondern bietet auch eine einfache und effiziente Möglichkeit die Informationen für bestimmte Zielgruppen aufzuteilen.

Verzeichnisdienste finden schon seit langer Zeit Verwendung, sind aber meist produktbezogen wie z.B. bei Lotus Notes[4], Novell eDirectory[5] oder dem Active Directory im Microsoft Server System[6]. Hierdurch wurde die Verwendung des Verzeichnisses auf eine geschlossene Gruppe eingeengt. Der erste Standard für Verzeichnisse war der OSI X.500 Standard. Dieser baute aber auf dem kompletten OSI Stack auf, was ihn sehr umfangreich und schwer handhabbar machte.

X.500 definiert zwei Subprotokolle namens DSP und DAP. Ersteres steht für *Directory System Protocol*, letzteres bedeutet *Directory Access Protocol*. Während das DSP zur Kommunikation der Server untereinander gedacht

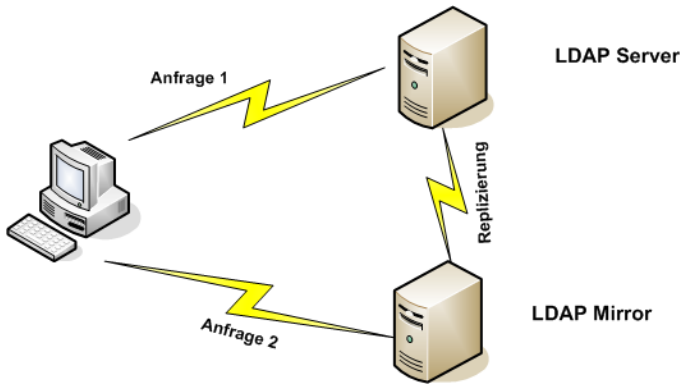
ist, soll das DAP zur Interaktion zwischen Benutzerprozess und Serverprozess dienen. Mit der zunehmenden Verbreitung des Internets erlangte das TCP/IP Protokoll immer mehr an Bedeutung. Selbst in den unternehmensweiten Netzwerken wurde das damals eher selten verwendete TCP/IP immer häufiger eingesetzt.

Die Entwicklung von entsprechenden Clients zur Verwaltung der Daten wurde aber durch die Komplexität des DAP-Protokolls erschwert: DAP setzt auf der Transportschicht des ISO Protokolls auf und nicht auf dem verbreiteten TCP. Als Ausweg aus dem DAP-Dilemma wurde im Juli 1993 im RFC1487[7] das Lightweight Directory Access Protocol spezifiziert.

2.1 Was ist LDAP ?

Das *Lightweight Directory Access Protocol* kurz LDAP ist, wie der Name bereits verrät ein „leichtes“ Kommunikationsprotokoll, das im Gegensatz zu X.500 aus der OSI-Welt auf dem einfacheren TCP/IP Stack aufbaut und den Zugriff auf den Verzeichnisdienst regelt, ohne das Netzwerk übermäßig zu belasten. Die Kommunikation erfolgt standardmäßig auf Port 389 und wird bereits von vielen Browser und Mailclients wie Netscape[1], Outlook[2] oder Mozilla unterstützt.

Der LDAP-Verzeichnisservice basiert auf einem Client-Server-Modell. Ein LDAP-Server enthält die Daten, die der Verzeichnis-Informationsbaum (DIT) bereit stellt. Der Client stellt seine Anfrage an den Server, woraufhin dieser mit einer Antwort reagiert. Da ein solcher DIT auch repliziert und aufgeteilt werden kann, ist es gleichgültig, an welchen LDAP-Server der Client seine Anfrage stellt. Er bekommt immer die gleiche Sicht auf das Verzeichnis. Dieses ist eine wichtige Eigenschaft eines globalen Verzeichnisservices wie LDAP.



[1] ab Version 4.0
[2] ab Version 98

2.1.1 Verschiedene Versionen

Die IETF (Internet Engineering Tast Forsk)[15] hat zu LDAP bereits mehrere RFC's verabschiedet. Das Dokument mit der Nummer RFC2251 „The Lightweight Directory Access Protocol (v3)"[22] spezifiziert die Details des Protokolls in der neuesten Version. Die Tatsache das LDAP in den verschiedenen Versionen nicht miteinander kompatibel ist, hat auch die Implementierung am Institut nicht gerade einfach gemacht. Wichtig für ein reibungsloses Funktionieren ist der Einsatz der neuesten Version v3.

Seit Ende 2001 wird OpenLDAP 1.2, das LDAP v2 implementiert, nicht weiter gepflegt[3]. Es befinden sich aber immer noch genügend Dokumente dazu im Internet. Selbst ein paar noch auf dem Markt befindlichen Bücher[11] beschreiben eine Implementierung der alte Version. Hier sollte also ein besonderes Augenmerk auf die Wahl der passenden Literatur gelegt werden.

Das aktuelle OpenLDAP 2.0 unterstützt LDAPv3. Es zeigt sich am deutlichsten bei der Definition der Attribute in den Schemata sowie bei der Definition der ACLs[4]. Sicherlich ein sehr wichtiger Punkt bei LDAPv3 ist die Verwendung von *Object Identifieres* für eigene Attribute und Objekte. Diese speziellen ID's sind lange Zahlenfolgen, die durch eine Punktnotation getrennt sind. Sie sind weltweit einmalig und können ab einer bestimmten Verzweigungstiefe für die Definition eigener Attribute genutzt werden. Ähnlich des Domain Name Service sind sie hierarchisch organisiert und können bei Bedarf von der IANA[8] beantragt werden. LDAPv3 ergänzt die Version 2 in vielerlei Hinsicht. Neuheiten umfassen auch:

- starke Authentifizierung durch SASL[10]
- optionaler Integritätsschutz und Verschlüsselung durch das auf SSL basierende TLS-Protokoll,
- Internationalisierung durch Unicode,
- Verweise auf andere LDAP-Server, so genannte Referrals und Continuations

2.2 LDAP Modelle

Obwohl man mit Verzeichnisdiensten vom Kaufhauskatalog bis zum weltweiten Telefonbuch allerlei schöne Dinge realisieren kann, ist es der Einsatz von LDAP zur Verwaltung von Mitarbeiterdaten und Netzwerkinformationen, der zur Zeit ein verstärktes Interesse an LDAP auslöst.

[3] Bei Interesse können Sie sich auf den Seiten der University of Michigan[9] die alten Dokumente noch anschauen.

[4] Access Control Lists

2.2.1 LDAP Informationsmodell

Das Informationsmodell dient dazu, die Form und Eigenschaften von Informationen in dem Verzeichnis festzulegen. Es basiert auf Einträgen, wobei ein Eintrag aus einer Ansammlung von Attributen besteht und einen global einzigartigen Namen, den Distinguished Name(DN), besitzt. Ein Attribut wiederum hat einen Typ und ein oder mehrere Werte(Values). Der Typ legt die Art der Information fest, die in den Values abgelegt werden kann. Der DN wird verwendet, um sich eindeutig auf den Eintrag zu beziehen.

Die Typen der Attribute sind gewöhnlich Kürzel, wie „cn“ für Common Name (allgemeiner Name) oder „mail“ für die Email-Adresse. Die Syntax von Werten hängt von der Attributart ab; z.B. könnte ein cn-Attribut den Wert „Felix Mustermann“ enthalten. Ein o-Attribut (Organisation) könnte den Wert „MeineFirma“ enthalten. Ein c-Attribut (Country) würde ein entsprechendes Domainkürzel enthalten. Ein typischer Eintrag in einem LDAP-Baum sieht etwa so aus:

```
cn=Felix Mustermann, ou=Verkauf, o=MeineFirma, c=DE
```

Dies wäre ein so genannter Distinguished Name mit den Attributen cn, ou, o und c. Das gesamte Namensformat ist im RFC2253 „Lightweight Directory Access Protocol (v3): UTF-8 String Representation“ festgelegt.[5]

Außerdem wird über die Syntax festgelegt, wie sich die einzelnen Zeichen der Attribute bei bestimmten Verzeichnis-Operationen, wie etwa dem Suchvorgang verhalten sollen. Bei einer Telefonnummer könnte über die Syntax folgendes festgelegt werden:

- Die Werte dürfen ausschließlich die Ziffern 0-9 umfassen sowie Leerzeichen und Bindestriche.
- Wenn mehrere Werte vorhanden sind, werden die Einträge lexikographisch geordnet.
- Leerzeichen und Bindestriche werden ignoriert, da bei einer Suchanfrage nur die Nummer-Zeichen miteinander verglichen werden. Somit ist es gleichgültig, ob eine Telefonnummer in der Form 9988776, 99 88 776 oder 9988-776 angegeben ist.

2.2.2 LDAP Namensmodell

Das Namensmodell dient in LDAP dazu die Referenzierung und Organisation von Informationen zu ermöglichen. In LDAP wird ein Verzeichnis in

[5] http://www.ietf.org/rfc/rfc2253.txt

Form eines Wurzelbaums verwaltet. Die Wurzel eines jeden Baumes ist hier das *root*-Element. Die Verzeichnisstruktur abstrahiert den Aufbau der von ihr dargestellten Organisation. Traditionsgemäß reflektierte diese Struktur die geographischen und/oder organisatorischen Grenzen. Die Anordnung basiert auf den jeweiligen eindeutigen DN's der Objekte. Der DN setzt sich aus einer Kette von durch Komma getrennten RDN's[6] zusammen, wobei jeder RDN einer Hierarchieebene im DIT entspricht.

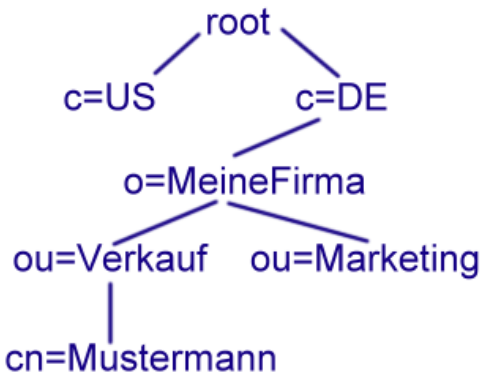
Gleich nach dem root-Element erscheinen als erste Kindobjekte des Baums die Einträge für Länder (z.B. c=DE für Deutschland). Das c steht hier für das Land (Country c). Unter ihnen sind die Einträge für Bundesländer und Organisationen dargestellt. Es hat sich gezeigt, dass die ebenfalls hierarchisch organisierte Domain des Unternehmens in umgekehrter Reihenfolge(z.B. www.zemm.de) zur Festlegung der ersten Einträge ideal genutzt werden kann.

