LDAP-UX Client Services B.04.10 Administrator's Guide

HP-UX 11i v1, v2 and v3

HP Part Number: J4269-90073
Published: E0407
Edition: Edition 7

Table of Contents

List of Figures

List of Tables

Preface: About This Document

The latest version of this document can be found on line at:

http://www.docs.hp.com

This document describes how to install and configure LDAP-UX Client Services product on HP-UX platforms.

The document printing date and part number indicate the document's current edition. The printing date will change when a new edition is printed. Minor changes may be made at reprint without changing the printing date. The document part number will change when extensive changes are made.

Document updates may be issued between editions to correct errors or document product changes. To ensure that you receive the updated or new editions, you should subscribe to the appropriate product support service. See your HP sales representative for details.

Intended Audience

This document is intended for system and network administrators responsible for installing, configuring, and managing the LDAP-UX Client Services. Administrators are expected to have knowledge of the LDAP-UX Client Services Integration product.

New and Changed Documentation in This Edition

This edition documents the following new information for the LDAP-UX Client Services version B.04.10:

- Support dynamic groups. This feature provides a reference to a dynamically managed group based on the user's status in an organization. A user can be added to or removed from a group dynamically based on his/her most current status.
- Enhance PAM_Authz to provide LDAP account and password security policy enforcement without requiring LDAP-based authentication. This feature supports applications which have already performed authentication, such as secure shell (SSH) or the r-commands.
- Enhance PAM_Authz to provide meaningful error messages. For example, if the pam_authz policy rule indicates that an account has been locked out or a password has expired, pam_authz can return an appropriate PAM error code instead of a general deny error code.
- Support an extension operation of TLS (Transport Layer Security) protocol called startTLS to secure communication between LDAP clients and the Netscape/Red Hat Directory Server.

Publishing History

Table 1 Publishing History Details

Document Manufacturing Part Number	Operating Systems Supported	Supported Product Versions	Publication Date
J4269-90016	11.0, 11i	B.03.00	September 2002
J4269-90030	11.0, 11i v1 and v2	B.03.20	October 2003
J4269-90038	11.0, 11i v1	B.03.30	July 2004
J4269-90040	11.0, 11i v1 and v2	B.03.30	September 2004
J4269-90048	11i v1 and v2	B.04.00	July 2005
J4269–90051	11i v1 and v2	B.04.00	August 2005
J4269-90053	11i v1 and v2	B.04.00	June 2006

Table 1 Publishing History Details *(continued)*

Document Manufacturing Part Number	Operating Systems Supported	Supported Product Versions	Publication Date
J4269-90063	11i v1 and v2	B.04.10	December 2006
J4269-90073	11i v1, v2 and v3	B.04.10	April 2007

What's in This document

This manual describes how to install, configure and administer the LDAP-UX Client Services software product.

The manual is organized as follows:

Chapter 1 **Introduction** Use this chapter to learn the LDAP-UX Client Services product features, components and client administration tools.

Chapter 2 **Installing And Configuring LDAP-UX Client Services** Use this chapter to learn how to install, configure, and use the LDAP-UX Client Services software.

Chapter 3 **LDAP Printer Configurator Support** Use this chapter to learn how to set up, configure, and use the printer configurator.

Chapter 4 **Dynamic Group Support** Use this chapter to learn how to set up, configure and use dynamic groups.

Chapter 5 **Administering LDAP-UX Client Services** Use this chapter to understand how to administer your LDAP-UX Clients to keep them running smoothly and expand them as your computing environment expands.

Chapter 6 **Command and Tool Reference** Use this chapter to learn about the commands and tools associated with the LDAP-UX Client Services product.

Chapter 7 **User Tasks** Use this chapter to learn how to change passwords and personal information.

Chapter 8 **Mozilla LDAP C SDK** Use this chapter to learn the Mozilla LDAP SDK software features and its major file components.

Typographical Conventions

This document uses the following conventions.

Book Title The title of a book. On the web and on the Instant Information CD, it may be a hot link to the book itself.

Emphasis Text that is emphasized.

Bold Text that is strongly emphasized.

Bold The defined use of an important word or phrase.

`ComputerOut` Text displayed by the computer.

`UserInput` Commands and other text that you type.

`Command` A command name or qualified command phrase.

`Variable` The name of a variable that you may replace in a command or function or information in a display that represents several possible values.

[ ] The contents are optional in formats and command descriptions. If the contents are a list separated by |, you must choose one of the items.

{ } The contents are required in formats and command descriptions. If the contents are a list separated by |, you must choose one of the items.

\ The continuous line symbol.

HP Encourages Your Comments

HP encourages your comments concerning this document. We are truly committed to providing documentation that meets your needs.

Please send comments to: netinfo_feedback@cup.hp.com

Please include document title, manufacturing part number, and any comment, error found, or suggestion for improvement you have concerning this document. Also, please include what we did right so we can incorporate it into other documents.

1 Introduction

LDAP-UX Client Services simplifies HP-UX system administration by consolidating account and configuration information into a central LDAP directory. This LDAP directory could reside on an HP-UX system such as Netscape Directory Server 6.x, Red Hat Directory Server 7.x or the account information could be integrated in Windows 2000/2003 Active Directory.

Information provided in this manual outlines the installation and administration tasks of LDAP-UX Client Services with HP-UX based LDAP directories such as Netscape Directory Server 6.x.

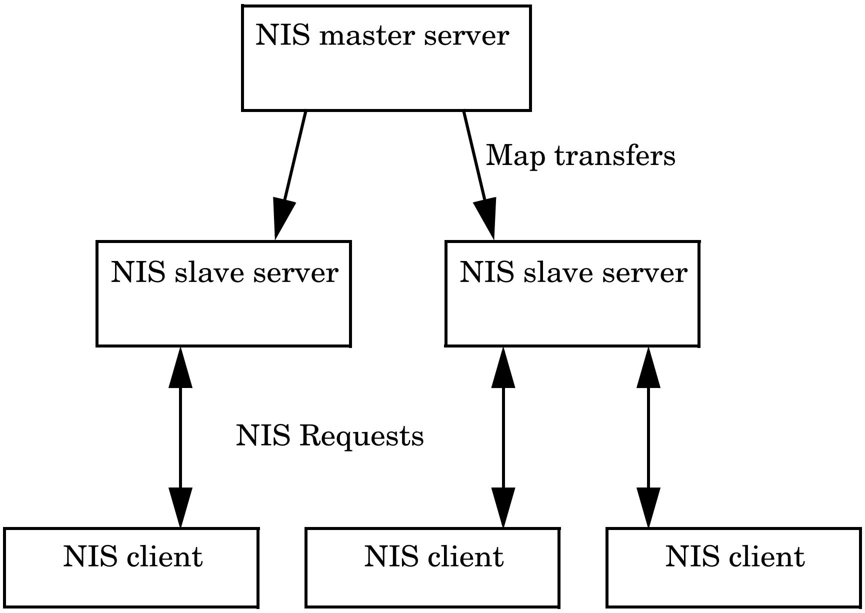
For information on the integration of LDAP-UX Client Services with Windows 2000/2003/2003 R2 Active Directory, see *LDAP-UX with Microsoft Windows Active Directory Administrator's Guide (J4269-90064)* at http://docs.hp.com/hpux/internet.

This chapter introduces LDAP-UX Client Services and briefly describes how it works.

Overview of LDAP-UX Client Services

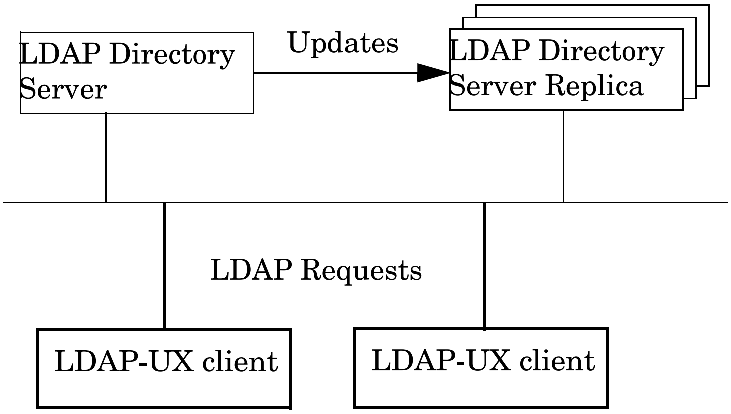
Traditionally, HP-UX account and configuration information is stored in text files, for example, /etc/passwd and /etc/group. NIS was developed to ease system administration by sharing this information across systems on the network. With NIS, account and configuration information resides on NIS servers. NIS client systems retrieve this shared configuration information across the network from NIS servers, as shown below:

Figure 1-1 A Simplified NIS Environment



LDAP-UX Client Services improves on this configuration information sharing. HP-UX account and configuration information is stored in an LDAP directory, not on the local client system. Client systems retrieve this shared configuration information across the network from the LDAP directory, as shown below. LDAP adds greater scalability, interoperability with other applications and platforms, and less network traffic from replica updates.

Figure 1-2 A Simplified LDAP-UX Client Services Environment



LDAP-UX Client Services supports the following name service data: passwd, groups, hosts, rpc, services, networks, protocols, publickeys, automount, netgroup. See the *LDAP-UX Integration B.04.10 Release Notes* for any additional supported services.

How LDAP-UX Client Services Works

LDAP-UX Client Services works by leveraging the authentication mechanism provided in the Pluggable Authentication Module, or PAM, and the naming services provided by the Name Service Switch, or NSS. See *pam*(3), *pam.conf*(4), and *Managing Systems and Workgroups* at http://docs.hp.com/hpux/os for information on PAM. For information on NSS, see *switch*(4) and "Configuring the Name Service Switch" in *Installing and Administering NFS Services* at http://docs.hp.com/hpux/communications/#NFS.

These extensible mechanisms allow new authentication methods and new name services to be installed and used without changing the underlying HP-UX commands. And, by supporting the PAM architecture, the HP-UX client becomes truly integrated in the LDAP environment. The PAM_LDAP library allows the HP-UX system to use the LDAP directory as a trusted server for authentication. This means that passwords may not only be stored in any syntax but also means that passwords may remain hidden from view (preventing a decryption attack on the hashed passwords). Because passwords may be stored in any syntax, HP-UX will be able to share passwords with other LDAP-enabled applications.

With LDAP-UX Client Services B.03.20 or later versions, the client daemon, `ldapclientd`, becomes the center of the product. It supports all NSS backend services for LDAP and data enumeration. It also supports PAM_LDAP for authentication and password change.

With LDAP-UX Client Services, HP-UX commands and subsystems can transparently access name service information from the LDAP directory through `ldapclientd`. The following table shows some examples of commands and subsystems that use PAM and NSS:

Table 1-1 Examples of Commands and Subsystems that use PAM and NSS

Commands that use NSS	Commands that use PAM and NSS
ls	login
nsquery[1]	passwd
who	ftp
whoami	su
finger[2]	rlogin
id	telnet
logname	dtlogin

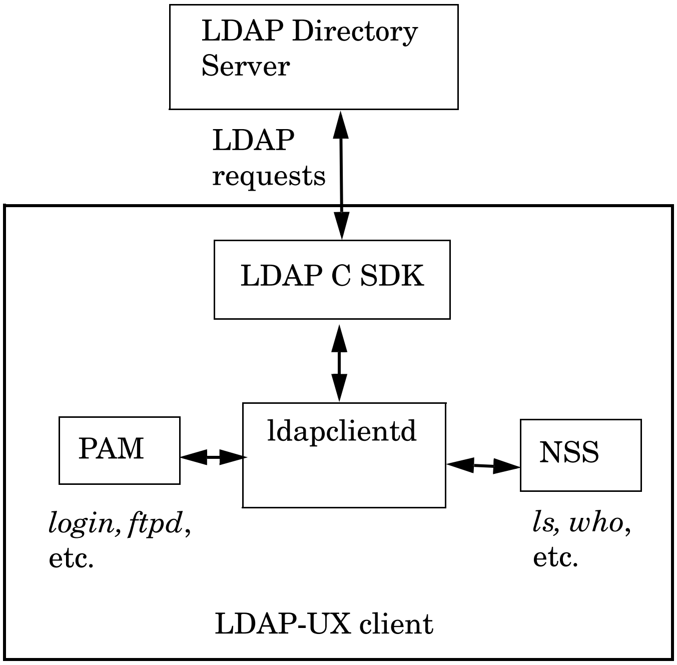
Table 1-1 Examples of Commands and Subsystems that use PAM and NSS *(continued)*

Commands that use NSS	Commands that use PAM and NSS
groups[2]	remsh
newgrp[2]	
pwget[2]	
grget[2]	
listusers[2]	
logins[2]	
nslookup	

[1] *nsquery*(1) is a contributed tool included with the ONC/NFS product.

[2] These commands enumerate the entire passwd or group database, which may reduce network and directory server performance for large databases.

Figure 1-3 A Simplified LDAP-UX Client Services Environment



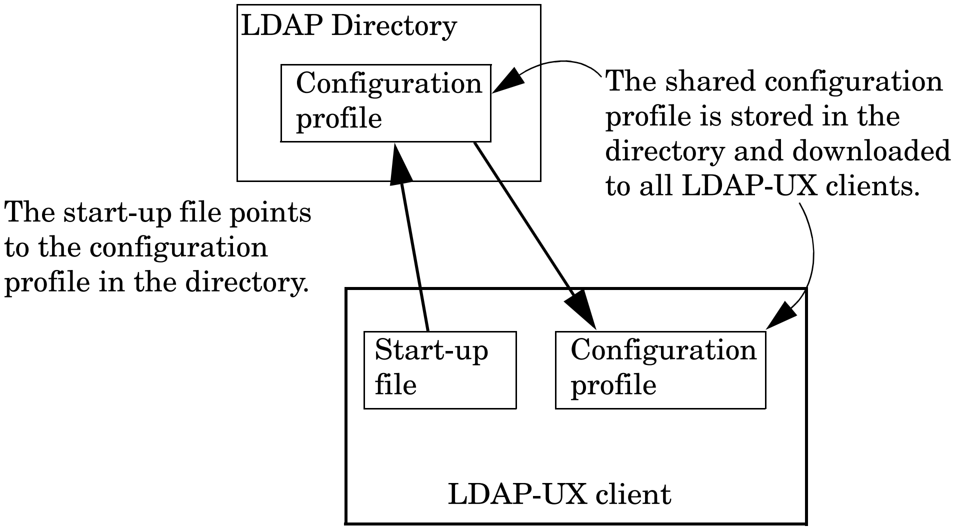
In addition, the *getpwent*(3C) and *getgrent*(3C) family of system calls get user and group information from the directory.

After you install and configure an LDAP directory and migrate your name service data into it, HP-UX client systems locate the directory from a "start-up file." The start-up file tells the client system how to download a "configuration profile" from the LDAP directory. The configuration profile is a directory entry containing configuration information common to many clients. Storing it in the directory lets you maintain it in one place and share it among many clients rather than storing it redundantly across the clients. Because the configuration information is stored in the directory, all each client needs to know is where its profile is, hence the start-up file. Each client downloads the configuration profile from the directory.

The profile is an entry in the directory containing details on how clients are to access the directory, such as:

- where and how clients should search the directory for user, group and other name service information.
- how clients should bind to the directory: anonymously or as a proxy user. Anonymous access is simplest. Configuring a proxy user adds some security, but at the same time it adds the overhead of managing the proxy user.
- other configuration parameters such as search time limits.

Figure 1-4 The Local Start-up File and the Configuration Profile


LDAP Directory
Configuration profile
The shared configuration profile is stored in the directory and downloaded to all LDAP-UX clients.
The start-up file points to the configuration profile in the directory.
Start-up file
Configuration profile
LDAP-UX client

The following chapter describes in detail how to install, configure, and verify LDAP-UX Client Services.

2 Installing And Configuring LDAP-UX Client Services

This chapter describes the decisions you need to make and the steps to install Netscape/Red Hat Directory Server and configure LDAP-UX Client Services. This chapter contains the following sections:

Before You Begin

This section lists some things to keep in mind as you plan your installation.

- Use the configuration worksheet to record your decisions and other information you'll need later for configuration in Configuration Worksheet (page 183).
- See the *LDAP-UX Integration B.04.10 Release Notes* (J4269-90065) at http://docs.hp.com/hpux/internet for last-minute information.
- You must have an LDAP directory. You can obtain the Directory Server for HP-UX version 6.x or 7.x from your local HP sales office or www.hp.com and view the documentation at http://docs.hp.com/en/internet.html.
- See the white paper *Preparing Your Directory for HP-UX Integration* at http://docs.hp.com/hpux/internet for advice on how to set up and configure your directory to work with HP-UX.
- Most examples here use the Netscape Directory Server for HP-UX version 6.x and assume you have some knowledge of this directory and its tools, such as the Directory Console and ldapsearch. If you have another directory, consult your directory's documentation for specific information.
- For details on how to integrate LDAP-UX Client Services with Windows 2000/2003/2003 R2 Active Directory, please refer to *LDAP-UX Client Services with Microsoft Windows Active Directory Administrator's Guide (J4269-90064)* at http://docs.hp.com/hpux/internet/#LDAP-UX%20Integration.
- The examples use a base DN of o=hp.com for illustrative purposes.

Summary of Installing and Configuring

The following summarizes the steps you take when installing and configuring an LDAP-UX Client Services environment.

- See Plan Your Installation (page 23).
- Install LDAP-UX Client Services on each client system. See Install LDAP-UX Client Services on a Client (page 28).
- Install and configure an LDAP directory, if not already done. See Configure Your Directory (page 29).
- Configure your LDAP server to support SSL or TLS if you attempt to enable SSL or TLS support with LDAP-UX. See "Configure the LDAP-UX Client Serivces with SSL or TLS Support" (page 45).
- Migrate your name service data to the directory. See Import Name Service Data into Your Directory (page 32).
- Install and set up the security database files on the LDAP-UX client system if you want to enable SSL support with LDAP-UX. See Configure the LDAP-UX Client Serivces with SSL or TLS Support (page 45).
- Run the setup program to configure LDAP-UX Client Services on a client system. Setup does the following for you:
  - — Extends your Netscape/Red Hat directory schema with the configuration profile schema, if not already done.
  - — Imports the LP printer schema into your LDAP directory server if you choose to start the LDAP printer configurator.
  - — Imports the publickey schema into your LDAP directory if you choose to store the public keys of users and hosts in the LDAP directory.
  - — Imports the automount schema into your LDAP directory server if you choose to store the AutoFS maps in the LDAP directory.
  - — Creates a start-up file on the client. This enables each client to download the configuration profile.
  - — Creates a configuration profile of directory access information in the directory, to be shared by a group of (or possibly all) clients.
  - — Downloads the configuration profile from the directory to the client.
  - — Start the product daemon, `ldapclientd`, if you choose to start it. Starting with LDAP-UX Client B.03.20 or later, the client daemon must be started for LDAP-UX functions to work. With LDAP-UX Client B.03.10 or earlier, running the client daemon is optional.

  See Configure the LDAP-UX Client Services (page 33).
- Modify the files `/etc/pam.conf` and `/etc/nsswitch.conf` on the client to specify LDAP authentication and name service, respectively. See Configure the LDAP-UX Client Services (page 33).
- Optionally modify the `disable_uid_range` flag in the `/etc/opt/ldapux/ldapux_client.conf` file to disable logins to the local system from specific ldap users.
- Optionally modify the `/etc/opt/ldapux/pam_authz.policy` and `/etc/pam.conf` files to verify the user access rights of a subset of users in a large repository needing access, if appropriate. See the pam_authz(5) man page for the command syntax.
- Verify each client is working properly. See Verify the LDAP-UX Client Services (page 65).
- See also Configure Subsequent Client Systems (page 68) for some shortcuts.

Plan Your Installation

Before beginning your installation, you should plan how you will set up and verify your LDAP directory and your LDAP-UX Client Services environment before putting them into production. Consider the following questions. Record your decisions and other information you'll need later in Configuration Worksheet (page 183).

- How many LDAP directory servers and replicas will you need?

  Each client system binds to an LDAP directory server containing your user, group, and other data. Multiple clients can bind to a single directory server or replica server. The answer depends on your environment, the size and configuration of your directory and how many users and clients you have.Write your directory server host and TCP port number in Configuration Worksheet (page 183). See the white paper *Preparing Your Directory for HP-UX Integration* at: http://docs.hp.com/hpux/internet for more information.

  See the *Netscape Directory Server Deployment Guide* for more information. You can add directory replicas to an existing LDAP-UX Client Services environment as described under Adding a Directory Replica (page 116). You may also want to review the LDAP-UX performance white paper at http://docs.hp.com/hpux/internet.

- Where will you get your name service data from when migrating it to the directory?

  You can get it from your files in the /etc directory or, if you are using NIS, from the same source files you create your NIS maps from, or you can get it from your NIS maps themselves. Write this information in Configuration Worksheet (page 183).

  See Import Name Service Data into Your Directory (page 32) for how to import your information into the directory and Name Service Migration Scripts (page 170) for details on the migration scripts.

  To add an individual user entry or modify an existing user entry in your directory, you can use the ldapmodify command or other directory administration tools such as the Netscape/Red Hat Directory Console. See also the *LDAP-UX Integration B.04.10 Release Notes* for additional contributed tools.

NOTE: You should keep a small subset of users in /etc/passwd, particularly the root login . This allows administrative users to log in during installation and testing. Also, if the directory is unavailable you can still log in to the system.

- Where in your directory will you put your name service data?

  Your directory architect needs to decide where in your directory to place your name service information. LDAP-UX Client Services by default expects user and group data to use the object classes and attributes specified by RFC 2307. The migration scripts by default create and populate a new subtree that conforms to RFC 2307. Example Directory Structure (page 25) shows a base DN of `ou=unix,o=hp.com`. Write the base DN of your name service data in Configuration Worksheet (page 183).

  If you prefer to merge your name service data into an existing directory structure, you can map the standard RFC 2307 attributes to alternate attributes. See LDAP-UX Client Services Object Classes (page 185) for more information.

- How will you put your user, group, and other data into your directory?

  LDAP supports group membership defined in the X.500 syntax (using the member or uniquemember attribute), while still supporting the RFC 2307 syntax (using the memberuid attribute). This new group membership syntax increases LDAP-UX integration with LDAP and other LDAP-based applications, and may reduce administration overhead eliminating the need to manage the memberuid attribute. In addition, a new performance improvement has been made through the addition of a new caching daemon which caches passwd, group and X.500 group membership information retrieved from an LDAP server. This significantly reduces LDAP-UX's response time to applications. In addition, the daemon re-uses connections for LDAP queries and maintains multiple connections to an LDAP server to improve performance.

The migration scripts provided with LDAP-UX Client Services can build and populate a new directory subtree for your user and group data.

If you merge your data into an existing directory, for example to share user names and passwords with other applications, the migration scripts can create LDIF files of your user data, but you will have to write your own scripts or use other tools to merge the data into your directory. You can add the posixAccount object class to your users already in the directory to leverage your existing directory data.

See Import Name Service Data into Your Directory (page 32) for how to import your information into the directory and Name Service Migration Scripts (page 170) for details on the migration scripts.

⚠ **CAUTION:** If you place a root login in the LDAP directory, that user and password will be able to log in as root to any client using LDAP-UX Client Services. Keeping the root user in /etc/passwd on each client system allows the root user to be managed locally. This can be especially useful if the network is down because it allows local access to the system.

It is not recommended that you put the same users both in /etc/passwd and in the directory. This could lead to conflicts and unexpected behavior.

- How many profiles do you need?

  A configuration profile is a directory entry that contains configuration information shared by a group of clients. The profile contains the information clients need to access user and group data in the directory, for example:

  — Your directory server hosts
  — Where user, group, and other information is in the directory
  — The method clients use to bind to the directory
  — Other configuration parameters such as search time limits

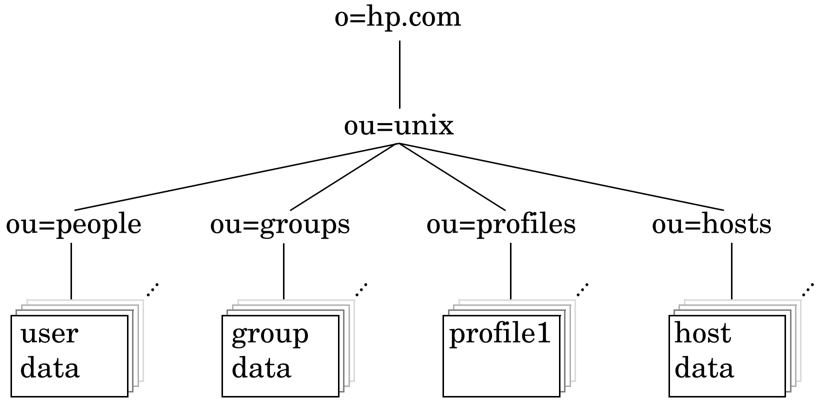
  If these parameters are the same for all your clients, you would need only one profile. You will need at least one profile per directory server or replica. In general, it is a good idea to have as few profiles as necessary to simplify maintenance. Look at the posixNamingProfile object class in LDAP-UX Client Services Object Classes (page 185) to see what is in a profile to decide how many different profiles you need.

  If you are familiar with NIS, one example is to create a separate profile for each NIS domain.

- Where in your directory will you put your profile?

  The profile contains directory access information. It specifies how and where clients can find user and group data in the directory. You can put the profile anywhere you want as long as the client systems can read it. For example, you might put it near your user data, or in a separate administrative area. You should put the profile in the same directory as your user and group data to simplify access permissions. Clients must have access to both the profile and the user and group data. The following example shows a configuration profile DN of `cn=profile1,ou=profiles,ou=devices,ou=unix,o=hp.com`.

Figure 2-1 Example Directory Structure



Write your configuration profile DN on the worksheet in Configuration Worksheet (page 183).

- By what method will client systems bind to the directory?

  Clients can bind to the directory anonymously. This is the default and is simplest to administer. If you need to prevent access to your data from anonymous users or your directory does not support anonymous access, you can use a proxy user. If you configure a proxy user, you can also configure anonymous access to be attempted in the event the proxy user fails.

  Write your client access method and proxy user DN, if needed, on the worksheet in Configuration Worksheet (page 183).

- How will you increase the security level of the product to prevent an unwanted user from logging in to the system via LDAP? What is the procedure to set up increased login security?

  The default is to allow all users stored in the LDAP directory to login. To disallow specific users to login to a local system, you will have to configure the disable_uid_range flag in /etc/opt/ldapux/ldapux_client.conf file. There are two sections in this file, the [profile] section and the [NSS] section. HP recommends that you do not edit the [profile] section. The [NSS] section contains the disable_uid_range flag along with two logging flags. For example, the flag might look like this: disable_uid_range=0-100, 300-450, 89.

  Another common example would be to disable root access This flag would look like this: disable_uid_range=0.

  When the disable_uid_range is turned on, the disabled uid will not be displayed when you run commands such as pwget, listusers, logins, etc.

NOTE: The passwd command may still allow you to change a password for a disabled user when alternative authentication methods, such as PAM Kerberos, are used since LDAP does not control these subsystems.

- What PAM authentication will you use? How will you set up /etc/pam.conf? What other authentication do you want to use & in what order?

  PAM is the Pluggable Authentication Module, providing authentication services. You can configure PAM to use ldap, Kerberos, or other traditional UNIX locations (for example files, NIS, NIS+) as controlled by NSS. See *pam*(3), *pam.conf*(4), and *Managing Systems and Workgroups* at http://docs.hp.com/hpux/os for more information on PAM.

  It is recommended you use HP-UX file-based authentication first, followed by LDAP or other authentication. /etc/pam.ldap is an example of this configuration. With this configuration, PAM uses traditional authentication first, searching /etc/passwd when any user logs in, then attempts to authenticate to the directory if the user is not in /etc/passwd. If you have a few users in /etc/passwd, in particular the root user, and if the directory is unavailable, you can still log in to the client as a user in /etc/passwd.

- Do you want to use TLS (Transport Layer Security) or SSL for secure communication between clients and Netscape/Red Hat Directory servers?

  LDAP-UX supports SSL or TLS with password as the credential, using either simple bind or DIGEST-MD5 authentication (DIGEST-MD5 is available for Netscape/Red Hat Directory Server only) to ensure confidentiality and data integrity between clients and servers. startTLS is a new extension operation of TLS protocol. You can utilize the StartTLS operation to set the TLS secure connection over a regular (an un-encrypted) LDAP port. The secure connection can also be established on an encrypted LDAP port when using SSL. By default, SSL and TLS are disabled. For detailed information, refer to "Configure the LDAP-UX Client Serivces with SSL or TLS Support" (page 45).

- What authentication method will you use when you choose to enable TLS?

  You have a choice between SIMPLE (the default), or SASL DIGEST-MD5 with TLS.

- What authentication method will you use when you choose to enable SSL?

  You have a choice between SIMPLE (the default), or SASL DIGEST-MD5 with SSL.

- What authentication method will you use when you choose to not enable SSL and TLS?

  You have a choice between SIMPLE (the default), or SASL DIGEST-MD5. SASL DIGEST-MD5 improves security, preventing snooping over the network during authentication.

  Using the DIGEST-MD5 authentication, the password must be stored in the clear text in the LDAP directory. Using the DIGEST-MD5 authentication requires the proxy credential level.

- Do you want to import the LDAP printer schema if you choose to start the printer configurator?

  LDAP-UX Client Services B.03.20 or later provides the integration with the LDAP printer configurator to simplify the LP printer management by updating LP printer configuration automatically on your client system. A new printer schema, which is based on *IETF<draft-fleming-ldap-printer-schema-02>*, is required to start the services.

IMPORTANT: If you attempt to use this new feature, in the `ldapclientd.conf` file, the `start` configuration parameter of the printer services section must be set to `yes`. If the `start` option is enabled, the printer configurator will start when `ldapclientd` is initialized. By default, the `start` parameter is enabled.

- Do you want to import the publickey schema into your LDAP directory if you choose to store and manage publickeys in the LDAP directory.

  LDAP-UX Client Services B.04.00 supports discovery and management of publickeys in an LDAP directory. Both public and private (secret) keys, used by the SecureRPC API can be stored in user and host entries in an LDAP directory server, using the `nisKeyObject` objectclass.

- Do you want to import the automount schema into your LDAP directory server if you choose to store and manage automount maps in the LDAP directory?

  LDAP-UX Client Services B.04.00 supports the automount service under the AutoFS subsystem. This new feature allows you to store or retrieve automount maps in/from an LDAP directory. LDAP-UX Client Services supports the new automount schema based on RFC2307-bis. The `nisObject` automount schema can also be used if configured via attribute mappings.

  The setup program will import the new automount schema into your Directory Server. An obsolete automount schema is shipped with the Netscape Directory Server version 6.x. You must manually delete the obsolete automount schema before the setup program can successfully import the new automount schema into the LDAP directory.

For the detailed information about AutoFS with LDAP support, see AutoFS Support (page 56).

- What name services will you use? How will you set up /etc/nsswitch.conf? What order do you want NSS to try services?

  NSS is the Name Service Switch, providing naming services for user names, group names, and other information. You can configure NSS to use files, ldap, or NIS in any order and with different parameters. See /etc/nsswitch.ldap for an example nsswitch.conf file using files and ldap. See *switch*(4) and "Configuring the Name Service Switch" in *Installing and Administering NFS Services* at http://docs.hp.com for more information.

  It is recommended you use files first, followed by LDAP for passwd, group and other supported name services. With this configuration, NSS will first check files, then check the directory if the name service data is not in the respective files. /etc/nsswitch.ldap is an example of this configuration.

- Do you need to configure login authorization for a subset of users from a large repository such as an LDAP directory? How will you set up the `/etc/opt/ldapux/pam_authz.policy` and `/etc/pam.conf` files to implement this feature?

  The pam_authz service module for PAM provides functionality that allows the administrator to control who can login to the system. These modules are located at /usr/lib/security/libpam_authz.1 on the HP 9000 machine and at `libpam_authz.so.1` on the Integrity (ia64) machine. pam_authz has been created to provide access control similar to the netgroup filtering feature that is performed by NIS. These modules are located at /usr/lib/security/libpam_authz.1 on the HP 9000 machine (libpam_authz.so.1 on the Integrity (ia64) machine). Starting with LDAP-UX Client Services B.04.00, pam_authz has been enhanced to allow system administrators to configure and customize their local access rules in a local policy file, `/etc/opt/ldapux/pam_authz.policy.` pam_authz uses these access control rules defined in the local policy file to control the login authorization. pam_authz is intended to be used when NIS is not used, such as when the pam_ldap or pam_kerberos authentication modules are used. Because pam_authz doesn't provide authentication, it doesn't verify if a user account exists.

  If the `/etc/opt/ldapux/pam_authz.policy` file does not exist in the system, pam_authz provides access control based on the netgroup information found in the `/etc/passwd` and `/etc/netgroup` files. If the `/etc/opt/ldapux/pam_authz.policy` file exists in the system, pam_authz uses the access rules defined in the policy file to determine who can login to the system.

  For detailed information on this feature and how to configure the `/etc/opt/ldapux/pam_authz.policy` file, see PAM_AUTHZ Login Authorization (page 98) or the `pam_authz(5)` man page.

- Do you want to configure the `/etc/opt/ldaux/pam_authz.policy` to enforce account and password policies, stored in an LDAP directory server.

  LDAP-UX provides pam_authz enhancement to support enforcement of account and password policies, stored in an LDAP directory server. This feature works in conjunction with SSH (Secure Shell), r-commands with rhost enabled where authentication is not performed via the PAM subsystem, but is performed by the command itself.

  For detailed information on this feature and how to configure the pam_authz.policy file, see “Security Policy Enforcement with Secure Shell (SSH) or r-commands” (page 110).

- How will you communicate with your user community about the change to LDAP?

  For the most part, your user community should be unaffected by the directory. Most HP-UX commands will work as always. However, for some LDAP directories (such as Netscape Directory Server 6.x), data in replica servers cannot be modified. The *passwd*(1) command will not work on clients configured to use such a directory replica. See To Change Passwords (page 177) for how you can use *ldappasswd*(8) in this situation.

Check the *Release Notes* for any other limitations and tell your users how they can work around them.

Install LDAP-UX Client Services on a Client

Use *swinstall*(1M) to install the LDAP-UX Client Services software, the NativeLdapClient subproduct, on a client system. See the *LDAP-UX Integration B.04.10 Release Notes* for any last-minute changes to this procedure. You don't need to reboot your system after installing the product.

NOTE: Starting with LDAP-UX Client Services B.03.20 or later, system reboot is not required after installing the product.

NOTE: For the HP 9000 and Integrity (ia64) client systems, you need to install the required patches. For the detailed information about the required patches, refer to "*LDAP-UX Integration B.04.10 Release Notes* at: *http://www.docs.hp.com.*

Configure Your Directory

This section describes how to configure your directory to work with LDAP-UX Client Services. Examples are given for Netscape Directory Server for HP-UX version 6.x. See the *LDAP-UX Integration B.04.10 Release Notes* for information on supported directories. If you have a different directory, see the documentation for your directory for details on how to configure it.

See *Preparing Your LDAP Directory for HP-UX Integration* at http://docs.hp.com/hpux/internet for more details on directory configuration.

1. Install the posix schema (RFC 2307) into your directory.

   If you have Netscape Directory Server for HP-UX version 4.0, or later, the posix schema is already installed.

   The schema is in the file /opt/ldapux/ypldapd/etc/slapd-v3.nis.conf. For information on the posix schema (RFC 2307), see http://www.ietf.org/rfc.html. RFC 2307 consists of object classes such as: posixAccount, posixGroup, shadowAccount, etc. posixAccount represents a user entry from /etc/passwd. posixGroup represents a group entry from /etc/group. And shadowAccount provides additional user information for added security.

2. Restrict write access to certain passwd (posixAccount) attributes of the posix schema.

   CAUTION: Make sure you restrict access to the attributes listed below. Allowing users to change them could be a security risk

   Grant write access of the uidnumber, gidnumber, homedirectory, and uid attributes only to directory administrators; disallow write access by all other users. You may want to restrict write access to other attributes in the passwd (posixAccount) entry as well.

   With Directory Server for HP-UX, you can use the Directory Console or ldapmodify to set up access control instructions (ACI) so ordinary users cannot change these attributes in their passwd entry in the directory.

   The following access control instruction is by default at the top of the directory tree for a 6.x Netscape directory. This ACI allows a user to change any attribute in their passwd entry:

   ```
   aci: (targetattr = "*") (version 3.0; acl "Allow self entry modification";

    allow (write)userdn = "ldap:///self";)
   ```

   You could modify this example ACI to the following, which prevents ordinary users from changing their uidnumber, gidnumber, homedirectory, and uid attributes:

   ```
   aci: (targetattr != "uidnumber || gidnumber || homedirectory || uid") (version
    3.0; acl "Allow self entry modification, except for important posix attributes";
    allow (write)userdn = "ldap:///self";)
   ```

   You may have other attributes you need to protect as well.

   To change an ACI with the Directory Console, select the Directory tab, select your directory suffix in the left-hand panel, then select the Object: Set Access Permissions menu item. In the dialog box, select the "Allow self entry modification" ACI and click OK. Use the Set Access Permissions dialog box to modify the ACI. See "Managing Access Control" in the *Netscape Directory Server Administrator's Guide* for complete details.

3. Restrict write access to certain group (posixGroup) attributes of the posix schema.

   Grant write access of the cn, memberuid, gidnumber, and userPassword attributes only to directory administrators; disallow write access by all other users.

   With Netscape/Red Hat Directory Server for HP-UX, you can use the Directory Console or ldapmodify to set up access control lists (ACL) so ordinary users cannot change these attributes in the posixGroup entry in the directory. For example, the following ACI, placed in the directory at `ou=groups,ou=unix,o=hp.com`, allows only the directory administrator to modify entries below `ou=groups,ou=unix,o=hp.com`:

```
aci: (targetattr = "*")(version 3.0;acl "Disallow modification of group
 entries"; deny (write) (groupdn != "ldap:///ou=Directory Administrators,
 o=hp.com");)
```

4. Grant read access of all attributes of the posix schema.

   Ensure all users have read access to the posix attributes.

   When using PAM_LDAP as your authentication method, users do not need read access to the userPassword attribute since the authentication is handled by the directory itself. Therefore, for better security, you can remove read access to userPassword from ordinary users.

5. Configure anonymous access, if needed. If you do not configure a proxy user, then the attributes of your name service data must be readable anonymously.
6. Create a proxy user in the directory, if needed.

   To create a proxy user with Netscape/Red Hat Directory Server for HP-UX, use the Directory Console, Users and Groups tab, Create button. For example, you might create a user `uid=proxyuser,ou=Special Users,o=hp.com`.

7. Set access permissions for the proxy user, if configured.

   Give the proxy user created above read permission for the posix account attributes.

