Qbe SAS SystemDocumentation

Christian Hofstädtler

Version 1.0.0-0194
14. Juli 2004

Über den Autor

Als Einzelprojektant war es nicht immer leicht dieses Projekt komplett durchzuziehen. Möglich schon gar nicht, Hilfe habe ich von einigen Seiten erhalten. Trotzdem hat dieses Projekt genug Spuren hinterlassen und Erfahrungen gebracht.

Dank an ...

Dr. Karl Filz
: Ein professioneller Projektleiter und Ratgeber, den man sich gar nicht besser wünschen könnte.

Dr. Gerhard Burian
: Unser Klassenvorstand der doch einige wichtige Ideen beigesteuert hat.

Andreas Stützner
: Ein Projektpartner der für alles zu haben war und ist.

Florian R.
: Für Anregungen in der Dokumentation.

Weiters ...
: P. Mies, E. Schuh, A. Krieger, C. Sudec, J. Hackl, L. Kranawetter

Inhaltsverzeichnis

Kapitel 1

Authentication Server

Der Authentication Server basiert auf der "Qbe Application Server" Software, die eine generische Engine zur Verfügung stellen soll, die folgendes bietet:

- Ausführung einer sogenannten Applikation für HTTP. Engine und Applikation werden in PHP implementiert, Hintergrunddienste vorzugsweise in Perl, oder auch C/C++.
- Einfache Einbindung von anderen Systemen in die Applikation.
- Modularität und klare Trennung von Teilkomponenten
- Transparentes Handling von verschiedenen Grundfunktionen, wie Menüsystem, SSL, einheitliches Seitenlayout ...

In der vorhandenen Implementierung sind Engine und Applikation nicht klar voneinander getrennt, da erst in einer späten Projektphase die große Wiederverwendbarkeit aufgedeckt wurde. Daher kann nur eine echte Anwendung (mit einigen Sub-Anwendungen) pro Installation ausgeführt werden, die Modularität ist ebenfalls teilweise nicht gegeben, wurde aber laufend bis zum Projektende verbessert.

1.1 Software-Teile

1.1.1 Verzeichnisstruktur

Verwendete Verzeichnisse auf dem AuthServer:

/qbe Qbe Application Software

/qbe-local Lokale Einstellungen für Qbe Application Software
Lokal bedeutet, lokal für diesen Server – sind mehrere Server vorhanden (z.B. in einem Cluster) können Dateien unter diesem Verzeichnis unterschiedlich sein.

/var/lib/mysql MySQL Database

/var/lib/ldap OpenLDAP Database (falls vorhanden)

/var/novell Novell eDirectory State (falls vorhanden)

/var/lib/nds Novell eDirectory Database (falls vorhanden)

/import/homes Benutzerverzeichnisse

/import/homes/qbe-systemstate Qbe Application Software Systemstatus

/import/homes/qbe-inetstate Qbe Systemstatus: Modul internet

/import/homes/.status Qbe Application Software Systemstatus (compatibility)

1.1.2 Verzeichnisse unter /qbe

etc Konfigurationsdateien und Vorlagen für automatisch erstellte Systemkonfigurationen.

data Enthält temporäre Daten.

sbin Programme, die die Hintergrunddienste der Applikation implementieren. Perl- und Shell-Skripte, C-Applikationen.
Achtung: keine Ordner - Modulspezifische Programme sollten `qbe-modulname-programmname` benannt werden.

status Enthält temporäre Statusinformationen.

web Dateien der Applikation, die für HTTP benötigt werden.
Ideal: nur Unterordner und defines.*.php.

web/htdocs Dateien die den benutzersichtbaren Teil der Applikation bilden. PHP Skripte, Grafiken, ...

1.1.3 Dienste

Qbe SAS setzt auf bereits lang existierenden, gut implementierten Diensten auf, diese sind:

Dienst	Implementation	Daemon
DNS	ISC BIND 9	named
DHCP	ISC DHCP 3	dhcpd
LDAP	Novell eDirectory 8.7	ndsd
	OpenLDAP 2	slapd
SQL Datenbank	MySQL 4.1	mysqld
Webserver (+SSL)	Apache 1.3	apache
Versionskontrolle	Subversion 1.0	mod_dav_svn im apache2

Notiz: es ist durchaus möglich, Qbe SAS mit dem OpenLDAP Server zu verwenden, jedoch wird in der HTL Novell eDirectory eingesetzt, da bei Tests in früheren Projektstadien es sich abgezeichnet hat, dass der OpenLDAP Daemon die Last von 1500 Benutzern nicht handlen könnte. Da nach der Umstellung auf Novell eDirectory noch Schemaerweiterungen dazugekommen sind, müsste das Schema wieder in eine OpenLDAP kompatible Form gebracht werden.

Die aktuellen Schemaerweiterungen sind im Anhang ersichtlich.

1.2 Konfiguration

Die Qbe SAS Konfiguration besteht aus mehreren einzelnen Konfigurationsdateien. Diese werden weiter unten ausführlich erklärt:

/qbe/web/defines.php Application Server Grundkonfiguration

/qbe/web/defines.app.php Konfiguration der Anwendung (hier: Qbe SAS)

/qbe/web/defines.local.php Serverspezifische Einstellungen für Cluster Installationen

/qbe/web/defines.security.php Sicherheitseinstellungen der Anwendung

/qbe/etc/perl/qbesystemconfig.pm Perl Konfiguration

/qbe/etc/modules/computer/dhcpd.template Vorlage für die DHCP Konfigurationsdatei

Weiters werden einige System-Konfigurationsdateien verwendet:

/etc/apache/httpd.conf Apache Konfiguration

/etc/apache/ssl* SSL Zertifikate für den HTTP Server

/etc/crontab Zeiteinstellungen für crond

/etc/dhcp3/dhcpd.conf Konfiguration des DHCP Servers, wird automatisch neu erstellt

/etc/php4/apache/php.ini PHP Konfiguration für den HTTP Server

/etc/php4/cgi/php.ini PHP Konfiguration für Background Tasks

/etc/samba/smb.conf Samba (CIFS Server) Konfiguration

1.2.1 /qbe/web/defines.php

setlocale(LC_ALL,"de_AT"); Setzt die Sprache für Ausgaben, Zeiformate, usw. in PHP auf de_AT – Deutsch (Österreich)

$sas_ldap_base Root-Name der LDAP Datenbank. z.B: "o=htlwrn,c=at"

$qbe_http_basepath Hauptverzeichniss der Qbe AppServer Dateien. Default: "/qbe/web/htdocs"

$qbe_http_server Voreinstellung des Servernamens.
Default: wird mit `$_SERVER['SERVER_NAME']` automatisch ermittelt

$qbe_ssl SSL serverseitig vorhanden, ja/nein. Default: true

1.2.2 /qbe/web/defines.security.php

Diese Datei enthält normalerweise die benutzten Passwörter (im Klartext) und sollte daher dem Benutzer `qbe`, Gruppe `www-data` gehören. Als Rechte sollte nur Benutzer `rw` und Gruppe `r` gesetzt sein.

$sas_mysql_server Hostname des MySQL Servers

$sas_mysql_database Die Datenbank, die für Qbe SAS verwendet werden soll

$sas_mysql_user Benutzername mit allen Rechten auf die Datenbank

$sas_mysql_password Zugehöriges Passwort

$sas_ldap_server Hostname des LDAP Servers, normalerweise "localhost"

$sas_ldap_adminuser Benutzername eines Users mit allen Rechten

$sas_ldap_adminpass Dazugehöriges Passwort

$sas_ldap_machineuser Benutzername eines Users mit eingeschränkten (nur-Lesen) Rechten

$sas_ldap_machinepass Dazugehöriges Passwort

1.2.3 /qbe/web/defines.local.php

In dieser Datei können alle Werte aus defines.php oder defines.security.php überschrieben werden. Die Datei ist in der Regel ein symbolic link auf `/qbe-local/web/defines.local.php` und enthält keine Einträge. In Cluster-Konfigurationen wird dort typischerweise die Variable `$qbe_http_globalservername` mit dem wirklichen Servernamen überschrieben.

1.2.4 /qbe/web/defines.app.php

Dies ist eine Konfigurationsdatei, spezifisch für die Applikation (hier: Qbe SAS).

$sas_version Qbe SAS Versionsnummer

$sas_codename Qbe SAS Codename der aktuellen Version

$qbe_http_globaldomain DNS-Domain in der die Server eingetragen sind - muss dem PHP Cookie-Domain Setting entsprechen. z.B: "htlwrn.ac.at"

$qbe_http_globalservername Vollständiger Servername, z.B: `"qbe-auth.".$qbe_http_globaldomain`

$sas_samba_domainsid Die Samba Domain-SID.
z.B: "S-1-5-21-1021225642-3915188714-2801850423"

$qbe_app_frontpage PHP-Skript, welches in der Startseite angezeigt wird. Default: leer.

$qbe_util_arp Pfad zum ARP Programm mit numerischen IP-Adressen. z.B: "/usr/sbin/arp -n "

1.3 Applikationsmodule

Die Applikationsmodule bestehen aus Dateien in diesen Verzeichnissen:

/qbe/etc/MODULNAME/ Enthält modulspezifische Konfigurationsdateien.

/qbe/web/htdocs/modules/MODULNAME/ Haupt-Modulordner, alle ausführbaren Webskripte, Grafiken, etc. liegen hier. Zusätzlich existiert eine defines.php, die Modulinformationen, Menübeschreibungen und globale Modulfunktionen enthält.

/qbe/web/htdocs/modules/MODULNAME/defines.php Enthält die Menüdefinitionen für das Modul und eventuell vorhandene globale Funktionen.

/qbe/web/htdocs/rpc/MODULNAME/ Ausführbare RPC Objekte, diese sollten die sas.inc.php nicht benutzen.

/qbe/sbin/qbe-MODULNAME-... Ausführbare Hintergrundprogramme

1.3.1 core

`core` stellt die Kernfunktionalität des Application-Servers zur Verfügung. Dem core-Modul gehören auch die Hauptkonfigurationsdateien sowie weitere Dateien ausserhalb des Modulverzeichnisses an:

admin/admin/finger.php Kann verwendet werden um die Un*x-Details eines Benutzers abzufragen.

admin/index.php Das Anmeldeformular

admin/logout.php Abmeldung des Benutzers

graphics/style.css Stylesheet für die Qbe SAS Seiten

graphics/qbe.sas.about.png Großes Qbe SAS Logo für die Release-Informationsseite

graphics/qbe.sas.topright.png Kleines Qbe SAS Logo für das Menü rechts oben

index.php Die Startseite – eine einfache Page die definierbare Inhalte darstellen kann. (Siehe Konfiguration.)

modules/defines.php Lädt alle aktiven Module.

sas.inc.php Master-Include, stellt die gesamte Basisfunktionalität (Seitenstart, -ende, Links, Menü, Hilfe, ...) zur Verfügung.