Bei der Implementierung reicht es oft aus nur das „o“ für Organisation zu verwenden. Die Untergliederung mit „st“ für Staat kommt aus dem amerikanischen und wird in Europa selten genutzt. Mit dem Attribut „ou“ wird eine Organizational Unit beschrieben, also eine nicht näher festgelegte Einheit oder Gruppe. LDAP bietet mehrere dieser strukturellen Attribute an, um eine möglichst genaue Abbildung der realen Zusammenhänge zu schaffen.

Alle weiteren Einträge unter der Organisation obliegen den eigenen Entwurfsvorstellungen. Sie können dann für Personen oder z.B. die Verkaufs- und Marketingabteilung genutzt werden. Abbildung 2.1 zeigt einen Verzeichnisbaum, der die traditionelle Namensgebung verwendet. Es gibt mittlerweile internationale standardisierte Daten-Schemata[7], die insbesondere zur Abbildung von Personen, also z.B. Kontaktdaten wie Email-Adresse und Telefonnummern, aber auch Benutzer-Account-Daten wie Login-ID und Passwort, sowie Zertifikaten, die im Rahmen von PKIs (Public Key Infrastructure) Verwendung finden, genutzt werden.

[6] relative Distinguished Name

[7] vgl. [RFC 2256], sowie die in [RFC 2798] spezifizierte Objektklasse inetOrgPerson



Die Tiefe des DIT ist nicht eingeschränkt. Es können also theoretisch so viele Hierarchieebenen gebildet werden, wie für die Abbildung der Struktur notwendig sind. Nachfolgend die wichtigsten Attributtypen:

- dc = Domain Component: Name der Domänenbestandteile
- dn = Common Name: Objektname
- d = Country: Ländername
- st = StateOrProvinceName: Name eines Bundeslandes
- l = LocalityName: Name einer Stadt
- o = OrganizationName: Firmenname
- ou = OrganizationalUnit: Name der Organisationseinheit
- uid = UserID
- gid = GroupID

Ein LDAP-Verzeichnis dient natürlich nicht nur zur Verwaltung von Personendaten, sondern auch maßgeblich zur Abbildung von verfügbaren Ressourcen wie Rechenleistung, Speicher und Drucker. Dazu gibt es ebenfalls einheitliche Schemata[8], die zur Verwaltung der Daten viele Attribute definieren.

Zusätzlich bietet LDAP auch die Möglichkeit über das spezielle Attribut „objectClass“ die verwendeten Attribute als notwendig oder optional zu deklarieren. Selbst die Objektklassen selbst stehen in einer Vererbungshierarchie, so dass z.B. die Objektklasse organizationalPerson alle Eigenschaften der Objektklasse Person erbt.
RFC 2307 beschäftigt sich mit der zentralen Verwaltung weiterer Netzwerkinformation mittels LDAP:

[8]Einen Überblick über standardisiertes LDAP-Schema bietet die Directory Schema Registry unter http://www.schemareg.org

- Benutzer und Gruppen (/etc/passwd, etc/groups)
- IP-Dienste (Zuordnung zwischen Diensten, Portnummern)
- IP-Protokolle (/etc/protocols)
- RPCs (Zuordnung von Remote-Procedure-Call-Nummern zu RPC-Diensten wie in /etc/rpc)
- NIS-Informationen
- Boot-Informationen (MAC-Adressen und Boot-Parameter)
- Mountpoints für Dateisysteme (/etc/fstab)
- Mail-Aliase.

2.2.3 LDAP Funktionsmodell

LDAP gibt die Operationen vor, die an den im Verzeichnis gespeicherten Objekten vorgenommen werden können. Dazu zählen folgende Vorgänge:

- Abfrage: Umfaßt die Such- und Vergleichfunktionen, um Informationen aus dem Verzeichnis zu beziehen.
- Update: Unter dieser Operation werden alle Vorgänge zusammengefaßt, die ein Objekt aktualisieren. Dazu zählen das Hinzufügen, Ändern, Löschen und Verschieben eines Objektes.
- Authentifizierung: Umfaßt alle Operationen zur Verbindungsherstellung und Trennung zu und von einem LDAP-Server sowie das Gewähren der Zugriffsrechte und der Schutz von Informationen.

Die häufigste Operation ist die Suche von Informationen im Verzeichnis. Aufgrund der großen Flexibilität in der Suche sind auch die Optionen sehr komplex. Sie bietet neben einem beliebig komplexen Suchfilter u.a. auch die zurückzugebenden Attribute und den Ort im Datenbaum an, ab dem die Suche anfangen soll (der sog. BaseDN). Zudem kann die Anzahl der Hierarchieebenen, die durchsucht werden sollen, bei jeder Suche festgelegt werden:

- Bei *base* wird nur das Basisobjekt, also der Startpunkt durchsucht
- *onelevel* durchsucht nur die direkt vom Basisobjekt abstammenden Einträge ohne das Basisobjekt selbst
- *subtree* durchsucht alle vom Basisobjekt abstammenden Einträge einschließlich des Basisobjekts selbst.

Eine komplexe Suche könnte also wie folgt lauten:

```
Finde die Emailadresse und die Handynummer von allen Personen
in der Organisationseinheit "Marketing", deren Nachname mit
"N" beginnt.

$ ldapsearch -h ldap.zemm.de. -b 'o=zemm' \
'(&(sn~=N)(department=Marketing))' mail mobilephone
```

Um die gefundenen Einträge über die *compare*-Funktion mit den eingegebenen Daten zu vergleichen, werden wie oben gezeigt reguläre Ausdrücke verwendet. Eine gute Referenz dazu bietet[12].

Bei Update-Operationen werden die Verzeichnisinhalte geändert. Die folgend definierten Operationen sind möglich:

- Add: Es werden neue Einträge im Verzeichnis erstellt.
- Delete: Vorhandene Einträge werden aus dem Verzeichnis entfernt.
- Modify: Es werden die Attribute und Werte eines Eintrags geändert. Dazu zählen das Hinzufügen, Löschen sowie Modifizieren von Attributen.
- ModifyDN: Hierbei wird ein kompletter DN-Eintrag an eine andere Stelle im DIT verschoben.

Die Authentifizierungs-Operationen werden beim Verbinden zum LDAP-Server durchgeführt. Die Sitzung selbst kann auf verschiedenen Sicherheitsleveln basieren. Es gibt eine unsichere anonyme Sitzung, eine durch den Client mit seinem Passwort authentifizierte Sitzung oder auch eine verschlüsselte Sitzung. Die folgenden Authentifizierungs-Operationen sind möglich:

- Bind: Hierdurch wird die Sitzung initiiert. Der Client lässt sich dabei vom Server authentifizieren.
- Unbind: Die Sitzung zwischen Client und Server wird beendet.
- Abandon: Der Client kann den Server auffordern, eine noch ausstehende Operation aufzugeben.

2.2.4 LDAP Sicherheitsmodell

LDAP ermöglicht die Integration von Sicherheitskonzepten. Hier bringt insbesondere die LDAP-Version 3 signifikante Verbesserungen. Für die Authentifizierung gibt es die oben aufgeführten Authentifizierungsverfahren am Server. Wenn der Client keinen DN und kein Passwort zur Authentifizierung angibt, eröffnet der LDAP-Server eine anonyme Sitzung. Für einen sicheren

Zugang (Access) unterstützt LDAP Transport Layer Security (TLS), womit die gesamte Kommunikation zwischen Client und Server verschlüsselt ablaufen kann. Für eine sichere Authentifizierung kann unter LDAP auf den Simple Authentication and Security Layer (SASL) aufgesetzt werden.

Mit verschiedenen Benutzerleveln sind üblicherweise verschiedene Zugriffsrechte verknüpft. LDAP bietet dazu ein entsprechendes Authorisierungsverfahren an. Der Datenbank-Administrator, der mit *rootDN* in der Konfigurationsdatei angelegt wird, hat immer alle Rechte. Für alle weiteren Nutzer erlaubt LDAP die Vergabe von Zugriffsrechten auf bestimmte Einträge und Attribute. Es gibt die Einträge „None“, „Search“, „Read“, „Write“, „Delete“ und „Compare“. Lezteres liefert wahr oder falsch für den Passwort-Check zurück, wodurch das Passwort zum Vergleich die Datenbank nie verlassen muss. Über Access Control Lists in der Konfigurationsdatei lassen sich so für jeden Eintrag die entsprechenden Rechte vergeben.

Single Sign-On (SSO) bedeutet somit, dass nach einmaliger Authentifizierung die Identität aller Benutzer allen Diensten, bzw. Anwendungen für einen bestimmten Zeitraum auf den authorisierten Bereichen gesichert zur Verfügung steht. Eine sehr gute Beschreibung zur Festlegung von Zugriffsrechten bietet RFC3829[26] und [1].

2.3 Klassen und Objekte

Jeder Eintrag in einem LDAP-Verzeichnis beschreibt ein Objekt, also eine konkrete Instanz einer Objektklasse. Eine Objektklasse ist eine verallgemeinerte Beschreibung für dieses und gleichartige Objekte. Die Objektklasse verfügt über ein Liste der zwingend vorgeschriebenen (mandatory) und der nicht verbindlich vorgeschriebenen (optionalen) Attribute. LDAP Objekt-Klassen können entweder:

- Abstrakt
- Strukturell oder
- Auxiliär sein können.

Abstrakte Objektklassen werden nur benutzt, um daraus andere Objektklassen abzuleiten. top ist eine solche abstrakte Objektklasse. Es gibt keine Einträge in einer abstrakten Objektklasse. Einträge müssen immer zu einer strukturellen Objektklasse gehören. Die meisten Objektklassen eines Schemas sind struktureller Natur; daher ist es logisch, dass die Default-Einstellung für Objektklassen *structural* ist. Auxiliäre Objektklassen können benutzt werden, um gleiche Attribute an verschiedene Einträge zu binden. Eine solche „Hilfsobjektklasse“ könnte z.B. dazu dienen, Einträge um ganz

spezielle Lieblings-Attribute zu bereichern.