   With Netscape Directory Server, for example, the following ACI gives a proxy user permission to compare, read, and search all posix account attributes except the userPassword attribute:

```
aci: (target="ldap:///o=hp.com")(targetattr!="userpassword")
  version 3.0; acl "Proxy userpassword read rights";
  allow (compare,read,search)
  userdn = "ldap:///uid=proxyuser,ou=Special Users,o=hp.com";)
```

8. The default ACI of Netscape Directory Server 6.11 allows a user to change his own common attributes. But, for Netscape Directory Server 6.21 or later, you need to set ACI that gives a user permission to change his own common attributes. By default, the Netscape Directory Server 6.21 or later provides the following ACI named `Enable self write for common attributes` that gives a user permission to change his own common attributes:

```
aci: (targetattr = "carLicense ||description ||displayName
 ||facsimileTelephoneNumber ||homePhone ||homePostalAddress ||initials
 ||jpegPhoto ||labeledURL ||mail ||mobile ||pager ||photo ||postOfficeBox
 ||postalAddress ||postalCode ||preferredDeliveryMethod ||preferredLanguage

 ||registeredAddress ||roomNumber ||secretary ||seeAlso ||st ||street
 ||telephoneNumber ||telexNumber ||title ||userCertificate ||userPassword
 ||userSMIMECertificate ||x500UniqueIdentifier")
 (version 3.0; acl "Enable self  write for common attributes"; allow (write)

 (userdn = "ldap:///self"))
```

   You can modify the default ACI and give appropriate access rights to change your own common attributes.

9. Index important attributes for better performance of Directory Server.

   Since many of your directory requests will be for the attributes listed below, you should index these to improve performance. If you don't index, your directory may search sequentially causing a performance bottleneck. As a rule of thumb, databases containing more than 100 entries should be indexed by their key attributes.

The following attributes are recommended for indexing:

- cn
- objectclass
- memberuid
- uidnumber
- gidnumber
- uid
- ipserviceport
- iphostnumber

To index these entries with Netscape/Red Hat Directory Server, use the Console, Configuration tab, Indexes tab, Add Attributes button.

10. Determine if you need to support enumeration requests. If you do, increase the Look-Through limit, the Size limit, and the All-IDs-Threshold in the Directory Server.

    Enumeration requests are directory queries that request all of a database, for example all users or all groups. Enumeration requests of large databases could reduce network and server performance. With large Netscape/Red Hat Directories and default configurations, enumerations may fail or provide incomplete data, but the default configuration also may prevent performance problems from enumerations.

    If you need to support enumerations with large Netscape/Red Hat Directories, increase the listed parameters as described in *Preparing Your LDAP Directory for LDAP-UX Integration* available at http://docs.hp.com/hpux/internet/#LDAP-UX%20Integration.

    The Look-through limit specifies the maximum number of directory entries to examine before aborting the search operation. The Size limit determines the maximum number of entries to return to any query before aborting. The All-IDs-Threshold specifies the number of entries that can be maintained for an index key. In general, it is bad practice to have an extremely large All-ID's threashold, as it can dramatically increase the size of your directory server's database. However, if you have a large number of posixAccounts, posixGroups or other form of RFC 2307 data that needs to be enumerated and you also have other large sets of data in your directory server, increasing the All-UID's threashold to above the maximum number of posixAccounts, posixGroups, or others, can dramatically increase enumeration performance.

    For information on these parameters and how to change them, see the *Red Hat Directory Server Administrator's Guide*. See also Minimizing Enumeration Requests (page 120).

11. If you want to enable SSL support with LDAP-UX, you need to turn on SSL in your directory server. For detailed information on how to set up and configure your Directory Server to enable SSL communication over LDAP, see “Managing SSL” Chapter in the *Red Hat Directory Server Administrator's Guide* at *http://docs.hp.com/en/internet.html*

Import Name Service Data into Your Directory

The next step is to import your name service data into your LDAP Directory. Here are some considerations when planning this:

- If you have already imported data into your directory with the NIS/LDAP Gateway product, LDAP-UX Client Services can use that data and you can skip to Configure the LDAP-UX Client Services (page 33).
- If you are using NIS, the migration scripts take your NIS maps and generate LDIF files. These scripts can then import the LDIF files into your directory, creating new entries in the directory. This only works if you are starting with an empty directory or creating an entirely new subtree in your directory for your data.

  If you are not using NIS, the migration scripts can take your user, group, and other data from files, generate LDIF, and import the LDIF into your directory.
- If you integrate the name service data into your directory, the migration scripts may be helpful depending on where you put the data in your directory. You could use them just to generate LDIF, edit the LDIF, then import the LDIF into your directory. For example, you could manually add the posixAccount object class to your existing entries under `ou=People` and add their HP-UX information there.

Steps to Importing Name Service Data into Your Directory

Here are the steps for importing your user and group data into your LDAP directory. Modify them as needed.

1. Decide which migration method and scripts you will use. Migration scripts are provided to ease the task of importing your existing name service data into your LDAP directory.

   See Name Service Migration Scripts (page 170) for a complete description of the scripts, what they do, and how to use them. Modify the migration scripts, if needed.
2. Back up your directory.
3. Run the migration scripts, using the worksheet in Configuration Worksheet (page 183).
4. If the method you used above did not already do so, import the LDIF file into your directory.

Configure the LDAP-UX Client Services

Below is a summary of how to configure LDAP-UX Client Services with Netscape Directory Server 6.x. For a default configuration, see Quick Configuration (page 34). For a custom configuration, see Custom Configuration (page 38) for more information.

NOTE: The setup program has only been certified with Netscape Directory Server 6.x, Red Hat Directory Server 7.x and Windows 2000/2003/2003 R2 Active Directory Sever. See the *LDAP-UX Integration B.04.10 Release Notes* (P/N J4269-90063).

NOTE: The LDAP-UX Client Services B.04.00 or later supports storage of automount maps and publickeys on Netscape /Red Hat Directory Server 6.x and 7.0/7.1. See the *LDAP-UX Integration B.04.10 Release Notes* (P/N J4269-90065).

- Run the Setup program. The setup program provides the following assistance:
  - Extends your Netscape/Red Hat directory schema with the configuration profile schema, if not already done
  - Imports the LDAP printer schema into your Directory Server if you choose to start the LDAP printer configurator
  - Imports the publickey schema into your Directory Server if you choose to store the public keys of users and hosts in an LDAP directory
  - Imports the new automount schema into your Directory Server if you choose to store the AutoFS maps in an LDAP directory
  - Provides the option to enable SSL for secure communication between LDAP clients and Directory servers
  - Optionally configures SASL Digest-MD5 authentication (for Netscape/Red Hat Directory only)
  - Creates a configuration profile entry in your directory server from information you provide
  - Updates the local client's start-up file (/etc/opt/ldapux/ldapux_client.conf) with your directory and configuration profile location
  - Downloads the configuration profile from the directory to your local client system
  - Configures a proxy user for the client, if needed
  - Starts the Client Daemon if you choose to start it

IMPORTANT: Starting with LDAP-UX Client Services B.03.20, the client daemon, `/opt/ldapux/bin/ldapclientd`, must be running for LDAP-UX functions to work. With LDAP-UX Client Services B.03.10 or earlier, running the client daemon, `ldapclientd`, is optional.

NOTE: The LDAP printer configurator can support any Directory Servers that support the LDAP printer schema based on *IETF<draft-fleming-ldap-printer-schema-02.txt>*.

However, the LDAP-UX Client Services only supports automatically importing the LDAP printer schema into the Directory Server by running the setup program.

If your directory server does not support the LDAP printer schema, you may experience problems when importing the printer schema.

- Configure the Pluggable Authentication Module (PAM) by modifying the file /etc/pam.conf. See /etc/pam.ldap for a sample.
- Configure the Name Service Switch (NSS) by modifying the file /etc/nsswitch.conf. See /etc/nsswitch.ldap for a sample.

- Optionally modify the disable_uid_range flag in the /etc/opt/ldapux/ldapux_client.conf file to disable logins to the local system from specific users.
- Optionally configure the authorization of one or more subgroups from a large repository such as an LDAP directory server. For the detailed information on how to set up the policy file, `/etc/opt/ldapux/pam_authz.policy`, see Policy File (page 101).

After you configure your directory and the first client system, configuring additional client systems is simpler. Refer to Configure Subsequent Client Systems (page 68) for more information.

Quick Configuration

You can quickly configure a Netscape/Rat Hat directory and the first client by letting most of the configuration parameters take default values as follows. For a custom configuration, see Custom Configuration (page 38).

The steps described below assume that you don't use SSL or TLS support with LDAP-UX. If you want to enable SSL support, see Custom Configuration (page 38).

1. Log in as root and run the Setup program:

   ```
   cd /opt/ldapux/config
   ./setup
   ```

   The Setup program asks you a series of questions and usually provides default answers. Press the Enter key to accept the default, or change the value and press Enter. At any point during setup, enter Control-b to back up or Control-c to exit setup.

2. Choose the Directory Server as your LDAP directory server (option 1).
3. Enter either the host name or IP address of the directory server where your profile exists, or where you want to create a new profile from Configuration Worksheet (page 183).
4. Enter the port number of the previously specified directory server that you want to store the profile from Configuration Worksheet (page 183). The default port number is 389.
5. If the profile schema has already been imported, setup skips this step. Otherwise, enter "yes" to extend the profile schema if the schema has not been imported with LDAP-UX Client Services object class `DUAConfigProfile`, See LDAP-UX Client Services Object Classes (page 185) for a detailed description of this object class.
6. If the LDAP printer schema has already been extended, setup skips this steps. Otherwise, enter "yes" to extend the LP printer schema if you choose to start the printer configurator. The LDAP printer configurator is a feature that simplifies the LP printer management by refreshing LP printer configurations on your client system. A new printer schema, which is based on *IETF<draft-fleming-ldap-printer-schema-02.txt>,* is required to start the services.
7. If the publickey schema has already extended, setup skips this step. Otherwise, enter "yes" to extend the publickey schema if you choose to store the public keys of users and hosts in the LDAP directory. A publickey schema, which is based on RFC 2307-bis is required to migrate the publickeys in the NIS+ credential table entries on the NIS+ server to the LDAP directory.
8. If the new automount schema has already been imported, setup skips to step 9.

   Otherwise, you will be asked whether or not you want to install the new automount schema which is based on RFC 2307-bis. Enter "yes" if you want to import the new automount schema into the LDAP directory server. Enter "no" if you do not want to import new automount schema into the LDAP directory server. Setup skips to step 9 if you enter "no".

9. Next, if the setup program detects the obsolete automount schema exists in the LDAP directory, it will prompt you for the information shown as follows:

   ```
   The obsolete automount schema exists in the directory.
   If you still want to use the new automount schema, you must
   perform the following steps:
    1. Exit this program
   ```

```
 2. Stop directory server
 3. Remove the obsolete automount schema:
    a. objectclass- automount
    b. attribute-automountInformation

    Note: for Netscape Directory Server, they are in 10rfc2307.ldif.
 4. Start directory and re-run setup program to install the new
automount schema.
Do you still want to use the new automount schema?
Press Yes will exit this program. {YES]:
```

Reply "`yes`" when asked do you still want to use the new automount schema. If you reply `yes`, it will take you to exit this program. You must re-run the setup program again to install the new automount schema after you exit this program and manually delete the obsolete automount schema. For detailed information on how to remove the obsolete automount schema, see Removing The Obsolete Automount Schema (page 59).

If you reply `no`, setup skips to step 9 and the new automount schema will not be imported.

Otherwise, you will be asked to enter the DN (Distinguished Name) and password of the directory user who can import the schema into the LDAP directory.

10. If you are creating a new profile, add all parent entries of the profile DN to the directory (if any). If you attempt to create a new profile and any parent entries of the profile do not already exist in the directory, setup will fail. For example, if your profile will be **`cn=profile1,ou=profiles,o=hp,com`**, then **`ou=profiles,o=hp.com`** must exist in the directory or setup will fail.
11. Next enter either the DN of a new profile, or the DN of an existing profile you want to use, from Configuration Worksheet (page 183).

    To display all the profiles in the directory, use a command like the following:

    ```
    ldapsearch -b o=hp.com objectclass=DUAConfigProfile dn
    ```

    If you are using an existing profile, setup configures your client, downloads the profile, and exits. In this case, continue with step 12 below.
12. If you are creating a new profile, enter the DN and password of the directory user who can create a new profile from Configuration Worksheet (page 183).
13. Next, it will prompt you for the following information:

    ```
    Select authentication method for users to bind/authenticate to
    the server
    1. SIMPLE
    2. SASL DIGEST-MD5
    To accept the default shown in brackets, press the Return key.

    Authentication method: [1]:
    ```

    Press the return key if you choose to accept SIMPLE authentication method, type 2 if you choose SASL DIGEST-MD5 authentication method for the following prompt:

    ```
    Authentication method: [1]:
    ```

14. Next enter the host name and port number of the directory where your name service data is, from Configuration Worksheet (page 183). For high availability, each LDAP-UX client can look for name service data in up to three different directory hosts. You can enter up to three hosts, to be searched in order.
15. Enter the base DN where clients should search for name service data from Configuration Worksheet (page 183).

16. You can quickly configure a Directory Server and the first client by accepting the remaining default configuration parameters when prompted.

    If you want to use the SASL DIGEST-MD5 authentication method, you need to configure a proxy user with its credential level.

    Using the SASL DIGEST-MD5 authentication, the password must be stored in the clear text in the LDAP directory.

    Configuration Parameter Default Values (page 36) shows the configuration parameters and the default values they will be configured with.

    Table 2-1 Configuration Parameter Default Values

Parameter	Default Value
Type of client binding	Anonymous
Bind time limit	5 seconds
Search time limit	no limit
Use of referrals	Yes
Profile TTL (Time To Live)	0 - infinite
Use standard RFC-2307 object class attributes for supported services	Yes
Use default search descriptions for supported services	Yes
Authentication method	Simple

    To change any of these default values, refer to Custom Configuration (page 38).

17. After entering all the configuration information, setup extends the schema, creates a new profile, and configures the client to use the directory.
18. Configure the Pluggable Authentication Module (PAM).

    Save a copy of the file /etc/pam.conf and edit the original to specify LDAP authentication and other authentication methods you want to use. See /etc/pam.ldap for a sample. You may be able to just copy /etc/pam.ldap to /etc/pam.conf. See *pam*(3), *pam.conf*(4), and *Managing Systems and Workgroups* at http://docs.hp.com/hpux for more information on PAM.

19. Configure the Name Service Switch (NSS).

    Save a copy of the file `/etc/nsswitch.conf` and edit the original to specify the ldap name service and other name services you want to use. See /etc/nsswitch.ldap for a sample. You may be able to just copy /etc/nsswitch.ldap to /etc/nsswitch.conf. See `nsswitch.conf(4)` for more information.

20. Optionally, configure the Pam Authorization Service module (pam_authz).

    LDAP-UX Client Services provides a sample configuration file, `/etc/opt/ldapux/pam_authz.conf.template`. This sample file shows you how to configure the policy file to work with pam_authz. You can copy this sample file and edit it using the correct syntax to specify the access rules you wish to authorize or exclude from authorization. For more detailed information on how to configure the policy file. see PAM_AUTHZ Login Authorization (page 98).

    The sample `/etc/pam.conf` file in the man page will show you how to configure the `/etc/pam.conf` file to work with pam_authz. For more detailed information about pam_authz, refer to the `pam_authz(5)` man page.

21. Optionally configure the disable_uid_range flag.

    Save a copy of the file `/etc/opt/ldapux/ldapux_client.conf` and edit the original to activate the disable_uid_range flag. Uncomment the flag in the [NSS] portion of the file and fill in the UID range. The format is disable_uid_range=uid#,[uid#-uid#], ... where uid# stands for uid number.

For example: disable_uid_range=0-100,300-450,89

Note:

- White spaces between numbers are ignored.
- Only one line of the list is accepted, however, the line can be wrapped.
- The maximum number of ranges is 20.

22. Verify the LDAP-UX Client Services (page 65).
23. Configure subsequent clients by running setup on those clients and specifying an existing configuration profile. Or for a simpler process see Configure Subsequent Client Systems (page 68).

Custom Configuration

Running the Setup program for a quick configuration, as described above, configures your client using default values where possible. If you would like to customize these parameters, proceed as follows.

If you want to use SSL or TLS, you must perform the following tasks before you run the custom configuration. See "Configure the LDAP-UX Client Serivces with SSL or TLS Support" (page 45) for details.

- Ensure that you have installed the certificate database files, *cert8.db or cert7.db and key3.db*, on your client system.
- If you choose to use TLS, set the `enable_starttls` parameter to `1` in the */etc/opt/ldapux/lldapux_client.conf* file to enable TLS. To use SSL, set `enable_starttls` to `0` to disable TLS. By default, TLS is disabled.

1. Perform the steps described in Quick Configuration (page 34).

   However, after step 11, you will be asked whether you want to use SSL or not if the value of the `enable_starttls` parameter is `0` (disabled) or undefined. Enter "yes" to the following question if you want to use SSL for the secure communication between LDAP clients and the Netscape/Red Hat Directory Server. Enter "no" to the following question if you don't want to use SSL. Skip to step 2.

   ```
   Do you want to use SSL (y/n)?
   ```

   Otherwise, if the value of the `enable_starttls` parameter is `1` (enabled), you will be asked whether you want to use TLS or not. Enter "yes" to the following question if you want to use TLS for the secure communication between LDAP clients and the Netscape/Red Hat Directory Server. Enter "no" to the following question if you don't want to use TLS. Skip to step 3.

   ```
   Do you want to use TLS (y/n)?
   ```

2. Next, it will prompt you for selecting the authentication method for users to bind/authenticate to the server.

   You have a choice between SIMPLE (the default), or SASL DIGEST-MD5 if you choose to not enable SSL. However, you have a choice between SIMPLE with SSL (the default), or SASL DIGEST-MD5 with SSL if you choose to enable SSL.

   LDAP-UX supports SASL DIGEST-MD5 authentication method for Netscape Directory Server 6.21 and Red Hat Directory Server 7.1 with SP2 version (B.07.10.20).

   If you select SASL DIGEST-MD5, two additional prompts will appear. The first will prompt you for a user mapping (UID, DN, or Other). The second will prompt you for a single realm to use when retrieving user authentication information. If no realm is specified, user information will be retrieved from the first realm the directory server offers.

   Skip to step 4.

3. Next, it will prompt you for selecting the authentication method for users to bind/authenticate to the server.

   You have a choice between SIMPLE (the default), or SASL DIGEST-MD5 if you choose to not enable TLS. However, you have a choice between SIMPLE with TLS (the default), or SASL DIGEST-MD5 with TLS if you choose to enable TLS.

   If you select SASL DIGEST-MD5, two additional prompts will appear. The first will prompt you for a user mapping (UID, DN, or Other). The second will prompt you for a single realm to use when retrieving user authentication information. If no realm is specified, user information will be retrieved from the first realm the directory server offers.

4. Specify the host name and optional port number where your directory is running. If you choose to use TLS, the default directory port number is 389. If you choose to use SSL, the default directory port number is 636.

   For high availability, each LDAP-UX client can look for user and group information in up to three different directory servers. You are able to specify up to three directory hosts, to be searched in order.

5. Reply "no" when asked if you want to accept the remaining default configuration parameters.
6. Select the client binding you want from Configuration Worksheet (page 183). This determines the identity that client systems use when binding to the directory to search for user and group information.
7. If you configured a proxy user, enter the DN and password of your proxy user, from Configuration Worksheet (page 183).

   If you want to use the SASL DIGEST-MD5 authentication method, you need to configure a proxy user with its credential level.

   Using the SASL DIGEST-MD5 authentication, the password must be stored in the clear text in the LDAP directory.

8. Enter the maximum time in seconds the client should wait for directory searches before aborting. Enter 0 for no time limit.
9. Enter whether or not you want directory searches to follow referrals. Referrals are a redirection mechanism supported by the LDAP protocol. Please see your directory manuals for more information on referrals.

NOTE: If you want your directory searches to follow referrals, you must allow anonymous access into your directories.

10. Enter the Profile TTL (Time To Live) value. This value defines the time interval between automatic downloads (refreshes) of new configuration profiles from the directory. Automatic refreshing ensures that the client is always configured using the newest configuration profile.

    If you want to disable automatic refresh or manually control when the refresh occurs, enter a value of 0. Download the Profile Periodically (page 69).

11. In this step, the setup program initiates a dialog where you can remap the standard object class attributes to alternate attributes. You may want to do this if the attributes in your directory do not conform to the object classes defined in RFC 2307.

    You can remap the attributes for any of the supported services: `passwd`, `shadow passwd`, `group`, `PAM`, `netgroup`, `rpc`, `protocols`, `networks`, `hosts`, `services` and `automount`.

NOTE: Make sure that the attribute names are entered correctly to avoid unpredictable results later.

Refer to RFC 2307 at http://www.ietf.org/rfc/rfc2307.txt for a description of the standard object classes and attributes.

At this point, the setup program will display the following dialog:

```
LDAP-UX Client Services supports the following services:
1.Password                              7.Networks
2.Shadow passwd                         8.Hosts
3.Group                                 9.Services
4.PAM (Pluggable Authentication Module)10.Printers
5.RPC                                  11.Automount
6 Protocols                            12.Netgroup
```

```
Each services uses a standard object class (defined by RFC 2307)
You can remap any of these attributes to alternate attributes.
Do you want to remap any of the standard RFC 2307 attributes?
```

Enter "yes" if you want to remap attributes for any of the supported services. Then go to the "Remapping Attributes for Services" (page 41) section for details of the procedures.

Otherwise, if you do not want to remap attributes for any of the supported services, then enter "no" to this prompt to continue to step 13 of the setup process.

12. In this step, the setup program initiates a dialog where you can create a custom search descriptor. A custom search descriptor allows you to specify a different search location or filter for retrieving entries for services supported by LDAP-UX Client. Each name service can have up to three different search descriptors. A custom search descriptor consists of three parts: a search base DN, scope, and filter. The client uses the search descriptors in order until it finds what it is looking for.

NOTE: If your search filters overlap, enumeration requests will result in duplicate entries being returned. For example, if one search filter searched a subset of your organization and a second search filter searched your entire organization, an enumeration request would return duplicate entries.

See the "Minimizing Enumeration Requests" section for more information.

To begin the process to create custom search descriptors, setup will prompt you for the following information:

```
LDAP-UX Client Services supports the following services:


1.Password                               7.Networks
2.Shadow passwd                          8.Hosts
3.Group                                  9.Services
4.PAM (Pluggable Authentication Module)10.Printers
5.RPC                                   11.Automount
6.Protocols                             12.Netgroup


You can create up to three custom search descriptors for each name
service to search different locations in the directory for user
and group information.
Do you want to create custom search descriptors? [No]:
```

Enter 'yes' if you want to create custom search descriptors for any of the supported services. Then enter the number of the service for which you want to create a custom search descriptor.

If, you do not want to create custom search descriptors, enter 'no' to this prompt to continue to step 13 of the setup process.

Creating the nisObject Search Filter

LDAP-UX Client Services uses the automount search filter for the automount service as default. If you want to create the `nisObject` search filter for the automount service to search a different location in the directory, use the following steps:

1. Type yes for the following question and press the return key:

   `Do you want to create custom search descriptors? [No]:` yes

2. Next, it will take you to the screen which shows you the following information:

   ```
   To accept the default shown in brackets, press the Return key.
   search base [dc=cup,dc=hp,dc=com]:
   search scope (base, one, sub) [sub]
   Search filter [(objectclass=automount)]
   ```

If you want to create the `nisObject` search filter for the automount service, then type `(objectclass=nisObject)` for the following prompt and press the Return key; otherwise press the return key to accept the default search filter, `objectclass=automount:`

`Search filter [(objectclass=automount)]:` (objectclass=nisObject)

13. You will be asked whether or not you want to start the client daemon. For LDAP-UX Client B.03.20 or later versions, the client daemon must be started for LDAP-UX functions to work. With LDAP-UX Client B.30.10 or earlier, the client daemon is optional, and should be turned on in order to provide better prformance (response time) and for the X.500 group membership to work.

Remapping Attributes for Services

This section describes detailed procedures on how to perform attribute mappings for automount, dynamic group and X.500 group membership services.

Attribute Mappings For Automount Service

By default, LDAP-UX Client Services uses the RFC2307-bis automount schema. The `nisObject` automount schema can also be used if configured via attribute mappings.

Use the following steps if you want to remap the automount attributes to the `nisObject` automount attributes:

1. Enter yes for the following question:

   `Do you want to remap any of the standard RFC 2307 attributes? [yes]:` yes

2. If you want to select the `automount` service, then enter 11 for the following question and press the return key:

   ```
   Specify the service you want to map? [0]:11
   ```

3. Next, it will take you to the screen which shows you the following information:

   ```
   Current Automount attribute names:


   1.automountMapName ->[automountMapname]
   2.automountKey -> [automountKey]
   3.automountInformation -> [automountInformation]
   Specify the attribute you want to map. [0]:
   ```

   You type 1 for the following question and press the return key:

   `Specify the attribute you want to map. [0]:`1

4. Next, type the attribute `nisMapName` that you want to map to the `automountMapName` attribute for the following question and press the return key:

   `automountMapName` -> `nisMapName`

5. Next, it will take you to the screen which shows you the following information:

   ```
   Current Automount attribute names:


   1.automountMapName ->[nisMapname]
   2.automountKey -> [automountKey]
   3.automountInformation -> [automountInformation]
   Specify the attribute you want to map. [0]:
   ```

   If you want to specify the attribute to map to the `automountKey` attribute, then type 2 for the following question and press the return key:

```
Specify the attribute you want to map. [0]:2
```

6. Next, type the attribute `cn` you want to map to the `automountKey` attribute and press the return key:

```
automountKey -> cn
```

7. Next, it will take you to the screen which shows you the following information:

```
Current Automount attribute names:


1.automountMapName ->[nisMapname]
2.automountKey -> [cn]
3.automountInformation -> [automountInformation]
Specify the attribute you want to map. [0]:
```

If you want to specify the attribute to map to the `automountInformation` attribute , then type 3 for the following question and press the return key:

```
Specify the attribute you want to map. [0]:3
```

8. Next, type the attribute `nisMapEntry` you want to map to the `automountInformation` attribute and press the return key:

```
automountInformation -> nisMapEntry
```

9. Next, it will take you to the screen which shows you the following information:

```
Current Automount attribute names:


1.automountMapName ->[nisMapname]
2.automountKey -> [cn]
3.automountInformation -> [nisMapEntry]
Specify the attribute you want to map. [0]:
```

You type 0 to exit this menu for the following question:

```
Specify the attribute you want to map. [0]:0
```

Attribute Mappings For Dynamic Group Support

If you are configuring dynamic group support, you need to remap the default group member attribute, `memberuid`, to `memberURL` (for Netscape/Red Hat Directory Server) or `nxsearchFilter` (for HP Openview Select Access). For detailed information about dynamic group support, see "Dynamic Group Support" (page 77).

Use the following steps to remap the `memberuid` attribute to the dynamic group attributes, `memberURL` or `nxsearchFilter`. For example, the following procedures are used to remap `memberuid` to `memberURL`:

1. Type yes for the following question:

```
Do you want to remap any of the standtard RFC 2307 attributes? [yes]:
yes
```

2. Select the `group` service by entering 3 for the following question and press the return key:

```
Specify the service you want to map? [0]: 3
```

3. Next, it will take you to the screen which shows you the following information:

```
Current Group attribute names:


1.cn ->[cn]
2.gidnumber  -> [gidnumber]
3.memberuid -> [memberuid]
```

```
4.userpassword -> [userPassword]
Specify the attribute you want to map. [0]:
```

If you want to specify the attribute to map to `memberuid`, then type 3 for the following question and press the return key:

```
Specify the attribute you want to map? [0]: 3
```

4. Type the attribute, `memberURL` or `nxsearchFilter`, that you want to map to the `memberuid` attribute and press the return key:

   ```
   memberuid —> memberURL
   ```

5. Next, it will take you to the screen which shows you the following information:

   ```
   Current Group.attribute names:

   1.cn ->[cn]
   2.gidnumber  -> [gidnumber]
   3.memberuid -> [memberURL]
   4.userpassword -> [userPassword]
   Specify the attribute you want to map. [0]:
   ```

   You type 0 to exit this menu for the following question:

   ```
   Specify the attribute you want to map. [0]:0
   ```

Attribute Mappings for X.500 Group Membership Support

If you want to configure X.500 group membership support, you should remap the group member attribute to `member` or `uniquemember` instead of using the default attribute, `memberuid`.

Perform the following steps for attribute mappings to set up X.500 group membership:

1. Type yes for the following question:

   ```
   Do you want to remap any of the startdard RFC 2307 attributes? [yes]:
   yes
   ```

2. Select the `group` service by entering 3 for the following question and press the return key:

   ```
   Specify the service you want to map? [0]: 3
   ```

3. Next, it will take you to the screen which shows you the following information:

   ```
   Current Group attribute names:

   1.cn ->[cn]
   2.gidnumber  -> [gidnumber]
   3.memberuid -> [memberuid]
   4.userpassword -> [userPassword]
   Specify the attribute you want to map. [0]:
   ```

   If you want to specify the attribute to map to `memberuid`, then type 3 for the following question and press the return key:

   ```
   Specify the attribute you want to map? [0]: 3
   ```

4. Type the `member` attribute that you want to map to the `memberuid` attribute and press the return key:

   ```
   memberuid —> member
   ```

5. Next, it will take you to the screen which shows you the following information:

   ```
   Current Group.attribute names:
   ```

```
1.cn ->[cn]
2.gidnumber  -> [gidnumber]
3.memberuid -> [member]
4.userpassword -> [userPassword]
Specify the attribute you want to map. [0]:
```

You type 0 to exit this menu for the following question:

```
Specify the attribute you want to map. [0]:0
```

NOTE: LDAP-UX supports DN-based (X.500 style) membership syntax. This means that you do not need to use the memberUid attribute to define the members of a POSIX group. Instead, you can use either the `member` or `uniqueMember` attribute. LDAP-UX can convert from the DN syntax to the POSIX syntax (an account name).

For Netscape/Red Hat Directory Server, the typical member attribute would be either `memberUid`, `member` or `uniqueMember`.

Configure the LDAP-UX Client Serivces with SSL or TLS Support

The LDAP-UX Client Services provides SSL (Secure Socket Layer) support to secure communication between LDAP clients and the LDAP directory server. An encrypted session is established on an encrypted port, 636. The LDAP-UX Client Services supports SSL with password as the credential, using either simple bind or DIGEST-MD5 authentication (DIGEST-MD5 is available for Netscape/Red Hat Directory Server only) to ensure confidentiality and data integrity between clients and servers. With SSL support, the LDAP-UX Clients provides a secure way to protect the password over the network. The directory administrator has the choice in selecting authentication mechanism, such as using simple password stored in the directory server as a hash syntax.

The LDAP-UX Client Services supports Microsoft Windows 2000, 2003 or 2003 R2 Active Directory Server (ADS) and Netscape/Red Hat Directory Server (NDS/RHDS) over SSL. For detailed information on how to set up and configure your Netscape/Red Hat Directory Server to enable SSL communication over LDAP, see *"Managing SSL Chapter*" in the *Administrator's Guide for Netscape/Red Hat Directory Server* at *http://www.redhat.com/docs/manuals/dir-server/*

TLS Support

Starting with LDAP-UX Client Services B.04.10, the product supports a new extension operation of TLS protocol called startTLS to secure communication between LDAP clients and the LDAP directory server. By default, an encrypted session is established on a un-encrypted port, 389. If an encrypted port is used, it will fail to establish the secure connection. The TLS protocol provides administrators better flexibility for using TLS in their environment by allowing the use of an un-encrypted LDAP port for communication between the clients and the server. LDAP-UX supports TLS with password as the credential, using either simple bind or DIGEST-MD5 authentication (DIGEST-MD5 is available for Netscape/Red Hat Directory Server only) to ensure confidentiality and data integrity between clients and servers.

The LDAP-UX Client Services supports Microsoft Windows 2003 or 2003 R2 Active Directory Server (ADS), Netscape Directory Server (NDS) 6.x and Red Hat Directory Server (RHDS) 7.0/7.1 over TLS.

Configuration Parameters

LDAP-UX Client Services provides the following parameter in the */etc/opt/ldapux/ldapux_client.conf* file to support TLS:

`enable_starttls` This integer variable controls whether the TLS feature is enabled or disabled. The valid values of this parameter are `1` and `0`. If you choose to use TLS, set this parameter to `1`. To disable TLS, set this variable to `0`. By default, TLS is disabled. If the enable_startTLS parameter is undefined or does not exist, it is processed as the TLS feature is disabled.

If you want to use SSL or TLS, you must perform the following tasks before you run the setup program:

- Ensure to have the certificate database files, *cert8.db* or *cert7.db* and *key3.db*, on your client system. See "Configuring the LDAP-UX Client to Use SSL or TLS" (page 45) for details.
- If you choose to use TLS, set the `enable_starttls` parameter to `1` in the */etc/opt/ldapux/lldapux_client.conf* file. To use SSL, set `enable_starttls` to `0`. By default, TLS is disabled.

Configuring the LDAP-UX Client to Use SSL or TLS

You can choose to enable SSL or TLS with LDAP-UX when you run the setup program. If you attempt to use SSL or TLS, you must install Certificate Authority (CA) certificate on your LDAP-UX Client and configure your LDAP directory server to support SSL or TLS before you run the setup program.

NOTE: If you already have the certificate database files, *cet7 or cert8.db* and *key3.db*, on your client for your HP-UX applications, you can simply create a symbolic link */etc/opt/ldapux/cert7.db* that points to *cert7.db* or */etc/opt/ldapux/cert8.db* that points to *cer8.db* and */etc/opt/ldapux/key3.db* that points to *key3.db*.

You can Download the certificate database from the Netscape Communicator or Mozilla browser to set up the certificate database into your LDAP-UX Client.

Steps to Download the CA Certificate from Mozilla Browser

The following steps show you an example on how to download the Certificate Authority (CA) certificate on your client system using Mozilla browser 1.4 for HP-UX:

1. Log in to your system as root.
2. Use Mozilla browser to connect to your Certificate Authority Server.

   The following shows an example of using a link to connect to your Certificate Authority Server:

   https://CA servername:port number/ca/
3. Click the `retrieval` tab in the *Netscape certificate management* window screen.
4. Click the "*import CA certificate chain*" link to take you to the *"import CA* certificate chain" window screen.
5. Check the *"import the CA certificate chain into your browser*" check box in the *"import CA certificate chain*" window screen. Then, click the `submit` button.
6. Check the "*Trust the CA to identify web sites*", "*Trust the CA to identify e-mail users*", and " *Trust the CA to identify software developers*" checkboxes in the *Downloading Certificate* window screen. Then click OK button.
7. The Netscape Directory CA certificate will be downloaded to the following two files on your LDAP-UX Client:

   /.mozilla/default/.slt/cert8.db*

   /.morilla/default/.slt/key3.db*
8. You can simply copy the */.mozilla/default/*slt/cert8.db* file to */etc/opt/ldapux/cert8.db* and */.mozilla/default/*slt/key3.db* file to */etc/opt/ldapux/key3.db.*
9. Set the file access permissions for*/etc/opt/ldapux/cert7..db* and */etc/opt/ldapux/key3.db* to be read only by root as follows:

```
-r-------- 1 root sys 65536 Jun 14 16:27 /etc/opt/ldapux/cert8.db
-r-------- 1 root sys 32768 Jun 14 16:27 /etc/opt/ldapux/key3.db
```

NOTE: You may use the unsupported `/opt/ldapux/contrib/bin/certutil` command line tool to create the certificate database files, *cert8.db* and *key3.db*. For detailed command options and their arguments, see *Using the Certificate Database Tool* available at *http://www.mozilla.org/projects/security/pki/nss/tools/certutil.html*.

NOTE: If your browser does not generate `cert7.db` or `cert8.db` and `key3.db` security database files, you must export the certificate (preferably the root certificate of the Certificate Authority that signed the LDAP server's certificate) from your certificate server as a Base64-Encoded certificate and use the `certutil` utility to create the`cert8.db` and `key3.db` security database files.

Steps to create database files using the certutil utility

The following steps show you an example on how to create the security database files, `cert8.db` and `key3.db` on your client system using the `certutil` utility:

1. Retrieve the Base64-Encoded certificate from the certificate server and save it.

   For example, get the Base64-Encoded certificate from the certificate server and save it as the `/tmp/mynew.cert` file. This file should look like:

```
--------------- BEGIN CERTIFICATE ------------------------------
-MIICJjCCAY+gAwIBAgIBJDANBgkghkiG9w0BAQQFADBxMQswCQYDVQQGEwJVUzEL
MAkga1UECBMCQ2ExEjAQBgNVBAcTCWN1cGVvsG1ubzEPMA0GA1UEChmgAhaUy29T
MRIwEAYDVQQLEw1RR1NMLUxkYXAxHDAaBgNVBAMTE0N1cnRpzmljYXR1IE1hbmFn
4I2vvzz2i1Ubq+Ajcf1y8sdafuCmqTgsGUYjy+J1weM061kaWOt0HxmXmrUdmenF
skyfHyvEGj8b5w6ppgIIA8JOT7z+F0w+/mig=
--------------- END CERTIFICATE ---------------------------------
```

2. Use the `rm` command to remove the old database files, `/etc/opt/ldapux/cert8.db` and `/etc/opt/ldapux/key3.db`:

   ```
   rm -f /etc/opt/ldapux/cert8.db /etc/opt/ldapux/key3.db
   ```

3. Use the `certutil` utility with the `-N` option to initialize the new database:

   ```
   /opt/ldapux/contrib/bin/certutil -N -d /etc/opt/ldapux
   ```

4. Add the Certificate Authority (CA) certificate or the LDAP server's certificate to the security database:

   - To use the `certutil` command to add a CA certificate to the database:

     For example, the following command adds the CA certificate, `my-ca-cert`, to the security database directory, `/etc/opt/ldapux`, with the Base64-Encoded certificate request file, `/tmp/mynew.cert`:

     ```
     /opt/ldapux/contrib/bin/certutil -A -n my-ca-cert -t \ "C,," -d
     /etc/opt/ldapux -a -i /tmp/mynew.cert
     ```

     NOTE: The `-t "C,,"` represents the minimum trust attributes that may be assigned to the CA certificate for LDAP-UX to successfully use SSL or TLS to connect to the LDAP directory server. If you have other applications that use the CA certificate for other functions, then you may wish to assign additional trust flags. See *http://www.mozilla.org/projects/security/pki/nss/tools/certutil.html* for additional information.

   - To use the `certutil` command to add the LDAP server's certificate to the security database:

     For example, the following command adds the LDAP server's certificate, `my-server-cert`, to the security database directory, `/etc/opt/ldapux`, with the Base64-Encoded certificate request file, `/tmp/mynew.cert`:

     ```
     /opt/ldapux/contrib/bin/certutil -A -n my-server-cert \
     -t "P,," -d /etc/opt/ldapux -a -i /tmp/mynew.cert
     ```

NOTE: The `-t "p,,"` represents the minimum trust attributes that may be assigned to the LDAP server's certificat for LDAP-UX to successfully use SSL or TLS to connect to the LDAP directory server. See *http://www.mozilla.org/projects/security/pki/nss/tools/certutil.html* for additional information.

Adjusting the Peer Certificate Policy

With SSL/TLS, not only communication between clients (LDAP-UX) and servers (the LDAP directory server) can be protected, but in addition, specific levels of assurance of the identities of the clients and servers can be validated. This section describes how to adjust this validation level.

The `peer_cert_policy` parameter in the `/etc/opt/ldapux/ldapux_client.conf` configuration file is a string variable used to control the validation level. There are three valid options for this parameter described below:

`WEAK` Performs no validation of SSL or TLS certificates. Communication between the client and server can be encrypted, however the client has no assurance that it is communicating with a trusted server.