Dateien innerhalb des Modulverzeichnisses:

about.php Eine graphisch ansprechende Informationsseite über Qbe SAS, Copyright-Informationen.

checklogin.php Check, ob der Benutzer angemeldet ist, falls nicht, Login und dann Weiterleitung auf Original-URL. Ist ein Qbe SAS Client mit der HTTP Client IP-Adresse registriert, wird dessen Authentifizierung benutzt.

chpass.php Frontend zum Passwort ändern, greift auf die User-Provider-Funktion zurück.

datenschutz.php Stellt Informationen über den eigenen Benutzer in halbwegs verständlicher Form dar und informiert über einige Grundsätze des Datenschutzes.

lookup.php Sucht den passenden Provider für das $subject und leitet den Benutzer auf die entsprechende URL.

lookup-helper.php Für Popup-Lookups enthält dieses File Javascript-Code, um das Original-Formular zu befüllen.

sendmsg.php Sendet (ohne Background Task) eine Nachricht an den Qbe SAS Clients des ausgewählten Benutzers.

Weiters enthält das **core** Modul einige **Background Tasks** , die sich um den Systemstatus usw. kümmern.

/qbe/web/syscheck.pl Dieses Perl Skript wird vom cron alle 5 Minuten aufgerufen und überprüft, ob die wichtigsten Dienste (sasd, ndsd, mysqld, apache, dhcpd und smbd) laufen, und schreibt mit diesen Informationen die Datei `/qbe/web/sysstate.php`. Diese sysstate.php wird vom Master Include eingebunden und ist für die Applikationen als Funktion `sysstate()` verfügbar.

/qbe/web/cron-10min.sh Dieses Shell Skript wird vom cron alle 10 Minuten aufgerufen und konfiguriert den DHCP neu bzw. meldet Benutzer ohne aktiven Client vom System ab.

/qbe/web/cron-daily.sh Dieses Shell Skript wird täglich vom cron aufgerufen und löscht die importierten EDVO Benutzer aus dem LDAP Directory. Weiters werden die Dateien unter /export/share-free/ gelöscht.

1.3.2 redir

Stellt erweiterte URL-Weiterleitungsfunktionen zur Verfügung.

Dateien innerhalb des Modulverzeichnisses:

outside.php Baut ein iframe mit dem Qbe SAS Template und der Original-URL auf.

ssl.php Leitet den Benutzer (falls SSL eingeschaltet ist) auf den HTTP-SSL Port des Application Servers weiter.

1.3.3 ldap

Providermodul, dass die Authentifizierung und Verwaltung von Benutzern im LDAP Verzeichnis ermöglicht.

Aufgrund anfänglich nicht modularer Implementierung sind viele Dateien des `ldap`-Modules über die alte Verzeichnisstruktur verteilt:

admin/login.php Meldet den, in den POST-Variablen `user` und `pass` spezifizierten Benutzer, an und speichert alle relevanten Daten in die $_SESSION Variable.

admin/activation.php Benutzer mit Erst-Passwort werden auf diese Seite umgeleitet, um ihr Passwort zu ändern. Dabei wird dann auch das Benutzerverzeichnis angelegt.

1.3.4 ldif

Providermodul für den Import von Benutzern in den LDAP Tree.

Dateien im Modulverzeichnis:

import.php Importiert Benutzerlisten im CSV-Format

import_passwords.php Importiert nur die Passwörter von Benutzerlisten (CSV)

1.3.5 computer

Stellt die Verwaltung der Computer-Objekte und der Notebook-Attribute der Benutzerobjekte zur Verfügung.

act.php Verwaltung bereits existierender Computer Objekte

add-client.php Fügt ein neues Computer Objekt hinzu

getip.php Zeigt die aktuelle IP und MAC-Adresse des HTTP Clients oder von anderen Computern

manage-clients.php Verwaltung bereits existierender Computer Objekte: Auflistung

Andere Dateien:

admin/tools/request_clearance.php Komplettes Verwaltungsinterface für die Notebooks der Benutzer

Als **Background Task** existiert nur die qbe_dhcpconf.pl, die vom cron-10min.sh aufgerufen wird, und die statischen DHCP Einträge exportiert.

1.3.6 client

Dieses Modul stellt ausschliesslich RPC-Objekte für den Qbe SAS Client zur Verfügung.

Die RPC Objekte befinden sich im `/rpc/client` Verzeichnis und sind im Kapitel **??** dokumentiert. Aliasnamen zur Kompabilität mit iLogin v2 <= 2.20 wurden mit Qbe Application Server Version 0.91 entfernt. Alte Clients können sich daher nicht mehr anmelden.

index.php Auswahlhilfe für die Qbe SAS Client Downloads

update.php Wertet den GET-Parameter "ver" aus und sendet entweder Statuscode 404 (Aktuelle Version ok) oder die neue Installations-Datei

version.php Enthält die aktuelle und die minimal notwendige Version des SAS Clients

1.3.7 filexs

Dieses Modul stellt im Webinterface eine Möglichkeit zur Verfügung, die Dateien im eigenen Benutzerverzeichnis zu verwalten. Folgende Aktionen sind möglich: Datei herunterladen (fileget), Datei abspeichern (fileput), Löschen (unlink bzw. rmdir), Umbenennen (rename) – alle Aktionen werden im setuid/setgrp-Bereich des jeweiligen Benutzers ausgeführt. Außerdem kann auf die "free"- und "alle"-Freigaben zugegriffen werden.

Alle zugehörigen Dateien befinden sich in den entsprechenden Verzeichnissen.

Dateien im Modulverzeichnis:

index.php Listet das ausgewählte Verzeichnis auf.

inc.php Modul-Include

act.php Frontend für den qbe-filexs Background-Task.

xfer-get.php Frontend für den qbe-filexs BgTask: Dateidownload (fileget)

xfer-put.php Frontend für den qbe-filexs BgTask: Dateiupload (fileput)

Das filexs Modul enthält nur den Pseudo-**Background-Task** "qbe-filexs" (ein setuid root-Binary), welcher sich um die eigentlichen Dateizugriffe kümmert. Damit liegen alle Sicherheitsprobleme und die Access-Control in diesem Background Task.

```
ch@xtc:/qbe/sbin -> ls qbe-filexs
-r-sr-sr-x    1 root     root         7643 Dec 19 13:10 qbe-filexs

Aufrufparameter:
/qbe/sbin/qbe-filexs user group action file [file2]
                     |    |     |      |    |
                     |    |     |      |    Ein zweiter Dateiname
                     |    |     |      Dateiname
                     |    |     Die auszuführende Aktion
                     |    Gruppenname oder "-"
                     Benutzer unter dem die Aktion ausgeführt
                     werden soll.
```

1.3.8 changelog

Stellt die Führung des System-Änderungsprotokolls durch Administratoren zur Verfügung – nur Administratoren können das ChangeLog einsehen. Die Einträge (bestehend aus Datum, Benutzername und Logtext) werden in der SQL-Tabelle sas.changelog gespeichert.

Dateien im Modulverzeichnis:

prettyprint.php Gibt das komplette ChangeLog als eine HTML Tabelle ohne weitere Stilinformationen aus.

latex.php Gibt das komplette ChangeLog als LaTeXlongtable aus.

index.php Listet das ChangeLog innerhalb des Templates auf und ermöglicht Administratoren die Eingabe von neuen Einträgen.

1.3.9 nagios

Ein typisches Custom-Modul, stellt für die Startseite (die Verwendung ist getrennt zu konfigurieren) Inhalt (Nagios Übersichtsbild) und einen Reverse Proxy zur Verfügung.

Dateien im Modulverzeichnis:

frontpage.php Custom Inhalt für Startseite

.htaccess Konfiguriert einen Reverse Proxy, basierend auf mod_rewrite, für das Bild dass die Nagios-Software erstellt

1.3.10 help

Implementiert das Hilfesystem. Die Seiten werden mit `index.php` dargestellt (welches nur im PopUp-Modus arbeitet), die einzelnen Hilfeseiten werden unter `topics` abgelegt.

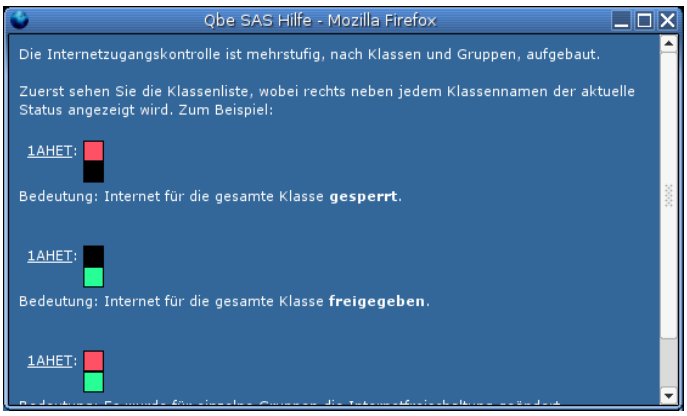


Abbildung 1.1: Screenshot des Hilfe-Fensters

1.3.11 dev

Zeigt einige Funktionen der Qbe AppServer Engine, mit Codebeispielen. Legt keine Menüeinträge an.

Dateien im Modulverzeichnis:

demo.php Zeigt Codebeispiele für gebräuchliche Funktionen.

masterinc.php Zeigt Funktionen aus dem Master Include File.

1.3.12 internet

Mit diesem Modul wird die Integration mit dem Qbe SAS Proxy (Kapitel 2) realisiert. Am Authentication Server können die einzelnen Klassen oder Gruppen freigeschaltet werden. Das Modul führt über jede Internetstatus-Änderung Protokoll, inklusive Zeit und IP-Adresse, welches dann durch Administratoren einsehbar ist. Damit kann man relativ leicht herausfinden, ob das Passwort eines Lehrers bekannt geworden ist.

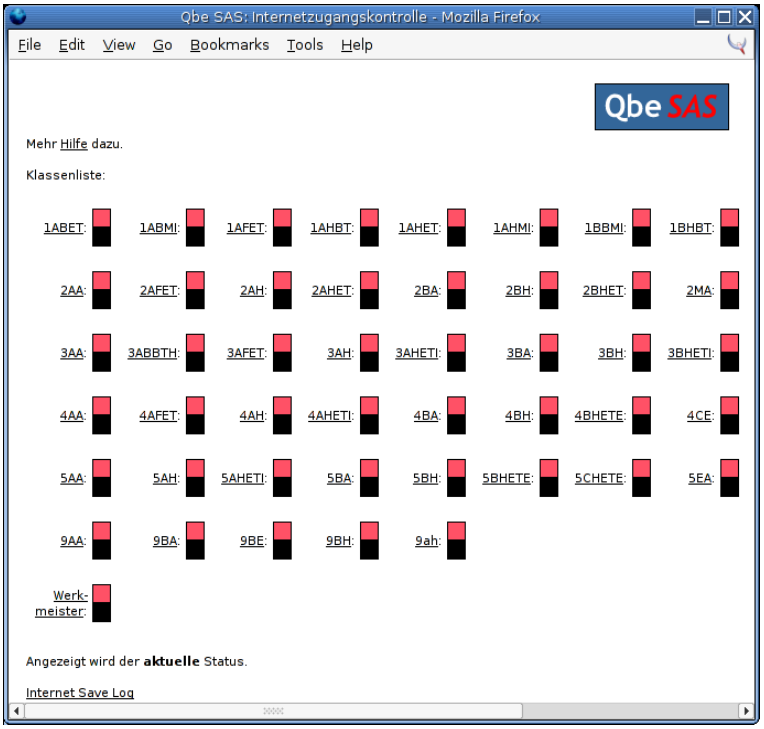


Abbildung 1.2: Internetzugangskontrolle

Dateien im Modulverzeichnis:

index.php Stellt eine grafisch ansprechende Ansicht über die Klassen und Gruppen dar, siehe Bild 1.2

save.php Validiert und speichert den neuen Internetstatus für Klassen bzw. Gruppen

pconly.php Erlaubt es einzelne Computer freizuschalten

log-inetsave.php Stellt die vergangenen Internetstatusänderungen in einer Tabelle dar

stats-traffic-class.php Summiert das Trafficvolumen auf Anfrage klassenweise

stats-traffic-overall.php Zeigt das gesamte Trafficvolumen aufgesplittet nach Abteilungen auf

stats-traffic-overall.chart.php Grafikausgabe für `stats-traffic-overall.php`

1.3.13 rfid

Implementiert die Elektronische Inventarverwaltung, ein weiteres Projekt der HTBLuVA Wiener Neustadt.