Die Objektklassen eines LDAP-Servers werden in einer zusammenfassenden Beschreibung abgelegt, dem Schema. Ein Schema beschreibt also, welche Objektklassen in einem Verzeichnis erlaubt sind, welche Attribute sie haben müssen, welche sie haben dürfen, und welche Syntax für die Attribute benutzt werden muss. Jeder LDAP-Server hat ein oder mehrere bekannte Standard-Schemas, auf das man immer zurückgreifen kann. Das heißt, dass jeder, der einen LDAP-Client programmieren will, erwarten kann, dass der LDAP-Server bestimmte Standard-Objektklassen und Attribute hat. Die am häufigsten verwendeten Schemata sind:

- core.schema
- cosine.schema
- inetorgperson.schema
- misc.schema
- nis.schema
- openldap.schema
- samba.schema
- sendmail.schema

Beispiel: Ein Schema könnte eine Objektklasse *person* definieren. Das *person*-Schema könnte dann zwingend vorschreiben, dass eine Person ein Attribut für ihren Nachnamen haben muss und dass dieses Attribut ein String ist. Optional darf der Eintrag *person* noch ein Attribut telephoneNumber für eine Telefonnummer haben, die aus Ziffern, Leerzeichen und Bindestrichen bestehen darf.

Damit ist schon eine Menge an Wissen bekannt, z.B. dass Personen-Objekte genau ein zwingendes Attribut für Nachname haben, nach denen gesucht werden könnte. Dabei ist es egal, was für eine Implementierung der LDAP-Server ist, jeder Client darf diese Daten erwarten. Um Beschreibungen für Objektklassen und Attributtypen vom LDAPv3 Format in das slapd.conf zu formatieren, kann man sich unter[14] einen Schema-Converter herunterladen.

2.4 Was ist das LDI-Format ?

Benutzerdaten oder Organisationsdaten lassen sich über Textdateien im LDI-Format in Schemata einpflegen. LDIF (Lightweight Data Interchange

Format) ist im RFC2849[25] definiert und kann für sehr viele unterschiedliche Zwecke verwendet werden.

Eintragungen in den DIT vornehmen ist ein Beispiel, ein anderes der Export eines bestehenden LDAP-Baums in eine LDIF-Datei. Dank des LDIF-Formats ist es leicht, LDAP-Daten zu bearbeiten. Eine nützliche Übersicht über LDIF gibt es bei[27]. Für unseren Mustermann würde eine passende LDIF Datei wie folgt aussehen:

```
dn: cn=Felix Mustermann,mail=felix.mustermann@beispiel.de
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
givenName: Felix
sn: Mustermann
cn: Felix Mustermann
mail: felix.mustermann@beispiel.de
telephoneNumber: 0789 1234 567
postalAddress: Ein Strasse 5
l: Karlsruhe
postalCode: 76131
c: Deutschland
title: Dipl Inf
o: MeineFirma
workurl: http://www.meinefirma.de
roomNumber: 125
userpassword: {CRYPT}saHW9GdxihkGQ
description: Ich bin in der Zeit vom 3.8.2004 bis einschließlich
17.8.2004 im Büro nicht zu erreichen. In dringenden
Fällen wenden Sie sich bitte an Herrn Beispielmann
unter 0789 5678 123.
```

Ein einzelner Eintrag im LDI-Format besteht aus zwei Teilen. Einen DN und einer Liste an Attributen. Der DN muss immer in der ersten Zeile eines LDIF Eintrags stehen und identifiziert diesen eindeutig über den Distinguished Name. Für unsere Datei also eine Kombination des CN (allgemeiner Name) und der Email-Adresse.

Dies ist nur dann nötig, wenn der Common Name(CN) nicht eindeutig ist. Um im ZeMM diesen Umstand zu lösen, hat man sich darauf geeinigt den DN eines Eintrags für Personen immer mit dem aus Unix bekannten Login-Namen zu verwenden. Da dieser zwingend eindeutig ist, kann man sich die umständliche Kombination mit einem weiteren Attribut ersparen.

Der zweite Teil eines LDIF-Eintrags besteht aus einer Liste von weiteren Attributen. Je nachdem welcher Objektklasse der Eintrag angehört, lassen sich so mehrere notwendige und optionale Einträge hinzufügen. Das in diesem Beispiel eingesetzte Passwort `{CRYPT}saHW9GdxihkGQ` wurde mit Hilfe eines Perl-Scripts erzeugt. Dazu muss in der Shell folgender Befehl eingegeben werden.

```
/usr/sbin/slappasswd -h {CRYPT} -s dasGeheimePasswort
```

Das entsprechende slappasswd Perl-Script wird bei der Installation von LDAP bereits mit installiert. Hat man das Passwort erzeugt, kann es danach per „copy and paste“ einfach in die LDIF Datei kopiert werden. Enthält eine LDIF Datei ein Attribut das nicht ASCII kodiert ist, so muss dieser Eintrag zuerst in das so genannte *base64* Format umgewandelt werden. Dies ist dann z.B. notwendig, wenn Bilder zu den Personen mit abgespeichert werden sollen. Die nachfolgende LDIF Datei zeigt einen solchen Eintrag.

```
                .
                .
                .
# Base64 encoded JPEG photo
jpegPhoto:: /9j/4AAQSkZJRgABAAAAAQABAAD/2wBDABALD
A4MChAODQ4SERATGCgaGBYWGDEjJR0oOjM9PDkzODdASFxOQ
ERXRTc4UG1RV19iZ2hnPk1xeXBkeFxlZ2P/2wBDARESEhgVG
```

Kapitel 3

LDAP am ZeMM

Das Zentrum für Multimedia an der Fakultät für Informatik betreibt 10 Server, die als Betriebssystem alle Debian Linux in der Version 2.2 nutzen. Im Clientbereich sind es hauptsächlich Windows 2000 und Windows XP Systeme. Im administrativem Bereich wird ebenfalls Debian Linux eingesetzt. Vereinzelt kommen auch Apple Notebooks zum Einsatz.

Es existieren eine ganze Reihe von verfügbaren Produkten, die hierarchische Verzeichnisdienste nach dem LDAP Standard anbieten. Mit OpenLDAP, dass unter der GPL[1] zur Verfügung steht, existiert eine freie Implementierung dieses offenen Standards. Sie unterstützt die anfangs bereits erwähnte LDAPv3 Implementierung. Im weiteren Verlauf dieser Studienarbeit wird nur Bezug auf Debian als Betriebssystem genommen.

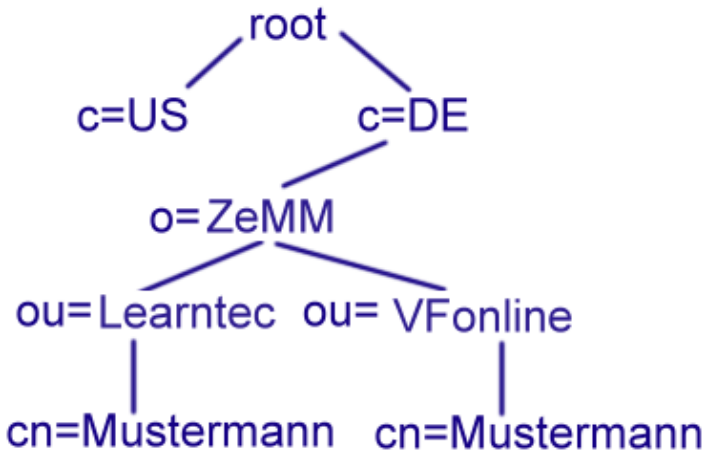
3.1 Modellierung der Strukturen

Nach einer umfangreichen Recherche der vorhandenen Dokumentationen und den bereits beschriebenen Problemen mit den inkompatiblen Versionen galt es zuerst die vorhandenen Strukturen des ZeMM im Directory Information Tree (DIT) abzubilden.

Wichtig für eine korrekte Modellierung der bestehenden Strukturen ist der Vergleich bestehender Daten auf den jeweiligen Servern. Im ZeMM mussten dazu von jedem Server die Benutzerdaten in die jeweilige LDIF-Datei umgewandelt werden, um sie danach mit den Daten der anderen Server zu vergleichen. Lokale User und Gruppen sollten auf dem Server bleiben, wohingegen Benutzer oder Gruppen mit Zugriffsrechten auf mehreren Servern nun zentral vom LDAP-Server verwaltet werden konnten. Dabei mussten insbesondere redundante Daten gelöscht werden.

[1] GNU General Public License, zu finden unter http://www.gnu.de/gpl-ger.html

Es sollten aber auch nicht Daten verloren gehen, wenn beispielsweise zwei Mitarbeiter den gleichen Nachnamen haben. Dazu ist es sinnvoll die gegebenen Abteilungen zuerst im LDAP-Baum abzubilden und die darin arbeitenden Mitarbeiten eine Hierarchieenbene tiefer anzusiedeln. Abbildung 3.1 zeigt einen solchen Baum.



Da wie bereits erwähnt die RD's im LDAP-Baum einzigartig sein müssen führt diese Struktur auch zu keinen Fehlern. Mann erhält die zwei Einträge, die sich in einem Attribut unterscheiden:

```
cn=Mustermann, ou=Lerntec, o=ZeMM, c=DE
cn=Mustermann, ou=VFonline, o=ZeMM, c=DE
```

Ist dies nicht möglich, da die Mitarbeiter sich nicht so einfach einer Abteilung fest zuweisen lassen, gibt es die Variante, alle Mitarbeiter in der Objektklasse Personen zu halten. Ähnlich der Gruppen-Struktur auf einem Unix System, die Mitarbeiter in den entsprechenden Gruppen als Teilnehmer aufnehmen, kann dies auch in einem LDAP-Baum geschehen. Am ZeMM wurde diese Variante verwendet, da es häufig neue Projekte gibt, die von verschiedensten Mitarbeitern und Hilfswissenschaftlern gelöst werden müssen. Mitglieder einer Gruppe sind dann alle Personen deren User-ID als Wert im Attribut von Group-ID aufgeführt sind.

3.2 Aufsetzten des Servers

Als Datenbank Backend benötigt der LDAP-Server *slapd* die Berkley DB[2] ab Version 4. In den meisten Fällen ist diese auf dem Debian System bereits installiert. Falls nicht, sollte dies vor der Installation von *slapd* erfolgen. Oft

[2]Berkley DB wird von Sleepycat Software kostenlos angeboten und kann unter folgender URL bezogen werden: http://www.sleepycat.com

wird aus Sicherheitsgründen auch eine verschlüsselte Verbindung zwischen Server und Client mittels TLS[3] gefordert. Ebenso unterstützt OpenLDAP die Authentifikation per Kerberos[4]. Beide Pakete sollten bei Bedarf ebenfalls vor der eigentlichen Installation von *slapd* installiert werden.

3.2.1 Software installieren

Die Installation des LDAP-Servers *slapd* auf den Debian Systemen ist denkbar einfach. Über das apt Programm[5] lassen sich auf der Console als root mit `apt-cache search ldap` alle Debian Pakete, die es zu LDAP gibt anzeigen. Mit `apt-get install slapd` lässt sich der Server dann installieren. Bei anderen Linux Distributionen und unter Windows kann ein LDAP-Server ähnlich leicht installiert werden. Dazu können die aktuellen Pakete auf den Seiten der OpenLDAP Software[6] herunter geladen werden.