`CERT` Verifies that the issuers of peer SSL or TLS certificates are trusted. Communication between the client and server can be encrypted and the client has some assurance that it is communicating with a trusted server. In this scenario, it is still possible for the server to have a certificate that has been issued for a different server if methods used to protect private keys of server certificates are not in place. `CERT` is the default mode of operation with LDAP-UX.

`CNCERT` Performs both the CERT check and also verifies that the common name or `subjectAltName` values embedded in the certificate matches the address used to connect to the LDAP server, as described in RFC 4513.

As mentioned above, the default mode of operation for LDAP-UX is `CERT`. Increasing certificate validation level to `CNCERT` requires additional and specific configuration steps. If not properly established, it can interfere with LDAP-UX and proper system operation. Because LDAP-UX can be used for host-name resolution (similar to DNS), LDAP-UX normally stores the IP address of LDAP servers in the configuration profile. This procedure assures that if LDAP-UX is asked to resolve a host name, it can do so without first needing to resolve the host name of the LDAP directory server (which could lead to a catch-22). However, since certificates normally embed the host name or fully qualified host name and LDAP-UX only has the IP address of the host, it is not possible for LDAP-UX to verify the host name on the certificate.

If you want to configure the `CNCERT` validation level with the `peer_cert_policy` parameter, you must manually execute the following configuration steps:

1. Update the `preferredserverlist` setting in the profile to contain the host name of the LDAP server such that it matches the host name specified in the LDAP server's certificate. See the "Modifying perferredserverList in the LDAP-UX Profile" section for details.
2. Select and execute one of the following steps:
   - Either LDAP-UX must not be used for host-name resolution by removing "`ldap`" from the "`hosts`" service in the `/etc/nsswitch.conf` file.
   - Or the host name and IP address must be provided by some other name resolution service, such as "`files`" or "`dns`", and that service must appear before "`ldap`" in the `/etc/nsswitch.conf` file for the "`hosts`" service.

Modifying preferredSererList in the LDAP-UX Profile

Use the following steps to modify the value of the `preferredServerList` attribute in the LDAP-UX configuration profile:

1. Run the following steps to find the name of the LDAP server used on the server certificate. Assuming this certificate has been installed in your local certificate database file, `/etc/opt/ldapux/cert8.db`:
   - Run the following commands to list all server certificates used by LDAP-UX:

     ```
     cd /etc/opt/ldapux
     ```

     ```
     certutil -d . -L
     ```

   - Run the following command to select the nickname of the certificate from the above list:

     ```
     cetutil -d . -L -n <selected nickname>
     ```

   - Select the first name component of the "`Subject:`" name. For example, if the "`Subject:`" string is "`CN=ldapserver.example.com, O=Example Corp`" then the name component would be "`ldapserver.example.com`".

NOTE: Depending on how your certificate administrator manages your network, the above server certificate may not be found in your `cert8.db` file. Instead you may only find certificates for any trusted Certificate Authorities. In this case, contact your certificate administrator for the LDAP server certificate details.

2. In a separate window, use the `ldapentry` tool to modify the value of the `preferredServerList` attribute with the LDAP server name found in step 1. See "Modifying a Profile" (page 118) for detailed information on changing LDAP-UX configuration profile settings manually.
3. Examine the "`preferedServerList`" attribute
4. Use the `nslookup` tool to verify the IP address specified in the preferred server list matches that of the name of the host name found in step 1 above.

   For example, if the `preferredserverlist` attribute value is 192.168.1.1:636 and "Subject" is `CN=ldapserver.example.com,O=Example Corp`, then

   ```
   $ nslookup 192.168.1.1
   Name Server:  dns-resolver.example.com
   Address:  192.169.1.254

   Trying DNS
   Name:  ldapserver.example.com
   Address:  192.168.1.1
   ```

Configure LDAP-UX Client Services with Publickey Support

LDAP-UX Client Services B.04.00 or later version supports discovery and management of publickeys in an LDAP directory. Both public and secret keys, used by the Secure RPC API can be stored in user and host entries in an LDAP directory server, using thenisKeyObject objectclass. Support for discovery of keys in an LDAP directory server is provided through the getpublickey() and getsecretkey() APIs. You can use chkey and newkey commands to manage user and host keys in an LDAP server. The chkey -s ldap command is used to change user's secure RPC public key and secret key in an LDAP directory. The newkey -u <username> -s ldap command is used to add new keys for users to an LDAP directory while the newkey -h <hostname> -s ldap command is used to create new keys for machines to an LDAP directory.

For detailed information on the newkey andchkey commands, refer to newkey(1M),chkey(1), getpublickey(3N),getsecretkey() and publickey(4) man pages.

HP-UX Enhanced Publickey-LDAP Software Requirement

Support for publickey through LDAP requires functionality enhancement in LDAP-UX Client Services and an enhancement in the ONC product. ONC with publickey LDAP support is available through the HP-UX Enhanced Publickey-LDAP Software Pack (SPK) web release.

To enable the publickey LDAP support, you must install the Enhanced Publickey-LDAP software bundle shown on Table 2-2 and LDAP-UX Client Services B.04.00 or later on your client systems. The software bundle contains all the required patches plus the enablement product for this new feature. For detailed information, refer to the *ONC with Publickey LDAP Support Software Pack Release Notes* at the following web site:

http://docs.hp.com/en/netcom.html

Navigate to NFS Services.

Table 2-2 Enhanced Publickey-LDAP Software Requirement

Operating System Supported	Software Bundle Version	Planned Release Date
HP-UX 11i v1	Enhkey B.11.11.01	June, 2006
HP-UX 11i v2	Enhkey B.11.23.01	October, 2006

You can download the Enhanced Publickey-LDAP software bundle from the following Software Depot web site:

- Go to http://www.hp.com/go/softwaredepot.
- Click on the Enhancement releases and patch bundles link.
- Select one of the following links:
  - — HP-UX Software Pack (Optional HP-UX 11i v1 Core Enhancements) for HP-UX 11i v1
  - — HP-UX Software Pack (Optional HP-UX 11i v2 Core Enhancements) for HP-UX 11i v2
- Select one of the following links:
  - — HP-UX Public Key LDAP link for HP-UX 11i v1
  - — PublicKey-LDAP link for HP-UX 11i v2

- Select and download one of the following software bundle, place it to your client system, /tmp is assumed:
  - `Enhkey B.11.11.01 HP-UX B.11.11 64+32 depot` for HP-UX 11i v1
  - `Enhkey B.11.23.01 HP-UX B.11.23 IA+PA depot` for HP-UX 11i v2
- Use swinstall to install the software bundle:
  - `swinstall -x autoreboot=true -s /tmp/ENHKEY_B.11.11.01_HP-UX_B.11.11_64_32.depot` for HP-UX 11i v1
  - `swinstall -x autoreboot=true -x reinstall=false -s /tmp/ENHKEY_B.11.23.01_HP-UX_B.11.23_IA_PA.depot` for HP-UX 11i v2

Extending the Publickey Schema into Your Directory

The publickey schema is not loaded in the Netscape/Red Hat Directory Server. If you are installing LDAP-UX B.04.00 or later version on your client system, the setup program will extend the publickey schema into your Directory Server. If you previously configured LDAP-UX B.03.30 or earlier version, and now update the product to version B.04.00 or later, you must re-run the setup program to extend the publickey schema into your LDAP directory. You do not need to re-run the setup program for the subsequent client systems. For detailed information on how to run the setup program to extend the publickey schema into an LDAP directory, see Quick Configuration (page 34).

Admin Proxy User

A special type of proxy user, known as an Admin Proxy has been added to LDAP-UX to support management of publickey information in an LDAP directory server. The Admin Proxy represents the HP-UX administrator's rights in the directory server and typically is used to represent root's privileges extended to the directory server. Only an Admin Proxy user is allowed to use the `newkey` tool to add host and user keys into the LDAP directory server, or to use the `chkey` tool to modify host keys in the LDAP directory server.

Configuring an Admin Proxy User Using ldap_proxy_config

You need to use a new `ldap_proxy_config` tool option`-A` to configure an Admin Proxy user. You must specify the `-A` option along with other options to perform operations applying to an Admin Proxy user. For example, you can use the `ldap_proxy_config -A -i` command to create an Admin Proxy user. See The ldap_proxy_config Tool (page 132) for details.

Password for an Admin Proxy User

In order to protect user's secret keys in the LDAP directory, the secret keys are encrypted using the user's password. This process is used in NIS as well as NIS+ environments. The host's secret key must also be encrypted. Since the host itself does not have its own password, root's password is used to encrypt the host's secret key. The `chkey` or `newkey` command prompts for root's password when changing or adding a key for a host. For this reason, you may wish to configure the Admin Proxy user in the LDAP directory to have the same password as the root user on the master host. Although it is not required that the Admin Proxy user and root user share the same password, it allows you to avoid storing the Admin Proxy user's password in the `/etc/opt/ldapux/acred` file. In such case, when you run the `ldap_proxy_config -A -i` command to configure the Admin Proxy user, you enter only Admin Proxy user's DN without the password. LDAP-UX will use the root's password given to the `chkey` and `newkey` commands as the Admin Proxy user's password to perform public key operations. However, the `ldap_proxy_config -A -v` command will not be able to validate the Admin Proxy user because no password is available to `ldap_proxy_config`. As a result, the message `“No password is provided. Validation is not performed"` will be displayed.

Setting ACI for Key Management

Before storing public keys in an LDAP server, LDAP administrators may wish to update their LDAP access controls such that users can manage their own keys, and the Admin Proxy user can manage host keys. This section describes how you set up access control instructions (ACI) for an Admin Proxy user or a user.

Setting ACI for an Admin Proxy User

With Netscape Directory Server 6.11 and 6.21, you can use the Netscape Console or `ldapmodify` to set up ACI, which gives an Admin Proxy user permissions to manage host and user keys in the LDAP directory.

An Example

The following ACI gives the permissions for the Admin Proxy user `uid=keyadmin` to read, write, and compare `nissecretkey` and `nispublickey` attributes for hosts and users:

```
dn:dc=org,dc=hp,dc=com

aci:(targetattr ="objectclass||nispublickey||nissecretkey")
 (version 3.0;acl "Allow keyadmin to change key pairs";
 allow (read,write,compare)
 userdn="ldap:///uid=keyadmin,ou=people,dc=org,dc=hp,dc=com";)
```

Setting ACI for a User

The default ACI of Netscape Directory Server 6.11 allows a user to change his own nispublickey and nissecretkey attributes. For Netscape Directory Server 6.21, you need to set up ACI which gives a user permission to change his own nissecretkey and nispublickey attributes. Use the Netscape Console or ldapmodify to set up ACI for a user.

An Example

The following ACI gives a user permission to change his own nissecretkey and nispublickey attributes for user keys:

```
dn:ou=People,dc=org,dc=hp,dc=com


aci:(targetattr ="nissecretkey||nispublickey")(version 3.0;
 acl "Allow key self modification";allow (write)
 (userdn = "ldap:///self");)
```

Configuring serviceAuthenticationMethod

serviceAuthenticationMethod is a newly supported attribute of the configuration profile, /opt/ldapux/ldapux_profile.ldif. It's function is the same as authenticationMethod, but it allows authentication configuration for specific name services. The serviceAuthenticationMethod attribute is created to resolve issues that may arise when the default authentication method is not considered secure enough for specific name services. For example, if the default authenticationMethod is configured as NONE then the newkey and chkey commands would not know how to properly bind to the directory server when changing or adding key pairs. LDAP-UX only supports the serviceAuthenticationMethod attribute for the keyserv service, since the keyserv service is the only one that currently needs modification of privileges in the directory server.

To perform newkey and chkey operations, LDAP-UX binds the Admin Proxy user to the LDAP directory using the authentication method specified in serviceAuthenticationMethod. LDAP-UX only supports serviceAuthenticationMethod for keyserv. Any other services configured in serviceAuthenticationMethod will be ignored.

Configuring serviceAuthenticationMethod is optional. If you do not configure serviceAuthenticationMethod, LDAP-UX binds the Admin Proxy user to the LDAP directory using the authentication method specified for the proxy user.

Authentication Methods

LDAP-UX Client Services supports the following authentication methods for the keyserv service:

- simple with SSL enabled
- SASL DIGEST-MD5 with SSL enabled
- simple with SSL disabled
- SASL DIGEST-MD5 with SSL disabled

NOTE: SSL settings for both authenticationMethod and serviceAuthenticationMethod must be set the same. It is not supported to have SSL enabled for authenticationMethod and SSL disabled for serviceAuthenticationMethod, or vice versa.

Procedures Used to Configure serviceAuthenticationMethod

Use the following steps on one of LDAP-UX client sytems to configure the serviceAuthenticationMethod attribute in the /etc/opt/ldapux/ldapux_profile.ldif file:

1. Login as `root`.
2. Use the `ldapentry` tool to modify the profile entry in the LDAP directory server to include `serviceAuthenticationMethod`. To do this, `ldapentry` requires the profile DN. You can find the profile DN from `PROFILE_ENTRY_DN` in `/etc/opt/ldapux/ldapux_client.conf` after you finish running the `setup` program. The following example edits the profile entry "`cn=ldapuxprofile,dc=org,dc=hp,dc=com`":

   For example:

   ```
   cd /opt/ldapux/bin
   ```

   ```
   ./ldapentry -m "cn=ldapuxprofile,dc=org,dc=hp,dc=com"
   ```

   After you enter the prompts for "Directory login:" and "password:", `ldapentry` will bring up an editor window with the profile entry. You can add the `serviceAuthenticationMethod` attribute.

   The value of the `serviceAuthenticatioMethod` entry depends on the authentication method you configure. The following shows the possible values of the `serviceAuthenticationMethod` attribute:

   - For SASL DIGEST-MD5 using the Distinguish Name (DN) to generate the DIGEST-MD5 hash, the data in the entry is:

     ```
     serviceAuthenticationMethod:keyserv:sasl/digest-md5:username=dn
     ```

   - For SASL DIGEST-MD5 using the UID attribute to generate the DIGEST-MD5 hash, the data in the entry is:

     ```
     serviceAuthenticationMethod:keyserv:sasl/digest-md5
     ```

   - For SASL DIGEST-MD5 with SSL enabled using the DN to generate the DIGEST-MD5 hash, the data in the entry is:

     ```
     serviceAuthenticationMethod:keyserv:tls:sasl/digest-md5:username=dn
     ```

   - For SASL DIGEST-MD with SSL enabled using the UID attribute to generate the DIGEST-MD5 hash, the data in the entry is:

     ```
     serviceAuthenticationMethod:keyserv:tls:sasl/digest-md5
     ```

   - For simple authentication, the data in the entry is:

     ```
     serviceAuthenticationMethod:keyserv:simple
     ```

   - For simple with SSL enabled, the data in the entry is:

     ```
     serviceAuthenticationMethod:keyserv:tls:simple
     ```

   For more information on `ldapentry`, refer to Command and Tool Reference (page 127).

NOTE: If you use TLS for secure communication between LDAP clients and the Netscape/Red Hat Directory Server, you need to use Directory Server Console to manually add the values of the `serviceAuthenticationMethod` attribute.

3. Go to `/opt/ldapux/config:`

   ```
   cd /opt/ldapux/config
   ```

4. Use `/opt/ldapux/config/get_profile_entry` to download the modified LDIF profile:

   ```
   ./get_profile_entry -s nss
   ```

5. Run the `/opt/ldapux/config/display_profile_cache` tool to check the configuration of the `serviceAuthenticationMethod` attribute:

   ```
   ./display_profile_cache
   ```

   For example:

If the serviceAuthenticationMethod:keyserv:sasl/digest-md5 entry is added to the profile entry in the LDAP directory, you can see the following information when you run the display_profile_cache tool:

```
serv-auth: keyserv:sasl/digest-md5
auth opts: username: uid
realm:
```

For subsequent LDAP-UX client systems that share the same profile configuration, use the following steps to download and activate the profile:

1. Login as root.
2. Go to /opt/ldapux/config:

   ```
   cd /opt/ldapux/config
   ```

3. Use /opt/ldapux/config/get_profile_entry to download the modified LDIF profile:

   ```
   ./get_profile_entry -s nss
   ```

4. Run the /opt/ldapux/config/display_profile_cache tool to check the configuration of the serviceAuthenticationMethod attribute:

   ```
   ./display_profile_cache
   ```

5. Restart the LDAP-UX Client daemon, ldapclientd, if you change the authentication method from non-SSL to SSL. Otherwise, skip this step.

Configuring Name Service Switch

Configure the Name Service Switch (NSS) to enable the LDAP support for publickey.

You can save a copy of /etc/nsswitch.conf file and modify the original to add ldap support to the publickey service. See /etc/nsswitch.ldap for a sample.

The following shows the sample file, /etc/nsswitch.ldap:

```
passwd:       files ldap
group:        files ldap
hosts:        dns files ldap
networks:     files ldap
protocols:    files ldap
rpc:          files ldap
publickey:    ldap [NOTFOUND=return] files
netgroup:     files ldap
automount:    files ldap
aliases:      files
services:     files ldap
```

AutoFS Support

AutoFS is a client-side service that automatically mounts appropriate file systems when users request access to them. If an automounted file system has been idle for a period of time, AutoFS unmounts it. AutoFS uses name services such as files, NIS or NIS+ to store and manage AutoFS maps.

LDAP-UX Client Services B.04.00 supports the automount service under the AutoFS subsystem. This new feature allows users to store AutoFS maps in an LDAP directory server. .

AutoFS Patch Requirement

In order to enable the LDAP support for AutoFS, you must install the AutoFS patch or Enhanced AutoFS version on your client system shown in Table 2-3:

Table 2-3 Patch Requirement

Operating System Supported	Patch ID/Version	Planned Release Date
`HP-UX 11i v1`	Enhanced AutoFS version B.11.11.0509.1	September, 2005
`HP-UX 11i v2`	PHNE_33100	August, 2005

Automount Schemas

This section describes the following three automount schemas:

- new automount schema

  An automount schema is based on RFC 2307-bis. This schema defines new `automountMap` and `automount` structures to represent the AutoFS maps and their entries in the LDAP directory.

- nisObject automount schema

  The `nisObject` automount schema defines`nisMap` and `nisObject` structures to represent the AutoFS maps and their entries in the LDAP directory. There are some limitations that you need to be aware of when using the `nisObject` automount schema.

- obsolete automount schema

  This is the schema that is shipped with Netscape Directory Server version 6.x.

The LDAP-UX Client Services supports the new automount schema. The `nisObject` automount schema can also be used if configured via attribute mappings. LDAP-UX does not support the obsolete automount schema. You must manually delete it before the setup program can successfully import the new automount schema into the LDAP directory server.

Read subsequent sections of this chapter for the detailed information about the automount schemas.

New Automount Schema

This schema is a new schema defined in RFC2307-bis. This schema defines new `automountMap` and `automount` structures to represent AutoFS maps and their entries in the LDAP directory. AutoFS maps are stored in the LDAP directory server using structures defined by this schema.

The RFC2307-bis automount schema is not loaded in the Netscape Directory Server. If you are installing LDAP-UX B.04.00 on your client system, the setup program will import the new automount schema into your Netscape Directory Server. If you previously configured LDAP-UX B.03.30 or an earlier version, and are now updating the product to version B.04.00, you must re-run the setup program to import the new automount schema into the LDAP directory. The subsequent client systems do not need to re-run the setup.

Schema

The following shows the RFC 2307-bis automount schema in the LDIF format:

```
objectClasses: ( 1.3.6.1.1.1.2.16
NAME 'automountMap'
DESC 'Automount Map information'
SUP top STRUCTURAL
MUST automountMapName
MAY description
X-ORIGIN 'user defined' )

objectClasses: ( 1.3.6.1.1.1.2.17
NAME 'automount'
DESC 'Automount information'
SUP top STRUCTURAL
MUST ( automountKey $ automountInformation )
MAY description
X-ORIGIN 'user defined' )

attributeTypes: ( 1.3.6.1.1.1.1.31
NAME 'automountMapName'
DESC 'automount Map Name'
EQUALITY caseExactIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE
X-ORIGIN 'user defined' )

attributeTypes: ( 1.3.6.1.1.1.1.32
NAME 'automountKey'
DESC 'Automount Key value'
EQUALITY caseExactIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE
X-ORIGIN 'user defined' )

attributeTypes: ( 1.3.6.1.1.1.1.33
NAME 'automountInformation'
DESC 'Automount information'
EQUALITY caseExactIA5Match
SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE
X-ORIGIN 'user defined' )
```

For Netscape Directory Server, each entry started by "`attributetypes:`" or "`objectclasses:`" must be one continuous line.

An Example

The following shows an example of a direct AutoFS map, `auto_direct`, stored in the LDAP directory server using new automount schema:

```
dn:automountMapName=auto_direct,dc=nishpind
objectClass: top
objectClass: automountMap
automountMapName: auto_direct


dn:automountKey=/mnt_direct/test1,\
automountMapname=auto_direct, dc=nishpind
objectClass: top
objectClass: automount
automountInformation:hostA:/tmp
automountKey: /mnt_direct/test1


dn:automountKey=/mnt_direct/test2,\
automountMapname=auto_direct, dc=nishpind
objectClass: top
```

```
objectClass: automount
automountInformation:hostB:/tmp
automountKey:/mnt_direct/test2
```

The nisObject Automount Schema

The `nisObject` automount schema defines`nisMap` and `nisObject` structures to represent the AutoFS maps and their entries. The AutoFS maps are stored in the LDAP directory server using the `nisMap`and `nisObject` structures.

An Example

The following shows an example of a direct AutoFS map, `auto_direct`, stored in the LDAP directory server using the `nisObject` automount schema:

```
dn:nisMapName=auto_direct,dc=nishpind
objectClass: top
objectClass: nisMap
nisMapName: auto_directdn:cn=/mnt_direct/test1,
nisMapName=auto_direct, dc=nishpind
objectClass: top
objectClass: nisObject
nisMapName: auto_direct
cn: /mnt_direct/test1
nisMapEntry:hostA:/tmp


dn:cn=/mnt_direct/test2, nisMapname=auto_direct, dc=nishpind
objectClass: top
objectClass: nisObject
nisMapName: auto_direct
cn: /mnt_direct/test2
nisMapEntry:hostB:/tmp
```

Limitations

The `nisObject` automount schema contains three attributes, `cn`, `nisMapEntry` and `nisMapName`. `cn` is an attribute that ignores case-matching. Consider the following example:

```
# an indirect map named auto_test
test1    server1:/source
TEST1    server2:/source
```

In the above example, because the `cn` attribute is case-insensitive, the LDAP considers "`cn=TEST1, nisMapName=auto_test`" to be a redefinition of "`cn=test1, nisMapName=auto_test`".

Using the `nisObject` automount map schema, capital letters are not significant. In other words, if two keys have names that are only different by the use of capital letters, then one of those entries will be rendered inoperable because the other one is the only one that can be retrieved.

NOTE: If you use the `nisObject` automount map schema, do not use any keys that have capital letters and only differ from other keys by those capital letters.

Obsolete Automount Schema

The obsolete automount schema is shipped with the Netscape Directory Server version 6.x. You must manually delete it before the setup program can successfully import the new automount schema into the LDAP directory server.

Removing The Obsolete Automount Schema

Perform the following steps to delete the obsolete automount schema:

1. Login to your Netscape Directory Server as `root`.
2. Stop your Netscape Directory Server daemon, `slapd`.

   ```
   /var/opt/netscape/servers/slapd-<server-instance>/stop-slapd
   ```

   For example:

   ```
   /var/opt/netscape/servers/slapd-ldapA.cup.hp.com/stop-slapd
   ```

3. Delete the following two entries in the `/var/opt/netscape/servers/slapd-<server-instance>/ \ config/schema/10rfc2307.ldif` file. These two entries contain the ' `automountInformation`' attributetype and the ' `automount`' objectclass. The data in these two entries define the obsolete automount schema. The complete two entries are:

   -
     ```
     attributeTypes:( 1.3.6.1.1.1.1.25 NAME 'automountInformation'
     DESC 'Standard LDAP attribute type'
     SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 X-ORIGIN 'RFC 2307')
     ```

   -
     ```
     objectClasses:( 1.3.6.1.1.1.2.9 NAME 'automount'
     DESC 'Standard LDAP objectclass' SUP top MUST (cn
     $automountInformation)MAY (description) X-ORIGIN 'RFC2307')
     ```

4. Restart the daemon, `slapd`. This is to ensure that the updated schema file is recognized by the Netscape Directory Server.

   ```
   /var/opt/netscape/servers/slapd-<server-instance>/restart-slapd
   ```

   For example:

   ```
   /var/opt/netscape/servers/slapd-ldapA.cup.hp.com/restart-slapd
   ```

After you delete the obsolete automount schema, you must re-run the setup program to import the new automount schema into the LDAP directory server.

Attribute Mappings

LDAP-UX Client Services B.04.00 supports attribute mappings between the new RFC 2307-bis automount schema and the `nisObject` automount schema. This feature allows the directory administrators to use the `nisObject` schema if they have already deployed it.

When both new automount schema and `nisObject` schema exist in the LDAP directory server, if you choose to use the `nisObject` automount schema, you must run the setup program using the custom configuration to perform the attribute mappings and search filter changes for the automount service. The attribute mappings include the following:

- Remap the new automount attributes to the`nisObject` automount attributes. The attribute mappings are done in step 11 of the Custom Configuration. For detailed information on how to remap the automunt attributes, see Custom Configuration (page 38).

  Table 2-3 shows the attribute mappings:

Table 2-4 Attribute Mappings

New Automount Attribute	nisObject Automount Attribute
automountMapname	nisMapname
automountKey	cn
automountInformation	nisMapEntry

- Change the `automount` search filter for the automount service to the `nisObjectsearch` filter. LDAP-UX Client Services uses the `automount` search filter for the automount service as a default. The search filter change can be done in step 12 of the Custom Configuration. If you want to create the `nisObject` search filter for the automount service to search a different location in the LDAP directory server, see Custom Configuration (page 38) for details.

If you want to perform attribute mappings or search filter changes by using the Custom Configuration, ensure that you do not accept the remaining default configuration parameters in step 5 of the Custom Configuration.

NOTE: You can use the `nisObject` automount schema without attribute mappings and search filter changes if only the `nisObject` automount schema exists in the LDAP directory.

Configuring Name Service Switch

Configure the Name Service Switch (NSS) to enable the LDAP support for AutoFS.

You can save a copy of `/etc/nsswitch.conf` file and modify the original to add LDAP support to the automount service. See `/etc/nsswitch.ldap` for a sample.

The following shows the sample file, `/etc/nsswitch.ldap`:

```
passwd:       files ldap
group:        files ldap
hosts:        dns files ldap
networks:     files ldap
protocols:    files ldap
rpc:          files ldap
publickey:    ldap [NOTFOUND=return] files
netgroup:     files ldap
automount:    files ldap
aliases:      files
services:     files ldap
```

AutoFS Migration Scripts

This section describes the migration scripts which can be used to migrate your AutoFS maps from files, NIS servers or NIS+ servers to LDIF files. After LDIF files are created, you can use the `ldapmodify` tool to import LDIF files to your LDAP directory server. These migration scripts use the new automount schema defined in RFC 2307-bis to migrate the AutoFS maps to LDIF. You need to import the new automount schema into your LDAP directory server before you use these migration scripts to migrate AutoFS maps.

Table 2-4 describes the migration scripts:

Table 2-5 Migration Scripts

Migration Script	Description
`migrate_automount.pl`	Migrates AutoFS maps from files to LDIF.
`migrate_nis_automount.pl`	Migrates AutoFS maps from the NIS server to LDIF.
`migrate_nisp_autofs.pl`	Migrates AutoFS maps from NIS+ server to the `nisp_automap.ldif` file.

Environment Variables

When you use the AutoFS migration scripts to migrate AutoFS maps, set the following environment variables:

`LDAP_BASEDN` The base distinguished name of the LDAP directory that the AutoFS maps are to be placed in.

`DOM_ENV` This only applies to the `migrate_nisp_autofs.pl` script. This variable defines the fully qualified name of the NIS+ domain where you want to migrate your data from.

`NIS_DOMAINNAME` This only applies to the `migrate_nis_automount.pl` script. This variable specifies the fully qualified name of the NIS domain where you want to migrate your data from. This variable is optional. If the NIS domain name is not specified, LDAP-UX uses the value of the `NIS_DOMAIN` parameter configured in the `/etc/rc.conf.d/namesvrs` file.

Examples:

The following command sets the fully qualified name of the NIS+ domain to "`cup.hp.com`":

```
export DOM_ENV="cup.hp.com"
```

The following command sets the fully qualified name of the NIS domain to "`india.hp.com`":

```
export NIS_DOMAINNAME="india.hp.com"
```

The following command sets the base DN to "`dc=cup, dc=hp, dc=com`":

```
export LDAP_BASEDN="dc=cup, dc=hp, dc=com"
```

General Syntax For Migration Scripts

The migration scripts use the following general syntax:

```
scriptnameinputfileoutfile
```

where

`scriptname` Is the name of the particular script you are using.

`inputfile` Is the fully qualified file name of the appropriate AutoFS map that you want to migrate. For example, `/etc/auto_master`.

`outputfile` This only applies to the `migrate_nis_automount.pl` and `migrate_automount.pl` scripts. This is optional and is the name of the file where the LDIF is written. stdout is the default output.

The migrate_automount.pl Script

This script, found in `/opt/ldapux/migrate`, migrates the AutoFS maps from files to LDIF.

Syntax

scriptnameinputfileoutputfile

Examples

The following commands migrate the AutoFS map `/etc/auto_direct` to LDIF and place the results in the `/tmp/auto_direct.ldif` file:

```
export LDAP_BASEDN="dc=nishpind"
migrate_automount.pl /etc/auto_direct /tmp/auto_direct.ldif
```

The following shows the `/etc/auto_direct` file:

```
#local mount point          remote server:directory
/mnt/direct/lab1                   hostA:/tmp
/mnt/direct/lab2                   hostB:/tmp
```

The following shows the `/tmp/auto_direct.ldif` file:

```
dn:automountMapName=auto_direct,dc=nishpind
objectClass: top
objectClass: automountMap
automountMapName: auto_direct

dn:automountKey=/mnt_direct/lab1,\ automountMapname=auto_direct, dc=nishpind
objectClass: top
objectClass: automount
automountInformation:hostA:/tmp
automountKey: /mnt_direct/lab1


dn:automountKey=/mnt_direct/lab2,\
automountMapname=auto_direct, dc=nishpind
objectClass: top
objectClass: automount
automountInformation:hostB:/tmp
automountKey:/mnt_direct/lab2
```

You can use the `/opt/ldapux/bin/ldapmodify` tool to import the LDIF file `/tmp/auto_direct.ldif` that you just created above into the LDAP directory. For example, the following command imports the `/tmp/auto_direct.ldif` file to the LDAP base DN "`dc=nishpind`" in the LDAP directory server `LDAPSERV1`:

```
/opt/ldapux/bin/ldapmodify -a -h LDAPSERV1 -D "cn=Directory Manager" -w <passwd> -f /tmp/auto_direct.ldif
```

Where options are:

-a Add a new entry into the LDAP directory

-h The LDAP directory host name

-D The Distinguish Name (DN) of the directory manager

-w The password of the directory manager

-f The LDIF file to be imported into the LDAP directory

The migrate_nis_automount.pl Script

This script, found in /opt/ldapux/migrate, migrates the AutoFS maps from the NIS server to LDIF.

Syntax

scriptnameinputfileoutputfile

Examples

The following commands migrate the AutoFS map /etc/auto_indirect to LDIF and place the results in the /tmp/auto_indirect.ldif file:

```
export LDAP_BASEDN="dc=nisserv1"
export NIS_DOMAINNAME="cup.hp.com"
migrate_nis_automount.pl /etc/auto_indirect  /tmp/auto_indirect.ldif
```

The following shows the /etc/auto_indirect file:

```
#local mount point          remote server:directory
lab1                            hostA:/tmp
lab2                            hostB:/tmp
```

The following shows the /tmp/auto_indirect.ldif file:

```
dn:automountMapName=auto_indirect,dc=nisserv1
objectClass: top
objectClass: automountMap
automountMapName: auto_indirect

dn:automountKey=lab1,\
automountMapname=auto_indirect, dc=nisserv1
objectClass: top
objectClass: automount
automountInformation:hostA:/tmp
automountKey: lab1


dn:automountKey=lab2, \
automountMapname=auto_indirect, dc=nisserv1
objectClass: top
objectClass: automount
automountInformation:hostB:/tmp
automountKey:lab2
```

You can use the /opt/ldapux/bin/ldapmodify tool to import the LDIF file /tmp/auto_indirect.ldif that you just created above into the LDAP directory. For example, the following command imports the /tmp/auto_indirect.ldif file to the LDAP base DN "dc=nisserv1" in the LDAP directory server LDAPSERV1:

```
/opt/ldapux/bin/ldapmodify -a -h LDAPSERV1 -D "cn=Directory Manager"
-w <passwd> -f /tmp/auto_indirect.ldif
```

The migrate_nisp_autofs.pl Script

This script, found in `/opt/ldapux/migrate/nisplusmigration`, migrates the AutoFS maps from the NIS+ server to the `nisp_automap.ldif` file.

Syntax

scriptnameinputfile

Examples

The following commands migrate the AutoFS map `/etc/auto_indirect` to LDIF and place the results in the `nisp_automap.ldif` file:

```
export LDAP_BASEDN="dc=nishpbnd"
export DOM_ENV ="cup.hp.com"
migrate_nisp_autofs.pl /etc/auto_indirect
```

The following shows the `/etc/auto_indirect` file:

```
#local mount point          remote server:directory
lab1                            hostA:/tmp
lab2                            hostB:/tmp
```

The following shows the `nisp_automap.ldif` file:

```
dn:automountMapName=auto_indirect,dc=nishpbnd
objectClass: top
objectClass: automountMap
automountMapName: auto_indirect

dn:automountKey=lab1, \
automountMapname=auto_indirect, dc=nishpbnd
objectClass: top
objectClass: automount
automountInformation:hostA:/tmp
automountKey: lab1


dn:automountKey=lab2, \
automountMapname=auto_indirect, dc=nishpbnd
objectClass: top
objectClass: automount
automountInformation:hostB:/tmp
automountKey:lab2
```

You can use the `/opt/ldapux/bin/ldapmodify` tool to import the LDIF file `nisp_automap.ldif` that you just created above into the LDAP directory. For example, the following command imports the `nisp_automap.ldif` file to the LDAP base DN "`dc=nishpbnd`" in the LDAP directory server `LDAPSERV1`:

```
/opt/ldapux/bin/ldapmodify -a -h LDAPSERV1 -D "cn=Directory Manager"
 -w <passwd> -f nisp_automap.ldif
```

Verify the LDAP-UX Client Services

This section describes some simple ways you can verify the installation and configuration of your LDAP-UX Client Services. You may need to do more elaborate and detailed testing, especially if you have a large environment.

If any of the following tests fail, see Troubleshooting (page 123).

1. Use the *nsquery(1)*[1] command to test the name service:

   ```
   nsquery lookup_type lookup_query [lookup_policy]
   ```

   For example, to test the name service switch to resolve a username lookup, enter:

   ```
   nsquery passwd username ldap
   ```

   where **username** is the login name of a valid user whose posix account information is in the directory. You should see output something like the following depending on how you have configured /etc/nsswitch.conf:

   ```
   Using "ldap" for the passwd policy.
   Searching ldap for jbloggs
   User name: jbloggs
   user Id: 10000
   Group Id: 2000
   Gecos:
   Home Directory: /home/jbloggs
   Shell: /bin/sh
   Switch configuration: Terminates Search
   ```

   This tests the Name Service Switch configuration in /etc/nsswitch.conf. If you do not see output like that above, check /etc/nsswitch.conf for proper configuration.

2. Use other commands to display information about users in the directory, making sure the output is as expected:

   ```
   pwget -n username
   nsquery hosts host_to_find
   grget -n groupname
   ls -l
   ```

NOTE: While you can use the following commands to verify your configuration, these commands enumerate the entire passwd or group database, which may reduce network and directory server performance for large databases:

```
pwget(with no options)
grget(with no options)
listusers
logins
```

3. Use the beq search utility to search for the following services: pwd (password), grp (group), shd (shadow password), srv (service), prt (protocol), rpc (RPC), hst (host), net (network), ngp (netgroup), and grm (group membership). An example beq command using name as the search key, grp as the service, and ldap as the library is shown below.

   ```
   ./beq -k n -s grp -l /usr/lib/libnss_ldap.1nss_status........ NSS_SUCCESS
   pw_name...........(iuser1)
   pw_passwd.........(*)
   pw_uid............(101)
   pw_gid............(21)
   ```

[1] *nsquery*(1) is a contributed tool included with the ONC/NFS product.

```
pw_age...........()
pw_comment.......()
pw_gecos.........(gecos data in files)
pw_dir...........(/home/iuser1)
pw_shell.........(/usr/bin/sh)
pw_audid.........(0)
pw_audflg........(0)
```

Refer to "beq Search Tool" in Chapter 4 for command syntax and examples.

4. Log in to the client system from another system using rlogin or telnet. Log in as a user in the directory and as a user in /etc/passwd to make sure both work.
5. Optionally, test your pam_authz authorization configuration:

   If the pam_authz is configured without the `pam_authz.policy` file, verify the followings:

   - logging into the client system from another system using rlogin or telnet with a user name that is a member of a +@netgroup in the directory to make sure the user will be allowed to log in.
   - logging in as a user that is a member of a -@netgroup to be sure that the user will not be allowed to login.

   If the pam_authz is configured with the `pam_authz.policy` file, verify the followings:

   - logging into the client system with a user name that is covered by an `allow` access rule in the policy file. Make sure the user will be allowed to log in.
   - logging in as a user that is covered by a`deny` access rule in the policy file. Make sure the user can not login to the client system.
6. Open a new *hpterm*(1X) window and log in to the client system as a user whose account information is in the directory. It is important you open a new hpterm window or log in from another system because if login doesn't work, you could be locked out of the system and would have to reboot to single-user mode. This tests the Pluggable Authentication Module (PAM) configuration in /etc/pam.conf. If you cannot log in, check /etc/pam.conf for proper configuration. Also check your directory to make sure the user's account information is accessible by the proxy user or anonymously, as appropriate. Check your profile to make sure it looks correct. See also Troubleshooting in this chapter for more information.
7. Use the *ls*(1) or *ll*(1) command to examine files belonging to a user whose account information is in the directory. Make sure the owner and group of each file are accurate:

   ```
   ll /tmp
   ls -l
   ```

   If any owner or group shows up as a number instead of a user or group name, the name service switch is not functioning properly. Check the file /etc/nsswitch.conf, your directory, and your profile.