1.3.14 sis

Enthält die Implementierung des Schul-Informations-Systems, ein weiteres Projekt der HTBLuVA Wiener Neustadt.

Kapitel 2

Proxy – Internetgateway

Der Qbe SAS Proxy stellt die Verbindung zum Internet her. Alle ausgehenden Verbindungen passieren den Proxy – teilweise durch Application Proxies oder IP-Forwarding. Zugriffe werden zentral über Qbe SAS kontrolliert und protokolliert. Der Proxy wird fast ausschliesslich mit Standard OpenSource Software implementiert.

2.1 Software

Folgende Standardsoftware wird verwendet:

- Debian GNU/Linux woody oder OpenBSD 3.2+ oder FreeBSD 4.x
- Linux Kernel 2.4.18+
- Squid Cache 2.4-STABLE oder 2.5-STABLE – HTTP Proxy
- Frox – FTP Proxy
- Perl 5.8 und Module Net::LDAP, File::Tail, DBI, MySql
- ISC BIND 9 – DNS

Der Proxy lädt ein Modul, welches die Benutzerauthentifizierung überprüft. Zusätzlich läuft nur ein Perl-Skript, das sich mit dem Volumenaccounting beschäftigt. Das Skript und das Modul werden üblicherweise in `/qbe/sbin/` abgelegt und können ggf. mit rsync vom AuthServer synchronisiert werden.

Notiz: Es ist nicht notwendig eine LDAP-Benutzerauthentifizierung für das System (Stichwort PAM) einzurichten. Idealerweise gibt es auf dem Qbe Proxy nur Systemaccounts und einen Systemverwalter (nicht `root`). `root` sollte sich (wie auch am Application Server) nicht übers Netzwerk anmelden können.

2.2 qbe-proxy-squidlog.pl – Auswertung der Squid Cache Logs

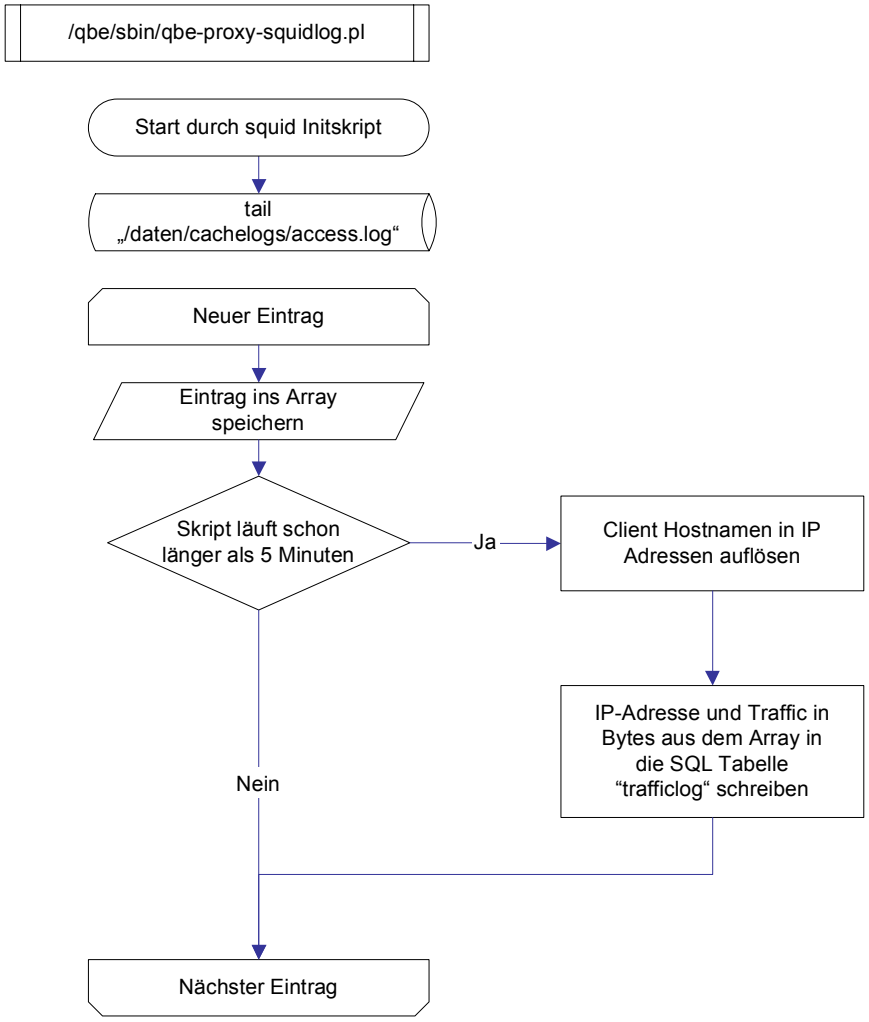
Das Perlskript liest die Squid-Logdatei kontinuierlich aus und wertet die Einträge aus, interne Zugriffe werden dabei ignoriert. Die Datentransfergröße wird (pro Client-IP) im RAM abgespeichert. Alle 5 Minuten werden die Daten im RAM in die MySQL-Tabelle sas.trafficlog gespeichert. Am AuthServer werden diese Daten dann zusammengezählt und in die LDAP-Datenbank hinzugefügt. Traffic, der nicht einem Benutzer zugeordnet werden kann, wird beim `nobody`-User dazugezählt. Für eine IP-basierende-Auswertung werden die Daten in eine getrennte Tabelle `sas.trafficip` gespeichert. Weiters kann am AuthServer eine Grafik über den Traffic-Verlauf erstellt werden.

Tabellenaufbau:

```
1 CREATE TABLE trafficlog (
2   ip varchar(60) NOT NULL default '',
3   traffic bigint(20) default NULL,
4   KEY client (ip)
5 ) TYPE=MyISAM;
6
7 CREATE TABLE trafficip (
8   ip varchar(20) NOT NULL default '',
9   traffic bigint(20) NOT NULL default '0',
10  PRIMARY KEY  (ip)
11 ) TYPE=MyISAM;
```

Ein typischer, temporärer Eintrag:

```
1 INSERT INTO trafficlog VALUES ('10.3.5.1',135084);
2 INSERT INTO trafficip VALUES ('10.1.40.1',8991453419);
```


/qbe/sbin/qbe-proxy-squidlog.pl
Start durch squid Initskript
tail „/daten/cachelogs/access.log“
Neuer Eintrag
Eintrag ins Array speichern
Skript läuft schon länger als 5 Minuten
Ja
Client Hostnamen in IP Adressen auflösen
IP-Adresse und Traffic in Bytes aus dem Array in die SQL Tabelle “trafficlog“ schreiben
Nein
Nächster Eintrag

2.2.1 Grafik-Auswertung nach Benutzern

Am Authentication Server wird jede Stunde einmal das Skript /qbe/sbin/qbe_trafficview.pl aufgerufen. Dieses aggregiert die Daten aus dem LDAP und kopiert die Daten pro Benutzer in die Tabelle `sas.trafficview`. Die Tabelle wird im UI mittels /modules/internet/stats-traffic-overall ausgewertet und als Balkendiagramm je Abteilung dargestellt.

```
1 CREATE TABLE trafficview (
2   userid varchar(10) NOT NULL default '',
3   traffic bigint(20) NOT NULL default '0',
4   abt varchar(5) NOT NULL default ''
5 ) TYPE=MyISAM;
```

Ein Beispieleintrag:

```
1 INSERT INTO trafficview VALUES ('cb',1123132,'Adm');
```

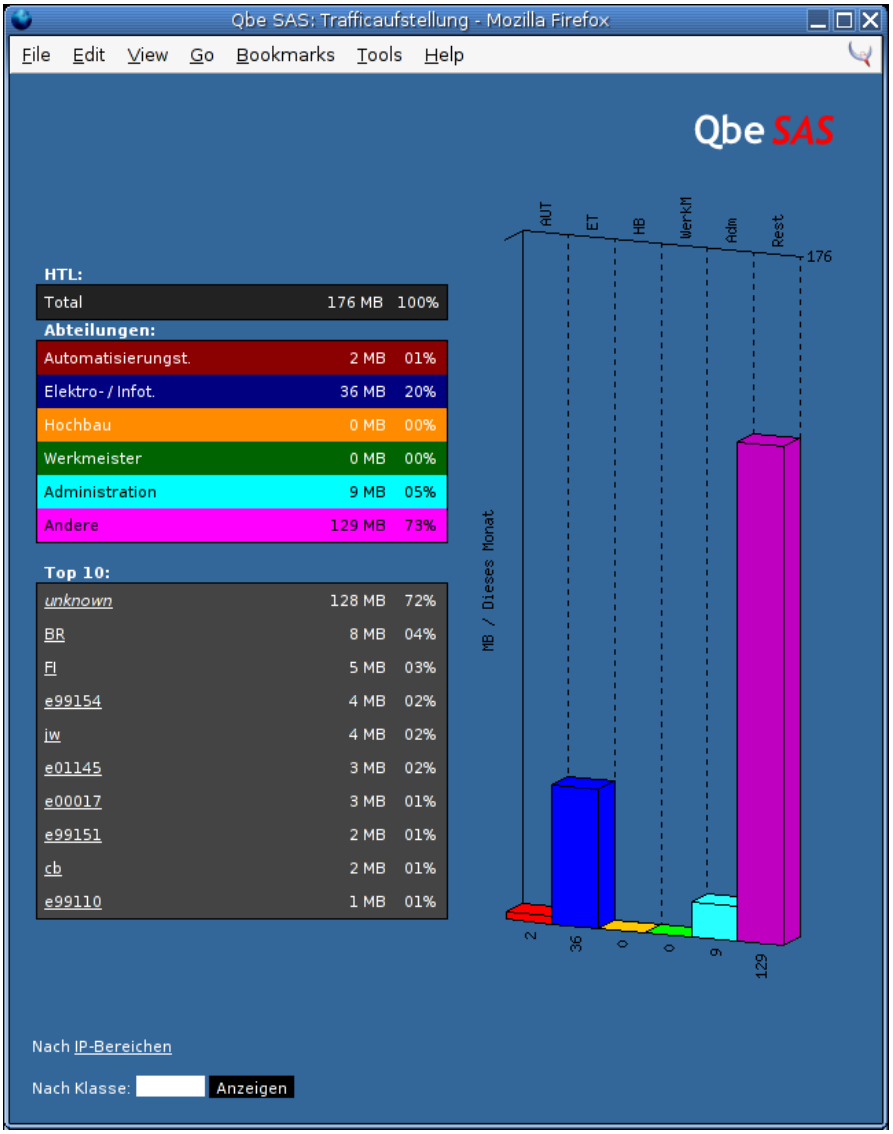


Abbildung 2.1: Graphische Auswertung

2.3 Zugriffssteuerung für Squid Cache

Frühere Qbe SAS Systeme setzten auf ein Perlskript welches regelmäßig die aktuell angemeldeten Benutzer abgefragt und in eine Datei geschrieben hat. Um die damit verbundenen Probleme zu lösen, wird jetzt ein Modul in den Squid Cache geladen. Das Modul heisst "ip_ldap" und wird als "external acl" eingetragen.