Da es im ZeMM bereits einen zentralen Fileserver gab, der mit Samba und FTP den Zugriff auf die institutsweiten Daten ermöglichte, wurde dieser als LDAP-Server ausgewählt. Für die Umsetzung der FTP-LDAP Schnittstelle war der Server zugleich auch Client.

3.2.2 Module die benötigt werden

Die zentrale Konfigurationsdatei des LDAP-Servers ist die *slapd.conf*. Wie bei Linux Systemen üblich, lässt sich der Server über diese Textdatei entsprechend konfigurieren. Zum besseren Verständnis, wie der LDAP-Server funktioniert, soll nun auf die Konfigurationsdatei etwas genauer eingegangen werden. Nach dem Aufruf der Datei unter */etc/ldap/slapd.conf* könnte dann eine erste und sehr einfach gehaltene Version wie in Abbildung 3.2 aussehen.

---

[3]Transport Layer Security kann über die von OpenSSL angebotenen Libaries gewährleistet werden. Die Software dazu finde Sie unter: http://www.openssl.org

[4]OpenLDAP unterstützt SASL/GSSAPI Authentifikation. Das vom MIT angebotene Kerberos V kann über folgende URL bezogen werden: http://web.mit.edu/kerberos/www/

[5]Informationen zum Gebrauch finden Sie unter: http://channel.debian.de/faq/ch-dpkgundco.html

[6]zu finden unter: http://www.openldap.org/software/download/

```
# This is the main ldapd configuration file. See slapd.conf(5) for more
# info on the configuration options.

# Schema and objectClass definitions
include         /etc/ldap/schema/core.schema
include         /etc/ldap/schema/cosine.schema
include         /etc/ldap/schema/nis.schema
include         /etc/ldap/schema/inetorgperson.schema
include         /etc/ldap/schema/openldap.schema
include         /etc/ldap/schema/kerberosobject.schema

# Schema check allows for forcing entries to
# match schemas for their objectClasses's
schemacheck     on

# Where the pid file is put. The init.d script
# will not stop the server if you change this.
pidfile         /var/run/slapd.pid

# List of arguments that were passed to the server
argsfile        /var/run/slapd.args

# Where to store the replica logs
# replogfile    /var/lib/ldap/replog

# Read slapd.conf(5) for possible values
loglevel    256

# ldbm database definitions

# The backend type, ldbm, is the default standard
database        ldbm

# The base of your directory
suffix          "o=zemm,c=de"

# Define the password used with rootdn. This is the Base64-encoded MD5 hash.
rootpw {CRYPT} ganzGeheim

# Where the database file are physically stored
directory       "/var/lib/ldap"
mode            0600

# Save the time that the entry gets modified
lastmod on

#######################################################
# The userPassword by default can be changed
# by the entry owning it if they are authenticated.
# Others should not be able to see it, except the
# admin entry below
#######################################################

#access to attr=userPassword
        by dn="cn=admin,o=zemm,c=de" write
        by anonymous auth
        by self write
        by * none

# The admin dn has full write access
#access to *
#       by dn="cn=admin,o=zemm,c=de" write
#       by anonymous auth
#       by self read
```

Die von OpenLDAP bei der Installation bereits mitgelieferten Schemata sollten nicht weiter verändert werden. Je nach Bedarf lassen sich wie hier aufgezeigt, weitere Schemata noch hinzufügen. Eines der am häufigsten genutzten Schemata ist das *inetorgperson.schema*. Es bietet zu den gängigen Benutzerdaten eine Fülle an weiteren Attributen wie ISDN Nummer, UserID, GroopID oder auch persönliche Beschreibungen.

Für den LDAP-Server gibt es eine Prozesserkennung (PID), die in der angegebenen Datei protokolliert wird. Neben der PID wird zusätzlich eine Argument-Datei angelegt. In ihr sind Informationen zu den Startparametern des Servers zu finden. Bei der Installation des LDAP-Servers am ZeMM wurde auf eine Replizierung der Daten mit einem zweiten Server in der ersten Implementierungsstufe verzichtet, der Eintrag konnte also getrost auskommentiert werden. Sollte dies nötig werden, kann man unter[1] eine gute Dokumentation dazu finden.

Angaben zum Domänen-Suffix, den der LDAP-Server repräsentieren soll sind besonders wichtig, da jede Anfrage ab dieser Wurzel gestartet wird. Das Passwort für den Administrator wurde hier mit *crypt* verschlüsselt angegeben. Es stehen natürlich auch andere Verfahren wie MD5 oder SSHA zur Verfügung. Die Konfigurationsdatei steht zu Bearbeitungszwecken nur dem Administrator des Servers zur Verfügung und stellt so kein Sicherheitsrisiko dar. Dies wird über den *mode*-Eintrag und der aus Unix bekannten 4 Byte langen Zahlenfolge erreicht.

Über die Access-Control-Lists lassen sich sehr detaillierte Zugriffsrechte vergeben. Als Beispiel soll in der obigen slapd.conf-Datei nur kurz der Zugriff auf die jeweiligen Passwörter gezeigt werden. Der jeweilige Besitzer darf so sein eigenes Passwort nach erfolgreicher Authentifizierung ändern, der Administrator hat Schreibrechte auf alle Daten. Sonstige User haben keine Rechte.

3.2.3 Integration der Daten in den Server

Da auf dem zentralen Server des Instituts die Mitarbeiterdaten bereits im Network Information System (NIS) abgelegt waren und die Authentifizierung über *passwd* und *shadow* realisiert wurde, musste nur noch ein Weg gefunden werden diese Daten in das entsprechende LDI-Format umzuwandeln. Die Syntax einer solchen LDIF-Datei wird in Kapitel 2 dargestellt.

Dazu bietet die Firma Padl LLC.[16] eine Sammlung von Migrationstools an, mit denen sich bestehende Daten auf dem Server bequem über das LDI-Format in die Datenbank importieren lassen. Es handelt sich dabei im wesentlichen um Perl-Scripte zur Migration von *users*, *groups*, *aliases*,

hosts, *netgroups*, *networks*, *protocols* und *services*. Weitere Informationen dazu werden auch im Anhang C beschrieben.

Insbesondere die erzeugten Dateien *useres.ldif* und *groups.ldif* mussten entsprechend unseren Entwurfsentscheidungen angepasst werden. Die bereinigten Daten konnten dann in die Datenbank übertragen werden. Das Hinzufügen über den Befehl *ldapadd* sieht dann wie folgt aus:

```
ldapadd -f /etc/ldap/groups.ldif -h server3.zemm.de (eine Zeile)
 -x -D 'cn=admin,o=zemm,c=DE' -W
```

Eine genauere Beschreibung der Kommandozeilen-Tools, die LDAP mit sich bringt, sowie die Migrations-Tools von PADL werden im Anhang C noch einmal genauer erklärt.

3.3 Auswahl eines geeigneten Editors

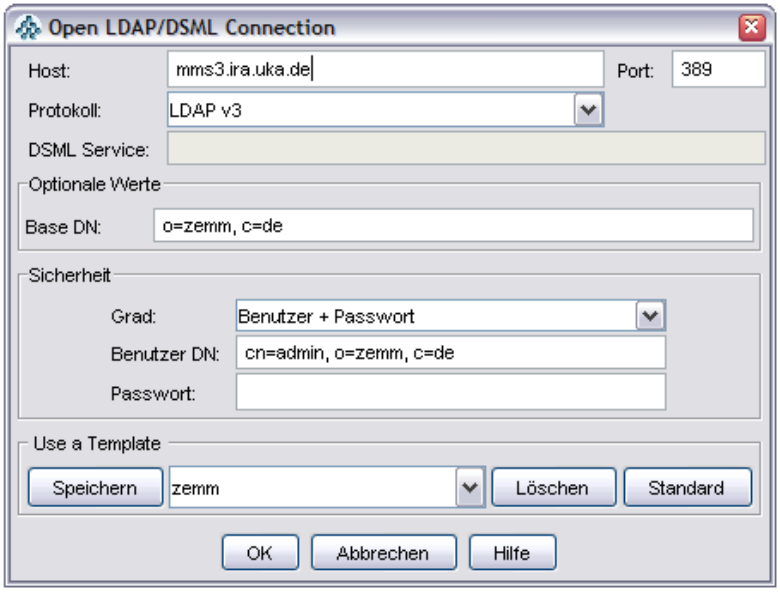
Um die Daten einmalig in den DIT zu laden und einfache Anfragen an diesen zu stellen mag es ausreichen, die Kommandozeilenwerkzeuge, die mit dem LDAP-Server geliefert wurden einzusetzen. Die Modifikation von Einträgen über LDIF wird jedoch schon komplizierter, so dass nur ein komplettes Löschen und neu schreiben des Eintrags die einfachste Lösung ist. Für den täglichen Gebrauch ist die Kommandozeile, wo sie ansonsten bei Linux eher die schnellere Variante ist, nicht geeignet. Auch der Administrator des ZeMM verlangte nach einem einfachen und schnell zu bedienenden Interface.

Der Wunsch nach einer Möglichkeit Daten im DIT einfacher warten zu können, ließ eine Fülle von LDAP-Editoren entstehen, die je nach eingesetzter Umgebung und Aufgabe unterschiedliche Ansätze verfolgten. Im Open-Source Bereich haben sich so eine Reihe mehr oder minder geeigneter grafischer Anwendungen etabliert, die sich im Hinblick auf ihre Leistungsfähigkeit doch sehr unterscheiden.

Für die Realisierung im ZeMM waren vor allem die Plattformunabhängigkeit und eine intuitive Bedienung entscheidend. Editoren wie gp[20], die auf GTK aufsetzen, konnten so wegen ihrer fehlenden Windows-Unterstützung nicht verwendet werden. Das KDE Pendant kldap[19] kam ebenfalls nicht in Frage.