If you want to verify that you set up X.500 group membership correctly, follow these steps:

1. Create a valid posix user and group. Add this user as a member of this group using the attribute "member" instead of "memberuid". Here is an example ldif file specifying `xuser2` as a member of the group `xgrpup1`:

   ```
   #cat example_ids.ldif
   dn: cn=xgroup1,ou=Groups,o=hp.com]
   objectClass: posixGroup
   objectClass: groupofnames
   objectClass: top
   cn: xgroup1
   userPassword: {crypt}*
   gidNumber: 999
   member: uid=xuser2,ou=People,o=hp.com
   dn: uid=xuser2,ou=People,o=hp.com
   ```

```
uid: xuser2
cn: xuser2
objectClass: top
objectClass: account
objectClass: posixAccount
userPassword: {crypt}xxxxxxxxxxxxx
loginShell: /bin/ksh
uidNumber: 9998
gidNumber: 999
homeDirectory: /home/xuser2
```

2. Make sure that the file `/etc/nsswitch.conf` specifies ldap for group service:

   ```
   cat /etc/nsswitch.conf
   ```

   ```
   :
   :
   group: files ldap
   :
   :
   ```

3. Verify:

   ```
   # grget -n xgroup1
   ```

   ```
   xgroup1:*:999: xuser2
   ```

   If xuser2 shows up as a member of xgroup1, then your setup is correct.

Configure Subsequent Client Systems

Once you have configured your directory and one client system, you can configure subsequent client systems using the following steps. Modify any of these files as needed.

1. Use swinstall to install LDAP-UX Client Services on the client system. This does not require rebooting the client system.
2. Copy the following files from a configured client to the client being configured:
   - /etc/opt/ldapux/ldapux_client.conf
   - /etc/opt/ldapux/pcred only if you have configured a proxy user, not if you are using only anonymous access
   - /etc/pam.conf
   - /etc/nsswitch.conf
   - /etc/opt/ldapux/acred if the /etc/opt/ldapux/acred file exists
   - cert7.db or cert8.bd and key3.db flles if SSL is enabled

   Set all file access mode permission to be the same as those of the first client being configured.
3. Download the profile by running get_profile_entry as follows:

   ```
   cd /opt/ldapux/config
   ./get_profile_entry -s nss
   ```

   Alternatively you could interactively run the setup program to download the profile from the directory and respond "no" when asked if you want to change the current configuration:

   ```
   cd /opt/ldapux/config
   ./setup
   ```

4. If you are using a proxy user, configure the proxy user by calling ldap_proxy_config as follows:

   ```
   cd /opt/ldapux/config
   ./ldap_proxy_config
   ```

5. Verify the LDAP-UX Client Services (page 65).

Download the Profile Periodically

Setup allows you to define a time interval after which the current profile is being automatically refreshed. The start time for this periodic refresh is defined by the time the setup program was run and the value defined for ProfileTTL. Therefore, it does not allow you to define a specific time of day when the profile should be downloaded (refreshed). For more detailed information, refer to the ldapclientd(1) man page.

If you would like to manually control when you want to download the profile, you can use the following steps:

1. When creating your profile entry using setup, set the ProfileTTL value to 0.
2. Using the command `get_profile_entry -s nss`, write a shell script that downloads the profile. Below is an example that downloads the profile from the directory. Modify this example for your environment. It also compares the new and old profiles and emails a status message:

   ```
   #!/bin/ksh
   cp /etc/opt/ldapux/ldapux_profile.ldif /etc/opt/ldapux/ldapux_profile.sav
   /opt/ldapux/config/get_profile_entry -s nss 2>&1>/tmp/profile.upd$$
   diff /etc/opt/ldapux/ldapux_profile.ldif \
   /etc/opt/ldapux/ldapux_profile.sav >> /tmp/profile.upd$$
   if [ -s /tmp/profile.upd$$ ]; then
      cat /tmp/profile.upd$$ | mailx -s "Profile cache
   refreshed." root@sys01
   else
      echo "No changes." | mailx -s "Profile cache refreshed."
   root@sys01
   fi
   rm -f /etc/opt/ldapux/ldapux_profile.sav
   rm -f /tmp/profile.upd$$
   ```

3. Create a *crontab*(1) file (or edit your existing crontab file) and specify how frequently you want the profile to be downloaded. For example, assuming the script above is in the file /ldapux/download_ldap_profile, the following crontab specification specifies that /ldapux/download_ldap_profile be executed nightly at midnight:

   ```
   0 0 * * * /ldapux/download_ldap_profile
   ```

4. Log in as root and schedule the job with the *crontab*(1) command. For example, assuming the crontab entry above is in the file crontab.profile, the following schedules the profile downloading:

   ```
   crontab crontab.profile
   ```

Use r-command for PAM_LDAP

An enhancement has been implemented to the LDAP-UX Client Services B.03.20, so that `r-commands` can work with LDAP account users whose password is hidden, or not in clear text or crypt syntax.

If you want to use this new fearture, use the following steps:

1. Uncomment out the following line in the `/etc/opt/ldapux/ldapux_client.conf` file:

   #**password_as = "x"**

2. On the HP-UX 11.0 or 11i v1 client system, modify account management session in `/etc/pam.conf` file for pam_ldap to add `rcommand` option as shown below:

   ```
   # Account management
   #
   ```

```
login     account sufficient  /usr/lib/security/libpam_unix.1
login     account required    /usr/lib/security/libpam_ldap.1 rcommand
su        account sufficient  /usr/lib/security/libpam_unix.1
su        account required    /usr/lib/security/libpam_ldap.1
dtlogin   account sufficient  /usr/lib/security/libpam_unix.1
dtlogin   account required    /usr/lib/security/libpam_ldap.1
dtaction  account sufficient  /usr/lib/security/libpam_unix.1
dtaction  account required    /usr/lib/security/libpam_ldap.1
ftp       account sufficient  /usr/lib/security/libpam_unix.1
ftp       account required    /usr/lib/security/libpam_ldap.1
OTHER     account sufficient  /usr/lib/security/libpam_unix.1
OTHER     account required    /usr/lib/security/libpam_ldap.1 rcommand
```

On the HP-UX 11i v2 client system, you will modify account management session in `/etc/pam.conf` file for pam_ldap to add "`rcommand`" option as follows:

```
# Account management
#
login     account required    libpam_hpsec.so.1
login     account sufficient  libpam_unix.so.1
login     account required    libpam_ldap.so.1 rcommand
su        account required    libpam_hpsec.so.1
su        account sufficient  libpam_unix.so.1
su        account required    libpam_ldap.so.1
dtlogin   account required    libpam_hpsec.so.1
dtlogin   account sufficient  libpam_unix.so.1
dtlogin   account required    libpam_ldap.so.1
dtaction  account required    libpam_hpsec.so.1
dtaction  account sufficient  libpam_unix.so.1
dtaction  account required    libpam_ldap.so.1
ftp       account required    libpam_hpsec.so.1
ftp       account sufficient  libpam_unix.so.1
ftp       account required    libpam_ldap.so.1
rcomds    account required    libpam_hpsec.so.1
rcomds    account sufficient  libpam_unix.so.1
rcomds    account required    libpam_ldap.so.1 rcommand
sshd      account required    libpam_hpsec.so.1
sshd      account sufficient  libpam_unix.so.1
sshd      account required    libpam_ldap.so.1
OTHER     account sufficient  libpam_unix.so.1
OTHER     account required    libpam_ldap.so.1
```

CAUTION: Setting user password to be returned as any string for the hidden password, and turning on the "rcommand" option for pam_ldap account management could allow users with active accounts on a remote host to rlogin to the local host on to a disabled account.

If you have security concerns, see "Security Policy Enforcement with Secure Shell (SSH) or r-commands" (page 110) section in chapter 5 and Appendix D, "Sample /etc/pam.conf File for Security Policy Enforcement" (page 193) for detailed information on how to configure access rules in the `/etc/opt/ldapux/pam_authz.policy` file, set global policy access permissions and configure the `pam_authz` library and the `rcommand` option under the account management section in the `/etc/pam.conf` file.

3 LDAP Printer Configurator Support

This chapter contains information describing how LDAP-UX supports the printer configurator, how to set up the printer schema, and how to configure the printer configurator to control its behaviors.

This chapter contains the following sections:

Overview

Management of network printing is complex, and printers themselves are more complicated. Instead of having printer configuration and information scattered over client systems and printer servers, they can be stored and managed from a single repository. LDAP is suited to build a backend printer configuration database. LDAP-UX enables the centralized management of printers, and the printer entries can easily be distributed to clients to reduce concerns about synchronization of configuration information. LDAP-UX comes with a printer configurator to consolidate printer configuration and control of printer devices into the LDAP Directory Server for a central location of printer management.

Definitions

Printer Services

HP-UX provides LP spooler system with the LP subsystem to manage printers and print services requests. The LP subsystem is a collection of 18 programs that operate on the resources (files and subdirectories) in LP spool directory to perform their functions, such as `lpadmin`, `rlpdaemon` programs, and `lp` command.

Printing Protocol

The LP spooler system has built-in support for sending jobs to other hosts that running `rlpdaemon`. `rlpdaemon` is a line printer daemon (LPD) for handling remote spool requests. This feaure enables the user to install a printer on one host and make it accessible from other hosts. It also works with printers/printservers that have network interfaces that support the LDP protocol. The LPD network printing protocol is the widely used network printing protocol in the UNIX world.

LP Printer types

The LP spooler supports the following three types of printers:

- A network printer which is a printer connected to a network interface or printserver.
- A remote printer is a printer configured on a system other than the one you are logged into when you submit a print request.
- A local printer which is a printer that is directly connected to your system.

NOTE: The LDAP printer configurator only supports the HP LP spooler system, remote printers, network printers and printerservers that support Line Printer Daemon (LPD) protocol. It does not support local printers.

How the LDAP Printer Configurator works

The Printer Configurator is a service daemon which provides the following functions:

- Periodically searches the existing printer entries stored in LDAP Directory Server
- Compares the search result with the master printer record file on each scheduled ldapsearch
- Adds the print configuration to client system for each new printer
- Deletes the printer from the client system for each removed printer
- Updates master printer record file

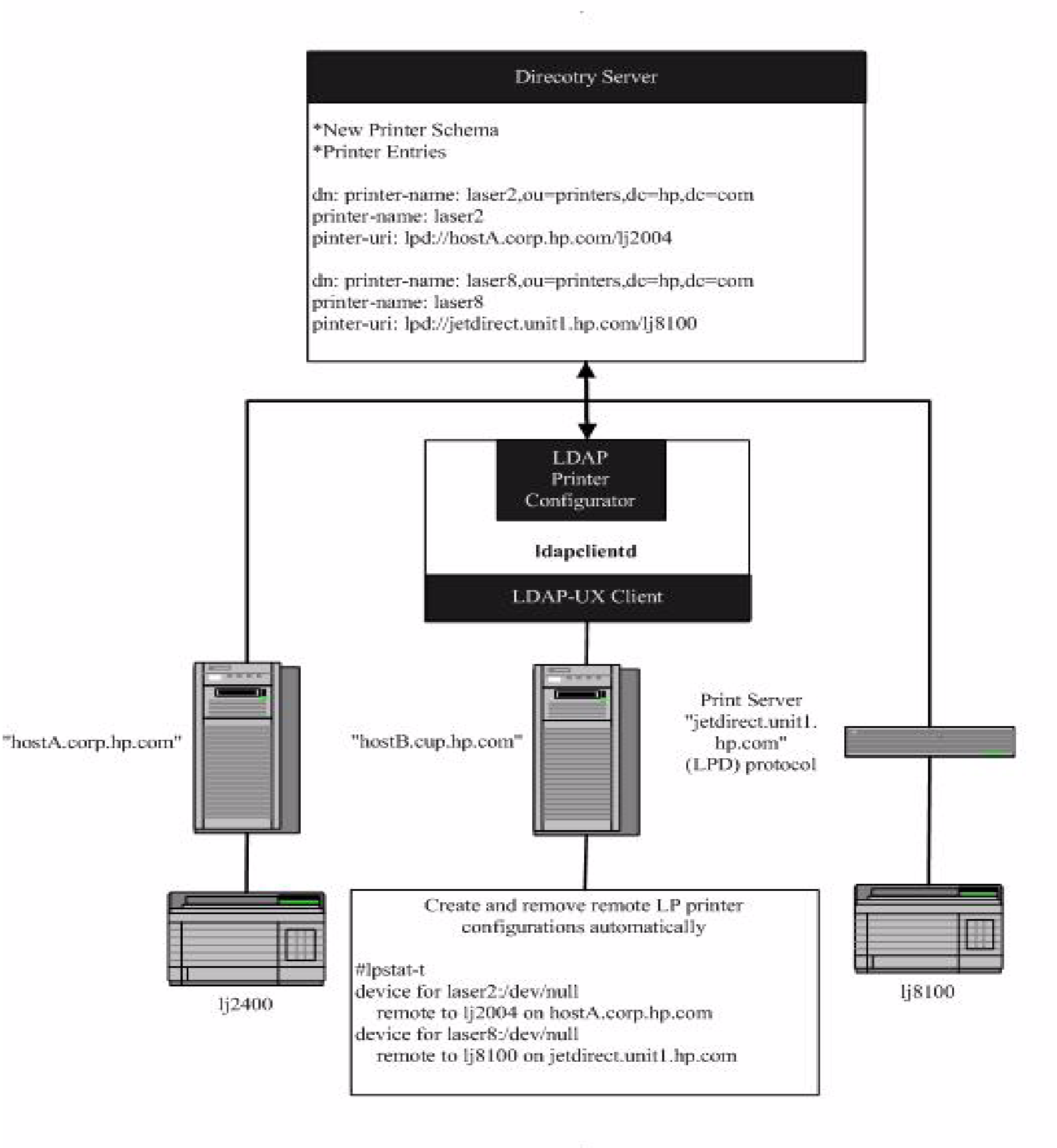
When `ldapclientd` is initialized, it will enable the printer configurator sevices at the same time. Once the printer configurator is up, it periodically searches for any existing printer entries in the LDAP Direcotry Server based on a predefined search filters. If there are any printer entries in the LDAP Directory Server, the printer configurator will extract the LP printer configuration from each printer entry.

Then, the printer configurator compares the printer configuration with the current LP printer configuration in the client system. The result of comparison will generate a list of new or removed printers. For a new printer, the printer configurator adds this printer to the LP printer spool of the client which is running the printer configurator. For a removed printer, the printer configurator deletes this printer from the LP printer spool of the client.

With the printer configurator, if a printer administrator attempts to remove or add a printer, all the administrator has to do is to add or delete the printer entry in the LDAP Directory Server. The printer configuration will be updated automatically without manually setting the printers on each client system.

NOTE: The system administrator manually adds or removes printers to the HP-UX system. The LDAP Printer Configurator will only add or remove printers that it has discovered in the LDAP directory according to the search filter defined for the printer.

Figure 3-1 Printer Configurator Architecture



```
dn: printer-name: laser2,ou=printers,dc=hp,dc=com
printer-name: laser2
pinter-uri: lpd://hostA.corp.hp.com/lj2004

dn: printer-name: laser8,ou=printers,dc=hp,dc=com
printer-name: laser8
pinter-uri: lpd://jetdirect.unit1.hp.com/lj8100
```

```
#lpstat-t
device for laser2:/dev/null
   remote to lj2004 on hostA.corp.hp.com
device for laser8:/dev/null
   remote to lj8100 on jetdirect.unit1.hp.com
```

Printer Configuration Parameters

The LDAP-UX Client Services provides four printer configuration parameters, `start`, `search_interval`, `max_printers` and `lpadmin_option`available for you to customize and control the behaviors of the printer configurator. These parameters are defined in the `ldapclientd.conf` file. For detailed information on these new parameters, refer to Administering LDAP-UX Client Services (page 87).

Printer Schema

The new printer schema, *IETF<draft-fleming-ldap-printer-schema-02.txt>*, is used to create the printer objects that are relevant to the printer configurator services. The draft printer schema can be obtained from IETF web site at *http://www.ietf.org*. For the detailed structure information of the new printer schema, see Appendix C. You must import the new printer schema into the LDAP Directory Server to create new printer objects.

NOTE: The LDAP printer configurator supports any Directory Servers that support the LDAP printer schema based on *IETF<draft-fleming-ldap-printer-schema-02.txt>*.

An Example

The following shows a typical printer object entry:

```
dn: printer-name=printer1,ou=printers,dc=cup,dc=hp,dc=com
objectclass: top
objectclass: printerabstract
objectclass: printerservice
objectclass: printerlpd
printer-name: lj81003
printer-uri: lpd://hostA.hp.com/lj81003
printer-location: 47L
printer-make-model: hp laser jet 81003
printer-service-person: John Louie
```

With the new printer schema, you are able to create printer objects for the LP printer configuration.The minimum information for a printer object entry is the local printer name, remote hostname, and the remote printer name. The remote hostname is the system or device that the remote printer is connected to. The remote hostname must be the fully qualified name.

The `printer-name` attribute provides information of local printer name, the `printer-uri` attribute identifies the remote hostname and the remote printer name information. URI stands for uniform resources identifier. The syntax of URI is based on RFC 2396. The following shows an example of the `printer-uri` attribute:

```
printer-uri: lpd://hostA.hp.com/lj2004
```

Managing the LP printer configuration

The LDAP-UX Client Services provide the printer configurator integration; the product daemon automatically updates the remote LP printer configuration of a client system based on the available printer objects in the Directory Server. The printer configurator provides the printer configuration management; it verifies if the printer configuration has any conflict with the LP printer configurations in the client system before it actually adds or deletes a printer.

Following are five examples to show how the LDAP printer configurator provides central management of printer services based on the printer objects stored in the Directory Server:

Example 1:

An administrator sets up a new printer located in the Engineering Lab and wants this printer to be shared. This printer is physically connected to a system `hostA` and is set up as a local printer `lj2004`. The administrator creates a new printer entry in the directory server as follow:

```
dn: printer-name=laser2,ou=printers,dc=hp,dc=com
printer-name: laser2
printer-uri: lpd://hostA.hp.com/lj2004
```

A new printer configuration for `laser2` is created automatically in every client system if the LDAP printer configurator is running. The print queue for `laser2` is enabled and ready to accept print jobs. Users can sent their print jobs to `laser2` by typing `lp -dlaser2` filename.

Example 2:

IT department would like to store additional service information in the printer object. The administrator modifies the printer object by adding more printer attributes. The modified content of the printer object is shown as below:

```
dn: printer-name=laser2,ou=printers,dc=hp,dc=com
printer-name: laser2
printer-uri: lpd://hostA.cup.hp.com/lj2004
printer-location: Engineering Lab
printer-model: Hewlett Packard laserjet Model 2004N
printer-service-person: David Lott
```

Since the local printer name, remote hostname, remote printer name, and the printing protocol information are still the same, the LDAP Printer Configurator will not change the current remote LP printer configuration for `laser2`.

Example 3:

The system `hostA.hp.com` is retired. The Laserjet 2004 printer is now connected to system `hostC` and set up as a local LP printer `lj2004`. The administrator should update the printer object by changing the value in `printer-uri` attribute. The following shows the updated information of print objects:

```
dn: printer-name=laser2,ou=printers,dc=hp,dc=com
printer-name: laser2
printer-model: Hewlett Packard laserjet Model 2004N
printer-service-person: David Lott
```

The current remote LP `laser2` printer configuration is removed from the client system, and the new `laser2` printer configuration with new remote hostname information is added to the client system. In fact, if either remote hostname or remote printer name of `printer-uri` attribute is modified, the printer configurator will remove the current remote LP printer configuration and create the new printer configuration with the updated resource information.

Example 4:

The remote LP printer, `laser2`, no longer supports LPD printing protocol. IPP printing protocol is implemented instead. The administrator updated the printer object by changing the printing protocol to IPP. The following shows the updated printer objects in the directory server:

```
dn: printer-name=laser2,ou=printers,dc=hp,dc=com
printer-name: laser2
printer-uri: ipp://hostC.hp.com/lj2004
printer-location: Engineering Lab
printer-model: Hewlett Packard laserjet Model 2004N
printer-service-person: David Lott
```

IPP printing protocol is not supported by the LP spool printing system. The only action that the LDAP printer configurator will take is to remove the current `laser2` printer configuration on the client system.

Example 5:

The administrator created a new printer object in the directory server as below:

```
dn: printer-name=laser8,ou=printers,dc=hp,dc=com
printer-name: laser8
printer-uri: lpd://hostD.hp.com/lj81003
```

In this example, the printer configurator adds a new remote LP `laser8` printer configuration to the client system.

However, if the user attempts to remove the `laser8` printer configuration manually, the printer configuration will no longer be managed by the printer configurator. The user has to recreate the printer configuration manually in case the laser8 printer is needed. The printer configurator does not try to create the printer configuration even though the printer object of `laser8` still exists in the directory server.

If the user manually adds a remote LP printer configuration to the client system, the new printer configuration will not be managed by the printer configurator. The user has to remove the printer configuration manually if the remote LP printer is no longer needed.

Limitations of Printer Configurator

- The new LDAP printer schema based on *IETF<draft-fleming-ldap-printer-schema-02>* is imported into the LDAP Directory Server to create the printer objects.
- LDAP-UX Client Services only suports the HP-UX LP spooler system, network printers, and printerservers that support Line Printer Daemon (LPD) protocol. The printer configurator does not support local printers.
- In a global management envoriment, it is hard to determine a default printer for the individual client system. The LDAP printer configurator treats every printer entry as the regular printer. The administrator or user requires to manually select a printer as a default printer for the client system.

4 Dynamic Group Support

This chapter contains information about how LDAP-UX Client Services supports dynamic groups, how to set up dynamic groups, and how to enable or disable dynamic group caches. This chapter includes the following sections:

Overview

A system administrator can associate some users with a group, and apply security policies (e.g. access control, password policies) to the group. As a result, all users belonging to the group inherit the specific policies, such as being able to access a file. In LDAP directories, there are two types of groups: static groups and dynamic groups. A static group defines all users statically. Each user must be added to the group individually and explicitly. Dynamic groups associate users with a group based on conditions. The condition can be specified by an LDAP URL or a search filter. When a user's data matches with the conditions, she/he belongs to the dynamic group. Dynamic groups offer the advantage of flexibility, and allow administrators to easily implement a role-based authorization policy based upon a company's organizational structure. Users can be added to or removed from a group dynamically based on his/her most current status (such a value of one or more attributes in the user's entry).

Since traditional POSIX-style groups are used largely to control file system access rights, dynamic groups in LDAP-UX offers a new and flexible method for defining file system access policies. For example, with file system access control lists (ACLs) it is possible to add group access permission for users that are a member of a particular group (say the "top secret" group). With dynamic groups, instead of needing to insert each individual member in the group, LDAP-UX discovers all users in the directory that have the "top secret" attribute associated with their entries. And when a user's attribute is no longer defined as "top secret", his/her group membership in the "top secret" is automatically revoked (no need to make manual changes to the group).

LDAP-UX Client Services B.04.10 or later supports dynamic groups and allows you to configure dynamic groups using the same syntaxes as the following directory servers and identity management:

- Netscape/Red Hat Directory Server
- Windows 2003 and 2003 Release 2 (R2) Active Directory Server
- HP Select Access and HP-UX Select Access for IdMI

Specifying an LDAP URL for a Dynamic Group

Netscape/Red Hat Directory Server defines the `memberURL` attribute and the `groupOfURLs` objectclass to represent the dynamic group. All POSIX users who can be found using the LDAP URL belong to the group.

Creating an HP-UX POSIX Dynamic Group

LDAP-UX Client Services only supports HP-UX POSIX dynamic groups. Use the following procedures to create an HP-UX POSIX dynamic groups:

1. Use the Directory Server Console to create a dynamic group. See the “Step1: Creating a Dynamic Group” section for details.
2. Add the `posixgroup` objectclass and `gidNumber` attribute information to the dynamic group entry created in step 1. See the “Step 2: Adding POSIX Attributes to a Dynamic Group” for details.

Step 1: Creating a Dynamic Group

You can use the Directory Server Console to create a dynamic group. For detailed information on how to use the Directory Server Console to create a dynamic group, refer to Chapter 5 “Advanced Entry Management” of the *Red Hat Directory Server Administrator's Guide* available at the following web site:

http://docs.hp.com/en/internet.html

The following shows an example of a dynamic group entry created using the Directory Server Console:

```
dn: cn=dyngroup,ou=groups,dc=example,dc=hp,dc=com

cn=dyngroup

objectClass: top

objectClass: groupofuniquenames

objectClass: groupofnames
```

objectClass: groupofurls

memberURL: ldap:///dc=example,dc=hp,dc=com??sub?(l=California)

The `memberURL` attribute in the above example specifies a sub-tree search starting at any level under dc=example, dc=hp, dc=com to find all entries matching (l=California). Any entries which have objectclass “`account`” and an attribute “`l`” with the value of “`California`” will be returned. With LDAP-UX, an additional criteria will be added that the user entry must be a POSIX account.

Step 2: Adding POSIX Attributes to a Dynamic Group

To create an HP-UX POSIX dynamic group, you must use the Directory Console, or the `ldapmodify` tool to add the following objectclass and attribute information to the dynamic group entry created in Step 1: Creating a Dynamic Group:

- `posixgroup` objectclass
- `gidNumber` attribute
- `cn` attribute if it does not exist in the group entry.

Adding Attributes to a Dynamic Group Using ldapmodify

Procedures

As an example, to create an HP-UX POSIX dynamic group, use the `ldapmodify` tool to add `posixgroup` and `gidNumber` information to the dynamic group entry created from the Directory Server Console as follows:

1. Create an LDIF update file.

   For example, the following LDIF update file, `new.ldif`, adds a `posixgroup` objectclass and the `gidNumber` attribute to the “`dn: cn=dyngroup,ou=groups,dc=example,dc=hp,dc=com`” entry:

   ```
   dn: cn=dyngroup,ou=groups,dc=example,dc=hp,dc=com
   changetype: modify
   add: objectClass
   objectClass: posixgroup
   -
   ```

```
add: gidNumber
gidNumber: 500
```

2. Use the `ldapmodify` tool to modify the existing entry with the LDIF file created in step 1.

   For example, the following command modifies the dynamic group entry in the LDAP directory server, `ldaphost1`, using the LDIF update file, `new.ldif`:

   ```
   ldapmodify –D “cn=Directory Manager" –w <passwd> –h ldaphost1 –p
   389 –f new.ldif
   ```

Examples

The following example is an HP-UX POSIX dynamic group entry with `objectClass: posixgroup` and `gidNumber: 500` information added:

`dn: cn=dyngourp,ou=groups,dc=example,dc=hp,dc=com`

`objectClass: groupofuniquenames`

`objectClass: groupofnames`

`objectClass: groupofurls`

objectClass: posixgroup

`objectClass: top`

`cn: dyngroup`

`memberURL: ldap:///dc=example,dc=hp,dc=com??sub?(l=California)`

gidNumber: 500

Changing an HP-UX POSIX Static Group to a Dynamic Group

To change an HP-UX POSIX static group to an HP-UX POSIX dynamic group, use the Directory Server Console to add the following objectclass and attribute information to the HP-UX POSIX static group:

- `groupofurls` objectclass
- `memberURL` attribute

For detailed information on how to use the Directory Server Console to modify a group, refer to *Red Hat Directory Server Administrator's Guide* available at the following web site:

http://docs.hp.com/en/internet.html

The following shows an example of an HP-UX POSIX static group entry:

```
dn: cn=all,ou=groups,dc=example,dc=hp,dc=com
objectClass: groupofuniquenames
objectClass: groupofnames
objectClass: posixgroup
objectClass: top
cn: all
gidNumber: 1000
memberuid: user1
```

After you add information for `groupofurls` and `memberURL` to the above HP-UX POSIX static group entry, the HP-UX POSIX dynamic group entry is as follows:

`dn: cn=all,ou=groups,dc=example,dc=hp,dc=com`

`objectClass: groupofuniquenames`

`objectClass: groupofnames`

objectClass: groupofurls

`objectClass: posixgroup`

`objectClass: top`

`cn: all`

memberURL: ldap:///dc=example,dc=hp,dc=com??sub?(l=California)

`gidNumber: 1000`

`memberuid: user1`

Now, the group "`all`" contains both static group member (i.e. user1) and dynamic members (i.e. all user entries which can be retrieved from the tree of `dc=example,dc=hp,dc=com` and have an attribute with `l=California`).

Specifying a Search Filter for a Dynamic Group

Instead of using memberURL and groupofurls to specify dynamic groups, HP OpenView Select Access and HP-UX Select Access for IdMI define the following new attributes and objectclass to support dynamic groups: .

- nxRole attribute
- nxSearchBaseDn attribute
- nxSearchFilter attribute
- nxSearchScope attribute
- nxRoleEntry objectclass

Creating an HP-UX POSIX Dynamic Group

Each dynamic group is configured with a search DN, search scope and search filter. LDAP-UX can support dynamic groups created by HP OpenView Select Access and HP-UX Select Access for IdMI if they are POSIX dynamic groups. Use the following procedures to create an HP-UX POSIX dynamic group:

1. Use the Select Access Policy Builder to create a dynamic group. See the "Step 1: Creating a Dynamic Group" section for details.
2. Add the posixgroup objectclass, gidNumber and cn attribute information to the dynamic group entry created in step 1. See the "Step 2: Adding POSIX Attributes to a Dynamic Group" for details.

Step 1: Creating a Dynamic Group

You can use the Select Access Policy Builder to create dynamic groups. For detailed information on how to use the Select Access Policy Builder to create a dynamic group, refer to the *Select Access Policy Builder Guide*. The *Select Access Policy Builder Guide* can be found in the /opt/OV/SelectAccess/docs directory after you install the HP-UX Select Access for IdMI product, SelectAccessIdMI.

The HP-UX Select Access for IdMI product can be downloaded from the following web site:

http://www.hp.com/go/softwaredepot

The following shows an example of a dynamic group entry:

```
dn: nxRole=Austine Managers,ou=groups,ou=Managing,dc=Example,dc=hp,dc=com
objectClass: nxRoleEntry
objectClass: top
nxSearchScope: sub
nxSearchBaseDn: ou=Managing,dc=Example,dc=hp,dc=com
nxRole: Austine Managers
nxSearchFilter: (l=Austine)
```

Step 2: Adding POSIX Attributes to a Dynamic Group

To create an HP-UX POSIX dynamic group, you can use the Directory Server Console or the ldapmodify tool to add information for the posixgroup objectclass, the gidNumber and cn attributes to the dynamic group entry created from Select Access Policy Builder. For more information on how to add attribute information to the dynamic group using ldapmodify, see the "Procedures" section in ""Adding Attributes to a Dynamic Group Using ldapmodify " (page 78).

Examples

The following shows an example of an HP-UX POSIX dynamic group entry with posixgroup, gidNumber and cn information added:

```
dn: nxRole=Austine Managers,ou=groups,ou=Managing,dc=Example,dc=hp,dc=com
```

```
objectClass: nxRoleEntry
objectClass: posixgroup
objectClass: top
nxSearchScope: sub
nxSearchBaseDn: ou=Managing,dc=Example,dc=hp,dc=com
nxRole: Austine Managers
nxSearchFilter: (l=Austine)
cn: AustMgrs
gidNumber: 2000
```

NOTE: Unlike Netscape/Red Hat Directory dynamic groups, Select Access dynamic groups require non-standard objectclass and attributes. You cannot change existing POSIX static groups to Select Access POSIX dynamic groups without importing those objectclass and attributes. This procedure is not supported.

Multiple Group Attribute Mappings

By default, LDAP-UX uses the `memberUid` attribute to retrieve group members. With the support of X.500 group member syntax, you can map the default group attribute, `memberUid`, to `member` or/and `uniquemember`, which you specify group members using user DNs. With dynamic group support, LDAP-UX allows you to map `memberUid` to `memberURL` (if you use Netscape/Red Hat Directory Server to create dynamic groups) or/and `nxSearchFilter` (if you use HP OpenView Select Access or HP-UX Select Access for IdMI to create dynamic groups).

You can run the setup tool and map `memberUid` to multiple attributes as needed. For example, the following output of `/opt/ldapux/config/display_profile_cache` shows that `memberUid` is mapped to both static group attributes, `memberUid`, `member` and `uniquemember`, and dynamic group attributes, `memberURL` and `nxSearchFilter`:

```
Group Service Configuration:

      Attribute:             is mapped to:
      -----------            -------------
      name:                       cn
      gid:                        gidnumber
      members:                    memberuid memberURL nxSearchFilter
                                  member uniquemember
```

LDAP-UX retrieves group members and processes groups that a specific user belongs to by looking into all configured attributes. If needed, you can create a group which include both static and dynamic members. When returning group members, LDAP-UX will return both static and dynamic members that belong to a specific group.

When processing dynamic group attributes, LDAP-UX combines the search filter of the passwd service from the profile with the search filter specified in `membeURL` (e.g. the last component in `memberURL`) or `nxSearchFilter` to retrieve group members. This is to make sure that group members returned are POSIX accounts and meet the configuration set for LDAP-UX.

Examples

The following is an example of the output of `/opt/ldapux/config/display_profile_cache`:

```
PASSWD Service Configuration

       Attribute:            is mapped to:
       ----------            --------------
```

```
          name:                     uid
          uid number:               uidnumber
          .....

          Search Descriptor
          search[0]:                dc=example,dc=hp,dc=com?sub?
                                    (objectclass=posixaccount)
```

The sample group entry is:

```
dn: cn=mygroup,ou=Groups,dc=example,dc=hp,dc=com
objectClass: groupofnames
objectClass: groupofuniquenames
objectClass: posixgroup
objectClass: groupofurls
objectClass: top
cn: mygroup
gidNumber: 100
memberUid: user1
member: uid=user2,ou=people,dc=example,dc=hp,dc=com
uniqueMember: uid=user3,ou=people,dc=example,dc=hp,dc=com
memberURL: ldap:///dc=example,dc=hp,dc=com??sub?(uid=p*)
```

When processing `memberURL` to retrieve dynamic members, LDAP-UX combines `(objectclass=posixaccount)` from passwd configuration with `(uid=p*)` as the search filter to search the tree of "`dc=example,dc=hp,dc=com`".

With the above attribute mappings, LDAP-UX will return `user1`, `user2`, `user3` and all users starting with "p" as group members.

Group Attribute Mappings

To enable the dynamic group feature support, you must run the setup program to remap the default group attribute, `memberuid`, to the dynamic group attribute, `memberURL` and/or `nxSearchFilter`. If neither `memberURL` nor `nxSearchFilter` is mapped to `memberUid`, LDAP-UX will not process dynamic groups.

The attribute mappings are done in step 11 of the Custom Configuration. For detailed information on how to remap the group attributes, see "Custom Configuration" (page 38).

Table 4–1shows attribute mappings between the default group attribute and alternate group attributes:

Table 4-1 Attribute Mappings

Default Group Attribute	Dynamic Group Attribute	Static X.500 Group Attribute
`memberuid`	`memberURL` or `nxSearchFilter`	`member` or `uniquemember`

If you want to perform group attribute mappings by using the Custom Configuration, ensure that you do not accept the remaining default configuration parameters in step 5 of the Custom Configuration.

Number of Group Members Returned

With dynamic membership support, as with regular (static) group membership support, the number of group members for a specific group returned by getgrnam()/getgrgid()/getgrent() on an HP-UX system is limited by internal buffer sizes. On HP-UX 11i v1 and v2 systems, the buffer size is 7296 bytes for 32bit applications and 10496 bytes for 64bit applications. This limitation is mainly impacted by the size of each member name. For detailed description, refer to the *Preparing your Directory for LDAP-UX Integration* white paper under the "Account and Group Management" collection available at the following web site:

http://docs.hp.com/en/internet.html

During the login process, information for getting group members is not requested. The login time will not be affected by processing group members.

Number of Groups Returned for a Specific User

When “ldap” is configured in the /etc/nsswitch.conf file as a data repository for the group service (see nsswitch.conf(4)), if an LDAP user logs into an HP-UX system, LDAP-UX is involved to return all groups that the user belongs to. The login application (e.g. login) initializes the user's group access based on the group information returned by LDAP-UX.

Information for getting groups that a specific user belongs to is requested by LDAP-UX during login via initgroups(). LDAP-UX returns at most 20 groups for a system limit on HP-UX 11i v1 and v2 systems. If the user belongs to more than 20 groups, only the first 20 groups are returned. The support of dynamic groups does not change the system limitation.

Depending on how you configure groups, if those 20 groups happens to be the last entries of thousands of dynamic groups, the login time could be long and performance could be impacted.

Based on the configuration of memberUid attribute mappings, LDAP-UX may return static and/or dynamic groups. The first memberUid mapped attribute determines if LDAP-UX returns static or dynamic groups first. If the first memberUid mapped attribute is a static group attribute (such as memberUid, member or uniquemember), LDAP-UX returns static groups first. If there are less than 20 static groups, LDAP-UX then returns dynamic groups for the rest groups. However, if the first memberUid mapped attribute is a dynamic group attribute (such as memberURL or nxSearchFilter), LDAP-UX returns dynamic groups first. If there are less than 20 dynamic groups, LDAP-UX then returns static groups for the rest groups.

With this design, a group containing both static and dynamic group attributes will be always processed, but will be limited to the first 20 groups.

For example, if a user belongs to 8 static groups and 20 dynamic groups, and you map memberUid to memberUid memberURL, LDAP-UX will return 8 static groups and 12 dynamic groups. If you map memberUid to memberURL memberUid, LDAP-UX will return 20 dynamic groups without any static groups.

Performance Impact for Dynamic Groups

The dynamic group is specified by either an LDAP URL or a search filter. Depending on how you configure dynamic groups, potentially, there could be a lot of LDAP searches involved. In that case, the performance of those applications calling `getgrnam()`, `getgrgid()` or `getgrent()(3C)` (e.g. the command "id", "groups", etc) will be affected.

In order to reduce the performance impact, LDAP-UX Client daemon, `ldapclientd,`, caches dynamic group information, including dynamic members that belongs to a specific group, and dynamic groups that a specific user belongs to. The caching will reduce the response time the `ldapclientd` daemon to return information. However, before the cache is established (i.e. the very first request) or when the cache expires, you may experience longer response time. See the "Configuring Dynamic Group Caches" (page 85) section for detailed information on dynamic group caching.

Enabling/Disabling enable_dynamic_getgroupsbymember

Processing dynamic groups that a specific user belongs to can potentially impact the user login time. To control the operation for processing dynamic groups a specific user belongs to, LDAP-UX Client Services supports the following configuration parameter, `disable_dynamic_getgroupsbymember`, in the `/etc/opt/ldapux/ldapux_client.conf` file:

enable_dynamic_getgroupsbymember

This integer variable controls whether to enable or disable the operation for processing dynamic groups that a specific user belongs to. The valid values of this option are 1 and 0.

By default, LDAP-UX returns dynamic groups that a user belongs to if the group attribute, `memberUid`, is mapped to `memberURL` or/and `nxSearchFilter`. If a user belongs to many dynamic groups, he/she may experience an unexpected delay when logging into an HP-UX client system.You can reduce the delay by disabling LDAP-UX of returning dynamic groups that a specific user belongs to unless he/she specifically uses the `newgrp` command. As a result, the user will not have access granted to those dynamic groups, and the "`id`" command will not show those groups. To disable it, set `enable_dynamic_getgroupsbymember` to 0. This parameter configuration does not affect the operation of processing dynamic members for a specific group. The default value is 1 to enable it.

NOTE: If the `enable_dynamic_getgroupsbymember` variable is set to 0, LDAP-UX will still return dynamic members for a specific group. If you don't want dynamic members returned, you must not include the `memberURL` and `nxSearchFilter` attributes in the `memberUid` group attribute mapping, which completely disables the dynamic group functionality with LDAP-UX.