2.4 Änderungen an der System-Konfiguration

Es werden hier kurz die notwendingen Änderungen an der Standardkonfiguration eines Debian-Systems beschrieben.

2.4.1 Squid: Initskript

Das Initskript des Squid Cache daemons `/etc/init.d/squid` muss angepasst werden, um `qbe-proxy-squidlog.pl` entsprechend zu starten bzw. zu beenden.

Ans Ende der `start()`-Routine gehört folgende Zeile:

```
1 # Qbe SAS Proxy
2 /qbe/sbin/qbe-proxy-squidlog.pl&
3 # END
```

An den Anfang der `stop()`-Routine muss folgendes hinzugefügt werden:

```
1 # Qbe SAS Proxy
2 killall "qbe-proxy-squidlog.pl"
3 # END
```

2.4.2 Squid: ACL

Um den Clients die Benutzung des Squid Caches zu erlauben, muss in der `/etc/squid/squid.conf` ein ACL-Eintrag hinzugefügt werden:

```
1 # angemeldete Benutzer
2 external_acl_type ldapacl ttl=45 \%SRC \%IDENT /usr/lib/squid/
    ip_ldap
3 acl proxy-users external ldapacl
```

Anschliessend muß diese ACL in der `allow`-Klasse eingetragen werden:

```
1 http_access allow proxy-users
```

2.5 WebSense EIM

Soll der "WebSense Employee Internet Manager" installiert werden (der für österreichische Schulen zum Zeitpunkt des Schreibens nur einer kostenlosen Registrierung bedarf), muss mindestens Squid 2.5-STABLE verwendet werden. Für Debian woody ist daher ein eigenes Paket notwendig.

Mit einem kleinen Patch kann das sid-Sourcepaket (hier: 2.5.4-3) verwendet werden. Es sind dann noch zwei Pakete aus unstable zu installieren, diese sind jedoch nicht plattformabhängig und funktionieren ohne Modifikation.

```
1  diff -r -u squid-2.5.4/debian/changelog squid-2.5.4-wo/debian/
       changelog
2  --- squid-2.5.4/debian/changelog     Fri Jan 23 09:12:42 2004
3  +++ squid-2.5.4-wo/debian/changelog Thu Jan 15 16:53:30 2004
4  @@ -1,3 +1,9 @@
5  +squid (2.5.4-4) stable; urgency=low
6  +
7  +  * ch special release
8  +
9  + -- Christian <ch@ionus.at>  Fri,  5 Dec 2003 04:10:56 +0100
10 +
11   squid (2.5.4-3) unstable; urgency=low
12
13     * debian/po/pt_BR.po
14 diff -r -u squid-2.5.4/debian/control squid-2.5.4-wo/debian/
       control
15 --- squid-2.5.4/debian/control   Fri Jan 23 09:12:42 2004
16 +++ squid-2.5.4-wo/debian/control   Thu Jan 15 17:13:49 2004
17 @@ -9,7 +9,7 @@
18   Architecture: any
19   Section: web
20   Priority: optional
21 -Pre-Depends: debconf (>= 1.2.9)
22 +Pre-Depends: debconf
23   Depends: ${shlibs:Depends}, netbase, adduser, logrotate (>=
       3.5.4-1)
24   Conflicts: squid-novm, sarg (<< 1.1.1-2)
25   Replaces: squid-novm
26 diff -r -u squid-2.5.4/debian/rules squid-2.5.4-wo/debian/rules
27 --- squid-2.5.4/debian/rules     Fri Jan 23 09:12:42 2004
28 +++ squid-2.5.4-wo/debian/rules Thu Jan 15 17:02:12 2004
29 @@ -353,7 +353,7 @@
30
31   checkroot:
32       $(checkdir)
33 -     test root = "`whoami`"
34 +#    test root = "`whoami`"
35
```

```
36 dist: binary
37     dpkg-source -b
```

2.6 Firewall Hinweis

Es soll hier ein Hinweis auf `iptables` (bzw. `ipf` oder `pf` unter FreeBSD/OpenBSD) gegeben werden, mit denen eine Firewall-Funktionalität aufgebaut werden kann. Dies ist dringend zu empfehlen. Es sollten auch keine anderen Dienste auf dem Qbe SAS Proxy laufen (z.B. Webserver, MySQL...) da diese ein nicht einschätzbares Sicherheitsrisiko beherbergen können.

2.7 Ideen für die Zukunft, bessere Skalierbarkeit

Die in dieser Version eingesetzte ACL-Kontrolle über eine Datei funktioniert zwar, führt jedoch teilweise zu obskuren Problemen. Da der Squid die ACL-Datei nur bei einem SIGHUP neu einliest, und dabei leider manchmal offene Verbindungen (warum er dies tut, ist mir unbekannt) unterbricht, wäre eine z.B. MySQL-basierende ACL besser. Dazu muss jedoch der Squid entsprechend erweitert werden und die Login/Logout Skripte am Authentication Server angepasst werden. Änderungen am Internet-Status der Benutzer könnte man mit einem LDAP-Event abfangen und damit die Datenbank aktualisieren.

Eine andere Möglichkeit wäre das Squid external-acl API zu verwenden. Dann läuft ein kleines Programm, dass nur mit dem LDAP Server spricht, sobald der squid einen Benutzer authentifizieren muss. Die Cache-Vorhaltezeit des Ja/Nein-Zustandes ist dann im Squid selbst konfigurierbar.

Kapitel 3

Webserver Integration

Qbe SAS ist vorbereitet um in Kooperation mit einem getrennten Webserver, die Webseite der Institution und die persönlichen Seiten der Systembenutzer anzuzeigen.

3.1 Software

Folgende Standardsoftware wird verwendet:

- Debian GNU/Linux woody oder OpenBSD 3.2+ oder FreeBSD 4.x
- Apache HTTP Server 1.3 oder 2.0
- NFS Client

3.2 Konfiguration

Da keine spezielle Software verwendet wird, beschränkt sich die Konfiguration auf das NFS Filesystem und den Apache Webserver.

Notiz: Es ist nicht notwendig, eine LDAP-Benutzerauthentifizierung einzurichten. Idealerweise gibt es auf dem Webserver nur Systemaccounts und einen Systemverwalter (nicht `root`). `root` sollte sich nicht über das Netzwerk anmelden können.

3.2.1 NFS

Datei `/etc/fstab` muss um folgenden Eintrag (in einer Zeile) ergänzt werden:

```
1 10.0.2.100:/export/homes /import/homes nfs rw,soft,
    timeo=60,async,nodev,noexec,nouser,nosuid 0 0
```

Dies weist das System an, beim Neustart automatisch das Dateisystem mit den Benutzerverzeichnissen via NFS vom AuthServer (hier: 10.0.2.100) zu importieren. Zusätzlich werden einige Parameter gesetzt die die Geschwindigkeit und Sicherheit positiv beeinflussen.

3.2.2 Apache httpd

In der Apache Konfigurationsdatei `httpd.conf` muss folgendes sinngemäß hinzugefügt werden (Beispiel für Apache Version 1.3):

```
1  ; Modul fuer Benutzerverzeichnisse laden
2  LoadModule userdir_module      modules/mod_userdir.so
3
4  ; Fuer root kein Benutzerverzeichnis, fuer alle anderen in ~/web/
5  <IfModule mod_userdir.c>
6      UserDir disabled root
7      UserDir /import/homes/*/web
8  </IfModule>
9  <Directory /import/homes/*/web>
10     AllowOverride All
11     Options Indexes Includes
12     Order allow,deny
13     Allow from all
14 </Directory>
```

Soll auch die Institutionsseite am AuthServer abgelegt werden, kann diese im Benutzerverzeichniss des Benutzers "web" geschehen. Dazu muss im `httpd.conf` zusätzlich folgendes eingetragen werden:

```
1  DocumentRoot "/import/homes/web"
```

Kapitel 4

Qbe SAS Client

Qbe SAS Client soll primär auf Clientsystemen mit Betriebssystem Windows 2000 und/oder Windows XP Professional laufen. Wenn möglich soll nur ein kleiner Teil für den Benutzer sichtbar sein (Anmeldung, Statusabfrage und Abmeldung) - die Funktion des Systems sollte nicht weiter in den Vordergrund gerückt werden.

Ebenfalls soll der gesamte Clientteil nur über einen definierten minimalen Befehlssatz mit dem Qbe Auth Server kommunizieren und keine direkten LDAP API Aufrufe durchführen, um maximale Portabilität (möglicherweise auch von LDAP weg) zu erreichen. Bestimmte Fehler am Client sollen nicht die Fähigkeit der Kommunikation mit dem Qbe Auth Server beeinträchtigen; Netzwerk-Disconnects sollen (besonders auf Laptops) transparent behandelt werden.

4.1 Betriebsarten

Qbe SAS Client übernimmt die Authentifizierung des Benutzers, der einen beliebigen Client PC im Netzwerk benutzen möchte. Dazu kann der Client in zwei verschiedenen Modi betrieben werden:

Network only Qbe SAS Client authentifiziert den Benutzer nur gegenüber dem Server. Der Benutzer gibt seinen Benutzernamen und Passwort getrennt von der Windows-Anmeldung ein.

System Logon Qbe SAS Client authentifiziert den Benutzer sowohl gegenüber Windows als auch dem Qbe Server - bereits beim Anmelden an die Windows Workstation. Bei Bedarf wird ein lokaler Benutzer mit konfigurierbaren Rechten angelegt, und bei der Abmeldung wieder gelöscht.

4.2 Serversuche

Der zu benutzende Server wird ausschließlich per DNS gefunden. Standardmäßig benutzt Qbe SAS Client den Servernamen "qbe-auth". Die Auflösung des Namens in eine IP Adresse wird vom Betriebssystem durchgeführt - Voraussetzungen dafür sind:

- DNS-Server läuft und ist richtig konfiguriert
  ("qbe-auth" ist als A-Record eingetragen.)
- Die Workstation kann den DNS-Server erreichen, und fragt mit dem richtigen Domainsuffix an. Die richtige Konfiguration der Workstation wird vorzugsweise mittels DHCP erreicht.