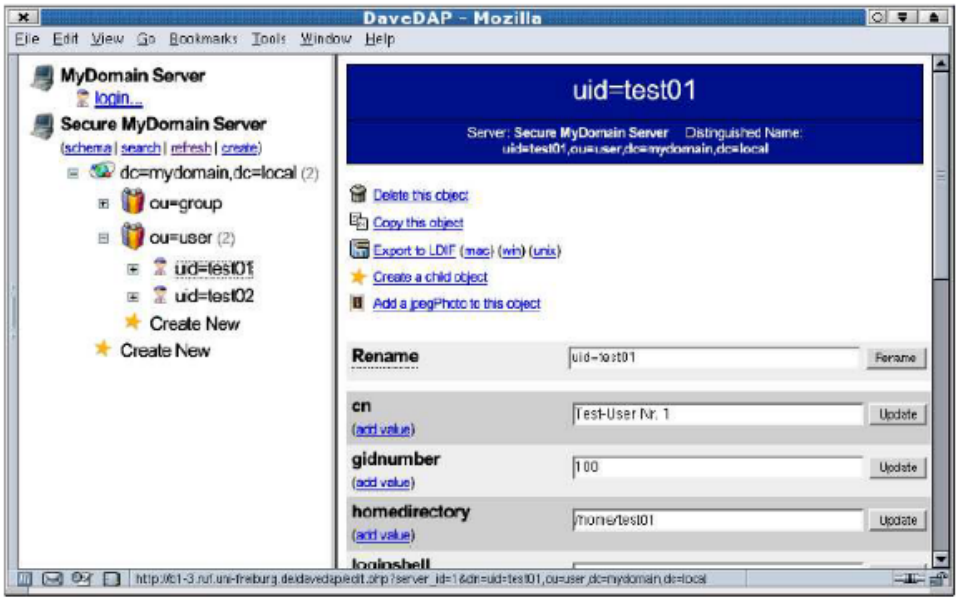
Das Programm JXplorer©von der Firma Pegacat[17] das vollkommen in Java geschrieben ist und sich auf allen gängigen Plattformen einsetzen lässt, überzeugte letztendlich alle Beteiligten am ZeMM. Es bietet eine aufgeräumte grafische Oberfläche, die sich sehr intuitiv bedienen lässt und steht

kostenlos zur Verfügung. Über eine gesonderte Login-Maske lässt sich einfach eine Verbindung zum LDAP-Server aufbauen. Abbildung 3.2 zeigt die für das ZeMM nötigen Einstellungen.



Eine weitere Möglichkeit bietet sich durch die Benutzung geeigneter Webschnittstellen. Da der Webserver des ZeMM allerdings auf einem anderen Server lief, kam eine solche Lösung nicht in Frage. Es soll hier dennoch auf die beiden Projekte auf den Seiten der SourceForge[2] verwiesen werden. Beide Projekte nutzen PHP in Verbindung mit LDAP und zeigen exakt auf, wie gut die Schnittstelle dazu funktioniert.

Das eine Projekt heißt Yala und steht für „Yet Another LDAP Administrator“. Das zweite Projekt heißt DaveDAP und ähnelt, wie sich in Abbildung 3.3 zeigt, der Benutzeroberfläche von JXplorer stark. Eine kurze Einführung im Umgang mit JXplorer findet sich in Anhang B. Weitere Editoren können im Literaturverzeichnis im Abschnitt „Referenzen aus dem Internet“ gefunden werden.



3.4 Installation der Software auf dem Client

In den meisten Mail-Clients und Browsern ist eine entsprechende LDAP-Schnittstelle bereits integriert. Damit der Webserver des ZeMM seine Autorisierung am LDAP-Sever durchführt, musste auf diesem zuerst die clientseitige Software installiert werden. Dazu mussten auf dem Server die Pakete libpam-ldap, libnss-ldap und ldap-utils installiert werden. Die Konfigurationsparameter der ldap.conf Datei unter /etc/ldap/ldap.conf werden in Tabelle 3.1 gezeigt. ACHTUNG: Es handelt sich hier nicht um die slapd.conf Datei zur Konfiguration des Servers!

Parameter	Bedeutung
host	Die IP-Nummer des abzufragenden LDAP-Servers
base	Der Bereich, ab dem gesucht werden soll
rootbinddn	Der Benutzer, mit dem die Abfragen, Änderungen und Eintragungen getätigt werden sollen
nss_base_passwd	Der Bereich, unter dem die Passwörter zu suchen sind
ssl	aktiviert und deaktiviert SSL

Kapitel 4

LDAP zur Verwaltung der Dienste

Ziel der Studienarbeit war nicht nur die entsprechende Modellierung der Daten im LDAP-Server und das Implementieren dieser, sondern vor allem die zentrale Verwaltung von Diensten, die verteilt über das Netzwerk des Instituts genutzt werden. Um dies zu gewährleisten, musste zuerst eine entsprechende Authentifizierung am LDAP-Server sichergestellt werden. LDAP verwendet dazu das PAM und NIS Modul, welche in Verbindung mit dem Namensservice die Authentifizierung realisieren. Danach konnte die Regelung von Zugriffen über SSH und FTP exemplarisch gelöst werden.

4.1 LDAP zur Authentifizierung

Wie bereits erwähnt, ist in dem in RFC2251[22] spezifizierten Netzwerk-Protokoll auch ein Authentifizierungsprozess definiert. Es handelt sich um die so genannte *bind* Operation, mittels derer sich ein LDAP-Client beim LDAP-Server anmeldet. Es gibt momentan zwei Methoden die Identität einer Person am LDAP Server zu überprüfen:

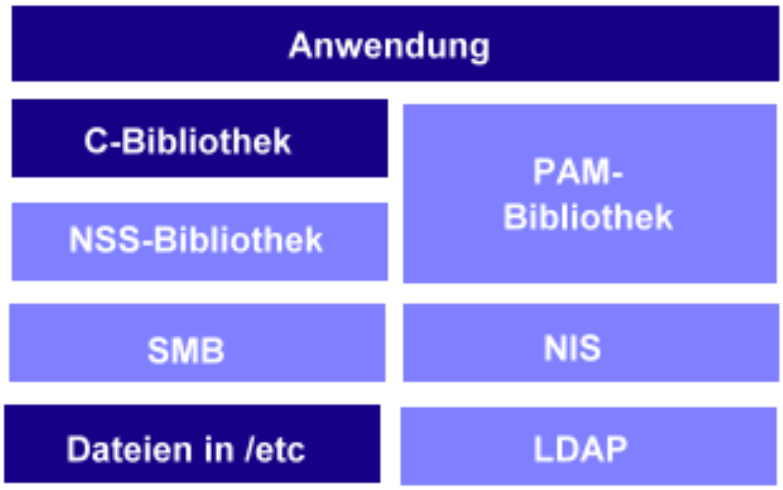
- Simple Bind: hierbei wird zusätzlich zum DN ein Passwort mitgeschickt. Dieses wird vom Server mit dem entsprechenden Passwort im LDAP-Baum verglichen. Da das Passwort ungeschützt über das Netz geschickt wird, also jederzeit abgehört werden kann, wird von der Verwendung von simple bind ohne vorherige Verschlüsselung der Verbindung abgeraten. Eine solche Verschlüsselung ist jedoch mit TLS möglich und ebenfalls im Standard spezifiziert.
- SASL (Simple Authentication and Security Layer) ist ein IETF-Standard und in RFC2222[21] definiert. Es ermöglicht den Authentifizierungs-Vorgang von der eigentlichen Anwendung in eine eigene Schicht zu kapseln, sodass nicht für jedes Anwendungsprotokoll ein eigener Authentifi-

zierungs-Mechanismus definiert werden muss. Auf die genauen SASL-Mechanismen soll hier nicht weiter eingegangen werden. Weitere Informationen finden sich dazu im entsprechenden RFC.

RFC2829[24] spezifiziert welche dieser Authentifikationsmechanismen von einer standardkonformen LDAPv3-Implementierung in jedem Fall unterstützt werden müssen. Diese sind:

- anonyme Authentifizierung (keine Authentifizierung, oder simple bind mit leerem Passwort)
- Passwortauthentifizierung mittels des SASL-Mechanismus DIGEST MD5
- durch TLS geschützte Passwortauthentifizierung mittels simple bind oder TLS geschützte Authentifizierung mittels des SASL-Mechanismus EXTERNAL

Bei Unix-Systemen geschieht die Integration der LDAP-Authentifizierung sehr elegant durch die Verwendung von zwei abgekapselten Bibliotheksschichten NSS und PAM, die in Abbildung 4.1 schematisch dargestellt werden.



Auf die Installation der Module und die Integration in das bestehende System werde ich in den beiden folgenden Abschnitten genauer eingehen, da sie für die erfolgreiche Migration des Systems am ZeMM von großer Bedeutung sind. Damit der Client diese beiden Module bei der Verbindung mit dem Server verwendet, müssen sie auch in die *ldap.conf*-Datei eingetragen sein.

4.1.1 LDAP mit NSS

Der Name Service Switch (NSS) ist eine Schicht innerhalb der C-Libraries des Betriebssystems, die es ermöglicht, Informationsdienste wie z.B. Accountdaten die normalerweise unter */etc/passwd* zu finden sind, nun über unseren Verzeichnisdienst zu versorgen. Dies geschieht vollkommen transparent. Neben dem normalerweise genutzten NIS (Network Information System) und DNS (Domain Name Service) kann anstelle dieser Dateien auch ein LDAP-Verzeichnis verwendet werden, um z.B. Benutzerdaten, Gruppendaten, IP-Dienste, etc. vorzuhalten. Das entsprechende LDAP-Schema ist in RFC2307[23] spezifiziert.

Um das entsprechende Modul auf dem Debian System zu nutzen, muss zuvor der Nameservice Caching Deamon installiert sein. Diesen kann man über *apt-get install nscd* installieren. Dannach können die Pakete libnss-ldap und ldap-utils über *apt-get install libnss-ldap ldap-utils* installiert werden. Das NSS als auch das PAM Modul werden ebenfalls von PADL LLC.[13] angeboten. Die entsprechende Dokumentation zum *nss_ldap* Modul kann unter [29] gefunden werden. Zur Konfiguration des Paketes müssen die Dateien */etc/nsswitch.conf* und */etc/libnss-ldap.conf* angepasst werden.

Änderungen hieran haben sofortige Auswirkung, d.h., es ist Vorsicht geboten, will man sich nicht ungewollt selber aussperren, wenn z.B. die Verbindung zum LDAP-Server noch nicht funktioniert. Dazu ist es ratsam, während der Testphase immer eine root-Shell offen zu haben, um im Notfall fehlerhafte Einstellungen wieder rückgängig machen zu können.