Configuring Dynamic Group Caches

To improve performance of dynamic groups, the ldapclient daemon, `ldapclientd`, caches dynamic group members to reduce the LDAP-UX client response time while retrieving dynamic group information. This cache is maintained in an independent memory space not shared with the cache for other service data.

To configure dynamic group caches, set the parameters defined in the `[dynamic_group]` section of the `/etc/opt/ldapux/ldapclientd.conf` file. See "ldapclientd.conf" (page 89) in the "Administering LDAP-UX Client Services" chapter for details.

5 Administering LDAP-UX Client Services

This chapter describes how to keep your clients running smoothly and expand your computing environment. It describes the following topics:

Using The LDAP-UX Client Daemon

This section describes the following:

- Overview of ldapclientd daemon operation.
- Configurable parameters and syntax in the `ldapclientd` configuration file, `ldapclientd.conf`.
- Command line syntax and options for the `ldapclientd` command.

Overview

The LDAP-UX client daemon enables LDAP-UX clients t o work with LDAP directory servers. It caches entries, supports multiple domains in the Windows 2000/2003 Active Directory Server (ADS), supports X.500 group membership, automatically downloads the configuration profiles, reuses connections to the LDAP Directory Server, and manages the remote LP printer configuration.

The client daemon enables LDAP-UX to use multiple domains for directory servers like Active Directory Server (ADS). The daemon also allows PAM Kerberos to authenticate posix users stored in multiple domains.

Automatic Profile Downloading updates the LDAP client configuration profile by downloading a newer copy from the directory server as the profileTTL (Time To Live) expires.

By default, the LDAP printer configurator is enabled, the client daemon, *ldapclientd*, automatically searches printer objects configured in the LDAP server and executes `lpshut`, `lpadmin` and `lpsched` commands to add, modify, and remove printers accordingly for the local system.

By default, `ldapclientd` starts at system boot time. The ldapclientd command can also be used to launch the client daemon manually, or control it when the daemon is already running. Please refer to the following section and the `ldapclientd` man page(s) for information about the `ldapclientd` command and its parameters.

IMPORTANT: Starting with LDAP-UX Client Services B.03.20 or later, the client daemon, `/opt/ldapux/bin/ldapclientd`, must be running for LDAP-UX functions to work. With LDAP-UX Client Services B.03.10 or earlier, running the client daemon, `ldapclientd`, is optional.

ldapclientd

Starting the client

Use the following syntax to start the client daemon. Note the use of upper and lower-case characters:

```
/opt/ldapux/bin/ldapclientd <[-d <level>] [-o<stdout|syslog|file[=size]>]
[-z]
```

Controlling the client

Use the following syntax to control the client daemon:

```
/opt/ldapux/bin/ldapclientd <[-d <level>]
[-o<stdout|syslog|file[=size]>]>
```

```
/opt/ldapux/bin/ldapclientd <[-D <cache>]|-E <cache>|-S [cache]>
```

```
/opt/ldapux/bin/ldapclientd <-f| -k| -L| -h| -r>
```

Client Daemon performance

Performance (client response time) is improved by the use of two techniques:

1. Reuse of connections to the LDAP Directory Server: This feature improves performance by reducing the overhead associated with opening and closing bindings to the directory server and significantly reduces network traffic and server load.
2. Enabling the client cache: Enabling the cache will allow the client to cache the reply information retrieved for the following maps:

   passwd
   group
   dynamic group
   netgroup
   X.500 group membership
   automount

Except for the dynamic group map, all of the above maps share a common memory space. The Dynamic Group map cache is created as an independent memory space. The length of time the reply data is held in the cache is determined by a Time To Live timer. This timer can be set for all maps or can be set independently for each of the maps listed above. The cache can also be flushed by specifying an option on the ldapclientd command. The cache space becomes available for new information after the Time to Live expires or the cache is flushed.

There are two categories of information that are held in the cache. The reply data for those requests that were successful, and replies when the information was not found. For example, when a specific user is trying to logon, the userID may or may not exist in the directory.

The Time to Live for replies that were found in the directory is set by a parameter `poscache_ttl` in the `ldapclient.conf` file and for replies where the information was not found by `negcache_ttl`.

For more information on the client daemon performance, see Client Daemon Performance (page 120).

Command options

Please refer to the ldapclientd man page(s) for option information.

Diagnostics

By default, errors are logged into syslog if the system log is enabled in the LDAP-UX client startup configuration file */etc/opt/ldapux/ldapux_client.conf*. Errors occuring before `ldapclientd` forks into a daemon process leaves an error message directly on the screen.

The following diagnostic messages may be issued:

Message: Already running.

Meaning: An attempt was made to start an LDAP Client Daemon when one was already running.

Message: Cache daemon is not running (or running but not ready).

Meaning: This message can mean several things:

1. Attempted to use the control option features of `ldapclientd` when no `ldapclientd` daemon process was running, to control.
2. Attempted to start, or control, `ldapclientd` without superuser's privilege.
3. The `ldapclientd` daemon process is too busy with other requests to respond at this time. Try again later.

Message: Problem reading configuration file.

Meaning: The */etc/opt/ldapux/ldapclientd.conf* file is missing or has a syntax error. If the problem is with its syntax, the error message will be accompanied by a line showing exactly where it could not recognize the syntax, or where it found a setting which is out of range.

Warnings

Whenever the system is rebooted, `ldapclientd` launches if `[StartOnBoot]` has the parameter `enabled=yes` in the file */etc/opt/ldapux/ldapclientd.conf* (the `ldapclientd` configuation file). Downloading profiles takes time, depending on the server's response time and the number of profiles listed in the LDAP-UX startup file */etc/opt/ldapux/ldapux_client.conf*.

ldapclientd.conf

The file *ldapclientd.conf* is the configuration file for */opt/ldapux/bin/ldapclientd*, the LDAP Client Daemon. Refer to the previous section for more information about the Client Daemon.

Missing settings

`ldapclientd` uses the *default values* for any settings which are not specified in the configuration file.

Configuration file syntax

```
# comment
[section]
setting=value
setting=value
. . .
[section]
setting=value
setting=value
. . .
```

Where:

`comment` ldapclientd ignores any line beginning with a # delimiter.

`section` Each section is configured by setting=value information underneath. The section name must be enclosed by brackets ("[ ]") as delimiters. Valid section names are:

- [StartOnBoot]

- [general]

- [passwd]

- [group]

- [dynamic_group]

- [netgroup]

- [uiddn]

- [domain_pwd]

- [domain_grp]

- [automount]

-[automountMap]

- [printers]

`setting` This will be different for each section.

`value` Depending on the setting, this can be `<yes|no|number>`.

Section details

Within a section, the following syntax applies:

[StartOnBoot] Determines if `ldapclientd` starts automatically when the system boots.

enable=<yes|no>

By default, this is enabled after LDAP-UX has been configured by the LDAP-UX setup program */opt/ldapux/config/setup*.

[general] Any cache setting defined here will be used as the default setting for all caches (passwd, group, netgroup, uiddn, domain_pwd, domain_grp, automount, automountmap, and dynamic_group). The cache_size setting defined here will be used for all caches except dynamic_group.

max_conn=<2-500>

The maximum number of connections `ldapclientd` can establish to the directory server (or multiple servers when in a multi-domain environment.

The default value is 100.

connection_ttl=<1-2147483647>

The number of seconds before an inactive connection to the directory server is brought down and cleaned up.

The default value is 300.

num_threads=<1-100>

The number of client request handling threads in `ldapclientd`.

The default value is 10.

socket_cleanup_time=<10-2147483647>

The interval, in seconds, before the next attempt to clean up the socket files created by any LDAP-UX client applications that were terminated abnormally.

The default value is 300.

cache_cleanup_time=<1-300>

The interval, in seconds, between the times when `ldapclientd` identifies and cleans up stale cache entries.

The default value is 10.

update_ldapux_conf_time=<10-2147483647>

This determines how often, in seconds, `ldapclientd` re-reads the */etc/opt/ldapux/ldapux_client.conf* client configuration file to download new domain profiles.

The default value is 600 (10 minutes).

cache_size=<102400-1073741823>

The maximum number of bytes that should be cached by `ldapclientd` for all services except dynamic_group. This value is the maximum, upper limit, of memory that can be used by `ldapclientd` for all services except dynamic_group. If this limit is reached, new entries are not cached until enough expired entries are freed to allow it.

The default value is 10000000.

state_dump_time=<0-2147483647>

As state, functions like a virtual between the client and LDAP server, is created for `setXXent()` request, and stays for the subsequent `getXXent()` requests. If no `get` requests are received in the specified time interval (in seconds), the state will be removed. The default value is 300 (in seconds).

max_enumeration_states=<0-95>[%]

The maximum number of states that `ldapclientd` allows. It means the number of enumeration `ldapclientd` will handle simultaneously. This number must be less than max_conn and it is configured as a percentage of max_conn. The minimum value is 0% and maximum value is 95%. The default value is 80%. A value of 0% disables enumeration.

poscache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. There is no `[general]` default value for this setting. Each cache section has its own default values (listed below). Specifying a value under `[general]` will override `poscache_ttl` defaults in other sections (where there is no specific `poscache_ttl` definitions for that section).

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache. There is no `[general]` default value for this setting. Each cache section has its own default value.

[passwd]

Cache settings for the `passwd` cache (which caches name, uid and shadow information).

enable=<yes|no>

`ldapclientd` only caches entries for this section, when it is enabled. If the cache is not enabled, `ldapclientd` will query the directory server for any entry request from this section. Since this impacts LDAP-UX client performance and response time, by default, caching is enabled.

poscache_ttl=<0-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. Since personal data can change frequently, this value is typically smaller than some others.

The default value is 120 (2 minutes)

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache.

The default value is 240 (4 minutes).

[group]

Cache settings for the group cache (which caches name, gid and membership information).

enable=<yes|no>

`ldapclientd` only caches entries for this section, when it is enabled. By default, caching is enabled.

poscache_ttl=<0-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. Since people are added and removed from groups occasionally, this value is not typically large. If dynamic_group caching is enabled, this value must be less than `poscache_ttl` of `[dynamic_group]`.

The default value is 240 (4 minutes)

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache. If dynamic_group caching is enabled, this value must be less than `negcache_ttl` of `[dynamic_group]`

The default value is 240 (4 minutes).

[dynamic_group]

This section describes the settings for the Dynamic Group cache. This cache manages dynamic group information including name, group ID and membership information. This cache is maintained in a independent memory space not shared with the cache for other maps.

enable=<yes|no>

`ldapclientd` only caches entries for this section, when it is enabled. Since this impacts LDAP-UX client performance and response time, caching is enabled by default.

poscache_ttl=<0-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. If group caching is enabled, this value must be greater than `poscache_ttl` of `[group]`.

The default value is 43200 (12 hours).

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache. If group caching is enabled, this value must be greater than `negcache_ttl` of `[group]`

The default value is 43200 (12 hours).

cache_size=<102400–1073741823>

This integer variable specifies the maximum number of bytes that should be cached by `ldapclientd`. This value is the maximum, upper limit, of memory that can be used by `ldapclientd`. If this limit is reached, new entries are not cached until enough expired entries are freed to allow it. The default value is 100000000 (10M).

NOTE: The cache_size option defined in the [general] section is used to configure for all other caches (passwdm netgroup, group, outomount, domain_pwd, domain_grp, uiddn).

[netgroup]

Cache settings for the netgroup cache.

enable=<yes|no>

`ldapclientd` only caches entries for this section, when it is enabled. By default, caching is enabled.

poscache_ttl=<0-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. Since people are added and removed from groups occasionally, this value is not typically large.

The default value is 240 (4 minutes)

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache.

The default value is 240 (4 minutes).

[uiddn]

This cache maps a user's UID to their DN from the directory.

enable=<yes|no>

`ldapclientd` only caches entries for this section, when it is enabled. By default, caching is enabled.

poscache_ttl=<0-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. Typically, once added into a directory, the user's DN rarely changes.

The default value is 86400 (24 hours).

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache.

The default value is 84400 (24 hours).

[domain_pwd]

This cache maps user names and UIDs to the domain holding its entry.

enable=<yes|no>

`ldapclientd` only caches entries for this section, when it is enabled. By default, caching is enabled.

poscache_ttl=<0-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. Since new domains are rarely added to or removed from the forest, the cache is typically valid for a long time.

The default value is 86400 (24 hours)

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache.

The default value is 86400 (24 hours).

[domain_grp]

This cache maps group names and GUIDs to the domain holding its entry.

enable=<yes|no>

`ldapclientd` only caches entries for this section, when it is enabled. By default, caching is enabled.

poscache_ttl=<0-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. Since new domains are rarely added to or removed from the forest, the cache is typically valid for a long time.

The default value is 86400 (24 hours).

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache.

The default value is 86400 (24 hours).

[automount]

Cache settings for the automount entry cache (which caches automount entries in automount maps).

A positive cache means that the automount entry data has been recently retrieved from the LDAP directory server and is stored in the positive cache locally.

A negative cache is used to store the automount entry data about non-existent information. For example, if a user requests information about an automount entry that does not exist, the LDAP directory server will not return an entry, all the negative result will be stored in the negative cache.

enable=<yes|no>

`ldapclientd` only caches entries for this section, when it is enabled. By default, caching is enabled.

poscache_ttl=<0-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. The default value is 1800 (30 minutes).

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache.

The default value is 1800 (30 minutes).

[automountMap]

Cache settings for the automount map cache.

enable=<yes|no>

`ldapclientd` only caches entries for this section, when it is enabled. By default, caching is enabled.

poscache_ttl=<0-2147483647>

The time, in seconds, before a cache entry expires from the positive cache. The default value is 1800 (30 minutes).

negcache_ttl=<1-2147483647>

The time, in seconds, before a cache entry expires from the negative cache.

The default value is 7200 (2 hours).

[printers]

Any printer setting defined here will be used by the LDAP printer configurator.

start=<yes|no>

Determines if the printer configurator service will start when `ldapclientd` is initialized. If it is enabled, the printer configurator will start when `ldapclientd` is initialized. By default, the `start` parameter is enabled.

search_interval=<1800-1209600>

Defines the interval, in seconds, before the printer configurator performs a printer search in the directory server. The default value is 86400 (in seconds). The minimum value is 1800 (30 minutes) and the maximum value is 1209600 (2 weeks).

max_printers= 50 (default value)

Defines the maximum printer objects that printer configurator services will handle. For example, a number of 100 printer entries is returned to the printer configurator after a scheduled printer search. If the `max_printers` value is set to 50, only the first 50 printer entries received by the printer configurator will be processed. For this configuration parameter, the value must be greater than 0 and the maximum value is unlimited. The default value is 50.

lpadmin_option

Defines the `lpadmin` options. Do not include the`-p`, `-orm` and `-orp` options in the option fields. The LDAP printer configurator provides the required information of printer name (`-p`), remote machine name (`-orm`) and remote printer name (`-orp`) during the run time. Do not include any other parameters, such as stderr or stdout redirection options. If the option fields of the `lpadmin_option` parameter are empty or the `lpadmin_option` parameter does not exist, the default `lpadmin` options are used. By default, `lpadmin_option=-mrmodel -v/dev/null -ocmrcmodel -osmrsmodel`.

Refer to the `lpadmin` man page for detailed information about allowed options and the syntax.

Configuration File

The LDAP client configuration file is automatically loaded when the product is installed. Refer to the man page for additional information.

If you update LDAP-UX Client Services from an older version, such as B.03.00 or B.03.10, the new configuration file will be */opt/ldapux/newconfig/etc/opt/ldapux/ldapclientd.conf*.

Integrating with Trusted Mode

This section describes features and limitations, PAM configuration changes and configuration parameter for integrating LDAP-UX with Trusted Mode.

Overview

LDAP-UX Client Services B.03.30 or later supports coexistence with Trusted Mode. This means that local-based accounts can benefit from the Trusted Mode security policies, while LDAP-based accounts benefit from the security policies offered by the LDAP server. This release of LDAP-UX also enables LDAP-based and local-based accounts to be audited on the Trusted Mode.

The coexistence of LDAP-UX and Trusted Mode supports certain security features, but also has limitations and usage requirements that you need to be aware of. For detailed information, see Features and Limitations (page 96).

Features and Limitations

This subsection describes features and limitations of integrating LDAP-UX with Trusted Mode.

Auditing

Integrating LDAP-UX with Trusted Mode enables accounts stored in the LDAP directory to login to a local host and to be audited on the Trusted Mode. The following describes the auditing features and limitations. To use these security features, you must enable the audit subsystem on the Trusted Mode local host:

- Auditing of both LDAP-based and local-based (*/etc/passwd*) accounts is possible. By default, auditing is disabled for all LDAP-based accounts. However, you can use the `audusr` (option `-a` or `-d`) command to alter the auditing flag for individual LDAP-based account.
- For LDAP-based accounts that are not yet known to the system, you can configure an initial setting for the auditing flag. You can configure this flag such that when an account becomes known to the system for the first time, auditing for that account is immediately enabled or disabled. This flag is defined as the `initial_ts_auditing` parameter in the */etc/opt/ldapux/ldapux_client.conf* file.
- You must manage Trusted Mode attributes for all accounts on each host. Trusted Mode attributes for LDAP-based accounts are not stored in the LDAP directory server. For example, enabling auditing for an account on host A does not enable auditing on host B.
- Audit IDs for LDAP-based accounts are unique on each system. Audit IDs are not synchronized across hosts running in the Trusted Mode.
- When an LDAP-based account name is changed, a new audit ID is generated on each host that the account is newly used on. The `initial_ts_auditing` flag is reset to the default value defined in the */etc/opt/ldapux/ldapux_client.conf* file.
- When an account is deleted from LDAP, the audit information for that account is not removed from the local system. If that account is re-used, the audit information from the previous account is re-used. You can choose to manually remove entries from the Trusted Mode database by removing the appropriate file under the */tcb/files/auth/...* directory, where "..." defines the directory name based on the first character of the account name.
- You can use the `audisp` command to display information about LDAP-based accounts. However, if an LDAP-based account has never logged in to the system (via telnet, rlogin, and so on), the `audisp -u <username>` command displays the message like "`audisp: all specified users names are invalid.`"

Password and Account Policies

The primary goal of integrating Trusted Mode policies and those policies enforced by an LDAP server is coexistence. This means that Trusted Mode policies are not enforced on LDAP-based accounts, and LDAP server policies are not enforced on local-based accounts. The password and account policies and limitations are described as followings:

- Accounts stored and authenticated through the LDAP directory adhere to the security policies of the directory server being used. These policies are specific to the brand and version of the directory server product deloyed. Examples of these policies include password expiration, password syntax checking, and account expiration. No policies of the HP-UX Trusted Mode product apply to accounts stored in the LDAP server.
- When you integrate LDAP-UX on an HP-UX 11i v1 or 11i v2 system with the Netscape/Red Hat Directory Server, if an LDAP-based user attempts to login to the system, but provides the incorrect password multiple times in a row (the default is three times in a row), Trusted Mode attempts to lock the account. However, the Trusted Mode attributes do not impact

LDAP-based accounts. So, if the user eventually provides the correct password, he or she can login.

PAM Configuration File

- If you integrate LDAP-UX Client Services with the Netscape/Red Hat Directory Server, you must define the `pam_ldap`library before the `pam_unix` library in the */etc/pam.conf* file for all services. You must set the control flag for both pam_ldap and pam_unit libraries to `required` under session management. Refer to Sample /etc/pam.ldap.trusted file (page 189) for the proper configuration.
- If you integrate LDAP-UX Client Services with the Windows 2000/2003 Active Directory Server, you must define the `pam_krb5` library before the `pam_unix` library in the */etc/pam.conf* file for all services. In addition, the control flag for both `pam_krb5` and `pam_unix`libraries must be set to `required` for `Session management`. Refer to *Appendix F and Appendix G* on *LDAP-UX Client Services B.04.10 With Microsoft Windows Active Directory Administrator's Guide* for the proper configuration.

Others

- The `authck -d` command removes the `/tcb/files/auth/...` files created for LDAP-based accounts. When the LDAP-based account logs into the system again, a new`/tcb/files/auth/...` file with new audit ID is recreated. Therfore, it is not recommended to run the `authck -d` command when you configure LDAP-UX with Trusted Mode.
- You cannot use the Trusted Mode management subsystem in SAM to manage LDAP-based accounts.
- The LDAP repository and */etc/passwd* repository must not contain accounts with the same login name or account number.
- Except for the audit flag, you cannot modify other Trusted Mode properties/policies for LDAP-based accounts. For example, attempting to lock an LDAP-based account by modifying the Trusted Mode field for that user does not prevent that account from logging in to the host. Instead, you must disable the account on the LDAP server itself. No runtime warning will be given that the local locking of the account has no effect. It is important that all system administrators are properly trained, so that administrative locks on accounts have the desired effect.

Configuration Parameter

LDAP-UX Client Services provides one configuration parameter, `initial_ts_auditing`, available for you to configure the initial auditing setting for the LDAP-based account. This parameter is defined in the */etc/opt/ldapux/ldapux_client.conf* file.

PAM_AUTHZ Login Authorization

The Pluggable Authentication Module (PAM) is an industry standard authentication framework that is supplied as an integrated part of the HP-UX system. PAM gives system administrators the flexibility of choosing any authentication service available on the system to perform authentication. The PAM framework also allows new authentication service modules to be plugged in and made available without modifying the PAM enabled applications.

The PAM framework, together with the PAM_AUTHZ service module supplied with LDAP-UX Client Services, provide support for `Account Management` services. These services allow the administrator to control who can login to the system based on netgroup information found in the `/etc/passwd` and `/etc/netgroup` files. PAM and PAM_AUTHZ can also be configured to utilize LDAP-UX Client Services to retrieve the information from a LDAP directory server to perform access of authorization.

Starting LDAP-UX Client Services B.04.00, PAM_AUTHZ has been enhanced to provide administrators a simple security configuration file to set up a local access policy to better meet their need in the organization. PAM_AUTHZ uses the access policy to determine which users are allowed to login to the system. A policy specifies which groups, ldap groups, users or other access control objects (such as objects defined by ldap search filters) are allowed to login to the system. This flexibility enables you to allow or deny access to a host or application based on a user's membership in a group, or role within a organization. For example, PAM and PAM_AUTHZ can define an access rule that utilizes a LDAP directory server to state that if '`userA`' works for manager '`Sam`' then the criteria is met. When the rule is evaluated, a request would be sent to the LDAP directory and if the attributes were found, the user could be granted or denied access.

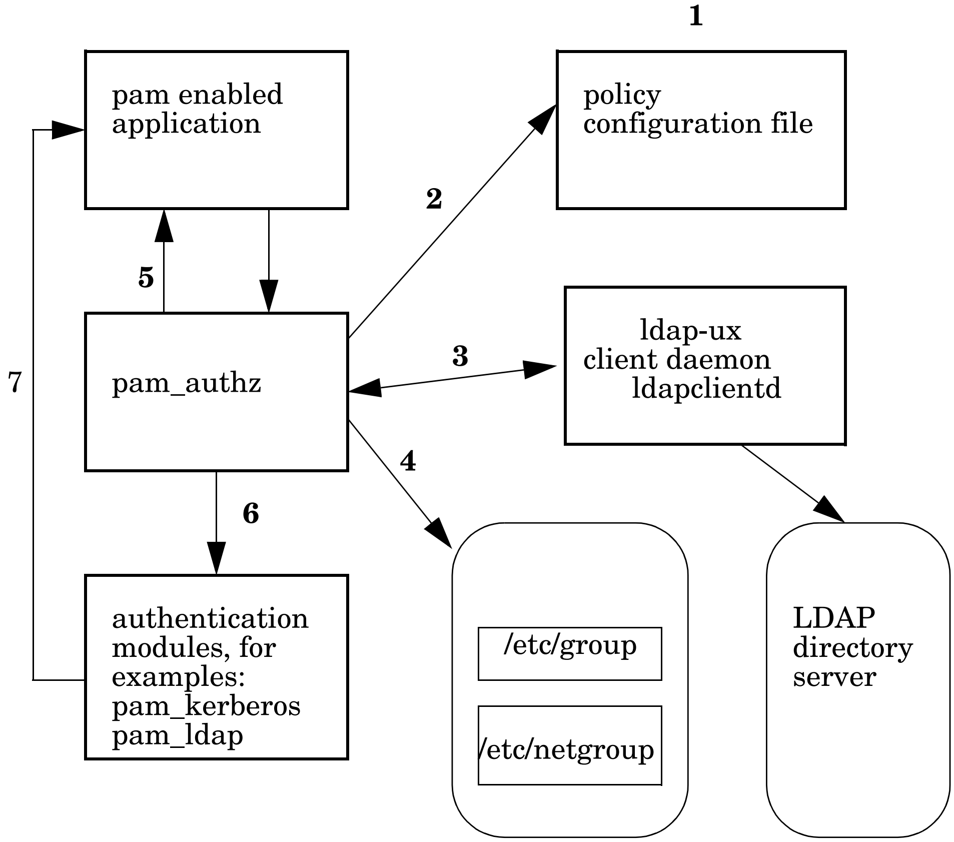
Policy And Access Rules

Access rules are the basic elements of access control. Administrators create access rules that restrict or permit a user's access permission. A policy is the collection of these different sets of access rules in a given order. This consolidated list of rules defines the overall access strategy of a local client machine. PAM_AUTHZ enables administrators to create an access policy by defining different types of access rules and to save the policy in a file.

How Login Authorization Works

The system administrator can define the access rules and store them in the policy file, `/etc/opt/ldapux/pam_authz.policy`. PAM_AUTHZ uses these access rules defined in the policy file to control the login authorization.

Figure 5-1 PAM_AUTHZ Environment


1
pam enabled application
policy configuration file
2
5
ldap-ux client daemon ldapclientd
3
7
pam_authz
4
6
authentication modules, for examples: pam_kerberos pam_ldap
/etc/group
/etc/netgroup
LDAP directory server

The following describes the policy validation processed by PAM_AUTHZ for the user login authorization shown in figure 5-1:.

PAM_AUTHZ Environment

1. The administrator defines a local policy file and saves all the defined access rules in the policy configuration file, `/etc/opt/ldapux/pam_authz.policy`.

2. PAM_AUTHZ service module receives an authentication request from PAM framework. It processes all the access rules stored in the /etc/opt/ldapux/pam_authz.policy file.

3. If a rule indicates that the required information is stored in a LDAP server, PAM_AUTHZ constructs a request message and sends to the LDAP client daemon, `ldapclientd`. The LDAP client daemon performs the actual ldap query and returns the result to PAM_AUTHZ. Then the access rule is evaluated and the final access right is returned.

4. If a rule indicates that the required information is in the UNIX files. PAM_AUTHZ retrieves user's information from `/etc/passwd`, `/etc/group` or `/etc/netgroup` file through `getpwname()` or `getgrname()` system calls. Then the rule is evaluated and the final access right is returned.

5. PAM_AUTHZ returns the corresponding pam result to PAM framework. The decision is returned to the application which called the PAM API.

6. If the user has the permission to login. then the decision is returned to the next PAM service module that is configured in `pam.conf` file, such as pam_ldap or pam_kerberos. If the user has no access right, then login is denied.

7. The PAM service module returns the authentication result to the application which called the PAM API.

PAM_AUTHZ Supports Security Policy Enforcement

PAM_AUTHZ supports enforcement of account and password policies, stored in an LDAP directory server. This feature works with SSH (Secure Shell), r-commands with rhost enabled where authentication is not performed via PAM (Pluggable Authentication Module) subsystem, but is performed by the command itself.

See the "Security Policy Enforcement with Secure Shell (SSH) or r-commands" (page 110) section for detailed information on how to configure access rules in the `/etc/opt/ldapux/pam_authz.policy` file, set global policy access permissions and configure the `pam.conf` file for security policy enforcement when using SSH key-pairs or r-commands.

Authentication using LDAP

The PAM framework is pluggable, the backend support for PAM's `Authentiaction`, `Account Management`, `Session Management` and `Password Management` services can be directed to an LDAP directory server. The LDAP-UX Client Services are plugged into the PAM framework by specifying the pam_ldap library, `libpam_ldap`, in the `/etc/pam.conf` configuration file. When the pam_ldap functions are invoked, the UNIX identity is translated into the distinguished name of an entry in the directory server that represents that user. To perform authentication, pam_ldap attempts to bind to the directory server as that identity. If the ldap_bind operation succeeds, then pam_ldap will return success to the PAM authentication subsystem.

When pam_ldap performs the ldap_bind operation, the LDAP server performs authentication of the user as well as determines if the LDAP account and password policy has passed. If the account is locked, the ldap_bind will fail. If the user's password has expired, the ldap_bind operation will return an error. An ldap_bind operation performs both `authentication` and `account management` operations.

Authentication with Secure Shell (SSH) and r-commands

For LDAP-UX B.04.00 or earlier versions, a user defined in an LDAP directory who tries to log on to a UNIX system using SSH key-pairs or the rhost enabled r-command will always be able to login even if this user's account has been locked or password has expired. These applications and commands do not need to call the PAM (Pluggable Authentication Module) authentication functions, but perform their own authentication instead. When this occurs, the ldap_bind operation is never performed. Thus, the LDAP directory server is never given the opportunity to perform security policy enforcement.

LDAP-UX Client Services B.04.10 provides PAM_AUTHZ features to support enforcement of account and password policies, stored in an LDAP directory server, for applications/commands (such as SSH or r-command) where authentication is not performed via PAM subsystem, but is performed by the command itself.

Policy File

The system administrator can define a local access policy and store all defined access rules in the policy file, `/etc/opt/ldapux/pam_authz.policy`. The PAM_AUTHZ service module uses this local policy file to process the access rules and to control the login authorization.

LDAP-UX Client Services provides a sample configuration file, `/etc/opt/ldapux/pam_authz.policy.template`. This sample file shows you how to configure the policy file to work with PAM_AUTHZ. You can copy this sample file and edit it using the correct syntax to specify the access rules you wish to authorize or exclude from authorization. For detailed information on how to construct an access rule in the policy file, see Constructing an Access Rule in pam_authz.policy (page 103).

NOTE: By default, the `allow:unix_local_user` access rule in the `/etc/opt/ldapux/pam_authz.policy.template` file is enabled.

Policy Validator

PAM_AUTHZ works as a policy validator. Once it receives a PAM request, it starts to process the access rules defined in `pam_authz.policy`. It validates and determines the user's login authorization based on the user's login name and the information it retrieves from various name services. The result is then returned to the PAM framework.

PAM_AUTHZ processes access rules in the order they are defined in the `pam_authz.policy`. It stops processing the access rules when any one of the access rules is evaluated to be true (match). That rule is called the "authorative" rule. If any access rule is evaluated to be false (no match), the rule is skipped. If all access rules in the policy file have been evaluated but the user's access right can not be determined, the user is restricted from login.

NOTE:

- If the user's login name is root or UID is 0, PAM_AUTHZ does not process the access rules defined in `pam_authz.policy`. The root user is always granted login access.
- The default `<action>` of PAM_AUTHZ is "`deny`" if no authorative rule is found.

The following describes situations where PAM_AUTHZ skips an access rule and does not process it:

- An access rule contains the wrong syntax.
- PAM_AUTHZ processes the `ldap_filter` and `ldap_group` types of access rules by querying the LDAP directory server through `ldapclientd` daemon. If LDAP-UX Client Services is not running, PAM_AUTHZ skips all the `ldap_filter` and `ldap_group` types of rules.

An Example of Access Rule Evaluation

The following shows an example of the `/etc/opt/ldapux/pam_authz.policy` file:

```
allow:unix_user:user1,user2,user3,user4
allow:unix_group:group1,group2
deny:unix_group:group11,group12
allow:netgroup:netgroup1,netgroup2
allow::ldap_group:ldapgroup1,ldapgroup2
allow:ldap_filter:(&(manager=Joeh) (department=marketing)(hostname=$[HOSTNAME]))
```

PAM_AUTHZ processes access rules in the order they are defined in the `pam_authz.policy` file. It stops evaluating the access rules when any one of the access rule is matched. In the above example, if the `user2` user attempts to login, it matches one of the user names in the first access rule, PAM_AUTHZ stops evaluating the rest of the access rules and allows the `user2` user to login. For another example, `user5` attempts to login and this user is only a member of `ldapgroup2`. PAM_AUTHZ validates user5's login access and when the fifth access rule is evaluated to be true, `user5` is granted the login access.

If the `user6` user reports to `Joeh`, the user's job is related to `marketing` and has a `hostname` attribute with the returned value, `HostSrv`, in his/her user entry in the LDAP directory. PAM_AUTHZ starts to validate user6's login access by evaluating all the access rule defined in `pam_authz.policy`. The sixth access rule is evaluated to be true, the `user6` is allowed to log in to the host, `HostSrv`.

Dynamic Variable Support

Dynamic variable support is a method by which an access rule can be defined where part or all of the policy criteria will be determined at the time the rule is evaluated. For example, the name of the computer from which the user attempts to logon can be substituted into the access rule to be evaluated. See the "Dynamic Variable Access Rule " (page 108) section for more information on how to define an access rule using dynamic variable support.

Constructing an Access Rule in pam_authz.policy

In the policy file, `/etc/opt/ldapux/pam_authz.policy`, an access rule consists of three fields as follows:

<action>:<type>:<object>

All fields are mandatory except for the `<object>` field when `unix_local_user` or `Other` is specified in the `<type>` field. If any field is missing or contains the incorrect syntax, the access rule is considered to be invalid and is ignored by PAM_AUTHZ.

These fields have the following limitations:

- No leading or trailing empty space is allowed in a field
- Fields are separated by a separator, :
- No leading or trailing empty space is allowed in a separator
- An access rule is terminated by a carriage return

Fields in an Access Rule

Table 5-1 shows a summary on all possible values and syntax of an access rule:

Table 5-1 Field Syntax in an Access Rule

<action>	<type>	<object>
`deny, allow, <pam_code>`	`unix_user`	A list of user name. It can be the multi-valued field. Each value is a character string that is separated by a separator "," (ASCII 2C HEX). Example: user1, user2, user3
`deny, allow, <pam_code>`	`unix_local_user`	No value is required.
`deny, allow, <pam_code>`	`unix_group`	A list of group name. It can be the multi-valued field. Each value is a character string that is separated by a separator `","` (ASCII 2C HEX). Example: group1, group2, group3
`deny, allow, <pam_code>`	`netgroup`	A list of netgroup name. It can be the multi-valued field. Each value is a character string that is separated by a separator `","(ASCII 2C HEX).` Example: netgroup1, netgroup2, netgroup3
`deny, allow, <pam_code>`	`ldap_group`	It is the Distinguished name of a ldap group with `groupofnames` objectclass or `groupofuniquenames` objectclass. It is a single-valued field. No separator is required. The syntax of DN is defined in RFC2253. Example: cn=ldapgroup1,cn=groups,dc=mydomain,dc=com
`deny, allow, <pam_code>`	`ldap_filter`	It is a single search descriptor that specifies one or more (attribute=value) or (attribute=$[variable_name]) pairs. $[variable_name] is a dynamic variable. It is a single value field. Only one search filter is allowed. No separator is required. The syntax of DN is defined in RFC2254. Example: (&(manager=Joeh)(department=sales)(hostcontrol=$[HOSTNAME]))

Table 5-1 Field Syntax in an Access Rule *(continued)*

<action>	<type>	<object>
`deny, allow, <pam_code>`	`other`	No value is required.
`status`	`<library_name>` The valid value for this field can be `rhds` or `ads`.	`<function_name>` Specifies the function name in `<library_name>` that is called to evaluate certain policy settings of the login user. Example: status:rhds:check_rhds_polcy See the "Account and Password Security Policy Enforcement " section for details.

The following describes three fields defined in an access rule in details:

<action> This field defines a user's final access permission if an access rule is evaluated to be true. Valid entries can be `allow`, `deny` and PAM return codes. `Allow` and `deny` are character strings and the value itself is not case sensitive. In additional to the general return codes, `allow` and `deny`, LDAP-UX Client Services B.04.10 or later, PAM_AUTHZ supports the meaningful PAM return codes to the application which called the PAM API. PAM_AUTHZ does not evaluate an access rule if no option is defined or if the `action` field contains an invalid string.

`<action>` field can be one of following values:

`allow`

This option indicates that a user is granted the login authorization.

`deny`

This option indicates that a user is denied the login authorization.

`<pam_code>`

One of the following meaningful PAM return codes can be specified in the `<action>` field, the PAM return codes are character strings:

- `PAM_SUCCESS`
- `PAM_PERM_DENIED`
- `PAM_MAXTRIES`
- `PAM_AUTH_ERR`
- `PAM_NEW_AUTHTOK_REQD`
- `PAM_AUTHTOKEN_REQD`
- `PAM_CRED_INSUFFICIENT`
- `PAM_AUTHINFO_UNAVAIL`
- `PAM_USER_UNKNOWN`
- `PAM_ACCT_EXPIRED`
- `PAM_AUTHTOK_EXPIRED`

For example, if the PAM_AUTHZ policy rule indicates that an account has been locked out or a password has expired, PAM_AUTHZ can return an appropriate PAM error code instead of a general deny error code.

<type> The value in this field represents the type of access rule. It defines what kinds of user information that PAM_AUTHZ needs to look for. The value also helps to determine the correct syntax in the following `<object>` field.

The following describes the valid values for this field:

`unix_user, unix_local_user, unix_group, netgroup, ldap_group`

Rules that have one of these specified as the `<type>` field are defining a static list access rule. For this rule, the `<object>` field is specified as a predefined list of identifiers. The identifiers are matched directly with data in the login request. This <type> field specifies where PAM_AUTHZ will look to determine if the login field is present in the appropriate data store, such as `/etc/passwd`, `/etc/group`, etc. If the login field is found, the rule is evaluated to be true. The final access right is determined by the `<action>` field. See the "Static List Access Rule" section for details.

`other`

PAM_AUTHZ ignores any access rules defined in the `<object>` field. The access rule is evaluated to be true immediately. For example,

```
allow:other
```

In the above example, all users are granted the login access to the machine. The primary usage of this type of rule is to toggle PAM_AUTHZ default `<action>`.

`ldap_filter`

In a role based access management, permission to access a resource can be controlled based on the user's role such as sales force, technical support or subscriber status and are typically defined by common business attributes of users based on company policies. The same concept is applied to the `ldap_filter` access rule. A search filter is defined in `<object>` field. A search filter consists of one or more (attribute=value) pairs. If the user entry is successfully retrieved from a directory server by using the search filter, the access rule is considered to be true. Examples of `ldap_filter` type of access rule are as follows:

```
allow:ldap_filter:(&(manager=paulw)(business
category=marketing))
```

In the above example, if a user reports to `paulw` and the user's job is related to `marketing`, then the user is granted the login access. The rule structure is very flexible about how to define access for certain groups of users.

```
PAM_ACCT_EXPIRED:ldap_filter:(nsAccountLock=TRUE)
```

In the above example, if a user account has been locked out and this access rule is evaluated to be true, the `PAM_ACCT_EXPIRED` code is returned by PAM_AUTHZ.

LDAP-UX Client Services B.04.10 or later, PAM_AUTHZ supports dynamic variable in the ldap_filter type of the access rule. A search filter can consist of one or more `(attribute=$[function_name])` pairs and is defined in the `<object>` field. The `[function_name]` is called and the return value is substituted into the search filter. Then the search filter is processed the same as the example above. For detailed information about dynamic variable support, see "Dynamic Variable Access Rule " (page 108).