In der aktuell vorhandenen Version besteht keine Möglichkeit den anfänglichen Servernamen zu ändern. Geplant ist, eine eigene DHCP-Option, die vom Qbe SAS Client ausgelesen werden kann - Voraussetzung dafür wäre, dass die Workstation die IP-Stack Konfiguration vom DHCP-Server bekommt.

```
; DNS-Beispieleintrag:
qbe-auth.htlwrn.ac.at.        IN        A                10.0.2.100
```

4.3 Einzelteile

Der Qbe SAS Client ist ein komplexes System, dass aus vielen kleineren Teilen besteht:

Teilsystem	Dateiname
Network Authentication Service	QbeSvc.exe
	QbeSAS.dll
System Logon	QbeGina.dll
Statusanzeige	QbeTray.exe
UI Components	SASClient.hta
	startup.hta
	Q.ico
Einstellungen	SASClient.reg
Installation	ServicePack.exe
	netQbe.inf
	QbePFC.exe

Qbe SAS Client kann bereits nur mit dem installiertem "qbesvc.exe" und der "QbeSAS.dll" im "Network only" Modus betrieben werden – der Xplat Client implementiert prinzipiell nichts anderes und kann auch unter Windows gestartet werden. Für den "System Logon" Modus bzw. für eine benutzerfreundlichere Umgebung, sollte der Client vollständig installiert werden.

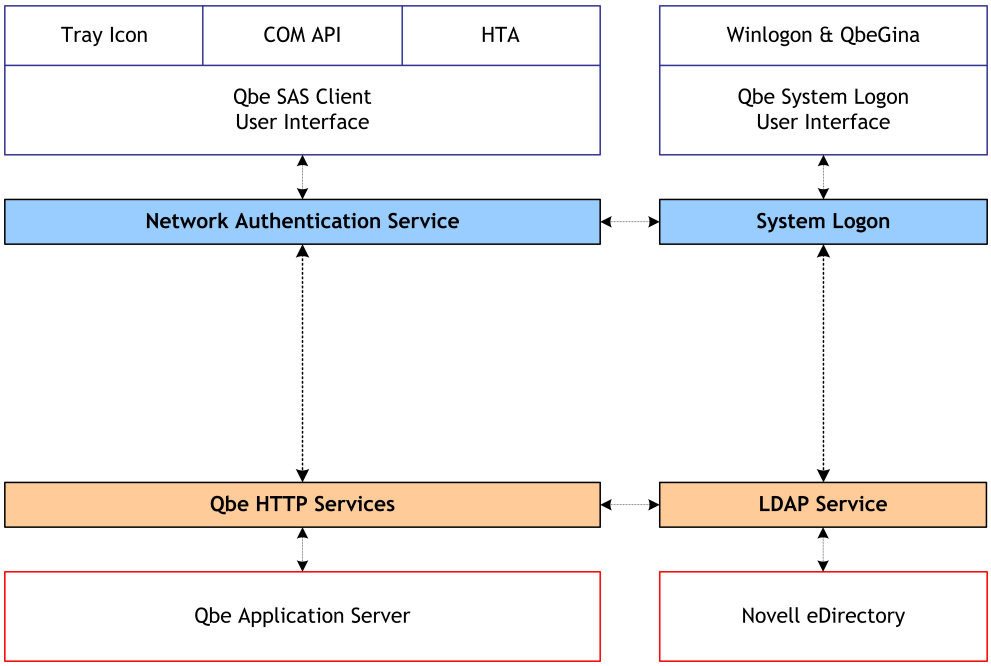


Abbildung 4.1: Beziehungsdiagramm Qbe SAS Client/Server

Im Beziehungsdiagramm wird deutlich, dass der Network Authentication Service alle grundlegenden Dienste für den SAS Client zur Verfügung stellt. Wird der System Logon Mode benutzt, wird ein zusätzlicher Teil systemwichtig – die SASGina. Dieser wird in das Windows Login eingebunden, und übernimmt die Windows Anmeldung und die Authentifizierung gegenüber dem LDAP Server.

4.3.1 Network Authentication Service

Dieser läuft als Windows-Dienst und implementiert sowohl einen HTTP-SSL Client für die Kommunikation mit dem Qbe-Auth Server, als auch einen HTTP Server (auf TCP/IP Port 7666), der die Interaktion mit dem Benutzer ermöglicht und Teile des Authentifizierungshandshakes abarbeitet. Auf den Network Authentication Service kann mit einem normalen Webbrowser zugegriffen werden, speziell wurde dies aber für den Internet Explorer (bzw. für die Microsoft HTML Engine) entwickelt und optimiert, da die restlichen Benutzerinterface-Teile auf der Microsoft HTML Engine aufsetzen.

Alle weiteren Teile kommunizieren ausschließlich mit dem Network Authentication Service, der alle relevanten Daten über den Benutzer im Speicher hält - diese Daten werden regelmäßig vom Server aufgefrischt.

Der HTTP Server ermöglicht es dem Server außerdem, bestimmte Parameter (Benutzername, Servername, Letzte Verbindung, ...) abzufragen, sowie Programme auszuführen (etwa

um einen Drucker zu installieren). Dies wird durch eine Abfrage der IP-Adresse geschützt.

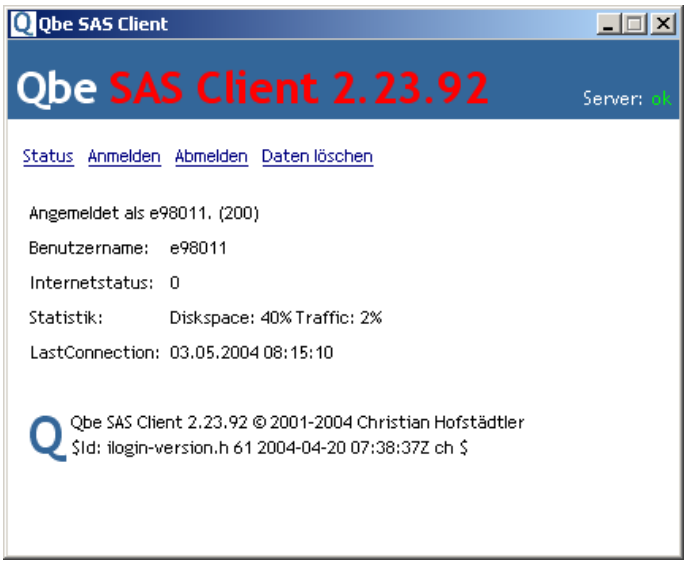


Abbildung 4.2: Screenshot Qbe SAS Client unter Windows

4.3.2 COM API and Authentication Helper

Das COM API ermöglicht es, auf den Network Authentication Service per Microsoft COM (ActiveX) zuzugreifen. Gleichzeitig erlaubt es eine Speicherung der Benutzerdaten (Username und Passwort) in der Registry, um die Anmeldung zu beschleunigen.

Bis zur Version 2.23.96-pre wurde eine DLL (geschrieben in Visual Basic 6) verwendet, die ein normales COM Objekt implementiert. Jetzt wird dies durch die Klasse QbeSAS::DataStore (in der .NET QbeSAS.dll) implementiert.

4.3.3 Statusanzeige

Die Statusanzeige besteht aus einem Windows System Tray Symbol, dass den aktuellen Internetstatus signalisiert, wichtige Ereignisse dem Benutzer mitteilt (Internet freigegeben) und schnellen Zugriff auf die Client Statusseite und Anmeldung erlaubt. Für die Statusseite wird die `SASClient.hta` gestartet, beim Start des QbeTray wird die `startup.hta` aufgerufen.

Energie-Events werden vom Tray Symbol behandelt. Bei einem kritischen Energieereignis (Standby/Ruhezustand) wird versucht, eine Abmeldung durch den Network Authentication Service zu erreichen. 30 Sekunden nach dem Aufwachen wird eine transparente Verbindungswiederherstellung eingeleitet.

4.3.4 UI Components

Die sogenannten UI Components bestehen lediglich aus Beschreibungsdateien für Microsoft HTA Applikationen, die auf den http Server im Network Authentication Service verweisen.

Die Datei `startup.hta` wird beim Starten des Clients geladen und verweist auf `/rpc/client/svc-frame?mode=startup` am Authentication Server.

Die Datei `SASClient.hta` wird beim Linksklick auf das Q-Trayicon geladen und verweist auf `/rpc/client/svc-frame` am Authentication Server.

4.3.5 Automatische Einstellungen

Qbe SAS Client installiert bei der Anmeldung einige Standardeinstellungen (z.B. Proxykonfiguration), die direkt aus der Datei "iLogin.reg" kommen. Diese wird mit Hilfe vom Windows Registrierungseditor (regedit.exe) automatisch in die Registry importiert.

4.3.6 Update

Da der Client bei jeder Anmeldung die Versionsnummer an den Server sendet, können alte/-problematische Client-Versionen einfach ausgeschlossen werden. Clients die den Statuscode `404 - Not found` erhalten, sollten keine weitere Anfrage an den Server schicken.

Neuere Windows Clients (ab 2.01) stellen die Benutzerschnittstelle erst nach einer Anfrage an den Server dar. Damit kann ein Update des Clients erzwungen werden. Workstations mit HDGUARD werden am Client abgefragt und das Zwangsupdate wird übersprungen.

4.4 Installation

Das Setup bietet keine weiteren Optionen an, und entpackt zuerst alle Dateien nach %SystemRoot%/System32/Qbe/Setup. Die weitere Reihenfolge:

- Eine eventuell vorhandene, alte Version wird deinstalliert
- .NET VM wird gegebenenfalls heruntergeladen und installiert
- %SystemRoot%/System32/Qbe/Setup wird komplett gelöscht
- PFC wird in das System kopiert und ausgeführt (siehe unten)
- Dateien von früheren Versionen die nach %SystemRoot%/System32 und %ProgramFiles%/iLogin installiert wurden werden gelöscht
- Alle Dateien werden nach %SystemRoot%/System32/Qbe/Setup kopiert

- Windows installiert den Qbe SAS Client nach %SystemRoot%/System32/Qbe
- QbePFC /s registriert die QbeSAS.DLL im Global Assembly Cache
- QbeGina wird nicht automatisch installiert sondern muss mit dem QbeLogin.exe installiert werden

4.4.1 PFC

Qbe "Pre-Flight-Check" ist eine .NET Anwendung (ClientSource/QbePFC), die vor der eigentlichen Installation ausgeführt wird und folgende Tasks erledigt:

- vorherigen Versionen von QbeSvc, QbeTray, iLogin, etc. schliessen
- QbeSvc aus der Registry entfernen
- HTA-Ausführungsschicht (MSHTA) und Internet Explorer beenden

Ab Client-Version 2.23.96-pre enthält QbePFC auch ein .NET-to-COM Registrierungsmodul, dieses wird mit dem Parameter "/c" angesprochen. Dies wurde notwendig da .NET-Objekte die als COM-Objekte verwendet werden, im .NET Global Assembly Cache registriert werden müssen. Das .NET-Objekt ist hier die Klasse QbeSAS::DataStore und wird von den MSHTA Seiten aufgerufen.

4.4.2 System-Updates

Der SAS Client (für Windows) benötigt Windows 2000 oder XP mit installiertem .NET Framework. Das aktuelle Servicepack auf den Zielsystemen ist nicht zwingend erforderlich, wäre aber von Vorteil. Das Client Setup prüft vor der eigentlichen Client-Installation ob das .NET Framework und die Microsoft C++ Runtimes in Version 7.1 vorhanden sind - wenn nicht wird beides heruntergeladen und installiert.