Die Konfiguration des NSS in */etc/nsswitch.conf* besteht aus jeweils einer Zeile pro Name Service, zuerst der Name des Service, dann die Reihenfolge, in der Lookups durchgeführt werden. Für die im ZeMM verwendete Konfiguration hat die nsswitch.conf Datei folgende Einträge:

```
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Service Switch functionality.
# If you have the ‘glibc-doc’ and ‘info’ packages installed, try:
# ‘info libc "Name Service Switch"’ for information about this file.

passwd:         ldap compat
group:          ldap compat
shadow:         ldap compat

hosts:          dns [!UNAVAIL=return] files
networks:       nis [NOTFOUND=return] files
```

```
protocols:      ldap [NOTFOUND=return] db files
services:       ldap [NOTFOUND=return] db files
ethers:         ldap [NOTFOUND=return] db files
rpc:            ldap [NOTFOUND=return] db files

netgroup:       ldap nis
```

Hier wird also zuerst der LDAP-Server befragt und nur im Fall, dass kein Eintrag vorhanden ist, in den lokalen Verzeichnissen des Clients nach einem lokalen Benutzer gesucht. Über den Befehl *getent shadow* lässt sich nun ganz einfach nachvollziehen, ob die Anfrage an den LDAP-Server erfolgreich war. Mann muss dazu nur die lokale shadow Datei mit dem Ergebnis der *getent* Anfrage vergleichen. Das Vorgehen zu NSS und weitere Details dazu können unter [30] gefunden werden.

4.1.2 LDAP mit PAM

Das Pluggable Authentication Module(PAM)[18] ist ein Rahmenwerk für Login-Dienste, welches in vier Module aufgeteilt ist:

1. Authentifizierung,
2. Session-Aufbau und Session-Loggen,
3. Accountverwaltung mit Login-Policies,
4. und Passwortpolicies.

Mit einem zentralen LDAP-Verzeichnis lässt sich über PAM ein zentraler Authentifizierungsdienst aufbauen, wobei ein Benutzer für beliebig viele Rechner ein einziges Passwort benötigt, was nicht nur für den Benutzer von Vorteil ist, sondern auch für den Administrator, der die Benutzer ebenfalls nur an einer zentralen Stelle managen muss. Das häufige Rücksetzen des Passworts, weil der Benutzer es vergessen hat wird so ebenfalls verhindert.

Auch für den Login-Prozess bei Windows-Rechnern kann eine solche LDAP-basierte Benutzerverwaltung verwendet werden. Dies wird durch den Einsatz von Samba[31] ermöglicht. Samba verfügt ebenfalls über eine LDAP-Schnittstelle, sodass wieder die Benutzer-Account-Daten für den Login-Prozess aus der zentralen LDAP-Benutzerverwaltung verwendet werden können. Eine sehr ausführliche Beschreibung zu Samba und LDAP als Primary Domain Controller für Windows Workstations findet sich unter[38]

Die Installation des libpam-ldap Moduls erfolgt ähnlich schnell wie die das libnss-ldap Moduls. Über *apt-get install libpam-ldap* kann das Modul

installiert werden. PAM kann auf verschiedene Arten konfiguriert werden, entweder in einem einzigen File für alle Applikationen /etc/pam.conf oder in einem separaten Verzeichnis /etc/pam.d/ mit je einem File pro Applikation. Bei Debian wird die leztere Variante verwendet. Nach der entsprechenden Konfiguration muss für eine erfolgreiche Passwort-Abfrage lediglich noch die Datei passwd unter */etc/pam.d/passwd* um die folgenden Zeilen erweitert werden.

```
password   required   pam_ldap.so ignore_unknown_user
password   optional   pam_unix.so nullok obscure min=4
                      ↪ max=8 try_first_pass
```

Hier wird folglich zuerst das LDAP Modul genutzt und nur im Fall einer fehlenden Antwort auf die lokale Datei verwiesen werden. Sollte der LDAP-Server einmal nicht antworten oder ausgefallen sein, kann sich der lokale Administrator auf dem Client immer noch einloggen. Eine genauere Beschreibung der Vorgehensweise ist ebenfalls unter[32] zu finden.

4.2 LDAP verwaltet SSH

OpenSSH ist eine freie Version der SSH Protokoll Suite von Netzwerk-Tools, auf die sich eine steigende Anzahl von Leuten im Internet verlassen. Im Gegensatz zu telnet oder ftp ermöglicht ssh eine verschlüsselte Verbindung zwischen einem Client und einem Server.

SSH wird am ZeMM auf allen Servern von den Administratoren und vielen Mitarbeitern zum remote-Login genutzt. Der ssh-Server *sshd* ist auf Debian im Grundpaket bereits enthalten. Weitere Informationen zu ssh und dessen Tools können unter [33] nachgeschlagen werden. Damit SSH in Zukunft den Benutzer und sein Passwort nicht mehr lokal auf dem jeweiligen Server(Client) verifiziert, sondern zentral über den LDAP-Server muss im Verzeichnis */etc/pam.d/* die entsprechende ssh Datei angepasst werden. Ähnlich den weiter oben bereits gezeigten Änderungen für die Passwortabfrage, wird die ssh Datei wie folgt ergänzt:

```
%PAM-1.0
auth       required     pam_env.so
# Woody's SSHD checks for /etc/nologin automatically,
# so there is no need for pam_nologin in /etc/pam.d/ssh
# auth       required     pam_nologin.so
auth       sufficient   pam_ldap.so
auth       required     pam_unix.so

account    sufficient   pam_ldap.so
```

```
account    required     pam_unix.so

session    sufficient   pam_ldap.so
session    required     pam_unix.so
session    optional     pam_lastlog.so # [1]
session    optional     pam_motd.so # [1]
session    optional     pam_mail.so standard noenv # [1]
session    required     pam_limits.so

password   sufficient   pam_ldap.so
password   required     pam_unix.so
```

Will man den Zugriff auf den Server nur einer bestimmten Gruppe an Benutzern erlauben, ist es nun möglich die betroffenen Mitarbeiter zentral auf dem LDAP-Server im passenden Eintrag für ssh aufzulisten. Es gibt folglich unter dem DN *ou=groups, o=zemm, c=de* einen Eintrag für ssh mit den entsprechenden User-ID's der Mitarbeiter, die auf den gewünschten Server Zugriff haben, falls die Access Control List's dies zulassen.

4.3 LDAP verwaltet Apache

Viele Anwendungen besitzen bereits eine integrierte LDAP-Schnittstelle für die Authentifizierung von Benutzern, wodurch eine anwendungsspezifische und proprietäre Benutzerverwaltung durch eine zentrale Benutzerverwaltung ersetzt werden kann. Alle derartigen Anwendungen können also auf dieselben Benutzerdaten zugreifen und Authentifizierung, sowie Autorisierung über diese abwickeln.

Als Beispiel hierfür sei das Modul mod_auth_ldap[28] des Webservers Apache[1] genannt, mittels dessen man sowohl die Authentifizierung der Benutzer, also deren Identitätsfeststellung, als auch die Autorisierung, also die Festlegung, welcher Benutzer auf welche Webseiten zugreifen kann, über LDAP abwickeln kann.

Der Apache-Webserver verfügt noch über ein zweites LDAP-Modul mod_ldap das ebenfalls unter [28] zu finden ist. Es bietet einen Cache für die LDAP-Suchen, sowie ein Management der bestehenden LDAP Verbindungen, durch welches weniger Bind-Vorgänge notwendig sind und für jede Suche LDAP Verbindungen bereitgehalten werden. Beide Features können die Performance der LDAP-Authentifizierung erheblich steigern.

[1] Das Modul mod_auth_ldap gehört ab der Apache-Version 2.0.41 zur Standard-Distribution.

4.4 LDAP verwaltet FTP

Am ZeMM wurde als FTP-Server der ProFTPD[35] eingesetzt. Dieser war bereits auf den entsprechenden Debian Systemen installiert und sollte an die Authentifizierung mit LDAP angepasst werden. Es stellte sich heraus, dass es dazu eine spezielle Version für Debian gibt, die bereits für diesen Anwendungsfall konzipiert ist.

Somit musste proFTPD zuerst wieder vom System deinstalliert werden, um danach über *apt-get install proftpd-ldap* eine neue Version zu installieren. Auch hier müssen die beiden Dateien */etc/proftpd.conf* und */etc/pam.d/proftpd* angepasst werden. Eine Liste der für ProFTPD verwendeten LDAP Direktive findet sich unter[36]. Für die Konfiguration am ZeMMM wurden unter anderem folgende Einträge in der proftpd.conf-Datei gemacht:

```
# Limit users to their web directory. Use the default search filter.
LDAPDoAuth      on "ou=People,o=zemm,c=de"

# Define the LDAP server to contact.
LDAPServer              mms3.ira.uka.de
LDAPAuthBinds           on
LDAPDoGIDLookups        on "ou=Group,o=zemm,c=de"
LDAPDoUIDLookups        on "ou=People,o=zemm,c=de"
```

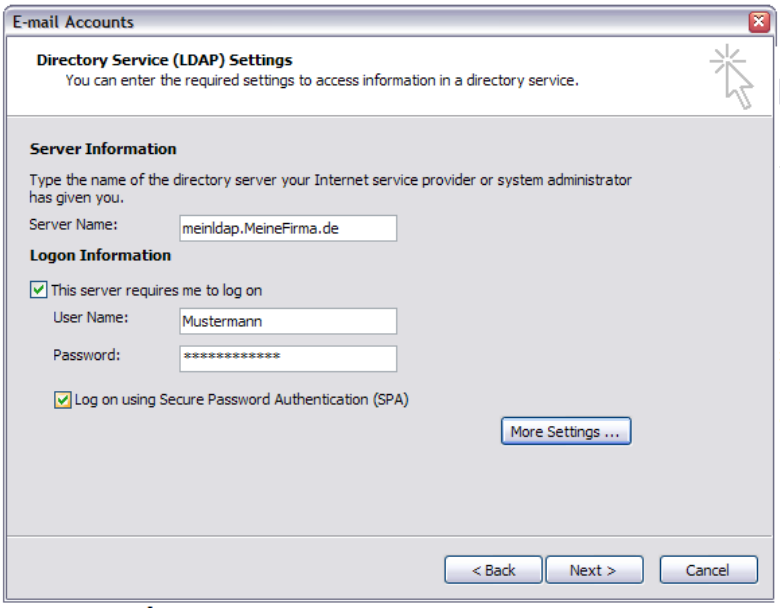
Der Eintrag „LDAPDoGIDLookups“ ermöglicht es nun, bestimmten Gruppen z.B. den Zugriff auf den Webserver zu ermöglichen. Weitere Informationen zu LDAP und ProFTPD finden sich auf den Seiten von[35] und sollen hier nicht weiter vertieft werden.