`status`

When `status` is specified as the `<action>` field, this defines a rule that is evaluated to perform account and password policy enforcement. This access rule defines a library, in the `<library_name>` field to be loaded, and a function in the `<function_name>` field that specifies a function to be invoked to perform policy evaluation for a particular directory server. See the "Security Policy Enforcement Access Rule " (page 110) section for detailed information on the supported values and usage of this access rule.

<object> The values in this field define the policy criteria that PAM_AUTHZ uses to validate with the login name. The values in this field are dependent on the option that is stated in the `<type>` field.

Static List Access Rule

When the value in the `<type>` field is one of `unix_user`, `unix_group`, `netgroup`, `ldap_group`, the rule is evaluated using a list of predefined values in the `<object>` field. Based on the value in the `<type>` field, pam_authz will call the appropriate service to determine if the item requested is present. If the requested information is found then the rule is evaluated to be true.

The following describes these values for this field in details:

unix_user

This option indicates that an administrator wants to control the login access by examining a user's login name with a list of predefined users. If the login name matches one of the user names in the list, the authorization statement is evaluated to be true. The final access right is determined by evaluating the `<action>` field. An example of a `unix_user` type of access rule is as follows:

```
allow:unix_user:myuser1,myuser2,myuser3
```

If a `myuser3` user attempts to login, the above access rule is evaluated to be true and the user is granted login access.

unix_local_user

This option indicates that an administrator wants to control the login access by examining a local user's login name with a list of user's accounts in the `/etc/passwd` file. If the login name matches one of the user accounts defined in `/etc/passwd`, the authorization statement is evaluated to be true. Otherwise, the rule is skipped. An example of a `unix_local_user` type of access rule is as follows:

```
allow:unix_local_user
```

As an example, if a user account, `myuser5`, is defined in `/etc/password`, the above access rule is evaluated to be true and this user `myuser5` is granted to login to the local host.

unix_group

This option specifies that an administrator wants to control the login access right using the user's group membership. You can specify a list of group name in the `<object>` field. PAM_AUTH retrieves the group information of each listed group by querying the name services specified in `nsswitch.conf`. That means the group entries may come from any sources (files, nis, ldap, etc). If the login user belongs to any groups in the list, the access rule is evaluated to be true. Otherwise, the rule is skipped. An example of a `unix_group` access rule is shown as follows:

```
deny:unix_group:myunixgroup10,myunixgroup11,myunixgroup12
```

A user tries to login and he is a member of `myunixgroup12`. The rule is evaluated to be true and the `<action>` is applied. The user is restricted from access to the machine even with a valid password.

netgroup

This option specifies that the access permission is determined by the user's netgroup membership. You must specify a list of netgroup name in the `<object>` field. If the user is a member of one of the netgroups specified in the netgroup list, then the access rule is evaluated to be true. PAM_AUTH obtains the netgroup information by querying the name services specified in `nsswitch.conf`. For example:

```
allow:netgroup:netgroup1,netgroup2,netgroup3
```

A user tries to login and he belongs to `netgroup1`. The above access rule is evaluated to be true. The user is granted login access.

ldap_group

This option specifies that an access rule is based on the non-POSIXGroup membership. PAM_AUTHZ supports ldap group with `groupOfNames` or groupOfUniqueNamesobjectclass. A list of ldap_group names is specified in the <object> field. The group membership information is stored in the LDAP directory server. An example of a ldap_group type of access rule is as follows:

```
deny:ldap_group:engineering_ldapgroup,support_ldapgroup,epartner_ldapgroup
```

PAM_AUTHZ retrieves group membership of each listed group from the directory server through LDAP-UX client services. Then, it examines if the user's Distinguished Name (DN) matches any value in the member or uniquemember attribute.

Dynamic Variable Access Rule

PAM_AUTHZ supports dynamic variables in the ldap_filter type of the access rule. A dynamic variable is defined in `<object>` (LDAP search filter) field, it can consist of one or more (attribute=$[variable_name]) pairs. The syntax of an access rule with the dynamic variable is:

<action>:ldap_filter:(attribute=$[variable_name])

For example, if an administrator has an attribute named `hostControl` defined in the directory, and wants to use this attribute to define which host a user can log on to. He may add the following access rule in the `/etc/opt/ldapux/pam_authz.policy` file:

```
allow:ldap_filter:(hostControl= hostA)
```

Where `hostA` is the value for the local host that the user must be granted access. If a user, `John`, has a `hostControl` attribute in his user entry in the LDAP directory and the value is `hostA`, then the access rule is evaluated to be true and this user is allowed to log in to the host, `hostA`.

In the above example, a dynamic variable `HOSTNAME` can be used. The previous access rule can be re-defined as follows:

```
allow: ldap_filter: (hostControl=$[HOSTNAME])
```

where `$[HOSTNAME]` represents a dynamic variable function which will be called to retrieve the local host name information. PAM_AUTHZ will then substitute its return value to the search filter.

Supported Functions for Dynamic Variables

In LDAP-UX Client Services B.04.10, PAM_AUTHZ provides the following default dynamic variable functions in the `libpolicy_commonauthz` library. These functions can be used as dynamic variables specified in the ldap_filter type of access rules::

HOSTNAME Returns the fully qualified host name of the local system from which the user attempts to log on. For example, hostA.hp.com.

HOSTIP Returns the IP address of the local system from which the user attempts to log on. For example, 12.10.2.105.

TERMINAL Returns the terminal type of the computer from which the user attempts to log on. For example, /dev/pts/0.

Some applications (such as ssh or remsh) do not pass the terminal dynamic variable value to PAM_AUTHZ.

TIMEOFTHEDAY Returns the current time of the computer system from which the user attempts to log on. For example, 20061015125535Z represents October 15, 2006 at 12:55 and 35 seconds GMT. `TIMEOFTHEDAY` follows the “UTC Time” syntax as described by RFC4517.

SERVICE Returns the name of the PAM service from which the user attempts to access. For example, common PAM service names include ftp, login, telnet.

RHOSTIP Returns the IP address of the remote host system from which the user starts the PAM enabled application, such as telnet.

RHOSTNAME Returns the name of the remote host system from which the user starts the PAM enabled application, such as telnet.

Examples

The following shows a sample access rule in the `pam_authz.policy` file:

```
allow:ldap_filter:(WorkstationIP=$[HOSTIP])
```

The above policy rule performs a security policy validation for users stored in the LDAP directory server. If a user, `Mary`, has a `WorkstationIP` attribute in her user entry in the LDAP directory and the value is `1.2.3.200`. If Mary attempts to log in to the host with the IP address, `1.2.3.200`, then the access rule is evaluated to be true and this user is granted login access.

Security Policy Enforcement with Secure Shell (SSH) or r-commands

PAM_AUTHZ has a limited ability to perform account and password security policy enforcement without requiring LDAP-based authentication. This section provides information on how to configure the security policy enforcement access rule, setup access permissions for global policy attributes and configure PAM configuration file to support enforcement of account and password policies, stored in an LDAP directory server, for applications such as SSH key-pair and r-commands with rhost enabled.

This feature is designed to support applications such as SSH (Secure Shell) and the r-commands (rlogin, rcp, etc..) with .rhost enabled. With these applications, authentication is not performed via PAM (Pluggable Authentication Module) subsystem, but is performed by the command itself. In these applications, when authentication is not performed by PAM, the LDAP directory server is not given the opportunity to provide security policy enforcement, which normally occurs during the LDAP authentication process.

To configure and use this feature for SSH key-pair or r-commands, you must perform the following tasks:

- Set security policy enforcement access rule in the `/etc/opt/ldapux/pam_authz.policy` file. See the "Security Policy Enforcement Access Rule " (page 110) section for details.
- Set access permissions for global policy attributes. See the "Setting Access Permissions for Global Policy Attributes" (page 111) section for details.
- Configure the pam_authz library and the `rcommand` option in the `/etc/pam.conf` file for the `sshd` and `rcomds` services under the account management section. See "Configuring PAM Configuration File" (page 112) section and Appendix D, "Sample /etc/pam.conf File for Security Policy Enforcement" (page 193) for details.

Security Policy Enforcement Access Rule

Specifying `status` in the `<action>` field of a pam_authz.policy access rule triggers use of the account and password security policy enforcement rule. When this rule is evaluated, PAM_AUTHZ will call the `<function_name>` in the library specified by the `<library_name>` field. PAM_AUTHZ returns the value which is one of the PAM return codes described in the "PAM Return Codes " (page 112) section below.

This access rule consists of the following three fields:

<action>:<library_name>:<function_name>

Fields in the Access Rule:

The following describes each field of the above access rule:

action When the `status` option is specified, PAM_AUTHZ returns whatever **<function_name>** in the **<library_name>** returns, which is one of the PAM return codes.

library_name This field specifies the name of the library to be loaded that supports the account and password policies for a particular directory server.

The following describes the valid values for this field:

- `rhds`: If this option is specified, PAM_AUTHZ loads the `/opt/ldapux/lib/libpolicy_rhds` library to process security policy configuration and examine the user's security policy status attributes, stored in the Netscape/Red Hat Directory Server.
- `ads`: If this option specified, PAM_AUHZ loads `/opt/ldapux/lib/libpolicy_ads` library to process security policy configuration and examine the user's security policy status attributes, stored in the Windows 2003 Active Directory Server.

function_name This field defines the function name in the specified **<library_name>** that PAM_AUTHZ uses to evaluate certain security policy settings with the login user.

The following describes the valid entries for this field:

- `check_rhds_policy`: If this option is specified, PAM_AUTHZ evaluates all the necessary account and password policies settings, stored in the Netscape/Red Hat Directory Server, for the login user.
- `check_ads_policy`: If this option is specified, PAM_AUTHZ evaluates all the necessary account and password policies settings, stored in theWindows 2000, 2003 or 2003 R2 Active Directory, for the login user.

NOTE: If the `status:rhds:check_rhds_policy` access rule is configured in the `/etc/opt/ldapux/pam_authz.policy` file, you must perform the following tasks:

- Define the `allow:unix_local_user` access rule in `pam_authz.policy` to allow the local user to login.
- Since the `status:rhds:check_rhds_policy` access rule is guaranteed to match and return a PAM return code. It is highly recommended to define the `status:rhds:check_rhds_policy` access rule at the end of the `pam_authz.policy` file. Otherwise, the access rules that are defined after the `status` access rule will not be evaluated.
- PAM_AUTHZ may display account and password policy attributes in the syslog file when the debug option is enabled. You can take proper action to protect the syslog file. For example, set the syslog file permissions, so that the file can only be accessed or viewed by the power user.

An Example of Access Rules

The following shows an example of the access rules defined in the `/etc/opt/ldapux/pam_authz.policy` file when configuring and using security policy enforcement for SSH key-pair or r-commands:

```
allow:unix_local_user
status:rhds:check_rhds_policy
```

Setting Access Permissions for Global Policy Attributes

In order for PAM_AUTHZ to support security policy enforcement with the Red Hat Directory server, PAM_AUTHZ needs access to the security policy configuration attributes. These global policy attributes are all defined under cn=config. Only authorized users can access them. If you use the PAM_AUTHZ enhancement to support the account and password policy enforcement, you must configure LDAP-UX with a proxy user and grant this proxy user read and search rights to search for specific attributes under cn=config. The following example ACI gives a proxy user permission to read and search all global policy attributes:

```
aci: (targetattr= "objectclass ||passwordLockout ||passwordUnlock
 ||passwordMaxFailure ||passwordExp ||passwordMustChange
 ||nsslapd-pwpolicy-local")
 (version 3.0; acl "Proxy global security policy attributes read and
 search rights";
 allow (read,search)
 (userdn = "ldap:///uid=proxyuser,ou=Special Users,o=hp.com");)
```

For more information about a list of security policy attributes supported by LDAP-UX, see "Directory Server Security Policies" (page 113).

Configuring PAM Configuration File

If you want to use PAM_AUTHZ to support enforcement of account and password policies, stored in the Netscape/Red Hat Directory Server, you must define the pam_authz library and the `rcommand` option in the `/etc/pam.conf` file for the `sshd` and `rcomds` services under the account management section. In addition, the control flag for the pam_authz library must be set to `required`. See Appendix D, "Sample /etc/pam.conf File for Security Policy Enforcement" (page 193) for proper configuration.

Evaluating the Netscape/Red Hat Directory Server Security Policy

The following is an example of the access rule in the `/etc/opt/ldapux/pam_authz.policy` file:

```
status:rhds:check_rhds_policy
```

If the above access rule is specified in the `pam_authz.policy` file, the `check_rhds_policy` routine in the `libpolicy_rhds` library is loaded and executed. PAM_AUTHZ constructs a request message that will be used to find the current security policy configuration as well as examine the specific user's security policy status attributes to determine if the user complies with the security policy. PAM_AUTHZ will search for the following information: :

- Global policy attributes under cn=config: `passwordLockout`, `passwordUnlock`, `passwordMaxFailure`, `passwordExp`, `passwordMustChange`, `nsslapdpwpolicy-local`.
- User specific policy attributes: `accountUnlockTime`, `passwordExpirationTime`, `pwdPolicySubEntry`, `passwordRetryCount`, `nsAccountLock`.
- If fine-grained policy is turned on and the sub-tree policy for this user has been configured,, then LDAP-UX searches for password policy attributes at the subtree and user level: `passwordLockout`, `passwordUnlock`, `passwordMaxFailure`, `passwordExp`, `passwordMustChange`.

PAM_AUTHZ performs the following major functionality by evaluating the necessary security policy settings and returns the corresponding PAM return code to the applications/commands which called the PAM API.

- Check to see if an account is inactivated or not.
- Check to see if an account is locked or not.
- Check to see if the password has expired or not.

PAM Return Codes

If the `status:rhds:check_rhds_policy` access rule is specified in the `/etc/opt/ldapux/pam_authz.policy` file for Netscape/Red Hat Directory Server, PAM_AUTHZ evaluates the necessary security policy settings and returns the possible PAM return codes as follows:

PAM_USER_UNKNOWN The code returned if the user is not found in the Directory Server or if there is any internal errors (such as an error returned by the server) to find the user's policy attributes.

PAM_ACCT_EXPIRED The code returned if the user account is inactive.

PAM_ACCT_EXPIRED The code returned if the user account has been locked out.

PAM_NEW_AUTHTOK_REQD The code returned if the user's password has expired.

PAM_SUCCESS The code returned if the user account is active and not locked, and user's password has not expired.

Directory Server Security Policies

Global Security Attributes

In the Netscape/Red Hat Directory Server, there are a number of attributes used to define the security policies. In order to support account and password security policy enforcement, PAM_AUTHZ is enhanced to support the global administrative security attributes listed in the table below.

These attributes are used to define the policy rules and are all defined under cn=config. Only authorized users can access them. If you use the PAM_AUTHZ enhancement to support the account and password policy enforcement, you must configure LDAP-UX with a proxy user and grant this proxy user read and search rights to search cn=config.

Table 5-2 Global Security Attributes

Attribute	Description
`passwordLockout`	This boolean attribute indicates whether users will be locked out of the directory after a given number of failed bind attempts. By default, users will not be locked out of the directory after a series of failed bind attempts.
`passwordUnlock`	This boolean attribute indicates whether users will be locked out of the directory for a specified amount of time or until the password is reset after an account lockout. If the `passwordUnlock` attribute is disabled and the `accountUnlockTime` attribute has a value of 0, then the account will be locked indefinitely.
`passwordMaxFailure`	This integer attribute indicates the maximum number of password failures after which a user will be locked out of the directory. By default, account lockout is disabled.
`passwordExp`	This boolean attribute indicates whether user passwords will expire after a given number of seconds. By default, user passwords do not expire. If this attribute is enabled, you can use the `passwordMaxAge` variable to set the number of seconds after which the password will expire.
`passwordMustChange`	This boolean attribute indicates whether users must change their passwords when they first bind to the Directory Server or when the password has been reset by the Directory Manager.
`nsslapd-pwpolicy-local`	Turns fine-grained (subtree and user level) password policy on and off. If this attribute has a value off, all entries (except for cn=Directory Manager) in the directory will be subjected to the global password policy, the server will ignore any defined subtree and user level password policy. If this attribute has a value on, the server will check for password policies at the subtree and user level and enforce those policies.

Security Policy Status Attributes

PAM_AUTHZ supports a list of attributes which hold the general security policy status information for a particular user in the Netscape/Red Hat Directory Server shown as table.

Table 5-3 Security Policy Status Attributes

Attribute	Description
`nsAccountLock`	This boolean attribute indicates whether an account is locked or not. If this attributes does not exist, the account is considered unlocked.
`passwordRetryCount`	This integer attribute specifies the number of consecutive failed attempts at entering the correct user password.

Table 5-3 Security Policy Status Attributes *(continued)*

`passwordExpirationTime`	This string attribute defines a date and time when a password is considered expired. The data and time are specified using the “Generalize Time” syntax as referenced in RFC 2252 and specified by the ISO x.208 standard. It uses the format YYYYMMDDHHMMSSTZ, where YYYY= 4 difit year, MM= 2 digit month, DD=2 digit day, HH=2 digit hour, MM=2 digit minute, SS=2 digit second and TZ=tme zone. In LDAP directory servers, they use the GMT time zone which is represented with the letter Z for Zone time. For example, 20060215165535Z represents February 15, 2006 at 16:55 and 35 seconds GMT.
`accountUnlockTime`	This string attribute defines a date and time when an account will be unlocked. The value is represented in the Generalized Time syntax described in the “`passwordExpirationTime`” attribute. If the attribute does not exist, the account is considered unlocked (assuming `nsAccountLock` does not also exist).
`pwdpolicysubentry`	This variable defines the location of the new password policy. The location is expressed in the DN format.

Adding One or More Users

You can add one or more users to your system as follows:

1. Add the user's posixAccount entry to your LDAP directory.

   You can use your directory's administration tools, the ldapadd command, or the ldapentry tool to add a new user entry to your directory. If you are adding a large number of users, you could create a passwd file with those users and use the migration tools to add them to your directory. See *Installing and Administering LDAP-UX Client Services* for information on these tools.

   To add the new user with the Netscape/Red Hat Directory Console, select the Directory tab. Select the directory location in the left panel where your user information is. Select the Object:New:Other... menu item. Select the posixAccount object class in the dialog box and select OK. Fill in the values for the user and select OK.

2. Add the user to the appropriate posixGroup entry.

   You can use your directory's administration tools, or the ldapmodify program to add the user to the appropriate group in the directory. Add the user name to the memberuid attribute. See *Installing and Administering LDAP-UX Client Services* for information on these tools.

   To add the new user with the Netscape/Red Hat Directory Console, select the Directory tab. Select the directory location in the left panel where your group information is. Double click on the group where you want to add the user, or select the group and select the Object:Open menu item. Select the memberuid attribute in the dialog box. Select the Edit:Add Value menu item. Fill in the user's uid (login) name in the new field and select the OK button.

3. Use *nsquery* (1) or *pwget*(1) to verify the information was added and is accessible to the client:

   ```
   nsquery passwd user
   pwget -n user
   ```

Adding a Directory Replica

Your LDAP directory contains configuration profiles downloaded by each client system and name service data accessed by each client system. As your environment grows, you may need to add a directory replica to your environment. LDAP-UX can take advantage of replica directory servers and the alternates if one of them fails. Follow these steps to inform LDAP-UX about multiple directory servers:

1. Create and configure your LDAP directory replica. For Netscape/Red Hat Directory Server for HP-UX, see the *Netscape Directory Server Deployment Guide*.
2. Edit an existing profile and modify the defaultServerList or preferredServerList attribute to specify a replica directory server. See Modifying a Profile (page 118).

   See LDAP-UX Client Services Object Classes (page 185) for a description of the defaultServerList or preferredServer attribute.
3. On all clients that are to use the replica server, edit the start-up file, /etc/opt/ldapux/ldapux_client.conf, to refer to the replica host. Modify the LDAP_HOSTPORT line to specify the replica server.
4. After modifying an existing profile, each client that regularly downloads its profile automatically will get the changes as scheduled. SeeDownload the Profile Periodically (page 69).

NOTE: Client systems using an LDAP directory replica may not be able to modify the directory replica. In this case, the *passwd*(1) command will not work on those systems. They can use the *ldappasswd*(8) command described under ldappasswd (page 137).

Displaying the Proxy User's DN

You can display the proxy user's distinguished name by running /opt/ldapux/config/ldap_proxy_config -p.

The following command displays the current proxy user:

```
ldap_proxy_config -p
PROXY DN: uid=proxy,ou=people,o=hp.com
```

Verifying the Proxy User

The proxy user information is stored encrypted in the file /etc/opt/ldapux/pcred. You can check if the proxy user can authenticate to the directory by running /opt/ldapux/config/ldap_proxy_config -v as follows:

```
cd /opt/ldapux/config
./ldap_proxy_config -v
File Credentials verified - valid
```

Creating a New Proxy User

If you need to create a new proxy user and change your client systems to use the new proxy user, use the following steps:

1. Add the new proxy user to your directory with appropriate access controls. See the steps "Create a proxy user" and "Set access permissions for the proxy user" under the procedure Configure Your Directory (page 29) for details.
2. Configure each client to use the new proxy user by running /opt/ldapux/config/ldap_proxy_config. See The ldap_proxy_config Tool (page 132) for details. See below for examples.

3. Run /opt/ldapux/config/ldap_proxy_config -p to display the proxy user you just configured and confirm that it is correct.
4. Run /opt/ldapux/config/ldap_proxy_config -v to verify the proxy user is working.

Example

For example, the following command configures the local client to use a proxy user DN of `uid=proxy,ou=people,o=hp.com` with a password of `abcd1234`:

```
cd /opt/ldapux/config
./ldap_proxy_config -i
uid=proxy,ou=people,o=hp.com
abcd1234
```

The following command displays the current proxy user:

```
./ldap_proxy_config -p
PROXY DN: uid=proxy,ou=people,o=hp.com
```

The following command checks to see if the proxy user can bind to the directory:

```
./ldap_proxy_config -v
File Credentials verified - valid
```

Displaying the Current Profile

You can display the profile in use by any client by running /opt/ldapux/config/display_profile_cache on that client. The current profile is in the binary file /etc/opt/ldapux/ldapux_profile.bin.

```
cd /opt/ldapux/config
./display_profile_cache
```

You can also find out from where in the directory the client downloaded the profile by displaying the file /etc/opt/ldapux/ldapux_client.conf and looking for the line beginning with PROFILE_ENTRY_DN, for example:

```
grep ^PROFILE_ENTRY_DN /etc/opt/ldapux/ldapux_client.conf
PROFILE_ENTRY_DN="cn=profile1,ou=hpuxprofiles,o=hp.com"
```

Creating a New Profile

To create a new profile, run /opt/ldapux/config/setup. When setup asks you for the distinguished name (DN) of the profile, give a DN that does not exist and setup will prompt you for the parameters to build a new profile. The setup program also configures the local client to use the new profile.

Alternatively, you could use your directory administration tools to make a copy of an existing profile and modify it.

You can also use the interactive tool create_profile_entry to create a new profile as follows:

```
cd /opt/ldapux/config
./create_profile_entry
```

Once you create a new profile, configure client systems to use it as described in Changing Which Profile a Client Is Using (page 118).

Modifying a Profile

You can modify an existing profile directly using your directory administration tools, for example with Netscape/Red Hat Console. See LDAP-UX Client Services Object Classes (page 185) for a complete description of the DUAConfigProfile object class, its attributes, and what values each attribute can have.

The `ldapentry` tool can also be used to modify the existing profile. This can be done with the following command:

```
$ /opt/ldapux/bin/ldapentry -m "DN_of_profile"
$ cd /opt/ldapux/config
$ ./get_profile_entry -s nss
```

After modifying a profile, each client that regularly downloads its profile automatically will get the changes as scheduled. See Download the Profile Periodically (page 69) for details.

Changing Which Profile a Client Is Using

Each client uses the profile specified in its start-up file /etc/opt/ldapux/ldapux_client.conf. To make a client use a different profile in the directory, edit this file and change the DN specified in the PROFILE_ENTRY_DN line. Then download the profile as described in Download the Profile Periodically (page 69).

Changing from Anonymous Access to Proxy Access

If you have anonymous access and you want to change to using a proxy user, do the following:

1. Create the proxy user in the directory. With Netscape/Red Hat Directory Server, you can use the Netscape Console.
2. Change the credentialLevel attribute in your profile to be "proxy" using your directory administration tools, for example the Netscape Console.

   If you want proxy access with anonymous access as a backup if proxy access fails, change credentialLevel to be "proxy anonymous".

3. Download the profile to the client. If you have an automated process to download the profile, you can wait until it executes. Or you can download the profile manually by running the following command:

   ```
   cd /opt/ldapux/config
   ./get_profile_entry -s nss
   ```

You can verify that the proxy user is configured with display_profile_cache and ldap_proxy_config. display_profile_cache displays the current configuration profile, including the credential level, which is either "proxy," "anonymous," or "proxy anonymous." ldap_proxy_config displays and verifies the proxy user the client is configured to use. See The display_profile_cache Tool (page 131), The ldap_proxy_config Tool (page 132), and The get_profile_entry Tool (page 131) for more information.

Changing from Proxy Access to Anonymous Access

If you are using proxy access and you want to change to using anonymous access, do the following:

1. Change the credentialLevel attribute in your profile to be "anonymous" using your directory administration tools, for example the Netscape/Red Hat Directory Console.
2. Download the profile to the client. If you have an automated process to download the profile, you can wait until it executes. Or you can download the profile manually as described in Download the Profile Periodically (page 69).

3. Remove the proxy information:

   ```
   cd /opt/ldapux/config
   ./ldap_proxy_config -e
   ```

4. Optionally, remove the proxy user from the directory if you no longer need it. With Netscape/Red Hat Directory Server, you can use the Directory Server Console.

Performance Considerations

This section lists some performance considerations for LDAP-UX Client Services. See the white paper *LDAP-UX Integration Performance and Tuning Guidelines* at:

http://docs.hp.com/hpux/internet/#LDAP-UX%20Integration

for additional performance information.

Minimizing Enumeration Requests

Enumeration requests are directory queries that request all of a database, for example all users or all groups. Enumeration requests of large databases could reduce network and server performance. For this reason, you may want to restrict the use of commands and applications that enumerate.

The following commands generate enumeration requests:

- *finger*(1)
- *grget*(1) with no options
- *pwget*(1) with no options
- *groups*(1)
- *listusers*(1)
- *logins*(1M)
- All netgroup calls

In addition, applications written with routines of families such as the *getpwent*, *getgrent*, *gethostent*, and *getnetent* family of calls can enumerate a map, depending on how they are written.

Client Daemon Performance

Compared to previous networked name service systems, LDAP directory servers support a number of new features. And the general purpose nature of LDAP allows it to support a variety of applications, beyond those just used by a networked OS. Although directory servers have excellent performance and scalability, the addition of these features, such as security, means that directory applications will benefit from a design that considers performance requirements. In order to maximize of the number of HP-UX clients that can be supported by an LDAP directory server, and also improve client response, the ldapclientd daemon supports both data caching and persistent network connections. Their use, benefits and side-effects are described below.

ldapclientd Caching

Caching LDAP data locally allows for much greater response time for name service operations. Caching means that data that has been recently retrieved from the directory server will be retrieved from a local store, instead of the directory server. Caching greatly reduces both directory server load and network usage. For example, when a user logs into the system, the OS typically needs to enquire about his/her account several times in the login process. This occurs as the OS identifies the user, gathers account information and authenticates the user. And further requests often occur as the account starts up new applications once a session is established. With caching, generally only one or two LDAP operations are required.

Caching is also critical to support certain types of applications that make frequent demands on the name service system, either because they are malfunctioning or need this specific type of information frequently.

ldapclientd also supports what is known as a negative cache. This type of cache is used to store meta-data about non-existent information. For example, if an application requests information about an account that does not exist, the directory server will not return an entry, and that negative result will be stored in a cache. Intuitively this type of cache would seem to be un-necessary. However, applications exist that may perform these operations frequently, either on purpose or because they are malfunctioning. For example, if a file is created with a group ID that does not exist, every time a user displays information about this file, using the ls command, a request to the directory server will be generated.

The ldapclientd daemon currently supports caching of passwd, group, netgroup and automount map information. ldapclientd also maintains a cache which maps user's accounts to LDAP DNs. This mapping allows LDAP-UX to support groupOfNames and groupOfUniqueNames for defining membership of an HP-UX group.

Although there are many benefits to caching, administrators must be aware of the side-effects of their use. Here are some examples to consider:

Table 5-4 Benefits and Side-Effects for Caching

Map Name	Benefits	Example Side-Effect
`passwd`	Reduces greatly the number of requests sent to a directory server during a login or other operation such as displaying files owned by that user.	Removing this information from the directory may not be visible to the operating system until after the cache has expired. In certain cases, this may allow a user to login to an HP-UX host, even after his account has been removed from the LDAP directory server. (In general this is not a problem when pam_ldap is used for authentication, since authentication requests are not cached.)
`group`	Frequent file system access may request information about groups that own particular files. Caching greatly reduces this impact.	Removing a member of a group may not be visible to the file system, until after the cache expires. During this window, a user may be able to access files or other resources based on his/her group membership, which had been revoked.
`netgroup`	netgroups can be heavily used for determining network file system access rights or user login rights. Caching this information greatly reduces this impact	Similar to groups, since netgroups are used to control access to resources, modification of these rights may not appear until after cache information has expired. Users may be allowed or denied login even their rights should allow / deny access,
`automount`	Frequent file system access to a directory may request automount information about a network file system. A positive AutoFS cache greatly reduces LDAP-UX Client response time while retrieving the automount data. Whenever a user attempts to access a directory that does not exist on the physical file system, the AutoFS system is called to determine if that directory is available via the network through AutoFS. A negative AutoFS cache is critical to assure that malfunctioning applications do not place redundant bogus requests on the directory server.	For the positive AutoFS cache, an alteration of the automount maps will sometimes not appear immediately. During this expiration window, a network file system may be granted access, when in fact the automount map should have unmounted from a network file system. For the negative AutoFS cache, an alteration of the automount maps will sometimes not appear immediately. During this expiration window, a user attempting to access a network file system may be denied access, when in fact the automount map should have set up a network file system mount.

NOTE: The `ldapclientd -f` command will flush all caches. Refer to the man page `ldapclientd (1M)` for more information.

It is possible to alter the caching lifetime values for each service listed above, in the /etc/opt/ldapux/ldapclientd.conf file. See below for additional information. It is also possible to enable or disable a cache using the -E or -D (respectively) options. These options may be useful in determining the effectiveness of caching or helpful in debugging.

ldapclientd Persistent Connections

Since the HP-UX can generate many requests to an LDAP server, the overhead of establishing a single connection for every request can create excessive network traffic and slow response time for name service requests. Depending on network latency, the connection establishment and tear-down can cause relatively severe delays for client response. However, a persistent connection to the directory server will eliminate this delay.

In the ldapclientd daemon, a pool of active connections is maintained to serve requests from the Name Service Subsystem (NSS). If the NSS needs to perform a request to the directory server, one of the free connections in this pool will be used. If there are no free connections in the pool, a new connection will be established, and added to the pool. If system activity is low, then connections that have been idle for a specified period of time (configurable in the ldapclientd.conf file) then those connections will be dropped, to free up directory server resources. Aside from ldapclientd connection time-out configuration, it is also possible to define a maximum number of connections that ldapclientd may establish. Setting a high number of connections means assures that ldapclientd will not become a bottleneck in performing name service operations to the directory server. However, a high number of connections from a large number of HP-UX clients to the same directory server may exhaust all available connection resources on that directory server. Setting a low number of maximum connections will reduce that resource requirement on the directory server, but may create a performance bottleneck in the ldapclientd.

Troubleshooting

This section describes troubleshooting techniques as well as problems you may encounter.

Enabling and Disabling LDAP-UX Logging

When something is behaving incorrectly, enabling logging is one way to examine the events that occur to determine where the problem is. Enable LDAP-UX Client Services logging on a particular client as follows:

1. Edit the local startup file /etc/opt/ldapux/ldapux_client.conf and uncomment the lines starting with #log_facility and #log_level by removing the initial # symbol. You can set log_level to LOG_INFO to log only unusual events. This is a good place to start. If LOG_INFO is not adequate to identify the problem, set log_level to LOG_DEBUG to log trace information. LOG_DEBUG will provide more information but will significantly reduce performance and generate large log files on active systems.
2. Edit the file /etc/syslog.conf and add a new line at the bottom:

   ```
   local0.debug <tab> /var/adm/syslog/local0.log
   ```

   where **`<tab>`** is the Tab key on your keyboard.
3. Restart the syslog daemon with the following command. (See *syslogd*(1M) for details.)

   ```
   kill -HUP 'cat /var/run/syslog.pid'
   ```

4. Once logging is enabled, run the HP-UX commands or applications that exhibit the problem.
5. Disable logging by commenting out the log_facility and log_level lines in the startup file /etc/opt/ldapux/ldapux_client.conf. Comment them out by inserting a "#" symbol in the first column.
6. Examine the log file at /var/adm/syslog/local0.log to see what actions were performed and if any are unexpected. Look for functions with "ldap_." These are standard LDAP function calls.

TIP: Enable LDAP logging only long enough to collect the data you need because logging can significantly reduce performance and generate large log files.

You may want to move the existing log file and start with an empty file: mv /var/adm/syslog/local0.log /var/adm/syslog/local0.log.save

Enabling and Disabling PAM Logging

When something is behaving incorrectly, enabling logging is one way to examine the events that occur to determine where the problem is. Enable PAM logging on a particular client as follows. See *pam*(1), *pam.conf*(4), and *Managing Systems and Workgroups* for more information on PAM.

1. Add the "debug" option to each line in /etc/pam.conf that contains libpam_ldap, for example:

   ```
   login account sufficient /usr/lib/security/libpam_unix.1
   login account required   /usr/lib/security/libpam_ldap.1 debug
   su    account sufficient /usr/lib/security/libpam_unix.1
   su     account required  /usr/lib/security/libpam_ldap.1 debug
   ...
   ```

2. Edit the file /etc/syslog.conf and add a new line at the bottom like the following:

   ```
   *.debug   <tab> /var/adm/syslog/debug.log
   ```

3. Restart the syslog daemon with the following command. (See *syslogd*(1M) for details.)

```
kill -HUP 'cat /var/run/syslog.pid'
```

4. Once logging is enabled, run the HP-UX commands or applications that exhibit the problem.
5. Restore the file /etc/syslog.conf to its previous state; otherwise, you may unintentionally enable logging in other applications.
6. Restart the syslog daemon with the following command. (See *syslogd*(1M) for details.)

   ```
   kill -HUP 'cat /var/run/syslog.pid'
   ```

7. Remove the "debug" options from /etc/pam.conf.
8. Examine the log file at /var/adm/syslog/debug.log to see what actions were performed and if any are unexpected. Look for lines containing "PAM_LDAP."

TIP: Enable PAM logging only long enough to collect the data you need because logging can significantly reduce performance and generate large log files.

You may want to move the existing log file and start with an empty file: mv /var/adm/syslog/debug.log /var/adm/syslog/debug.log.save. Then restore the file when finished.

Directory Server Log Files

You can view log files to see if any unusual events have occurred with your directory. The Directory Server for HP-UX logs information to files under

/var/opt/Netscape/server4/slapd-<*serverID*>/logs

where slapd-<*serverID*> is the name of your directory server.

The error logs contain start-up, shut-down, and unusual events. The access logs contain all requests. See the *Netscape Directory Server Administrator's Guide* for details.

User Cannot Log on to Client System

If a user cannot log in to a client system, perform the following checks.

- Use a command like *pwget*(1) with -n, or *nsquery*(1) [2] to verify that NSS is working:

  ```
  pwget -n username
  nsquery passwd username
  ```

  If the output shows ldap is not being searched, check /etc/nsswitch.conf to make sure ldap is specified. If *username* is not found, make sure that user is in the directory and, if using a proxy user, make sure the proxy user is properly configured.

  If *nsquery*(1) displays the user's information, make sure /etc/pam.conf is configured correctly for ldap. If /etc/pam.conf is configured correctly, check the directory's policy management status. It could be the directory's policy management is preventing the bind because, for example the user's password has expired or the login retry limit has been exceeded. To check this try an ldapsearch command and bind as the user, for example:

  ```
  cd /opt/ldapux/bin
  ./ldapsearch -h servername -b "basdDN" uid=username (get user's DN)
  ./ldapsearch -h servername -b "baseDN" -D "userDN" -w passwd \ uid=username
  ```

  where *userDN* is the DN of the user who cannot log in and *username* is the login of the user. If you cannot bind as the user, check if any directory policies are preventing access.

[2] *nsquery*(1) is a contributed tool included with the ONC/NFS product.

See below for an example of determining the user's bind DN.

- Display the current configuration profile and check all the values to make sure they are as you expect:

  ```
  cd /opt/ldapux/config
  ./display_profile_cache
  ```

  In particular, check the values for the directory server host and port, the default search base DN, and the credential level. Also, if you have remapped any standard attributes to alternate attributes, or defined any custom search descriptors, make sure these are correct and exist in your database. If any of these are incorrect, correct them as described in Modifying a Profile (page 118).

- If you are using a proxy user, make sure the configuration is correct as described under Verifying the Proxy User (page 116).
- Make sure the client system can authenticate to the directory and find a user in the directory by searching for one of your user's information in the directory. Use the ldapsearch command and information from the current profile.

  If you are using a proxy user (determined by the credentialLevel attribute in the configuration profile), try searching for one of your user's information in the directory as the proxy user with a command like the following:

  ```
  cd /opt/ldapux/bin
  ./ldapsearch -h servername -b "baseDN" -D "proxyuser" -w \ passwd uid=username
  ```

  using the name of your directory server (from display_profile_cache), search base DN (from display_profile_cache), proxy user (from ldap_proxy_config -p), proxy user password, and a user name from the directory.

  For example:

  ```
  cd /opt/ldapux/bin
  ./ldapsearch -h sys001.hp.com -b "ou=people, o=hp.com" \
  -D "uid=proxyuser,ou=special users,o=hp.com" -w passwd \ uid=steves
  ```

  You should get output like the following:

  ```
  dn: uid=steves,ou=people o=hp.com
  uid: steves
  cn: Steve Sy
  objectclass: top
  objectclass: account
  objectclass: posixAccount
  loginshell: /bin/ksh
  uidnumber: 2875
  gidnumber: 191
  homedirectory: /home/steves
  gecos: Steve Sy, building 5, x50
  ```

  If you don't, your proxy user may not be configured properly. Make sure you have access permissions set correctly for the proxy user. See the steps "Create a proxy user" and "Set access permissions for the proxy user" under the procedure Configure Your Directory (page 29) for details on configuring the proxy user.

  You can also try binding to the directory as the directory administrator and reading the user's information.

  If you are using anonymous access, (determined by the value of the credentialLevel attribute in the configuration profile), try searching for one of your user's information in the directory with a command like the following:

```
./ldapsearch -h servername -b "o=hp.com" uid=username
```

using the name of your directory server (from display_profile_cache), search base DN (from display_profile_cache), and a user name from the directory.

You should get output similar to the previous example. If you don't, anonymous access may not be configured properly. Make sure you have access permissions set correctly for anonymous access. See the steps "Configure anonymous access" and "Set access permissions for anonymous access" under Configure Your Directory (page 29) for details on configuring anonymous access.