4.4.3 .NET Framework

Die .NET Framework Version wird mittels Registry-Key überprüft, andernfalls wird das Setup heruntergeladen und gestartet. Das SAS Client Setup wird dabei nicht unterbrochen, sondern wartet nur auf das .NET Framework Setup.

```
URL: http://qbe-auth.htlwrn.ac.at/modules/client/files/dotnetfx.exe

ch@xtc:/qbe/web/htdocs/modules/client/files/ -> ls -la dotnetfx.exe
-rw-rw-r--  ch   sysops   24277024 Mar  4  2003 dotnetfx.exe
```

4.5 Windows Client kompilieren

Um die Qbe SAS Client Sourcen zu kompilieren, muss folgende Software auf dem Entwicklungssystem gegeben sein:

- Microsoft Visual Studio .NET 2003 mit C++ und C# Unterstützung
- Microsoft Platform SDK, February 2003
- GCC 3.x aus dem Cygwin Paket
- Nullsoft Install System 2.0
- Mono MCS für den mono/Xplat Build

Es muss die Umgebungsvariable `%DEVELDRIVE%` auf das Installationslaufwerk obiger Software gesetzt sein. Zum Beispiel: `set DEVELDRIVE=C:` wenn die Software auf C: installiert wurde. Weiters darf keines der Visual Studio Setups den PATH oder LIBS/INCLUDE etc. verändert haben. Diese Variablen werden durch das build-Skript automatisch für das Microsoft Platform SDK gesetzt.

Das Build-Skript setzt folgende Installationspfade voraus:

Visual Studio "%DEVELDRIVE%/Programme/Microsoft Visual Studio 2003"

Cygwin/GCC kein Pfad, muss bereits im %PATH% eingetragen sein

Platform SDK "%DEVELDRIVE%/Programme/Microsoft SDK"

NSIS 2.0 "%DEVELDRIVE%/Programme/NSIS20"

4.5.1 NSIS Patch

Der SAS Client Installer erfordert folgenden Patch des "NSISdl" Modules. Damit wird die Proxy-Konfiguration nicht mehr automatisch aus den Internet Explorer Einstellungen ausgelesen.

```
1 --- nsisdl.cpp~ 2004-02-02 00:55:08.000000000 +0100
2 +++ nsisdl.cpp  2004-02-03 20:39:18.000000000 +0100
3 @@ -310,7 +310,7 @@
4          char *buf=main_buf;
5          char *p=NULL;
6
7 -        HKEY hKey;
8 +/*        HKEY hKey;
```

```
9         if (RegOpenKeyEx(HKEY_CURRENT_USER,"Software\\Microsoft\\
              Windows\\CurrentVersion\\Internet Settings",0,KEY_READ,&
              hKey) == ERROR_SUCCESS)
10        {
11          DWORD l = 4;
12 @@ -335,9 +335,10 @@
13          buf[8192-1]=0;
14          RegCloseKey(hKey);
15        }
16 -
17 +*/
18        DWORD start_time=GetTickCount();
19 -       get=new JNL_HTTPGet(JNL_CONNECTION_AUTODNS,16384,(p&&p[0])?
       p:NULL);
20 +//       get=new JNL_HTTPGet(JNL_CONNECTION_AUTODNS,16384,(p&&p
       [0])?p:NULL);
21 +       get=new JNL_HTTPGet(JNL_CONNECTION_AUTODNS,16384,NULL);
22        int           st;
23        int           has_printed_headers = 0;
24        int           cl;
```

4.5.2 Kompilierungsskript

Um die Erstellung des Qbe SAS Client Installers drastisch zu vereinfachen, wurden alle Komponenten mit einem Makefile versehen. Die Makefiles werden durch das Skript "buildx.bat" in der richtigen Reihenfolge aufgerufen, welches wiederum über eines der build-w2k.bat oder build-wxp.bat aufgerufen wird. Die build-xxx.bat Skripte starten zuerst das Platform SDK `setenv.bat` mit Parametern, um das Build-Environment entsprechend herzurichten. Anschliessend wird buildx.bat ausgeführt... Die Zieldateien werden im Verzeichnis BIN/RETAIL/PLATFORMNAME erstellt.

4.5.3 Versionsnummern

Die Versionsnummer wird in der Datei Common/ilogin-version.h definiert. Die Datei wird von sehr vielen Komponenten benutzt um die Qbe SAS Client Ziel-Version zu ermitteln und darzustellen.

4.6 Qbe SAS Xplat Client

Der komplexe Aufbau und die enge Verwebung mit dem Windows Betriebssystem machen es (mit heutigen Mitteln) unmöglich, den normalen Qbe SAS Client unter z.B. Linux zu verwenden.

Auf der anderen Seite macht die .NET Architektur dies sehr einfach. So kann, mit der Annahme dass es pro PC nur einen Benutzer gibt, der gleiche Sourcecode verwendet werden, um ein mit Mono ausführbares Binary zu erstellen. Dieses Binary ("QbeService.exe") enthält dann die minimalste Funktionalität, die der Qbe SAS Client mitbringen muss. Da das User Interface hier ebenfalls über HTTP ausgeliefert wird, sieht der Client für den Endanwender relativ ähnlich zum Windows Client aus.

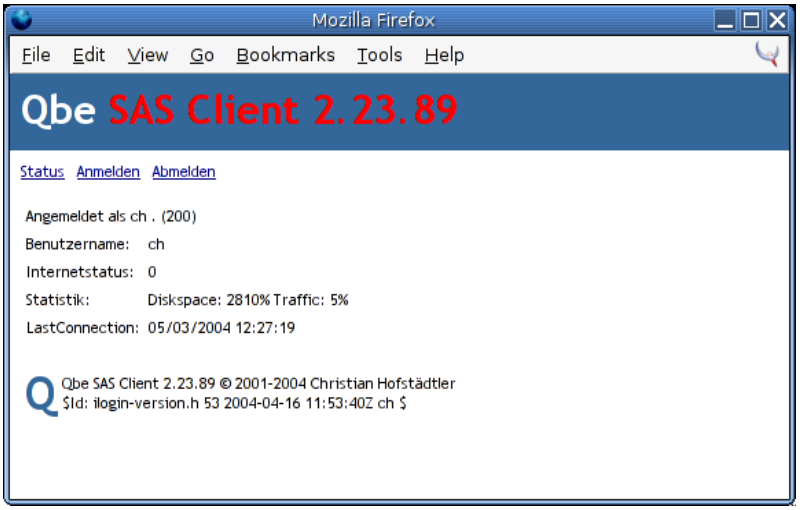


Abbildung 4.3: Screenshot: Qbe SAS Client unter Linux: Benutzeroberfläche

Zusätzlich zu dem QbeService.exe sind im Source-Tree ein paar Skripte, die die Verwendung mit Gnome als X11-Umgebung erleichtern können.

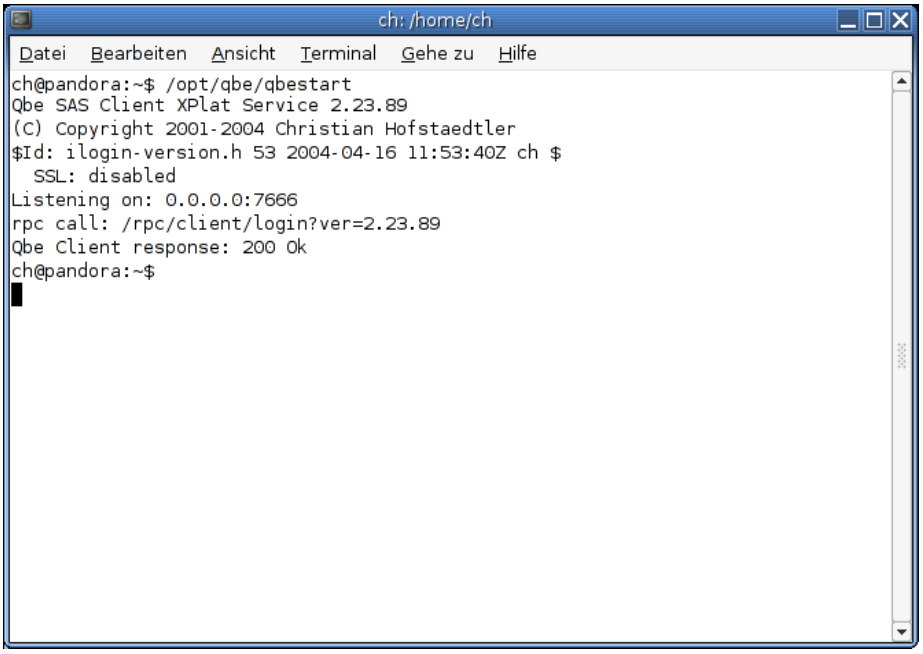


Abbildung 4.4: Screenshot: Qbe SAS Client unter Linux: Terminalfenster

4.7 System Logon

Die QbeGina und unterstützende Dateien werden mit dem "Windows Logon Enabler" Setup installiert. Die aktuelle Version implementiert eine transparente LDAP-Authentifizierung. Benutzer werden automatisch angelegt bzw. gelöscht. Homedirectory wird entsprechend eingestellt, der Profilpfad wird auf %HOMEDRIVE%/profile gesetzt. Im Anmeldedialogfeld kann mit der Tastenkombination CTRL-ALT-DEL die LDAP-Authentifizierung übersprungen werden – dann wird die normale MSGina.dll für diese Session aktiv.

Die Installation legt die Gruppe "Qbe SAS Users" an und trägt folgende Registry-Änderungen ein:

```
1 ; Registry Aenderungen
2 [HKEY_LOCAL_MACHINE/Software/Microsoft/Windows NT/CurrentVersion/
      Winlogon]
3 GinaDLL="QbeGina.DLL"
4 ComputerName="ComputerName"
5
6 [HKEY_LOCAL_MACHINE/SYSTEM/CurrentControlSet/Services/QbeNP]
7
8 [HKEY_LOCAL_MACHINE/SYSTEM/CurrentControlSet/Control/
      NetworkProvider/Order]
9 ProviderOrder+="QbeNP"
```

4.7.1 Arbeitsweise

Beim Windows **Boot** wird statt dem Windows Login Fenster ein Fenster mit der Schrift "Qbe SAS - Strg-Alt-Entf zum anmelden." angezeigt.

Drückt man dann Strg-Alt-Entf, beginnt die QbeGina die Policy-Konfiguration vom Server herunterzuladen. Dann gelangt man zur **Anmeldung** . Die QbeGina fragt hier den Benutzer um seinen Accountnamen und das Passwort, lässt aber keine Auswahl zwischen lokaler Anmeldung oder Netzwerk-Anmeldung zu. Um eine lokale Anmeldung zu erzwingen, kann man in diesem Fenster nochmals Strg-Alt-Enf drücken (damit wird ein Winlogon SAS Event ausgelöst) und man kommt zum normalen Windows Anmeldedialog.