4.5 LDAP als zentrale Adressbuch

Eine sehr nützliche Eigenschaft eines zentralen LDAP-Servers im internen Netz eines Instituts oder einer Organisation ist die Speicherung von Kontaktdaten an genau einer Stelle. Hierdurch werden gleich zwei Probleme gelöst. Zum einen müssen die Kontakte nicht mehr auf jeden Client redundant gehalten werden und zum anderen sind die Daten aus dem LDAP-Server zu jeder Zeit aktuell.

Alle gängigen Email-Programme können den LDAP-Verzeichnisdienst verwenden. Dabei ist es möglich im jeweiligen Adressbuch private als auch allgemeine Kontakte zu verwalten. Die Implementierung am ZeMM bietet somit jedem die Möglichkeit seine privaten Kontakt lokal auf dem Rechner zu belassen und die allgemeinen Kontakt über den LDAP-Server zu beziehen.

Bei der Eingabe der Kontaktdaten in das Empfänger-Feld des Email-Programms wird automatisch mit dem LDAP-Server eine Verbindung aufgebaut und die cn-, sn-, givenname- und post-Felder durchsucht. Für den Benutzer läuft dies absolut transparent ab. Wird ein passender Eintrag im DIT gefunden, ergänzt das Email-Programm den Namen um die dazugehörige Adresse. Abbildung 4.2 zeigt die Konfigurationsmaske von Outlook[2] zum Einrichten der LDAP-Verbindung. Diese Maske erreichen Sie über Extras → Konten verwalten → Hinzufügen → Verzeichnisdienst



[2]Das hier verwendetet Outlook ist die Version 2003

Kapitel 5

Zusammenfassung und Ausblick

Immer mehr Großunternehmen und Organisationen im öffentlichen Bereich setzen heute angesichts der zahlreichen Fusionierungen und der sich dadurch rasch wandelnden Systemlandschaften auf Unabhängigkeit in der zentralen Benutzerverwaltung. Um in Zukunft den stetig wachsenden administrativen Aufwand zu bewältigen, stellt LDAP eine stabile und flexible Lösung dar. OpenLDAP braucht dabei die kommerzielle Konkurrenz nicht zu scheuen. Es hat sich bei der Implementierung am ZeMM als eine ausgereifte Lösung mit umfangreichen Sicherheitsmechanismen dargestellt. Der Server läuft stabil und authentifiziert bereits die Mitarbeiter beim Zugriff via SSH und FTP auf verschiedenen Servern. Das Personenverzeichnis ist ebenfalls aufgesetzt und kann von jedem Mitarbeiter über ein LDAP-fähiges Mail-Programm abgerufen werden.

Einzig die Abbildung gewachsener Strukturen lässt sich, je nach der verwendeten Software, nicht immer konsequent umsetzen. So konnte die Realisierung einer LDAP-Exim Lösung zum autorisierten Verschicken von Mails bis zum Ende der Studienarbeit nicht zufriedenstellend realisiert werden, da die genutzte Exim[1] Version so stark an die Bedürfnisse des Administrators angepasst waren, dass sie sich nicht mehr zur Kombination mit LDAP eigneten. Die Intergration von Samba ist bereits vorbereitet. Dazu müssen lediglich die passenden Konfigurationsdateien des Samba-Servers angepasst werden.

In machen Bereichen muss somit der Administrator weiterhin die Konfiguration lokal auf dem Client vornehmen. Leider ist das Zusammenspiel zwischen Exim und LDAP auch nicht sehr gut dokumentiert. Hier sollte

[1] Exim ist ein Mail Transfer Agent und wird mit vielen Linuxversionen vertrieben

in Zukunft eher auf *sendmail*[2] umgestiegen werden. Seit der Version 8.11.0 kann Sendmail auf LDAP-Verzeichnisse zugreifen.

Die stetig steigende Zahl an Systemen und Lösungen die den offenen Standard unterstützen, zeigt deutlich wie sehr die Unternehmen eine zentrale Stelle zur Verwaltung von Diensten und Benutzerdaten wünschen. Sicherlich darf der enorme Aufwand zur Integration eines solchen Systems in die laufende Infrastruktur nicht unterschätzt werden, über kurz oder lang wird sich dies aber nicht vermeiden lassen, will das Unternehmen oder Institut die zu verwaltenden Daten vernünftig und zeiteffizient administrieren.

Bedeutend für die weitere Entwicklung ist sicherlich das OpenLDAP Projekt, das es auch kleinen und mittelständigen Unternehmen ermöglicht die Vorteile eines Verzeichnisdienstes zu nutzen. Die anfänglichen Probleme durch fehlende Sicherheitsmechanismen sind seit LDAPv3 aus dem Weg geräumt. Somit können auch kritische Daten und Dienste über das Verzeichnis abgefragt werden. Damit diese sensiblen Daten auch immer verfügbar sind, ist die Replizierung auf mehrere Server ein wichtiger Faktor zur Steigerung der Stabilität und sollte nach einer erfolgreichen Implementierung unbedingt durchgeführt werden.

[2]Eine Dokumentation zu LDAP und Sendmail findet sich unter[1]

Anhang A

Verwendete Literatur

Die aktuellsten, und meiner Meinung nach, besten Bücher

- LDAP system administration / Carter, Gerald , 2003
- LDAP verstehen, OpenLDAP einsetzen / Klünter, Dieter; Laser, Jochen , 2003
- Understanding and deploying LDAP directory services / Howes, Timothy A.; Smith, Mark C.; Good, Gordon S. , 2003
- Online: LDAPv3 HOWTO unter http://www.bayour.com/LDAPv3-HOWTO.html
- Online: Deutsche Seite zu LDAP http://www.verzeichnisdienst.de/

Weitere Literatur

- LDAP für Java-Entwickler / Wegener, Jörg; Zörner, Stefan , 2004
- Samba 3 - Wanderer zwischen den Welten / Kühnel, Jens , 2004
- LDAP directories explained / Arkills, Brian , 2003
- Implementing LDAP von Mark Wilcox
- LDAP: Programming Directory-Enabled Applications with Lightweight Directory Access Protocol von Howes und Smith
- LDAP unter Linux von Banning, Addison-Wesley (baut auf LDAP v2 auf!)
- Understinding LDAP als IBM Redbook (kann online gefunden werden unter: http://www.redbooks.ibm.com/abstracts/sg244986.html)
- LDAP Programming with Java von Rob Weltman und Tony Dahbura

- Deploying OpenLDAP von Tom Jackiewicz, erscheint erst Ende 2004
- Whitepaper: Unter folgender Adresse können weitere Referenzen aus dem Internet bezogen werden: http://www.zdnet.de/suchen/index.htm?collection=whitepapers&query=ldap
- Reguläre Ausdrücke von Jeffrey E. F. Friedl, O'Reilly 2003

Anhang B

Liste der für LDAP wichtigen RFCs

rfc1274.txt COSINE and Internet X.500 Schema (PS)
rfc2079.txt X.500 Attribute Type and an Object Class to Hold URIs (PS)
rfc2247.txt Using Domains in LDAP DNs (PS)
rfc2251.txt LDAPv3 Protocol (PS)
rfc2252.txt LDAPv3 Attribute Types (PS)
rfc2253.txt LDAPv3 Disinguished Name (PS)
rfc2254.txt LDAPv3 Search Filters (PS)
rfc2255.txt LDAPv3 URL Format (PS)
rfc2256.txt X.500(96) Schema for LDAPv3 (PS)
rfc2293.txt Tables and Subtrees in the X.500 Directory (PS)
rfc2294.txt O/R Address hierarchy in the X.500 DIT (PS)
rfc2307.txt LDAP Network Information Services Schema (E)
rfc2377.txt LDAP Naming Plan (I)
rfc2587.txt Internet X.509 PKI LDAPv2 Schema (PS)
rfc2589.txt LDAPv3: Dynamic Directory Services Extensions (PS)
rfc2596.txt Use of Language Codes in LDAP (PS)
rfc2649.txt LDAPv3 Operational Signatures (E)
rfc2696.txt LDAP Simple Paged Result Control (I)
rfc2713.txt LDAP Java schema (I)
rfc2714.txt LDAP CORBA schema (I)
rfc2798.txt LDAP inetOrgPerson schema (I)
rfc2829.txt LDAPv3: Authentication Methods (PS)
rfc2830.txt LDAPv3: Start TLS (PS)
rfc2849.txt LDIFv1 (PS)
rfc2891.txt LDAPv3: Server Side Sorting of Search Results (PS)
rfc3045.txt Storing Vendor Information in the LDAP root DSE (I)
rfc3062.txt LDAP Password Modify Extended Operation (PS)
rfc3088.txt OpenLDAP Root Service (E)

rfc3112.txt LDAP Authentication Password Schema (I)
rfc3296.txt Named Subordinate References in LDAP (PS)
rfc3377.txt LDAP(v3): Technical Specification (PS)
rfc3383.txt IANA Considerations for LDAP (BCP)

Legende: STD Standard
DS Draft Standard
PS Proposed Standard
I Information
E Experimental
FYI For Your Information
BCP Best Common Practice

Anhang C

PADL Tools nutzen

Die LDAP Migration Tools sind eine Sammlung von PERL-Skripten, die bei der Konvertierung von vorhandenen Datenbanken ins LDAP Format (bzw. ins LDI-Format) behilflich sind. So kann man z.B. die /etc/passwd Datei ins LDI-Format überführen. Die LDAP Migration Tools sind erhältlich unter[13]. Die Migration Tools sind vor allem dann hilfreich, wenn man den LDAP Server z.B. zur Benutzer Authentifizierung in Anspruch nehmen will. Soll der LDAP Server nur als Adressbuch verwendet werden, sind die Tools nicht notwendig.

Die LDAP Migration Tools sind nach der Installation über *apt-get install migrationtools* unter *usr/share/migrationtools/* zu finden. Über die entsprechenden Scripte lassen sich so folgende Dateien aus dem Server in das LDIF Format umwandeln:

- passwd, shadow
- netgroup
- group
- hosts
- network
- services
- aliases
- automount
- fstab
- rpc

Wichtig für eine sichere und stabile Implementierung der Daten in den Directory Information Tree, ist die Auswahl der Daten, die beim späteren Einsatz des LDAP-Servers auch verwendet werden. Es sollten nur diejenigen integriert werden, die beim Einsatz des LDAP-Servers auch benötigt werden. Am ZeMM wurde so auf die Integration von automount, aliases, fstab und rpc verzichtet. Exemplarisch lässt sich dann über folgenden Befehl eine entsprechende LDIF-Datei für *netgroup* erzeugen:

```
aus dem Verzeichnis /usr/share/migrationtools

./migrate_netgroup /etc/netgroup /etc/ldap/netgroup.ldif
```

Das Ergebnis lässt sich dank der Formatierung in ASCII, mit jedem beliebigen Editor(z.B. vi) betrachten. Danach müssen die Daten nur noch über das *ldapadd*-Kommando in den DIT übertragen werden. Dies könnte dann für die eben erzeugte netgroup.ldif-Datei wie folgt aussehen:

```
ldapadd -f /etc/ldap/netgroup.ldif -h meinserver.de (eine Zeile)
 -x -D 'cn=admin,o=MeineFirma,c=DE' -W
```

ldapadd fügt das Objekt dem Verzeichnis hinzu. Anwender können die so im LDAP abgelegten Objekte via ldapsearch abfragen. Um vernünftig mit dem Kommando arbeiten zu können, sollte man in /etc/ldap/ldap.conf für den allgemeinen Fall folgende Werte vorgeben:

```
URI ldap://localhost:389/
BASE dc=samba,dc=org
```

Danach liefert ldapsearch -x die eingefügten Objekte aus dem Verzeichnis.