- Enable PAM logging as described under Enabling and Disabling PAM Logging (page 123) then try logging in again. Check the PAM logs for any unexpected events.
- Enable LDAP-UX logging as described under Enabling and Disabling LDAP-UX Logging (page 123), then try logging in again. Check the log file for any unexpected events.
- If you are using Netscape/Red Hat Directory Server, use the Netscape/Red Hat Directory Console to authenticate to the directory as the directory administrator. Check the ACIs for the proxy user. Make sure the proxy user or anonymous can view the attributes listed below. If not, change the ACI to allow this. Make sure all users can read their own information. If they cannot, change the ACI to allow this.

  Make sure all users have the following attributes and can read them:

  — cn
  — loginshell
  — uid
  — uidnumber
  — gidnumber
  — memberuid
  — homedirectory
  — gecos

6 Command and Tool Reference

This chapter describes the commands and tools associated with the LDAP-UX Client Services. It includes the following sections:

The LDAP-UX Client Services Components

The LDAP-UX Client Services product, comprising the following components, can be found under /opt/ldapux and /etc/opt/ldapux, except where noted. LDAP-UX Client Services libraries are listed on table 6-2 , 6–3 and 6-4.

Table 6-1 LDAP-UX Client Services Components

Component	Description
/etc/opt/ldapux/ldapux_client.conf	The LDAP-UX start-up file, specifies where the directory is, where in the directory the profile data is, and logging.
/etc/pam.ldap	A sample PAM configuration file. The actual PAM configuration file is /etc/pam.conf.
/etc/nsswitch.ldap	A sample Name Service Switch configuration file. The actual NSS configuration file is /etc/nsswitch.conf.
/etc/opt/ldapux/ldapux_profile.bin	The configuration profile translated from ldapux_profile.ldif, in binary format, used by the client. See also display_profile_cache below.
/etc/opt/ldapux/ldapux_profile.ldif	The configuration profile downloaded from the LDAP directory, in LDIF format.
/opt/ldapux/config/setup	Program to configure LDAP-UX Client Services.
/opt/ldapux/config/get_profile_entry	Program to download a configuration profile from a directory.
/opt/ldapux/config/display_profile_cache	Program to display the current configuration profile.
/opt/ldapux/config/create_profile_entry	Program to create a new configuration profile.
/opt/ldapux/config/create_profile_schema /opt/ldapux/config/create_profile_cache	Programs called by the setup program.
/opt/ldapux/config/ldap_proxy_config	Program to configure and verify the proxy user.
/opt/ldapux/bin/ldapdelete /opt/ldapux/bin/ldapmodify /opt/ldapux/bin/ldapsearch /opt/ldapux/bin/ldapentry /opt/ldapux/bin/ldap_del_entry /opt/ldapux/bin/ldap_new_entry /opt/ldapux/bin/ldap_mod_entry	Tools to delete, modify, and search for entries in a directory. See the "LDAP Directory Tools" section and the *Netscape Directory Server Administrator's Guide* for details.

Table 6-1 LDAP-UX Client Services Components *(continued)*

Component	Description
/opt/ldapux/bin/ldifdiff	Tool to generate LDIF change records from two input files.
/etc/opt/ldapux/ldapclientd.conf	The ldapclientd daemon configuration file.
/opt/ldapux/bin/ldapclientd	The ldapclientd daemon binary.
/opt/ldapux/bin/ldappasswd	Tool to modify user password in a directory.
/opt/ldapux/bin/ldapschema	Tool to query and extend directory server schema. See the "Schema Extension Utility" section for details.
/opt/ldapux/migrate	A set of scripts for migrating user, group, and other information into a directory. See Name Service Migration Scripts (page 170) for more information.
/opt/ldapux/share	Man pages.
/opt/ldapux/contrib/bin/perl	perl, version 5, used by migration scripts.
/opt/ldapux/ypldapd	Files for the NIS/LDAP Gateway product. See *Installing and Administering NIS/LDAP Gateway*.
/opt/ldapux/contrib/bin/beq	Search tool that bypasses the name service switch and queries the backend directly based on the specified library.
/opt/ldapux/contrib/bin/certutil	Command-line tool that creates and modifies the Netscape Communicator *cert7.db* and *key3.db* database files.

NOTE: For LDAP C SDK libraries info, refer to Mozilla LDAP C SDK (page 179) for details.

Table 6-2 shows LDAP-UX Client Services libraries on the HP 11.0 or 11i v1 machine:

Table 6-2 LDAP-UX Client Services Libraries on the HP-UX 11.0 or 11i v1 PA machine

Files	Description
/usr/lib/libldap_send.1 (32-bit) /usr/lib/libldap_util.1 (32-bit) /usr/lib/libnss_ldap.1 (32-bit) /usr/lib/libldapci.1 (32-bit) /usr/lib/libldap.1 (32-bit) /usr/lib/security/libpam_ldap.1 (32-bit) /usr/lib/security/libpam_authz.1 (32-bit) /usr/lib/pa20_64/libldap.1 (64-bit) /usr/lib/pa20_64/libldap_send.1 (64-bit) /usr/lib/pa20_64/libnss_ldap.1 (64-bit)	LDAP -UX Client Services libraries.

Table 6-3 shows LDAP-UX Client Services libraries on 32 or 64 bit of the HP-UX 11i v2 PA machine:

Table 6-3 LDAP-UX Client Services Libraries on the HP-UX 11i v2 PA machine

Files	Description
/usr/lib/libldap_send.1 (32-bit) /usr/lib/libldap_util.1 (32-bit) /usr/lib/libnss_ldap.1 (32-bit) /usr/lib/libldapci.1 (32-bit) /usr/lib/libldap.1 (32-bit) /usr/lib/security/libpam_ldap.1(32-bit) /usr/lib/security/libpam_authz.1 (32-bit) /usr/lib/pa20_64/libldap.1 (64-bit) /usr/lib/pa20_64/libldap_send.1 (64-bit) /usr/lib/pa20_64/libnss_ldap.1 (64-bit) /usr/lib/security/pa20_64/libpam_ldap.1 (64-bit) /usr/lib/security/pa20_64/libpam_authz.1 (64-bit)	LDAP -UX Client Services libraries.

Table 6-4 shows LDAP-UX Client Services libraries on 32 or 64 bit of the HP-UX 11i v2 IA machine:

Table 6-4 LDAP-UX Client Services Libraries on the HP-UX 11i v2 IA machine

Files	Description
/usr/lib/hpux32/libldap_send.so.1 (32-bit) /usr/lib/hpux32/libldap_util.so.1 (32-bit) /usr/lib/hpux32/libnss_ldap.so.1 (32--bit) /usr/lib/hpux32/libldapci.so.1 (32-bit) /usr/lib/hpux32/libldap.so.1 (32-bit) /usr/lib/security/hpux32/libpam_ldap.so.1 (32-bit) /usr/lib/security/hpux32/libpam_authz.so.1 (32-bit) /usr/lib/hpux64/libldap.so.1 (64-bit) /usr/lib/hpux64/libldap_send.so.1 (64-bit) /usr/lib/hpux64/libnss_ldap.so.1 (64-bit) /usr/lib/security/hpux64/libpam_ldap.so.1 (64-bit) /usr/lib/security/hpux64/libpam_authz.so.1 (64-bit) /usr/lib/libldap_send.1 (32-bit) /usr/lib/libldap.1 (32-bit) /usr/lib/libnss_ldap.1 (32--bit) /usr/lib/security/libpam_ldap.1 (32-bit) /usr/lib/security/libpam_authz.1 (32-bit) /usr/lib/pa20_64/libldap_send.1 (64-bit) /usr/lib/pa20_64/libldap.1 (64-bit) /usr/lib/pa20_64/libnss_ldap.1 (64--bit) /usr/lib/security/pa20_64/libpam_ldap.1 (64-bit) /usr/lib/security/pa20_64/libpam_authz.1 (64-bit)	LDAP -UX Client Services libraries.

Client Management Tools

This section describes the following programs for managing client systems. Most of these are called by the setup program when you configure a system.

display_profile_cache	Displays the currently active profile.
create_profile_entry	Creates a new profile in the directory.
get_profile_entry	Downloads a profile from the directory to LDIF, and creates the profile cache.
ldap_proxy_config	Configures a proxy user.

The following tools are called by the setup program and are not typically used separately.

create_profile_schema	Extends the schema in the directory for profiles.
create_profile_cache	Creates a new active profile from an LDIF profile. This is also called by get_profile_entry.

The create_profile_entry Tool

This tool, found in /opt/ldapux/config, creates a new profile entry in an LDAP directory from information you provide interactively. The directory schema must have the DUAConfigProfile extensions.

Syntax

```
create_profile_entry
```

The create_profile_cache Tool

This tool, found in /opt/ldapux/config, creates a binary profile file from an LDIF profile file, thus activating the profile for the client. (You can download a profile to LDIF from the directory with get_profile_entry.) Typically you run the setup program instead of running this program directly. See also Download the Profile Periodically (page 69).

Syntax

```
create_profile_cache [-i infile] [-o outfile]
```

where ***infile***is the LDIF file containing a profile, by default /etc/opt/ldapux/ldapux_profile.ldif and ***outfile***is the name of the binary output file, by default /etc/opt/ldapux/ldapux_profile.bin. The LDIF file must contain an entry for the object class DUAConfigProfile.

Examples

The following command creates the binary profile file /etc/opt/ldapux/ldapux_profile.bin from the existing LDIF file /etc/opt/ldapux/ldapux_profile.ldif:

```
create_profile_cache
```

The following command creates the binary profile file my_profile.bin from the existing LDIF file profile1.ldif:

```
create_profile_cache -i profile1.ldif -o my_profile.bin
```

Note that you must copy the file **my_profile.bin** to **/etc/opt/ldapux/ldapux_profile.bin** to activate the profile.

The create_profile_schema Tool

This tool, found in /opt/ldapux/config, extends the schema of a Netscape Directory Server 6.x with the DUAConfigProfile object class using the information you provide interactively. Typically you run the setup program instead of running this program directly.

Syntax

```
create_profile_schema
```

The display_profile_cache Tool

This tool, found in /opt/ldapux/config, displays information from a binary profile (cache) file. By default, it displays the currently active profile in /etc/opt/ldapux/ldapux_profile.bin.

Syntax

```
display_profile_cache [-i infile] [-o outfile]
```

where ***infile***is a binary profile file, /etc/opt/ldapux/ldapux_profile.bin by default, and ***outfile***is the output file, stdout by default.

NOTE: The binary profile contains mappings for all backend commands (even unused ones) all of which are displayed by display_profile_cache. The actual client configuration can be reviewed in the configuration profile LDIF file: `/etc/opt/ldapux/ldapux_profile.ldif`.

Examples

The following command displays the profile in the binary profile file /etc/opt/ldapux/ldapux_profile.bin to stdout:

```
display_profile_cache
```

The following command displays the profile in the binary profile file my_profile.bin and writes the output to the file profile:

```
display_profile_cache -i my_profile.bin -o profile
```

The get_profile_entry Tool

This tool, found in /opt/ldapux/config, downloads a profile from an LDAP directory into an LDIF file and calls create_profile_cache to create a binary profile file, thereby activating it on the client. This tool looks in the local client configuration file /etc/opt/ldapux/ldapux_client.conf for the profile DN.

Syntax

```
get_profile_entry -s service [-o outfile]
```

where ***service***is the name of a supported service, typically NSS, and ***outfile***is the name of a file to contain the LDIF output, by default /etc/opt/ldapux_profile.ldif.

Examples

The following command downloads the profile for the Name Service Switch (NSS) specified in the client configuration file /etc/opt/ldapux/ldapux_client.conf and places the LDIF in the file /etc/opt/ldapux/ldapux_profile.ldif:

```
get_profile_entry -s NSS
```

The following command downloads the profile for the Name Service Switch (NSS) specified in the client configuration file /etc/opt/ldapux/ldapux_client.conf and places the LDIF in the file profile1.ldif:

```
get_profile_entry -s NSS -o profile1.ldif
```

The ldap_proxy_config Tool

This tool, found in /opt/ldapux/config, configures a proxy user or an Admin Proxy user for the client accessing the directory. It stores the encrypted proxy user information in the file`/etc/opt/ldapux/pcred.` The encrypted Admin Proxy user information is stored in the file `/etc/opt/ldapux/acred`. If you are using only anonymous access, you do not need to use this tool. You must run this tool logged in as root.

Syntax

`ldap_proxy_config [`*`options`*`]`

where ***`options`***can be any of the following:

`-A` Action applies to the Admin Proxy user. This option must be specified with other option to apply the operation for the Admin Proxy user.

`-e` erases the currently configured proxy user from the file `/etc/opt/ldapux/pcred`. Has no effect on the proxy user information in the directory itself.

`-i` uses the `-i` option to configure the proxy user interactively from stdin. Use `-A -i`options to configure an Admin Proxy user.

If you use `ldap_proxy_config -i` to configure the proxy user using the simple authentication, type the command with -i then press Return. Next type the proxy user DN then press Return. Finally type the proxy user's credential or password and press Return.

If you configure the proxy user using the SASL DIGEST-MD5 with DN authentication (i.e. use the DN to generate the DIGEST-MD5 hash), type the command with -i then press Return. Next type the proxy user DN then press Return. Next type the proxy user's credential or password and press Return. Finally press Return.

If you configure the proxy user using the SASL DIGEST-MD5 with UID authentication (i.e. use the UID attribute to generate the DIGEST-MD5 hash), type the command with -i then press Return. Next type the proxy user DN then press Return. Next type the proxy user's credential or password and press Return. Finally type the proxy user's UID and press Return.

When you use the `ldap_proxy_config -A -i` command to configure an Admin Proxy user interactively from stdin, the configuration procedures are similar to the procedures used by the `ldap_proxy_config -i` command for a proxy user.

When configuring an Admin Proxy user, if you only enter the Admin Proxy user's DN without password, the root's password will be used instead.

`-f` ***`file`*** configures the proxy user from ***`file`***. ***`file`***must contain two lines: the first line must be the proxy user DN, and the second line must be the proxy user credential or password.

△ **CAUTION:** After using this option you should delete or protect the file as it could be a security risk.

`-d` ***`DN`*** sets the proxy user distinguished name to be ***`DN`***. To use this option, the */etc/opt/ldapux/pcred* file must exist.

`-c` ***`passwd`*** sets the proxy user credential or password to be ***`passwd`***. To use this option, the */etc/opt/ldapux/pcred* file must exist.

`-p` prints the distinguished name of the current proxy user.

`-v` verifies the current proxy user and credential by connecting to the server.

-h displays help on this command.

With no options, ldap_proxy_config configures the proxy user as specified in the file /etc/opt/ldapux/pcred.

For the proxy user, if you switch the authentication method between simple and DIGEST-MD5, you need to use the `ldap_proxy_config -e` command to delete /etc/opt/ldapux/pcred, then use the `ldap_proxy_config -i` command to reconfig the proxy user.

For the Admin Proxy user, if you switch the authentication method between simple and DIGEST-MD5, you need to use the `ldap_proxy_config -A -e` command to delete /etc/opt/ldapux/acred, then use the `ldap_proxy_config -A -i` to reconfig the Admin Proxy user.

Examples

The following example configures the proxy user as **uid=proxyuser1,ou=special users,o=hp.com** with the password **prox1pw** and creates or updates the file /etc/opt/ldapux/pcred with this information, the proxy user uses the simple authentication:

```
ldap_proxy_config -i
uid=proxyuser1,ou=special users,o=hp.com
prox1pw
```

The following example configures the proxy user as **uid=proxyusr2,ou=special users,o=hp.com** with **password prox2pw** and creates or updates the file /etc/opt/ldapux/pcred with this information, the proxy user uses the SASL DIGEST-MD5 authentication and uses the DN to generate the DIGEST-MD5 hash:

```
ldap_proxy_config -i
uid=proxyusr2,ou=special users,o=hp.com
prox2pw
CR>
```

The following example configures the proxy user as **uid=proxyusr3,ou=special users,o=hp.com**, UID **proxyusr3 and password prox3pw** and creates or updates the file /etc/opt/ldapux/pcred with this information, the proxy user uses the SASL DIGEST-MD5 authentication and uses the UID to generate the DIGEST-MD5 hash:

```
ldap_proxy_config -i
uid=proxyusr3,ou=special users,o=hp.com
prox3pw
proxyusr3
```

The following example configures the Admin Proxy user as **uid=adminproxy,ou=special users,o=hp.com** with the password **adminproxpw** and creates or updates the file /etc/opt/ldapux/acred with this information, the Admin Proxy user uses the simple authentication:

```
ldap_proxy_config -A -i
uid=adminproxy,ou=special users,o=hp.com
adminproxpw
```

The following example configures the Admin Proxy user as **uid=adminproxy2,ou=special users,o=hp.com** with **password admin2pw** and creates or updates the file /etc/opt/ldapux/acred with this information, the Admin Proxy user uses the SASL DIGEST-MD5 authentication and uses the DN to generate the DIGEST-MD5 hash:

```
ldap_proxy_config -A -i
uid=adminproxy2,ou=special users,o=hp.com
admin2pw
CR>
```

The following example configures the Admin Proxy as **uid=adminproxy3,ou=special users,o=hp.com**, UID **adminproxy3 and password admin3pw** and creates or updates the file /etc/opt/ldapux/acred with this information, the Admin Proxy user uses the SASL DIGEST-MD5 authentication and uses the UID to generate the DIGEST-MD5 hash:

```
ldap_proxy_config -A -i
uid=adminproxy3,ou=special users,o=hp.com
admin3pw
adminproxy3
```

The following example displays the current proxy user:

```
ldap_proxy_config -p
PROXY_DN: uid=proxyuser,ou=special users,o=hp.com
```

The following example checks the configured proxy user information and checks whether or not the client can bind to the directory as the proxy user:

```
ldap_proxy_config -v
File Credentials verified - valid
```

The following example configures the proxy user as uid=proxyuser,ou=special users,o=hp.com with the password prox12pw and creates or updates the file /etc/opt/ldapux/pcred with this information:

```
ldap_proxy_config -d "uid=proxyuser,ou=special users,o=hp.com" -c prox12pw
```

The following example configures the proxy user with the contents of the file proxyfile and creates or updates the file /etc/opt/ldapux/pcred with this information:

```
ldap_proxy_config -f proxyfile
```

The file proxyfile must contain two lines: the proxy user DN on the first line and password on the second line.

LDAP Directory Tools

This section briefly describes the `ldapentry`, `ldappasswd`, `ldapsearch`, `ldapmodify` and `ldapdelete`.

For detailed information about `ldapsearch`, `ldapmodify`, and `ldapdelete`, refer to the *Red Hat Directory Server for HP-UX Administrator's Guide* available at http://docs.hp.com/en/internet.html

ldapentry

`ldapentry` is a script tool that simplifies the task of adding, modifying and deleting entries in a Directory Server. It supports the following name services: passwd, group, hosts, rpc, services, networks, and protocols.

`ldapentry` accepts run-time options either on the command line, or via environment variables, which can be defined locally, in the configuration profile or are read in from the configuration profile. The add and modify functions open an entry into an editor with a pre-defined template to aid the user in providing the necessary directory attributes. The template file is customizable and can be found in /etc/opt/ldapux/ldapentry.templates.

The `ldapentry` command also accepts options through environment variables, configuration files, and the LDAP configuration profiles.

Configuration Variable

Configuration variables can be defined in the following locations (from most specific to most general):

1. as shell environment variables
2. in a user 'rc' configuration file (`~/.ux_ldap_admin_rc`)
3. in a global configuration file (`/etc/opt/ldapux/ldapclient.conf`)
4. in the configuration profile (`/etc/opt/ldapux/ldapux_profile.ldif`)

The order of evaluation is that any settings on more specific locations will overwrite any settings on more general locations.

Environment Variables

The following environment variables can be defined:

LDAP_BINDDN The DN of the LDAP user allowed to add, delete, or modify the entry.

LDAP_BINDCRED The password for the above specified LDAP user. It is recommended to not store the password in any configuration file, the user will be prompted for it when running ldapentry.

LDAP_HOST Host name of LDAP directory server.

LDAP_BASEDN The DN of the search base which tells `ldapentry` where to start the search for the entry. In case of adding an entry, LDAP_BASEDN determines the insert base.

LDAP_SCOPE The scope of LDAP search (sub, one, base). Will default to sub if LDAP_BASEDN is defined, but LDAP_SCOPE is not. You must define LDAP_BASEDN, if you define LDAP_SCOPE.

INSERT_BASE This DN tells ldapentry where to insert new entries. This value will default to LDAP_BASEDN or a default discovered by the configuration profile. INSERT_BASE is only used when adding entries.

EDITOR The editor to use when an entry is added or modified.

Syntax

```
ldapentry -<a|m|d> [options] <service value | dn>
```

where

-a Adds a new entry to the directory.

-m Modifies an existing entry in the directory.

-d Deletes an existing entry in the directory.

options

-f Forces command execution with warning override.

-v Displays verbose information.

-b Specifies the DN of the search/insert base which defines where `ldapentry` starts the search/insert for the entry.

This option is optional if the `LDAP_BASED` variable is set. If specified, this option overwrites the `LDAP_BASEDN` variable setting.

-h Specifies the host name of the LDAP directory. If not specified, `ldapentry` uses the local host.

-p Specifies the TCP port number that the LDAP directory uses. The default is 389.

-D Specifies the distinguished name (DN) of an administrator who has the authority to add, modify, or delete entries in the LDAP directory.

This option is optional if the `LDAP_BINDDN` environment variables has been set. If specified, this option overwrites the `LDAP_BINDDN` variable setting.

service

The name of the service that will determine the type of entry to edit. Can be either passwd, group, hosts, rpc, services, or networks.

value

The name of the entry recognized by the directory to be added, modified, or deleted.

dn

The full distinguished name of the entry to add, modify or delete.

Refer to the ldapentry(1) man page for more detailed information.

Examples

The following configuration variables are defined in the user's configuration file as ~/.ux_ldap_admin_rc:

```
LDAP_BINDDN="cn=Directory Manager" LDAP_HOST="myhost"
```

The Command

```
ldapentry -a passwd UserA
```

will try to bind to the directory on server myhost as Directory Manager, prompt for the credentials, and retrieve the service search descriptor from the profile LDIF file based on the service name passwd. It will then open the template file with the editor defined by the environment variable EDITOR and collect the input to pass it to ldapmodify to add the new entry.

The Command

```
ldapentry -m "uid=UserA, ou=People, o=hp.com”
```

will try to bind to the directory on server myhost as Directory Manager, prompt for the credentials, and use the entered DN to retrieve the entry from the directory. It will then populate a template with the retrieved information, and collect the changes to pass to ldapmodify for execution.

NOTE: Although the ldapentry tool will allow the users to modify any information on the EDITOR window, the directory server has the final decision on accepting the modification. If the user makes an invalid LDIF syntax, violates the directory's schema or does not have the priviledge to perform the modificaiton, the ldapentry tool will report the error after the EDITOR window is closed when it tries to update the directory server with the information. The user will be given the option to re-enter the EDITOR and correct the error.

ldappasswd

This section describes the `ldappasswd` command and its parameters. The `ldappasswd` command, installed in `/opt/ldapux/bin`, is needed on clients that use an LDAP directory replica because the replica cannot be modified by the `passwd(1)` command, or any other command.

Syntax

```
ldappasswd [options]
```

where ***options*** can be any of the following:

-b *basedn* specifies ***basedn*** as the base distinguished name of where to start searching.

-h *host* specifies ***host*** as the LDAP server name or IP address.

-c generates an encrypted password on the client. Use this parameter for directories that do not automatically encrypt passwords. The default is to send the new password in plain text to the directory. Netscape/Red Hat Directory Server for HP-UX supports automatic encryption of passwords.

-v prints the software version and exits.

-p *port* specifies ***port*** as the LDAP server TCP port number.

-D *binddn* specifies ***binddn*** as the bind distinguished name.

-w *passwd* specifies ***passwd*** as the bind password (for simple authentication).

-l *login* specifies ***login*** as the uid of the account to change; defaults to the current user.

Examples

The following is a command the directory administrator can use to change the password in the directory for the user **steves**:

```
ldappasswd -h sys001.hp.com -p 389 -b "ou=people,o=hp.com"
-D "cn=directory manager" -w passwd -l steves
```

ldapsearch

You use the `ldapsearch` command-line utility to locate and retrieve LDAP directory entries. This utility opens a connection to the specified server using the specified distinguished name and password, and locates entries based on the specified search filter. Search results are returned in LDIF format. For detailed information, refer to the *Red Hat Directory Server for HP-UX Configuration, Command, and File Reference* available at the following web site:

http://docs.hp.com/en/internet.html

Syntax

```
ldapsearch -b basedn  [optional_options][filter]
[optional_list_of_attributes]
```

where

filterfilter — Specifies an LDAP search filter. Do not specify a search filter if you supply search filters in a file using the -f option.

optional_options — Specifies a series of command-line options. These must be specified before the search filter, if used.

optional_list_of_attributes — are spaces-separaed attributes that reduct the scope of the attributes returned in the search results. This list of attributes must appear after the search filter. Refer to the *Red Hat Directory Server Administrator's Guide* for details.

ldapsearch Options

This section lists the most commonly used `ldapsearch` command-line options. For more information, refer to *Red Hat Directory Server for HP-UX Configuration, Command and File Reference*.

-b Specifies the starting point for the search. The value specified here must be a distinguished name that currently exits in the database.

-D Specifies the distinguished name (DN) with which to authenticate to the server. If specified, this value must be a DN recognized by the Directory Server, and it must also have the authority to search for the entries.

-h Specifies the hostname or IP address of the Directory Server. If you do not specify a host, `ldapsearch` uses the local host.

-l Specifies the maximum number of seconds to wait for a search request to complete.

-P Specifies the TCP port number that the Directory Server uses. The default is 389.

-s Specifies the scope of the search. The scope can be one of the following:

- base: Search only the entry specified in the –`b` option or defined by the `LDAP_BASEDN` environment variable.
- one: Search only the immediate children of the entry specified in the `-b` option.
- sub: Search the entry specified in the `-b` option and all of its descendants. Perform a subtree search starting at the point identified in the `-b` option. This is the default.

-w Specifies the password associated with the distinguished name that is specified in the -D option.

-x Specifies that the search results are sorted on the server rather than on the client. In general, it is faster to sort on the server rather than on the client.

-f Specifies the file containing the search filter(s) to be used in the search. Omit this opiton if you want to supply a search filter directly to the command-line.

ldapmodify

You use the `ldapmodify` command-line utility to add or modify entries in an existing LDAP directory. `ldapmodify` opens a connection to the specified server using the distinguished name and password you supply, and adds or modifies the entries based on the LDIF update statements contained in a specified file. Because `ldapmodify` uses LDIF update statements, `ldapmodify` can do everything `ldapdelete` can do. For detailed information, refer to the *Red HatDirectory Server for HP-UX Administrator's Guide* available at the following web site:

http://docs.hp.com/en/internet.html

Syntax

```
ldapmodify [optional_options]
```

where

optional_options Specifies a series of command-line options.

ldapmodify Options

The section lists the most commonly used.`ldapmodify` options. For more information, refer to *Red Hat Directory Server for HP-UX Configuration, Command and File Reference*.

-a Allows you to add LDIF entries to the directory without requiring the `changetype:add` LDIF update statement. This provides a simplified method of adding entries to the directory.

-B Specifies the suffix under which the new entries will be added.

-D Specifies the distinguished name (DN) with which to authenticate to the server. If specified, this value must be a DN recognized by the Directory Server, and it must also have the authority to search for the entries.

-f This option specifies the file containing the LDIF update statements used to define the directory modification. If you do not supply this option, the update statements are read from `stdin`.

-h Specifies the hostname or IP address of the Directory Server. If not specified, `ldapmodify` uses the local host.

-p Specifies the TCP port number that the Directory Server uses. The default is 389.

-q Causes each add to be performed silently as opposed to being echoed to the screen individually.

-w Specifies the password associated with the distinguished name that is specified in the `-D` option.

ldapdelete

You use the `ldapdelete` command-line utility to delete entries from an existing LDAP directory. `ldapdelete` opens a connection to the specified server using the distinguished name and password you provide, and deletes the entry or entries. For details, see the *Red Hat Directory Server for HP-UX Administrator's Guide* available at the following web site:

http://docs.hp.com/en/internet.html

Syntax

```
ldapdelete [optional_options]
```

where

optional_options Specifies a series of command-line options.

ldapdelete Options

The section lists `ldapdelete` options most commonly used. For detailed information, refer to *Red Hat Directory Server for HP-UX Configuration, Command and File Reference*.

-D Specifies the distinguished name (DN) with which to authenticate to the server. If specified, this value must be a DN recognized by the Directory Server, and it must also have the authority to delete the entries.

-h Specifies the name of the host on which the Directory Server is running. If you do not specify a host, `ldapdelete` uses the local host.

-P Specifies the TCP port number that the Directory Server uses. The default is 389.

-dn Specifies the `DN` of the entry to be deleted.

-w Specifies the password associated with the distinguished name that is specified in the `-D` option.

Schema Extension Utility

Overview

A directory schema is a collection of attribute type definitions, object class definitions and other information supported by a directory server. Schema controls the type of data that can be stored in a directory server. Although there are some recommended schemas that came originally from the X.500 standards, mostly for representing individuals and organizations, there is no universal schema standard in place for every possible application. Also, there is no standard method for installing the schema definition on a directory server. To support a particular schema definition, LDAP developers are required to manually create schema definition files in the specific format tailored for each version of a supported directory server. They also have to create a custom install program for each variety of directory servers.

To address these issues, LDAP-UX Client Services B.04.10 supports the schema extension utility. This tool queries the current status of the LDAP schema on an LDAP directory server and extends the LDAP server schema with new schema definitions. This tool allows creation of a schema definition in a general format, that can be installed on a number of different directory servers types (such as Netscape/Red Hat Directory Server, Windows Active Directory Server, etc…). A user with valid directory server administration privileges can use this tool to query and extend schema definitions stored in an XML schema definition file into the LDAP directory server.

The Benefits of the Schema Extension Tool

The schema extension tool provides the following benefits:

- Assists application developers to easily install their application schemas to the LDAP directory server.
- Supports automated schema integration into the directory server environment.
- Extends the LDAP directory server schema with new schema definitions dynamically using the schema extension tool, or stores schema extension instructions in the specified file (usually in LDIF format) so the schema can be extended into the directory server manually.
- Reduces user effort in schema extension.
- Simplifies schema management.

How Does the Schema Extension Utility Work

The schema extension utility, `/opt/ldapux/bin/ldapschema`, automatically maps a custom schema definition in a general purpose format to the schema definition format required by the specific LDAP directory server. The Netscape/Red Hat Directory Server and Windows Active Directory Server (ADS) are fully supported by the `ldapschema` tool.

The schema extension utility extends the LDAP directory server with new object classes and attribute types specified in a schema definition file. This utility extends only object classes and attribute types that are not yet defined in a Directory Server schema. No new matching rules or syntaxes can be installed on a Directory Server using this tool. If any attribute types specified in the new schema definition use matching rules or syntaxes that are not defined in the LDAP directory server, the schema extension tool maps these attribute types using alternate matching rules and syntaxes the directory server supports. If no alternate matching rule or syntax is found on an LDAP directory server, the default substitute matching rule or syntax will be used instead. See the "Mapping Unsupported Matching Rules and LDAP Syntaxes" for details.

The schema definitions are stored in an XML format file. This allows you to specify a general schema definition that can be extended on different types and versions of directory servers. See the "Schema Definition File", "Defining Attribute Types" and "Defining Object Classes" sections for details.

For this release of LDAP-UX Client Services, the setup tool has not been integrated with `ldapschema`. You will continue to use the setup tool to extend the Netscape/Red Hat Directory

Server schema with printer, public key and automount schemas. For Windows Active Directory Server, you will continue to run the setup tool to extend the directory server with the automount schema.

Operations Performed by the Schema Extension Utility

The schema extension utility, `ldapschema`, supports the following two modes of operation:

1. Query Schema Status

   Based on the set of attribute types and object classes defined in the input schema definition file, this tool queries their status on the directory server schema without applying any changes to the LDAP directory server. `ldapschema` checks if new attribute types and object classes specified in the input schema file are already defined on the directory server. This tool also determines if definitions installed on the LDAP directory server match definitions specified in the schema file being queried.

2. Extend a Directory Server with Schema Definitions

   This utility supports the extend mode of operation. It can add attribute types and object classes defined in the input schema file that are not yet installed on the LDAP server to that server's schema. Only new valid attribute types and object classes can be added to the LDAP server schema. To execute the `ldapschema` utility in the extend mode, most LDAP directory servers require specifying the distinguished name and password of an administrator who has permissions to modify the schema on that server.

DTD and XML Files Used by ldapschema

The `ldspschema` tool uses the following XML files to perform its operations:

- LDAP Schema Definition Files

  This tool queries and extends the LDAP directory server schema with the input schema definitions stored in an XML schema definition file. Several predefined files (such as rfc3712.xml and rfc2256.xml, etc...) are stored in the `/etc/opt/ldapux/schema` directory. But the schema definition file can be stored in any directory with any file name. The file name is passed to the tool as one of the required arguments. See the "Schema Definition File" (page 148) section for details.

- Documentation Type Definition (DTD) Template

  LDAP-UX provides the predefined Document Type Definition template, `/etc/opt/ldapux/schema/schema.dtd`. Each schema definition file must adhere to DTD template specified in `/etc/opt/ldapux/schema/schema.dtd` file. Every XML file used by the `ldapscheam` utility must include `/etc/opt/ldapux/schema/schema.dtd` as its DTD. This DTD file is used by `ldapschema` to validate new attribute types and object classes before they can be added to the LDAP directory server. See the "Schema Definition File" (page 148) section for details.

  WARNING! Do not modify the `schema.dtd` file, or create your own DTD template file. Modifying this file will cause `ldapschema` to fail.

- Supported Matching Rules and Syntaxes File

  The `ldapschema` utility performs LDAP directory server schema search to obtain the complete list of schema syntaxes and matching rules that the Directory Server supports. Netscape/Red Hat Directory Server provides a list of supported matching rules and syntaxes as part of the schema search.

  However, some directory servers (such as Windows Active Directory Server) do not provide a list of supported syntaxes and/or matching rules as part of the directory server schema search. To support Windows ADS, LDAP-UX provides the predefined LDAP directory server definition file, /etc/opt/ldapux/schema/schema-ads.xml, which contains a list of schema syntaxes that Windows Active Directory Server supports.

If you choose to use the ldapschema tool with the directory server other than Netscape/Red Hat Directory Server or Windows Active Directory Server, and the LDAP directory server doesn't provide a list of supported matching rules and syntaxes as part of the directory server schema search. Then, you need to define your own supported matching rules and syntaxes file. See the “LDAP Directory Server Definition File” section for detailed information on how to create an XML file containing supported matching rules and syntaxes for your directory server.

- Mapping Rules For Unsupported Matching Rules and Syntaxes File

  If matching rules and/or LDAP syntaxes used in attribute type definitions in the schema definition file are not supported on the LDAP directory server, the ldapschema tool maps them using alternate matching rules and syntaxes the LDAP server supports. LDAP-UX provides the /etc/opt/ldapux/schema/map-rules.xml file which defines a list of default substitution matching rules and syntaxes, and alternate matching rules and syntaxes. See the “Mapping Unsupported Matching Rules and LDAP Syntaxes” (page 159) section for details.

ldapschema — The Schema Extension Tool

The `ldapschema` utility allows schema developers to define LDAP schemas using a universal XML syntax, greatly simplifying the ability to support different directory server variations. It can be used to query the current status of the LDAP schema on the LDAP directory server, as well as extend the LDAP directory server schema with new attribute types and object classes. The `ldapschema` utility was designed to support directory servers from several vendors and is currently supported with Netscape Directory Server/Red Hat Directory Server and Microsoft Windows Active Directory Server.

Syntax for ldapschema

```
ldapschema -q <schema> -T <ds_type> -V <ds_version> [options]

ldapschema -e <schema> -T <ds_type> -V <ds_version> [options]
```

Required Command Options

The following describes required options:

-q <schema> Queries the schema status on the LDAP directory without applying any changes to the LDAP directory server. The schema definitions can be obtained from the file name specified in the `<schema>` argument. `ldapschema` checks if any attribute types and/or object classes of the LDAP schema are already installed on the LDAP server. Also, determines if definitions installed on the LDAP server match definitions specified in the schema file being queried. See the “Schema Definition File” section for details.

-e <schema> Extends the LDAP directory server schema with attribute types and object classes defined in the specified schema. Schema definition is obtained from the schema file. See the “Schema Definition File” section for details. On most LDAP directory servers this option requires specifying the `-D binddn` option and either the `-j filename` or the `-w -` option to specify the credentials of an administrator who has permissions to modify the schema on the directory server.

-T ds_type Specifies type of LDAP directory server.

The following types of LDAP directory servers are fully supported by `ldapschema`:

Table 6-5 Supported Directory Servers

Type of Directory Server	ds_type
Windows Active Directory Server	ads
Netscape/Red Hat Directory Server	rhds

The `ldapschema` utility may work with other types of LDAPv3 directory servers., although its behavior has not been verified.

The following table lists names of LDAPv3 directory servers which are reserved for future support:

Table 6-6 Reserved LDAPv3 Directory Servers

Type of Directory Server	ds_type
openLDAP Directory Server	openldap
Oracle Information Directory	oracle

Table 6-6 Reserved LDAPv3 Directory Servers *(continued)*

Novell e-Directory Server	eDirectory
IBM Tivoli Directory Server	ibm
MAC OS X Directory Server	mac
Sun One Directory Server	sun
Computer Associates Directory Server	ca
iPlanet Directory Server	iPlanet

-V ds_version The version of the LDAP directory server. The `strcasecmp()` function compares the version specified by this –V option and the version defined in the XML files the `ldapschema` utility processes. The version specified by the –V option and the version defined in the XML files must be consistent. For example, the schema definition file contains the following object class definition:

```
<objectClassDefinition>
       <oid>1.2.345.6.789</oid>
       <name>sampleObject</name>
       <must>sampleAttributeA</must>
       <must only="rhds"
       versionGreaterOrEqual="6.2">sampleAttributeB</must>
<objectClassDefinistion>
```

If the `ldapschema` utility is called with `<ds_version>` set to “6.2.1”, the `sampleObject` definition has two mandatory attributes, `sampleAttributeA` and `sampleAttributeB`. The strcasecmp(“6.2.1”, “6.2”) returns a positive integer, so `sampleAttributeB` is included in the definition of the object class `sampleObject`.

On the other hand, if the `ldapschema` utility is called with <ds_version> set to “6.02.1”, the sampleObject definition has only one mandatory attribute, `sampleAttributeA`. The strcasecmp(“6.02.1”, “6.2”) returns a negative integer, so `sampleAttributeB` is not included in the definition of the object class `sampleObject`.

The `ldapschema` utility ignores `<ds_version>` if the LDAP directory server version-specific attributes “`versionGreaterOrEqual`” and “`versionLessThan`” are not used in the XML files being processed (i.e., the schema definition files, the LDAP directory server definition file and the mapping rules file). If the XML files include any definitions with “`versionGreaterOrEqual`” attribute set, `strcasecmp()` must return zero or a positive integer to include directory-specific information in the LDAP schema definition. If the XML files include any definitions with “`versionLessThan`” attribute set, `strcasecmp()` must return a negative integer to include directory-specific information in the LDAP schema definition. Also, “`versionGreaterOrEqual`” and “`versionLessThan`” can be used simultaneously to define a range of version of the LDAP directory server. See “Defining Directory Specific Information” (page 154) section for details.