Kann die QbeGina nach der Benutzerdatenabfrage die Verbindung zum LDAP Server nicht mehr wiederherstellen, wird der Benutzer gefragt, ob er eine lokale Anmeldung versuchen möchte (Bild 4.7).

Die Arbeitsweise der Anmeldung am Netzwerk und an der Windows Workstation ist aus der Abbildung 4.5 ersichtlich.

Beim **Abmelden** , wird geprüft ob der Benutzer Mitglied der Gruppe "Qbe SAS Users" ist. Falls dies zutrifft wird der Benutzer und sein lokales Profilverzeichnis gelöscht.

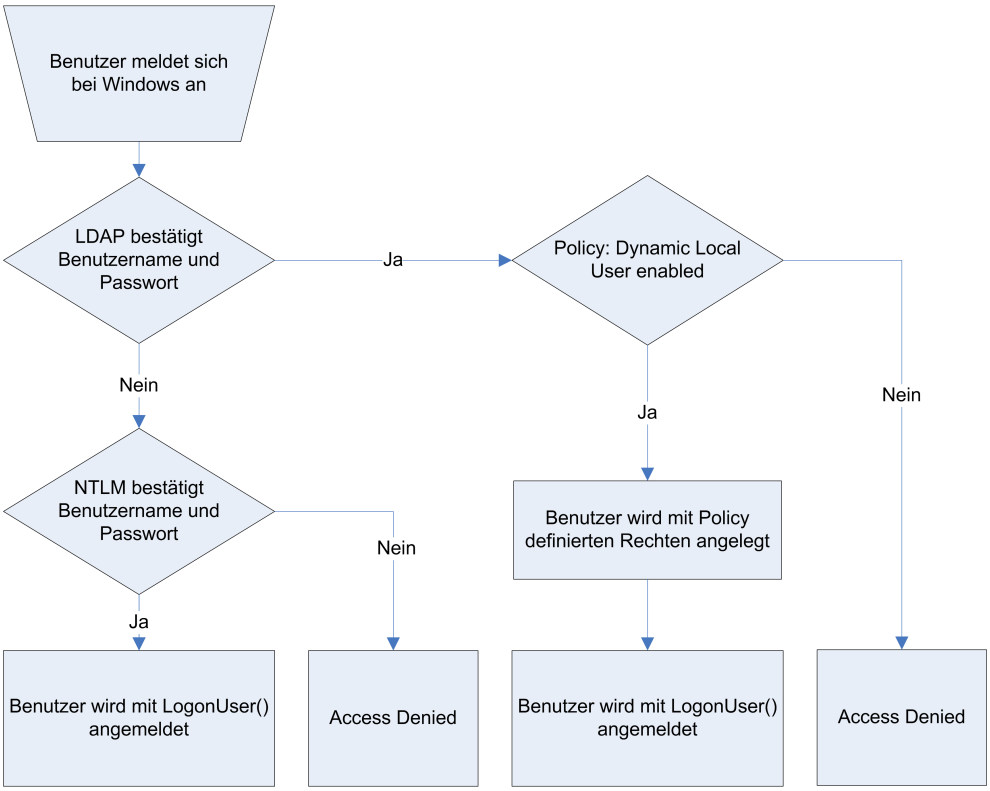


Abbildung 4.5: Ablaufdiagramm: QbeGina Benutzeranmeldung

4.7.2 Konfiguration durch Policy-Objekt

Jedes Workstation Objekt kann das Attribut `qbePolicyName` mit einem Verweis auf ein Policy-Objekt enthalten. Dort sind dann folgende Parameter konfigurierbar:

Dynamic Local User Legt fest, ob ein lokaler Benutzer angelegt wird falls dieser nicht existiert. Ohne lokalen Benutzer kann keine Anmeldung stattfinden.

Local User Gruppe Legt fest, in welche Gruppe neue Benutzer hinzugefügt werden. Dies kann ein normaler Gruppenname sein, oder einer der folgenden vordefinierten Namen, die dann in die sprachspezifischen Windows Namen konvertiert werden.

- **BUILTIN/Administrators** Die Administratorengruppe (Administators, Administratoren, ...)
- **BUILTIN/Power Users** Die Hauptbenutzergruppe (Power Users, Hauptbenutzer, ...)
- **BUILTIN/Users** Die normale Benutzergruppe (Users, Benutzer, ...)
- **BUILTIN/Guests** Die Gastgruppe (Guests, Gäste, ...)

Profile Path Der Profilpfad für den Benutzer. Ein `%s` wird durch den Benutzernamen ersetzt.

Login Script Pfad zu einem ausführbaren Loginskript. Vorzugsweise wird dafür die QbeLoginScript.exe verwendet. An den kompletten Pfadnamen wird der Benutzername und das Passwort angehängt.

4.7.3 Screenshots QbeGina

Abbildung 4.6: Screenshot: QbeGina Fenster vor der Anmeldung

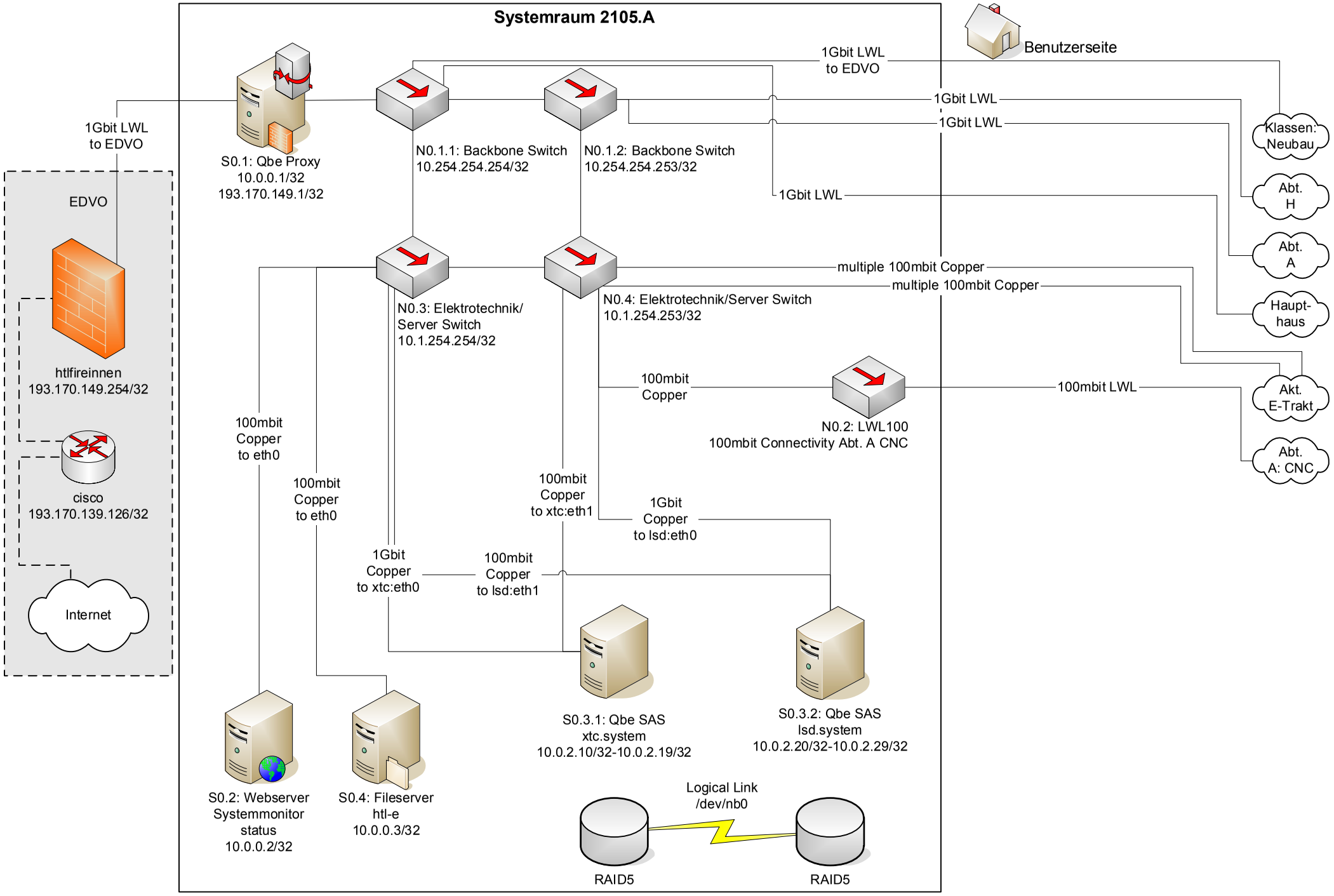
Abbildung 4.7: Screenshot: QbeGina Anmeldung wenn die LDAP Verbindung verloren geht

Abbildung 4.8: Screenshot: QbeGina Benutzeranmeldefenster

Anhang A

Netzwerkübersicht HTL

Übersicht über die Server im HTL Netzwerk sowie der Anbindung an das Internet und die Clients.


Systemraum 2105.A
Benutzerseite
1Gbit LWL
to EDVO
1Gbit LWL
1Gbit LWL
1Gbit LWL
1Gbit LWL
to EDVO
EDVO
S0.1: Qbe Proxy
10.0.0.1/32
193.170.149.1/32
N0.1.1: Backbone Switch
10.254.254.254/32
N0.1.2: Backbone Switch
10.254.254.253/32
Klassen:
Neubau
Abt.
H
Abt.
A
Haupt-
haus
multiple 100mbit Copper
multiple 100mbit Copper
N0.3: Elektrotechnik/
Server Switch
10.1.254.254/32
N0.4: Elektrotechnik/Server Switch
10.1.254.253/32
htlfireinnen
193.170.149.254/32
100mbit
Copper
100mbit LWL
Akt.
E-Trakt
N0.2: LWL100
100mbit Connectivity Abt. A CNC
Abt.
A: CNC
100mbit
Copper
to eth0
100mbit
Copper
to eth0
100mbit
Copper
to xtc:eth1
1Gbit
Copper
to lsd:eth0
cisco
193.170.139.126/32
1Gbit
Copper
to xtc:eth0
100mbit
Copper
to lsd:eth1
Internet
S0.3.1: Qbe SAS
xtc.system
10.0.2.10/32-10.0.2.19/32
S0.3.2: Qbe SAS
lsd.system
10.0.2.20/32-10.0.2.29/32
S0.2: Webserver
Systemmonitor
status
10.0.0.2/32
S0.4: Fileserver
htl-e
10.0.0.3/32
Logical Link
/dev/nb0
RAID5
RAID5

Anhang B

eDirectory Schema und Tree

Die Qbe SAS Anwendung benötigt einige neue Attribute und Objekte. Diese Schemaerweiterungen wurden mit der OID Nummer der HTL angelegt.