Ganz wichtig ist beim Anlegen der Daten, dass z.B. Dateien wie *passwd*, die auf jedem Server existieren, zuvor von redundanten Einträgen befreit werden. Es sollten am Besten alle *passwd*-Dateien der Server miteinander verglichen werden und zuletzt eine LDIF-Datei mit allen Daten in den DIT übertragen werden.

Neben dem ldapsearch und dem ldapadd gibt es auch noch die Befehle ldapmodify. Er dient z.B. dazu Passwörter im DIT neu zu setzen.

```
ldapmodify -x -D cn=admin,o=zemm,c=de -W -f passwort.ldif
```

Alternativ kann ldapmodify auch direkt die zu ändernden Attribute auf der Kommandozeile angeben. Hierbei wird folgender Befehl aufgerufen und anschließend das Passwort dazu eingegeben.

```
ldapmodify -x -D cn=admin,o=zemm,c=de -W

Enter LDAP password:

    *******
```

Dannach müssen die Änderungen nach der folgenden Syntax in genau dieser Reihenfolge angegeben werden:

```
dn: cn=Mustermann,ou=People,o=zemm,c=de
changetype: modify
replace: telephoneNumber
telephoneNumber: +49 1234 567-10
```

Abschließend sollen noch die wichtigsten Optionen für die Kommandozeilen-Tools aufgezeigt werden:

Optionen	Beschreibung
-n	Zeige nur die mögliche Veränderung an, führe aber nichts aus.
-h *host*	Verbindung zum LDAP-Server der auf *host* läuft
-p *port*	Nutze den Port *port* anstelle von 389
-Z	Benutze SSL beim Verbindungsaufbau
-x	Aktiviert und deaktiviert SSL
-w *passwd*	dient zur Übergabe des Passworts in der Komandozeile
-W	dient zur Übergabe das Passworts mit Abfrage
-D *binddn*	Verbindung zum LDAP-Server über den *binddn*
-f *file*	Lese das LDIF vom *file* anstelle der Standardeingabe

Anhang D

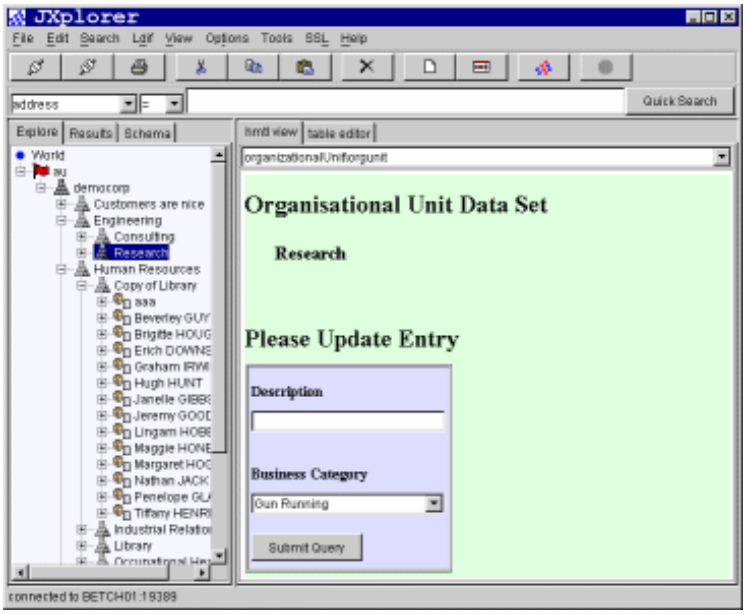
JXplorer verwenden.

JXplorer[17] ist eine Software mit erweiterter Sicherheitsintegration und Unterstützung für die schwierigeren und unverständlicheren Teile des LDAP-Protokolls. Die Lauffähigkeit wurde auf Windows, Solaris, Linux und OS390 geprüft und sollte auf jedem Betriebssystem das Java unterstützt laufen. Die zentralen Eigenschaften umfassen:

- Standard LDAP-Abfragen: add/delete/copy/modify
- komplexe Abfragen: Baumkopie und Baumlöschung
- wahlweise GUI-bassierte Suchfilter erzeugen
- SSL- und SASL-Authentisierung
- steckbares editors/viewers
- steckbare Sicherheitsversorger
- HTML-Templates für die Datenanzeige
- volle i18n-Unterstützung
- Formatunterstützung für LDIF Dateien
- in weiten Teilen an die Bedürfnisse des Benutzers anpassbar.
- Drag-and-Drop Unterstützung für Einträge

Die Oberfläche erscheint sehr aufgeräumt im Vergleich zu anderen Editoren. Über den „connect“-Button auf der linken Seite des Fensters lässt sich wie in Kapitel 3.3 dargestellt eine Verbindung einrichten. JXplorer ist bereits so vorkonfiguriert, das keine weiteren Änderungen für den ersten Einsatz nötig sind. Seit der Veröffentlichung von Version 3.0 auch eine teilweise deutsche Version der Software verfügbar.

Die Suchmaske im oberen Teil des Fensters ermöglicht es, direkte Suchanfragen an die ausgewählten Attribute zu stellen. Über den Reiter „Schema“ lassen sich sogar die im LDAP-Server verwendeten Schemata anzeigen. Änderungen können im Hauptfenster direkt vorgenommen werden. So ist das Editieren und Anlegen von Daten für den Administrator mit wenigen „klicks“ möglich. Weiter Informationen zu dieser Software finden Sie auf[2].



D.1 Weitere LDAP Editoren und Projekte

LDAP Browser/Editor ist ebenfalls in Java geschrieben. Wurde aber seit 2001 nicht weiter entwickelt:
http : *//www.iit.edu/ gawojar/ldap/*

ActiveX LDAP Client: Gibt es bereits in der Version 3.11 und wird stetig weiterentwickelt, ist allerdings ein kommerzielles Produkt und nicht gerade preiswert:
http : *//www.ldapservices.com/*

slix's eduserver : Ist ein Netzwerk-Server für Schulen mit heterogenem Netzwerk, zum Arbeiten mit Windows und Linux Workstations. Als Standardlösung und Musterserver für den Schulalltag vorkonfiguriert.
http : *//www.slixs.at/content.php?review*.22

frood : ist ein weiterer Open Source Client der von Source Forge[2] gehostet wird:

http : *//frood.sourceforge.net/*

web2ldap : ist eine deutsche Entwicklung und bietet einen webbasierten LDAP-Client:
http : *//www.web2ldap.de/*

softera : ist eine Firma die eine ganze Reihe an kommerziellen Produkten zu LDAP anbietet:
http : *//www.softerra.com/products/products.php*

phpLDAPadmin : ist nochmal ein Open Source Projekt das ein webbasiertes Frontend auf PHP Basis zur Verwaltung einer LDAP Verzeichnisses bietet:
http : *//sourceforge.net/projects/phpldapadmin/*

Novell bietet zu seinem eDirectory Produkt eine Testseite an, auf der die LDAP Funktionalität getestet werden kann:
http : *//www.nldap.com/NLDAP/*

LDAP Browser : Ist ein Browser Plugin für den Internet Explorer um LDAP Verzeichnisse zu bearbeiten:
http : *//www.directory − applications.com/browser.html*

Literaturverzeichnis

[1] LDAP system administration / Carter, Gerald , 2003

[2] *http : //www.sourceforge.net/*

[3] *http : //www.openldap.org/*

[4] *http : //www.lotus.com/world/germany.nsf*

[5] *http : //www.novell.com/de − de/products/edirectory/*

[6] *http : //www.microsoft.com/germany/ms/windowsserver*2003/

[7] *http : //ietf.org/rfc/rfc*1478.*txt*

[8] *http : //www.iana.org/*

[9] *http : //www.umich.edu/ dirsvcs/ldap/doc/guides/slapd/toc.html*

[10] das SASL-Modul wird von Cyrus angeboten. Alle weiteren Informationen dazu finden Sie unter: http://www.cyrus.org/sasl

[11] LDAP unter Linux von Banning, Addison-Wesley

[12] Reguläre Ausdrücke von Jeffrey E. F. Friedl, O'Reilly 2003

[13] *http : //www.padl.com/*

[14] *http : //www.arturschneider.de/ldap.de.html*

[15] *http : //www.ietf.org/*

[16] *http : //www.padl.com/OSS/MigrationTools.html*

[17] *http : //www.pegacat.com/jxplorer/*

[18] *http : //www.padl.com/*

[19] *http : //www.mountpoint.ch/oliver/klap*

[20] *http : //biot.com/gq/*

[21] $http://ietf.org/rfc/rfc2222.txt$

[22] $http://ietf.org/rfc/rfc2251.txt$

[23] $http://ietf.org/rfc/rfc2307.txt$

[24] $http://ietf.org/rfc/rfc2829.txt$

[25] $http://ietf.org/rfc/rfc2849.txt$

[26] $http://ietf.org/rfc/rfc3829.txt$

[27] LDAP verstehen, OpenLDAP einsetzen, dpunkt2003

[28] $http://httpd.apache.org/docs-2.0/de/mod/$

[29] $http://www.padl.com/OSS/nss_ldap.html$

[30] $http://wiki.debian.net/index.cgi?NSS-LDAPSetup$

[31] $http://www.samba.org/$

[32] $http://wiki.debian.net/index.cgi?PAM-LDAPSetup$

[33] $http://www.openssh.com/de/index.html$

[34] $http://homex.subnet.at/\ max/ldap/$

[35] $http://www.proftpd.de/$

[36] $http://www.proftpd.de/Direktiven_Liste.28.0.html$

[37] $http://www.ldapguru.com/$

[38] $http://samba.idealx.org/index.en.html$