B.1 Neue eDirectory Attribute

inetStatus, loggedonHost, loggedonMac, traffic und lastActivity werden in den Benutzer-, Gruppen-, Klassen-, und Computer-Objekten verwendet. Die qbePolicy*-Attribute werden nur für die Computer-Policy-Objekte (qbeHostPolicy) verwendet.

```
1 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.10 NAME 'inetStatus'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
2 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.11 NAME 'loggedonHost'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{64512} SINGLE-VALUE )
3 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.12 NAME 'loggedonMac'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} SINGLE-VALUE )
4 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.13 NAME 'traffic' SYNTAX
     1.3.6.1.4.1.1466.115.121.1.36{64512} SINGLE-VALUE )
5 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.14 NAME 'lastActivity'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
6 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.15 NAME '
    qbePolicyDynamicUserGroup' SYNTAX
    1.3.6.1.4.1.1466.115.121.1.15{64512} )
7 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.16 NAME '
    qbePolicyDynamicUserEnabled' SYNTAX
    1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
8 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.17 NAME 'qbePolicyName'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )
9 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.18 NAME '
    qbePolicyLoginScript' SYNTAX
    1.3.6.1.4.1.1466.115.121.1.15{64512} SINGLE-VALUE )
```

```
10 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.19 NAME '
     qbePolicyHomeDrive' SYNTAX1.3.6.1.4.1.1466.115.121.1.15{64512}
     SINGLE-VALUE )
11 attributeTypes: ( 1.2.826.0.1.16224.0.0.0.20 NAME '
     qbePolicyHomeDriveDir' SYNTAX
     1.3.6.1.4.1.1466.115.121.1.15{64512} SINGLE-VALUE )
```

Bei einer Neuinstallation sollten die Attribute inetStatus, loggedonHost, loggedonMac, traffic und lastActivity auf qbe* umbenannt werden. Dies würde die Schemaverwaltung vereinfachen, zieht jedoch auch Änderungen in den PHP Skripten nach sich.

B.2 Neue eDirectory Objekte

qbeOrganizationalUnit und qbeGroup werden für die Klassen-/Gruppen-Verwaltung verwendet. qbeIpDevice, qbeOwnedObject und qbeHostPolicy für die ComputerVerwaltung.

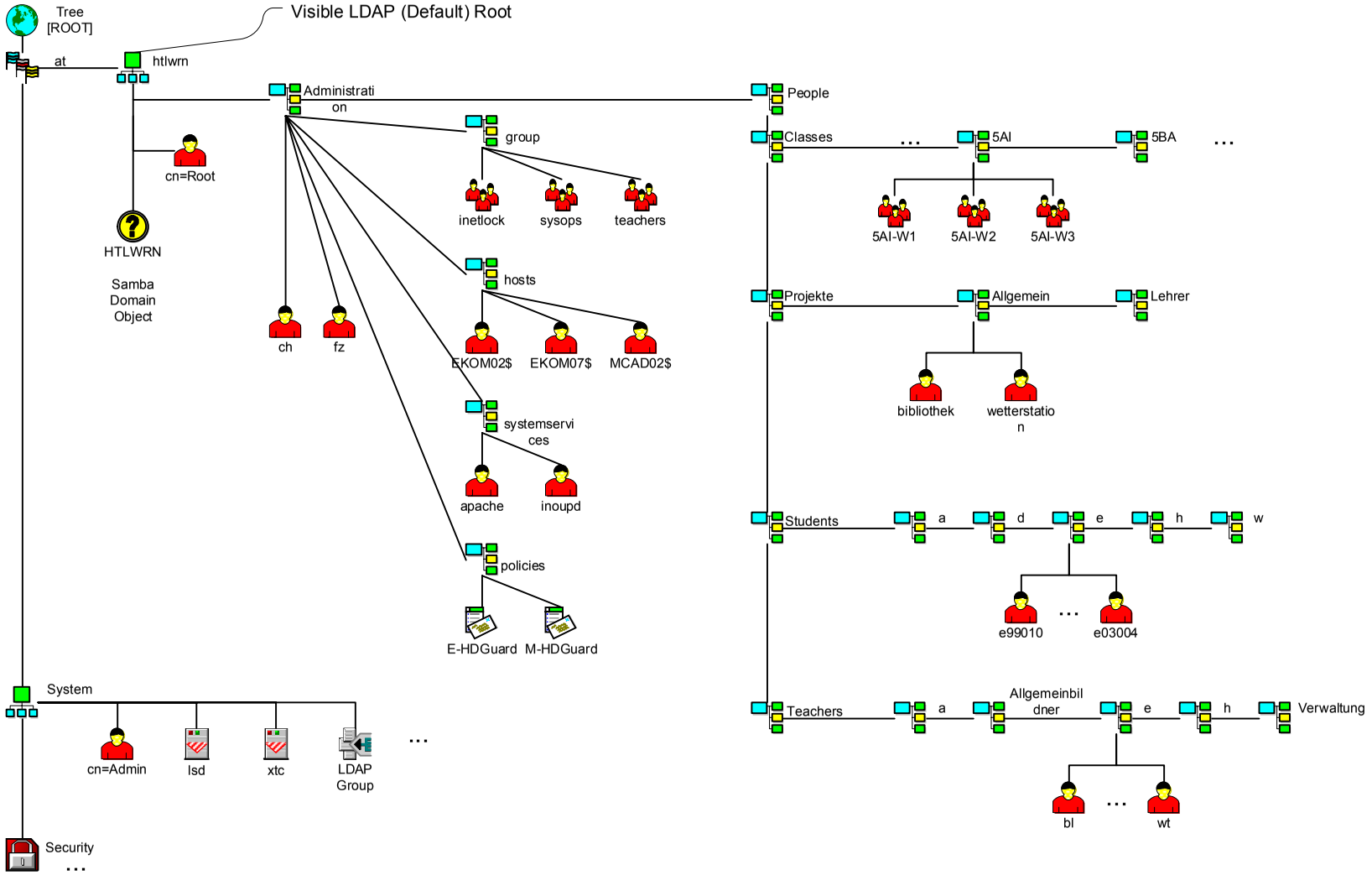
```
1 objectClasses: ( 1.2.826.0.1.16224.0.0.0.1 NAME '
     qbeOrganizationalUnit' SUP organizationalUnit AUXILIARY MAY (
     cn $ inetStatus ) X-NDS_NOT_CONTAINER '1' )
2 objectClasses: ( 1.2.826.0.1.16224.0.0.0.2 NAME 'qbeGroup' SUP
     groupOfNames AUXILIARY MAY inetStatus X-NDS_NOT_CONTAINER '1' )
3
4 objectClasses: ( 1.2.826.0.1.16224.0.0.0.3 NAME 'qbeIpDevice'
     AUXILIARY MAY ( macAddress $ ipHostNumber $ qbePolicyName ) X-
     NDS_NOT_CONTAINER '1' )
5 objectClasses: ( 1.2.826.0.1.16224.0.0.0.4 NAME 'qbeOwnedObject'
     AUXILIARY MAY owner X-NDS_NOT_CONTAINER '1' )
6 objectClasses: ( 1.2.826.0.1.16224.0.0.0.5 NAME 'qbeHostPolicy'
     AUXILIARY MUST qbePolicyDynamicUserEnabled MAY (
     qbePolicyDynamicUserGroup $ qbePolicyHomeDrive $
     qbePolicyHomeDriveDir $ qbePolicyLoginScript ) X-
     NDS_NOT_CONTAINER '1' )
```

B.3 eDirectory Tree Übersicht

Im folgenden findet sich eine Übersicht des eDirectory Verzeichnisses. Hier mit Treename "HTL" und Basiskontext "o=htlwrn,c=at".

Der Kontext "o=System" ist nicht via LDAP sichtbar, kann daher auch nicht einfach von Applikationen verändert/beschädigt werden.

NDS Tree: HTL


Tree
[ROOT]
Visible LDAP (Default) Root
at
htlwrn
Administrati
on
People
group
Classes
5AI
5BA
cn=Root
inetlock
sysops
teachers
5AI-W1
5AI-W2
5AI-W3
HTLWRN
Samba
Domain
Object
hosts
Projekte
Allgemein
Lehrer
ch
fz
EKOM02$
EKOM07$
MCAD02$
bibliothek
wetterstatio
n
systemservi
ces
apache
inoupd
Students
a
d
e
h
w
policies
e99010
e03004
E-HDGuard
M-HDGuard
System
Teachers
Allgemeinbil
dner
Verwaltung
cn=Admin
lsd
xtc
LDAP
Group
bl
wt
Security

Anhang C

Cluster-Installation HTL

Der HTL-Cluster für den Authentication Server besteht aus zwei identischen Intel-Servern. Der Cluster-Name lautet auf qbe-auth.htlwrn.ac.at. Die Intel-Server wurden xtc.system.htlwrn.ac.at (Primärer Server) und lsd.system.htlwrn.ac.at (Sekundärer Server) getauft.

Als Cluster Monitor-Software wird heartbeat eingesetzt. Die Intel-Server sind vom Modell SR2300 und mit dem Intel SE7501WV2 sowie dem Intel Raid-Controller SRCZCR ausgestattet. Je Server sind 5 Seagate 73GB SCSI Hotswap Disks eingebaut und werden als RAID 5 mit Hotspare verwendet.

Novell eDirectory wird nicht vom heartbeat verwaltet, es läuft immer auf beiden Cluster Nodes. Alle anderen Dienste werden vom heartbeat gestartet/gestoppt.

Um das Daten-Volume `/raid` auf beiden Cluster-Nodes konsistent zu halten, wird ein Netzwerk-RAID1 mit drbd aufgebaut. Damit wird nicht direkt die Partition `/dev/sda7` sondern das drbd Device `/dev/nb0` gemountet. Als Filesystem wird journaling ext3 eingesetzt.

Partitionen:

```
1 Device    Boot   Id   Type
2 /dev/sda1        12   System Diagnostics
3 /dev/sda2   *    83   Linux          /boot (ext2)
4 /dev/sda3         5   Extended
5 /dev/sda5        83   Linux          / (ext3)
6 /dev/sda6        82   Linux swap
7 /dev/sda7        83   Linux          /raid via drbd (ext3)
```

Konfiguration Netzwerk-RAID1 /etc/drbd.conf:

```
1 resource drbd0 {
2   protocol=A            # Protokolltyp A
3   fsckcmd=fsck.ext3 -p -y   # ext3 fsck verwenden
4   disk {
```

```
5           disk-size=132255081 # Partition Objektgroesse in Byte
6     }
7     net {
8       sync-rate=100000          # Maximale
            Synchronisationsgeschwindigkeit
9     }
10    on xtc {
11      device=/dev/nb0
12      disk=/dev/sda7       # Partitiondevice
13      address=192.168.177.1    # IP Adresse des ersten Clusternode
14      port=7788
15    }
16    on lsd {
17      device=/dev/nb0
18      disk=/dev/sda7       # Partitiondevice
19      address=192.168.177.2    # IP Adresse der zweiten Clusternode
20      port=7788
21    }
22  }
```

Die Cluster-Applikationen und IP-Adressen:

Filesystem::/dev/nb0::/raid::ext3::noatime,usrquota,grpquota Das /raid Datendateisystem

qbe-sas-daemon Hintergrunddienste vom Application Server

10.0.2.100 Application Server IP-Adresse

172.16.0.1/16 Application Server IP für unbekannte Clients

apache::/etc/apache/httpd.conf Application Server Webserver

samba CIFS Server

mysql SQL Server

vsftpd FTP Server

nfs-kernel-server Network File System (NFS) Server für die www

dhcp3-server DHCP Server

10.0.2.104 SubVersion IP-Adresse

apache2 SubVersion Server

Index