Additional Options (Optional)

The following describes a list of options which are optional:

`-h hostname` Specifies the LDAP directory server host name or IP address. (Default: localhost)

`-p <port>` Specifies the LDAP directory server TCP port number. (Default: 389 for regular connections, 636 for SSL connections.)

`-D <binddn>` Specifies Distinguished Name (DN) of an administrator who has permissions to read and modify LDAP directory server schema.

`-j <filename>` Specifies an administrator's password in the file (for simple authentication).

`-w-` Inputs an administrator's password from the prompt (for simple authentication).

`-Z` Establishes an SSL-encrypted connection.

`-ZZ` Specifies StartTLS request.

`-ZZZ` Enforces startTLS request (requires successful server response).

`-P path` Specifies path to SSL certificate database. (Default: /etc/opt/ldapux)

`-3` Verifies the host name in SSL certificates.

`-s-` Disables syntax substitution in attribute types. Normally, if an attribute type uses an LDAP syntax not supported on the LDAP directory server, it is mapped to use a higher level (more inclusive) syntax supported by that server. If this option is specified, any attribute types that use unsupported LDAP syntax will not be added to the LDAP directory server schema. See "Mapping Unsupported Matching Rules and LDAP Syntaxes" section for more details.

`-m-` Disables matching rule substitution in attribute types. Normally, if an attribute type uses a matching rule not supported on the LDAP directory server, it is mapped to use a higher level (less specific) matching rule supported by that server. If this option is specified, any attribute types that use unsupported matching rules will not be added to the LDAP directory server schema. See the "Mapping Unsupported Matching Rules and LDAP Syntaxes" section for more details.

`-f <filename>` Stores schema extension instructions in the specified file (usually in LDIF format). Do not apply any changes to the LDAP directory server schema. This option requires specifying the `-e` option.

`-F` Forces installation of schema even if it contains any invalid attribute type or object class definitions, or some components specified in the schema file are already present in the LDAP directory server.

Security

For security reasons, the LDAP administrator's password may not be specified on the command line. It can be specified at the prompt (-w - option), in a file (-j <filename> option), or using the `LDAP_BINDCRED` environmental variable described in the "Environment Variables" section below.

Environment Variables

The `ldapschema` utility supports the following environment variables:

LDAP_BINDDN The Distinguished Name (DN) of an administrator who has permissions to read and modify LDAP directory server schema.

LDAP_BINCRED The password for the privileged LDAP directory user.

LDAP_HOST The host name of the LDAP directory server. The `LDAP_HOST` variable uses the "hostname:port" format. If the port is not specified, default port number is 389 for regular connections, or 636 for SSL connections.

Options specified on the command line override the values in environment variables. For example, if the `-j /home/secret.txt` option is specified on the command line, and the `LDAP_BINDCRED` environmental variable is set, the password of the LDAP directory server administrator is obtained from the `/home/secret.txt` file.

Examples

This section describes examples using the `ldapschema` tool.

An Example for Querying the Schema Status

The following command queries the status of RFC 2307 schema on Red Hat directory server, `ldaphost`, with version 7.1:

```
ldapschema -q /etc/opt/ldapux/schema/rfc2307.xml -h ldaphost -T rhds
 -V 7.1
```

The LDAP directory server version number bears no effect unless also specified in the XML files being processed. Version specification must follow the same format as version specification used in the `/etc/opt/ldapux/schema/rfc2307.xml` and `/etc/opt/ldapux/map-rules.xml` files.

An Example for Extending the New Schema into the Directory Server

The following procedures are used to extend Red Hat Directory Server, `ldaphost`, with custom `Sample` schema:

1. Create the schema definition file, `/etc/opt/ldapux/schema/sample.xml`, which contains attribute type and object class definitions for the `Sample` schema.
2. This step is recommended. Query the current status of the `Sample` schema on the server by running the following command:

   ```
   ldapschema -q /etc/opt/ldapux/schema/sample.xml -h ldaphost
   -T rhds -V 7.1 -D "<binddn>" -j /tmp/secret.txt
   ```

   The directory manager password can be specified at the prompt (-w - option) or in a file (-j <password_file> option).
3. Based on the results produced by Step 2, correct any invalid definitions.
4. Extend the Red Hat Directory Server schema with new `Sample` schema elements by executing the following command:

   ```
   ldapschema -e /etc/opt/ldapux/schema/sample.xml -h ldaphost
   -T rhds -V 7.1 -D "<binddn>" -j /tmp/secret.txt
   ```

Note that LDAP directory server version number bears no effect unless also specified in the XML files being processed. Version specification must follow the same format as version specification used in the `/etc/opt/ldapux/schema/sample.xml` and `/etc/opt/ldapux/schema/map-rules.xml` files.

Schema Definition File

The ldapschema utility queries and extends LDAP directory server based on the XML schema definition file. When using the ldapschema tool, the schema argument used with the -q or -e option must correspond to the XML file containing the appropriate schema definition.

Several predefined files (such as rfc3712.xml, rfc2256.xml, etc...) are stored in the /etc/opt/ldapux/schema directory. But the schema definition file can be stored in any directory with any file name.

Each schema definition file must adhere to Document Type Definition (DTD) template specified in /etc/opt/ldapux/schema/schema.dtd file. Every XML file used by the ldapschema utility must include /etc/opt/ldapux/schema/schema.dtd as its DTD. See Line 2 in the "A Sample RFC3712.xml File " (page 149) section below.

⚠ **WARNING!** Every XML file used with ldapschema utility must include /etc/opt/ldapux/schema/schema.dtd file as its DTD template. Do not modify this file, or create your own DTD template file. The /etc/opt/ldapux/schema/schema.dtd file is created to validate attribute type and object class definitions before they can be added to the LDAP directory server schema. Altering the format of any schema elements in this file will cause ldapschema to fail.

The schema definition file, enclosed by <schemaDefinition> tags, specifies schema name, schema description and schema source, followed by any number of attribute type and object class definitions. The schema name, schema description and schema source XML tags are optional.

The following describes the schemaName, schemaDescription, and schemaSource tags:

<schemaName>	Optional, specifies the name of schema definition file.
<schemaDescription>	Optional, contains a brief one line schema description.
<schemaSource>	An optional field used to specify the X-ORIGIN field of extended attribute types and object classes, if used.

In the schema definition file, after general schema information is specified, attribute type definitions, if any, must be specified followed by any object class definitions.

A Sample RFC3712.xml File

A sample `rfc3712.xml` file below defines two attribute types, `printer-name` and `printer-aliases`, followed by one object class, `printerLPR`, as specified in RFC3712:

```
Line 1:  <?xml version="1.0" encoding="UTF-8"?>
Line 2: <!DOCTYPE schemaDefinition SYSTEM "/etc/opt/ldapux/schema/schema.dtd">
Line 3:
LINE 4:  <schemaDefinition>
Line 5:
Line 6:  <schemaName>rfc3712</schemaName>
Line 7:  <schemaDescription>Printer Services Schema</schemaDescription>
Line 8:  <schemaSource>RFC3712</schemaSource>
Line 9:
Line 10: <attributeTypeDefinition>
LINe 11:       <oid>1.3.18.0.2.4.1135</oid>
Line 12:       <name>printer-name</name>
Line 13:      <desc>A site-specific administrative name of this printer</desc>
Line 14:       <equality>caseIgnoreMatch</equality>
Line 15:       <substr>caseIgnoreSubstringsMatch</substr>
Line 16:       <syntax>1.3.6.1.4.1.1466.115.121.1.15</syntax>
Line 17:       <length>127<length>
LIne 18:       <singleValued/>
Line 19: </attributeTypeDefinition>
Line 20:
Line 21: <attributeTypeDefinition>
LINe 22:       <oid>1.3.18.0.2.4.1108</oid>
Line 23:       <name>printer-aliases</name>
Line 24:       <desc>Names in addition to the printer-name</desc>
Line 25:       <equality>caseIgnoreMatch</equality>
Line 26:       <substr>caseIgnoreSubstringsMatch</substr>
Line 27:       <syntax>1.3.6.1.4.1.1466.115.121.1.15</syntax>
Line 28:       <length>127<length>
Line 29: </attributeTypeDefinition>
Line 30:
inee 31: <objectClassDefinition>
LINe 32:       <oid>1.3.18.0.2.6.253</oid>
Line 33:       <name>printerLPR</name>
Line 34:       <desc>LPR information</desc>
Line 35:       <type>AUXILIARY</type>
Line 36:       <must>printer-name</must>
Line 37:       <may>printer-aliases</may>
Line 38: </objectClassDefinition>
Line 39:
Line 40: </schemaDefinition>
```

NOTE: Line 1–2 are required in every schema definition file. Attribute type and object class definitions closely follow the format specified in RFC 2252. Values specified for all XML tags except the `<dsSpecific>` tag must not be quoted. Only the description field (enclosed by <desc>...</desc> tags) can contain spaces.

Defining Attribute Types

Each attribute type definition, enclosed by <attributeTypeDefinition> tags, can contain the following case-sensitive tags, in the order specified:

<oid> Required. Exactly one numeric id must be specified. The <oid> value must adhere to RFC 2252 format specification.

<name> Required. At least one attribute type name must be specified. Do not use quotes around the name values. The <name> value must adhere to RFC 2252 format specification.

<displayName> Optional. At most one display name can be specified. This tag specifies a display name of the attribute type used by LDAP clients and administrative tools. Currently, <displayName> applies only to Active Directory Server (ADS) to specify lDAPDisplayName and adminDisplayName if different from the <name> value.

<desc> Optional. At most one description can be specified. Do not use quotes around the description value.

<obsolete> Optional, use only if applicable. Obsolete attribute types cannot be used in definitions of any other attribute types or object classes. At most one obsolete flag can be specified.

<subTypeOf> Optional, use if an attribute type has a super-type. At most one super-type can be specified. The specified super-type must already exist on the LDAP directory server, or its definition must be specified in the same schema definition file.

<equality> Optional. At most one equality rule can be specified.

<ordering> Optional. At most one ordering rule can be specified.

<substr> Optional. At most one substring matching rule can be specified.

<syntax> Required if an attribute type has no super-type. At most one LDAP syntax value can be specified.

<length> Optional indication of the maximum length of a value of this attribute. RFC 2252 specifies this value in curly braces following the attribute type’s syntax. For instance, "1.3.6.4.1.1466.0{64}” can be expressed using the following tags:

```
<syntax>1.3.6.4.1.1466.0</syntax>
<length>64</length>
```

At most one syntax length value can be specified. <length> must contain a positive integer value.

<singleValued> Optional, use if the SINGLE-VALUE flag is set. At most one singleValued flag can be specified.

<collective> Optional, use if the COLLECTIVE-VALUE flag is set. At most one collective flag can be specified.

<noUserModification> Optional, use if NO-USER-MODIFICATION flag is set. At most one noUserModification flag can be specified.

<usage> Optional, must contain one of the following possible values:

- userApplications
- directoryOperation
- distributedOperation
- dSAOperation

At most one usage value can be specified..

`<indexed>`	Optional, use if an attribute type requires indexing. At most one `indexed` flag can be specified.
`<dsSpecific>`	Optional, use to specify any directory-specific information about the attribute type. See "Defining Directory Specific Information" (page 154) section for details.

Attribute Type Definition Requirements

To add the new schema to the LDAP directory server, each attribute type definition must meet the following requirements:

- The attribute type has a `<oid>` tag with one numeric id value which adheres to RFC 2252 format specification.
- The attribute type has at least one `<name>` tag with the attribute type name. Each name must adhere to RFC 2252 format specification.
- No other attribute types in the schema definition file or on the LDAP directory server have the same OID value.
- No other attribute types in the schema definition file or on the LDAP directory server have the same name values.
- The specified super-type used by the attribute type must already exit on the LDAP directory sever or its definition must be specified in the same schema definition file.
- The attribute type specifies either an LDAP syntax value or a super-type. Some directory servers, for example ADS, do not support attribute type inheritance. For such directory servers, the LDAP syntax for the sub-type attribute is obtained from the super-type definition and the super-type/sub-type relationship is ignored
- The matching rules and syntaxes used by this attribute type are supported by the LDAP directory server. See the "Mapping Unsupported Matching Rules and LDAP Syntaxes" section for details.
- The inheritance hierarchy has no cycles (no circular dependencies exist in the super-class/sub-class relationships).
- If the attribute type has a super-type, they both have the same value defined in the `<usage>` tag.

Defining Object Classes

Each object class definition, enclosed by the `<objectClassDefinition>` tags, can contain the following case-sensitive tags, in the order specified:

`<oid>` Required. Exactly one numeric id must be specified. The <oid> value must adhere to RFC 2252 format specification.

`<name>` Required. At least one object class name must be specified. Do not use quotes around the name values. The <name> value must adhere to RFC 2252 format specification.

`<displayName>` Optional. At most one display name can be specified. This tag specifies a display name of the object class used by LDAP clients and administrative tools. Currently, <displayName> applies only to Active Directory Server (ADS) to specify lDAPDisplayName and adminDisplayName if different from the <name> value.

`<desc>` Optional. At most one object class description can be specified. Do not use quotes around the description value.

`<obsolete>` Optional, use only if applicable. Obsolete object class cannot be used in definitions of any other object classes. At most one obsolete flag can be specified.

`<subClassOf>` Optional, use if an object class has super-classes. The specified super-class must already exist on the LDAP directory server, or its definition must be specified in the same schema definition file. If the LDAP directory server allows only one super-class, then only the first `<subClassOf>` value will be used.

`<type>` Optional, must contain one of the following possible values: `STRUCTURAL`, `AUXILIARY`, `ABSTRACT`. At most one type value can be specified.

`<must>` Optional, use if an object class has mandatory attributes. The specified attributes must already exist on the LDAP directory, or its definition must be specified in the same schema definition file.

`<may>` Optional, use if an object class has optional attributes. The specified attributes must already exist on the LDAP directory server, or its definition must be specified in the same schema definition file.

`<rdn>` Optional, defines the recommended attribute to use for the relative distinguished name for new entries created with this object class. Currently, `<rdn>` applies only to Active Directory Server (ADS). At most one RDN can be specified.

`<extendAuxiliaryClass>` Optional, applies only to `AUXILIARY` object classes. This tag is used to extend an object class already defined in the LDAP server schema with this new `AUXILIARY` object class. Currently, `<extendAuxiliaryClass>` applies only to Active Directory Server (ADS) to include the new AUXILIARY class as an “auxiliaryClass” in the definition of another object class already defined in the LDAP server schema.

`<dsSpecific>` Optional, use to specify any directory-specific information about the object type. See “Defining Directory Specific Information” (page 154) section for details.

Object Class Definition Requirements

To add the new schema to the LDAP directory server, each object class definition must meet the following requirements:

- The object class definition contains a `<oid>` tag with one numeric id value which adheres to RFC 2252 format specification.
- The object class definition has at least one `<name>` tag with the object class name. Each name must adhere to RFC 2252 format specification.
- No other object classes in the schema definition file or on the LDAP directory server have the same numeric id value.
- No other object classes in the schema definition file or on the LDAP directory server have the same name value.
- The super-class(es) used by the object class must be defined.
- The attribute(s) used by the object classes must be defined.
- The inheritance hierarchy has no cycles (no circular dependencies exist in the super-class and sub-class relationships).
- An `ABSTRACT` object class can specify only `ABSTRACT` object class(es) as its super-class(es).
- An `AUXILIARY` object class can specify `ABSTRACT` or `AUXILIARY` object class(es) as its super-class(es).
- A `STRUCTURAL` object class can specify `ABSTRACT` or `STRUCTURAL` object class(es) as its super-class(es).

Predefined Schema Definition Files

The following LDAP schema definition files are delivered with the LDAP-UX product:

- `/etc/opt/ldapux/schema/rfc2256.xml`
- `/etc/opt/ldapux/schema/rfc2307.xml`
- `/etc/opt/ldapux/schema/rfc2307-bis.xml`
- `/etc/opt/ldapux/schema/rfc2926.xml`
- `/etc/opt/ldapux/schema/rfc3712.xml`

These files are provided as examples to demonstrate how to define new LDAP schema definition files to use with the ldapschema utility. Since these files define attribute types and object classes that come pre-installed on most LDAP directory servers they are not intended for extending the LDAP directory server schema. Instead, these files are provided for reference when creating the new schema definition files to query and extend the LDAP directory server schema with the new attribute type and object class definitions.

Defining Directory Specific Information

Attribute type and object class definitions can be extended with directory-specific information using the <dsSpecific> tag. This is useful to maintain a single schema definition file for different types and versions of LDAP directory servers.

An Example of Defining Directory Specific Information in the Attribute Type Definition

This section takes an example to illustrate how directory specific information can be specified in a single attribute type definition to support Netscape/Red Hat Directory Server and Windows Active Directory Server specific definitions simultaneously.

The following is an example of the attribute type definition with directory specific information using the <dsSpecific> tag:

```
<attributeTypeDefinition>
      <oid>1.23.456.7.89101112.1.314.1.51.6<oid>
      <name>sampleAttribute</name>
      <displayName vendor="ads">
      versionGreaterOrEqual="2003">my-sample-attribute</displayName>
      <equlaity>caseIgnoreMatch</equality>
      <syntax>1.3.6.1.4.1.1466.115.121.1.15</syntax>
      <dsSpecific vendor="rhds" versionGreaterorEqual="6.2"
                                versionLessThan="7.1"
           <field attr="X-ORIGIN">'Custom Schema'</field>
      </dsSpecific>
      <dsSpecific vendor="ads" versionLessThan="2003">
           <field attr="systemOnly">TRUE</field>
           <field attr="rangeLower">256</field>
      </dsSpecific>
      <dsSpecific vendor="ads" versionGreaterOrEqual="2003">
           <field attr="rangeLower">512</field>
      </dsSpecific>
 </attributeTypeDefinition>
```

For the above example, on Red Hat Directory Server 6.2 through 7.0, the X-ORIGIN flag for sampleAttribute will be set to 'Custom Schema' as specified in the dsSpecific field. On Red Hat Directory Server 6.1 and earlier, or 7.1 and later, the X-ORIGIN flag for sampleAttribute will be set to the value specified in the <schemaSource>

On Active Directory Server 2000, the sampleAttribute is added using the same display name as specified by the <name> value, with the rangeLower attribute set to 256, and the systemOnly attribute set to TRUE.

On Active Directory Server 2003, the sampleAttribute is added using "my-sample-attribute" display name, with the rangeLower attribute set to 512, and the systemOnly attribute set to FALSE, which is the default value.

Table 6-7 Directory Specific Information

Attribute	RHDS 6.2–7.0	RHDS 7.1	ADS 2000	ADS 2003
Name	sampleAttribute	sampleAttribute	sampleAttribute	sampleAttribute
Display Name	N/A	N/A	sampleAttribute	my-sample-attribute
X-ORIGIN	'Custom Schema'	As Specified in <schemaSource>	N/A	N/A
systemOnly	N/A	N/A	TRUE	FALSE (default)
rangeLower	N/A	N/A	256	512

Also, the 1.3.6.1.4.1.1466.115.121.1.15 syntax is not supported on the Windows ADS, it is mapped to the corresponding Directory String syntax supported on Windows ADS, which is `attributeSyntax = 2.5.5.12,oMSyntax=64`. See “Mapping Unsupported Matching Rules and LDAP Syntaxes” (page 159) section for details.

An Example of Defining Directory Specific Information in the Object Class Definition

Directory specific information can be specified in the object class definitions as well as in optional and mandatory attributes.

The following is an example of the object class definition with directory specific information using the `<dsSpecific>` tag and XML attributes, `not` and `only`:

```
<objectClassDefinition>
      <oid>1.23.456.7.89101112.1.314.1.51.7<oid>
      <name>sampleObject</name>
      <must only="ads">serverRole</must>
      <must not="ads">userPassword</must>
      <may>sampleAttribute</may>
      <dsSpecific vendor="ads">
          <field attr="systemOnly">TRUE</field>
      </dsSpecific>
</objectClassDefinition>
```

For the above example, on Windows Active Directory Server, this object class has a mandatory attribute type, `serverRole`, and an optional attribute type, `sampleAttribute`. On all other types of directory servers, this object class has a mandatory attribute type, `userPassword` and an optional attribute, `sampleAttribute`. On Windows Active Directory Server, this object class has the `systemOnly` attribute set to `TRUE`.

NOTE: Directory-specific attributes and values specified in `<dsSpecific>` fields are not validated. You need to ensure that the values specified in these fields are legitimate and adhere to the LDAP directory server rules. The field value must be specified exactly as it is to appear in the attribute type or object class definition, using single and double quotes as applicable.

Attributes and values specified in the `<dsSpecific>` fields override the default attribute type and object class configurations. For example, on Windows Active Directory Server the default value of the `isDefunct` attribute is set to `False`. If the following `<dsSpecific>` attribute is defined, the specific setting will override the default setting and will result in the element being defunct.

```
<dsSpecific vendor="ads">
    <field attr="isDefunct">TRUE</field>
</dsSpecific>
```

LDAP Directory Server Definition File

In order to properly install new attribute types in an LDAP directory server schema, the ldapschema utility needs to determine whether the LDAP server supports the matching rules and LDAP syntaxes used by the new attribute type definitions. The `ldapschema` utility performs an LDAP search for supported matching rules and syntaxes on the LDAP server. However, some types of directory servers do not provide this information as part of the search.

You can perform the following commands to determine if your directory server returns information about supported matching rules and LDAP syntaxes

1. To determine <schema DN>, run the following command:

   ```
   /opt/ldapux/bin/ldapsearch —b "" —s base “(objectclass=*)”
   subsechemasubentry
   ```

2. To obtain a list of supported matching rules and LDAP syntaxes, run the following command using schema DN information obtained from step 1:

   ```
   /opt/ldapux/bin/ldapsearch —b "<schema DN>" —s base “(objectclass=*)”  \
   matchingRules ldapSyntaxes
   ```

If the latter LDAP search in step 2 does not return a complete list of supported matching rules and LDAP syntaxes, the directory server definitions must be specified in the `/etc/opt/ldapux/schema/schema-<ds_type>.xml` file. The `<ds_type>` value must correspond to the same value specified with the `-T` option on the `ldapschema` command line. The case defined in `<ds_type>` must match identically to the case specified in the `-T` argument.

The LDAP directory server definition, enclosed by `<dsSchemaDefintion>` tags, optionally specifies schema description, followed by any number of supported matching rules and LDAP syntaxes definitions. For example, LDAP-UX provides the `/etc/opt/ldapux/schema/schema-ads.xml` file which can be used to obtain a list of syntaxes and matching rules that Windows ADS supports. Run `ldapschema` with the `–T ads` option, the corresponding directory server definition is obtained from the `/etc/opt/ldapux/schema/schema-ads.xml` file.

After general schema information is specified, supported matching rules, if any, must be specified followed by any supported LDAP syntaxes definitions.

An Example of the Directory Server Definition File

The example below defines two syntaxes with <oid> values of 2.5.5.1 and 2.5.5.2 supported on Windows ADS:

```
Line 1:  <?xml version="1.0" encoding="UTF-8"?>
Line 2: <!DOCTYPE dsSchemaDefinition SYSTEM "/etc/opt/ldapux/schema/schema.dtd">
Line 3
LINE 4:  <dsSchemaDefinition>
LINE 5:
Line 6:  <schemaDescription>ADS Syntaxes</schemaDescription>
Line 7:
Line 8:  <syntaxDefinition vendor="ads">
LINe 9:        <oid>2.5.5.1</oid>
Line 10:       <dessc>Distinguished Name</desc>
Line 11:       <oMSyntax>127</oMSyntax>
Line 12: </syntaxDefintion>
Line 13:
Line 14: <syntaxDefinition vendor="ads">
LINe 15:       <oid>2.5.5.2</oid>
Line 16:       <desc>Object Identifier</desc>
Line 17:       <oMSyntax>6</oMSyntax>
Line 18: </syntaxDefintion>
LINE 19:
```

```
Line 20: </dsSchemaDefintion>
```

Lines 1-2 are required in every LDAP directory server definition file. LDAP syntax and matching rule definitions closely follow the format specified in RFC 2252. Values specified for all XML tags must not be quoted. Only the description field (enclosed by <desc>...<desc> tages) can contain spaces.

NOTE: Only LDAP syntaxes and matching rules fully supported by the LDAP directory server can be specified in this file. The `vendor`, `versionGreaterOrEqual` and `versionLessThan` attributes can be used to specify directory specific information.

See the `/etc/opt/ldapux/schema/schema-ads.xml` file for an example of LDAP directory server definition files.

Defining Matching Rules

Each `<syntaxDefinition>` tag can contain the following case-sensitive tags, in the order specified:

`<oid>` Required. Exactly one numeric id must be specified.

`<name>` Required. At least one matching rule name must be specified. Do not use quotes around the name values.

`<desc>` Optional. At most one description can be specified.

`<obsolete>` Optional, use it only if it is applicable. Obsolete matching rules cannot be used in definitions of any other attribute types. At most one obsolete flag can be specified.

`<syntax>` Required. The syntax used by the matching rule definition must also be supported on the LDAP directory server. At most one LDAP syntax value can be specified per matching rule definition.

Defining LDAP Syntaxes

Each `<syntaxDefinition>` tag can contain the following case-sensitive tags, in the order specified:

`<oid>` Required. Exactly one numeric id must be specified.

`<desc>` Optional. At most one description can be specified.

`<oMSyntax>` Required on Windows ADS only, ignored on other types of LDAP directory servers

Mapping Unsupported Matching Rules and LDAP Syntaxes

If matching rules and/or LDAP syntaxes used in attribute type definitions in the schema definition file are not supported on the LDAP directory server, the `ldapschema` tool maps them to alternate matching rules and syntaxes the LDAP server supports. LDAP-UX provides the `/etc/opt/ldapux/schema/map-rules.xml` file which defines a list of default substitution matching rules and syntaxes, and alternate matching rules and syntaxes.

The matching rules are specified in `<equality>`, `<ordering>` or `<substr>` in the attribute type definition. The LDAP syntax is specified in the `<syntax>` tag of the attribute type definition.

The purpose of the mapping rules file is to allow an LDAP schema to be installed on an LDAP directory server even if some of matching rules and LDAP syntaxes used in the definition of that schema are not supported by the directory server. The `/etc/opt/ldapux/schema/map-rules.xml` file uses the following mapping rules guideline:

- Map more restrictive syntaxes to less restrictive syntaxes.
- Map more specific matching rules to less specific matching rules.

For example, the Integer syntax contains a subset of characters of the IA5 string syntax. Therefore, it is acceptable to map the Integer syntax to the IA5 string syntax, since the IA5 string syntax is a super-set of the integer syntax.

Examples of Alternate Matching Rules and Syntaxes in /etc/opt/ldapux/map-rules.xml

The following shows examples of alternate matching rules and syntaxes defined in the `/etc/opt/ldapux/map-rules.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mappingPolicies SYSTEM "/etc/opt/ldapux/schema/schema.dtd">

<mappingPolicies>
<defaultMatchingRulesReplacements>
      <defaultMatchingRule>
            <matchingRule>caseIgnoreMatch</matchingRule>
      </defaultMatchingRule>
</defaultMatchingRulesReplacements>

<defaultSyntaxesReplacements>
      <defaultSyntax only="ads">
            <syntax>2.5.5.12</syntax>
            <desc>Active Directory String syntax.</desc>
            <oMSyntax>64</oMSyntax>
      </defaultSyntax>

      <defaultSyntax not="ads">
            <syntax>1.3.6.1.4.1.1466.115.121.1.15</syntax>
            <desc>Directory String syntax.</desc>
      </defaultSyntax>
</defaultSyntaxesReplacements>

<matchingRulesReplacements>
      <matchingRules>
            <matchingRule>IntegerMatch</matchingRule>
            <subRule>
                <matchingRule>numericStringMatch</matchingRule>
            </subRule>
      </matchingRules>
</matchingRulesReplacements>

<syntaxesReplacements>
   <syntaxes>
      <syntax>1.3.6.1.4.1.1466.115.121.1.26</syntax>
      <desc> IA5 String Syntax.</desc>
      <equivSyntax>
```

```
            <syntax>2.5.5.5</syntax>
            <desc>Active Directory IA5 String LDAP Syntax.</desc>
            <oMSyntax>22</oMSyntax>
      </equivSyntax>
      <subSyntax>
            <syntax>1.3.6.1.4.1.1466.115.121.1.15</syntax>
            <desc>Directory String syntax.</desc>
      </subSyntax>
   </syntaxes>
</syntaxesReplacements>
</mappingPolicies>
```

How Does ldapschema Map Unsupported Matching Rules and LDAP Syntaxes

If any mapping rules or the syntax used by an attribute type are not supported on the LDAP server, the `ldapschema` utility checks if the appropriate substitution rule is specified in the `/etc/opt/ldapux/map-rules.xml` file. If it is specified, `ldapschema` locates the first available matching rule or syntax supported on the LDAP server, and uses it in the attribute type definition instead. If the substitution rule is not specified, or none of the substitution matching rules or syntaxes are supported on the LDAP directory server, `ldapschema` checks if the default substitution can be used.

The "`vendor`", "`versionGreaterOrEqual`" and "`versionLessThan`" XML attributes can be used to specify directory-specific information stored in `<defaultMatchingRule>` and `<defaultSyntax>` tags. If the default substitution is not supported on the LDAP server, the attribute type cannot be added to the LDAP directory server schema.

Examples

For example, an attribute type with `IA5String` syntax (`1.3.6.1.4.1.1466.115.121.1.26`) is installed on Windows ADS, where this IA5 String syntax is not supported. `ldapschema` will try using the first specified equivalent or substitution syntax supported by the target LDAP directory server. The specified equivalent syntax of `2.5.5.5` syntax with the `oMSyntax` value of 22 is supported on windows ADS and will be used in place of the original syntax value.

As another example, assume an attribute type with a Boolean equality rule is being installed on the LDAP server where this matching rule is not supported. Since no substitution policy is specified for this matching rule in the example above, the default substitution matching rule, `caseIgnoreMatch`, would be used instead, if the LDAP server supports it. If the LDAP server does not support `caseIgnoreMatch`, that attribute type cannot be installed on the LDAP server, unless its definition is modified to use another supported equality matching rule.

If the `-s-` option is specified in the `ldapsechema` tool, syntax substitution in attribute types is disabled. Any attribute types with unsupported LDAP syntaxes will not be added to the LDAP directory server schema. The `-m-` option with the `ldapschema` tool disables matching rule substitution. Any attribute types with unsupported matching rules will not be added to the LDAP directory server schema.

Return Values From ldapschema

The `ldapschema` tool returns the following values:

`0` The operation is successful.

`-1` The operation fails.

In addition, `ldapschema` prints to STDOUT the overall status of the schema being queried or extended. Based on the schema status, any combination of the following messages is displayed. Detailed explanations of each message are specified in the square brackets following the message body text.

Schema Status Messages

`SCHEMA_NEW`

The `<schema>` file contains attribute types and object classes that are not defined in the LDAP directory server schema.

[The **`SCHEMA_NEW`** message indicates all attribute types and object classes defined in the <schema> file are new to the LDAP directory server. The **`SCHEMA_NEW`** message indicates none of the specified definitions are currently installed in the LDAP server schema.]

`SCHEMA_FOUND`

Subset of attribute types and/or object classes defined in the `<schema>` file are already part of the LDAP server schema.

[The `SCHEMA_FOUND` message indicates one or more attribute type or object class definitions specified in the <schema> file are already installed in the LDAP server schema. Such elements will be excluded from being extended on the LDAP server. Only attribute types and object classes with new and unique numeric oids and names can be added to the LDAP server schema. Check the messages containing ATTRIB_FOUND and OBJECT_FOUND described below for details.

The ldapschema utility may install any remaining new elements that are not already defined in the LDAP server schema if both of the following two conditions are met:

- The LDAP schema defined in the <schema> file is compatible with the LDAP server schema. The two schemas are compatible if the definitions of any elements found in the LDAP server schema match their definitions specified in the <schema> file.

  If the `SCHEMA_MISMATCH` message is displayed, the two schemas are not compatible. This means one or more elements installed on the LDAP server have definitions different from those specified in the <schema> file. Installation of any remaining new elements is not recommended. See definition of the `SCHEMA_MISMATCH` message below.

  If the `SCHEMA_MISMATCH` message is not displayed, the two schemas are compatible. The schema specified in the <schema> file partially exists on the LDAP server schema, and can be extended with any remaining new valid attribute type and object class definitions.

- The LDAP schema defined in the <schema> file is valid.

  If the `SCHEMA_INVALID` message is displayed, one or more definitions specified in the <schema> file are invalid and cannot be added to the LDAP server schema. Such definitions need to be corrected before the new schema elements can be extended on the LDAP server.

If the SCHEMA_INVALID message is not displayed, the schema definition in the <schema> file is valid. It partially exists on the LDAP server schema, and can be extended with any remaining new valid attribute type and object class definitions.]

SCHEMA_EXISTS

No changes to the LDAP server schema are needed. All attribute types and object classes defined in the <schema> file are already part of the LDAP directory server schema.

[The SCHEMA_EXISTS message indicates the schema specified in the <schema> file is already installed on the LDAP directory server. All attribute types and object classes defined in the <schema> file are already part of the schema on the LDAP directory server. Only attribute types and object classes with new and unique numeric oids and names can be added to the LDAP server schema. Check the messages containing ATTRIB_FOUND and OBJECT_FOUND described below for details. Since the definitions specified in the <schema> file are already installed in the LDAP server schema, the ldapschema utility will make no changes to the LDAP directory server schema.]

SCHEMA_OK

All attribute types and object classes specified in the <schema> file are valid.

[The SCHEMA_OK message indicates the definitions of attribute types and object classes specified in the <schema> file have valid XML format and conform to the DTD template and the LDAP directory server schema policies. This message also indicates no mismatching/incompatible definitions specified in the <schema> file are installed on the LDAP server.]

SCHEMA_INVALID

The <schema> file contains one or more invalid definition of attribute types and/or object classes. Review the messages above and correct any errors in the schema definition file.

[The SCHEMA_INVALID message indicates some of the attribute types and/or object classes specified in the <schema> file have invalid definitions. This condition occurs if the definition does not conform to the LDAP directory server schema policies or the DTD template. Review the "Defining Attribute Types" and "Defining Object Classes" sections for details. Also, check the messages containing ATTRIB_INVALID, ATTRIB_UNRESOLVED, OBJECT_INVALID and OBJECT_UNRESOLVED described below for details.

Any invalid elements and any elements that depend on them will be excluded from being extended on the LDAP server. For example, if an attribute type 'sampleAttributeA' has an invalid <usage> value, and an object class 'sampleObjectO' includes 'sampleAttributeA' as a mandatory or an optional attribute, neither 'sampleAttributeA' nor 'sampleObjectO' can be added to the LDAP server schema until the <usage> value is corrected. Running the ldapschema utility in verbose mode (the -v option) can provide additional information about invalid attribute type and object class definitions. HP recommends correcting any invalid definitions before extending the LDAP directory server schema with any remaining new valid definitions.]

SCHEMA_MISMATCH

The <schema> file contains one or more attribute types or object classes already installed in the LDAP server schema with incompatible (i.e., mismatching) definitions. Review the messages above and verify definitions of any mismatching schema elements. Any remaining schema elements defined in the `<schema>` file cannot be added to the LDAP server schema unless the force flag ("-F" option) is specified.

[The `SCHEMA_MISMATCH` message indicates one or more attribute types or object classes defined in the <schema> file are already installed on the LDAP directory server, however, their definitions do not match. This means that some attribute type or object class definitions specified in the <schema> file do not match the LDAP server schema definitions of the elements with the same numeric oids or names.

Check the messages containing ATTRIB_MISMATCH and OBJECT_MISMATCH described below for the exact instances of attribute types and object classes, respectively, causing the schema mismatch.

The mismatch is caused by any differences in element definitions, such as equality matching rule, single-valued setting, attribute syntax, object class type, attribute types an object class includes, etc. For example, if an attribute type 'sampleAttributeA' installed on the LDAP directory server specifies IA5 String syntax, but the definition of 'sampleAttributeA' in the <schema> file specifies Unicode String syntax, the two attribute types are mismatching. HP does not recommend installing schemas containing mismatching definitions. If the <schema> file defines any new valid attribute types or object classes that are not present in the LDAP directory server schema and you would like to install them anyway, use the force flag (the -F option) to add them to the LDAP server schema.]

`SCHEMA_REJECTED`

The `<schema>` file contains no valid attribute type or object class definitions that can be added to the LDAP directory server schema. It defines elements already installed in the LDAP directory server schema, or contains invalid definitions that hence cannot be installed. Review the messages above and correct any errors in the schema definition file.

[The `SCHEMA_REJECTED` message indicates no attribute type or object class definitions specified in the <schema> file meet the requirement of being both new and valid, and, therefore, cannot be added to the LDAP server schema. Any invalid definitions need to be corrected before they can be added to the LDAP directory server schema.

Check the messages containing ATTRIB_INVALID, ATTRIB_UNRESOLVED, ATTRIB_MISMATCH, OBJECT_INVALID, OBJECT_UNRESOLVED, OBJECT_MISMATCH, SCHEMA_INVALID and SCHEMA_MISMATCH for details on which attribute type and object class definitions prevent the schema from being installed.

If the <schema> file contains any mismatching or invalid definitions, HP does not recommend installing the schema on the LDAP server.]

Attribute Type Status Messages

`ATTRIB_INVALID`

Attribute type definition is missing a numeric oid. Edit the schema definition file to specify one <oid> tag and its value for every `<attributeTypeDefiniton>` definition.

[This message indicates the <oid> tag and its value need to be specified in the `<attributeTypeDefiniton>` definition in the <schema> file.]

ATTRIB_INVALID

Attribute type definition is missing a name. Edit the schema definition file to specify at least one <name> tag and its value for every `<attributeTypeDefiniton>` definition.

[This message indicates the <name> tag and its value need to be specified in the `<attributeTypeDefiniton>` definition in the <schema> file.]

ATTRIB_INVALID

Attribute type “ <attribute name>” has an invalid numericoid. Edit the schema definition file to specify an RFC 2252 compliant <oid> value for this attribute type. Valid numericoid must consist of digits (0-9) that can be separated by a period (.). Leading zeroes are not allowed. See RFC 2252 for details.

This message indicates the <oid> tag and its value need to be corrected in the `<attributeTypeDefiniton>` definition in the <schema> file. The <oid> value must be compliant with RFC 2252. See RFC 2252 for details.

ATTRIB_INVALID

Attribute type “`<attribute name>`” has an invalid name. Edit the schema definition file to specify an RFC 2252 compliant <name > value for this attribute type. Valid name characters include letters (A-z), digits (0-9), semicolons (;) and dashes (-). Valid name must begin with an alphabet letter (A-z). See RFC 2252 for details.

[This message indicates the <name > tag and its value need to be corrected in the `<attributeTypeDefiniton>` definition in the <schema> file. The attribute type name value must be compliant with RFC 2252. See RFC 2252 for details.]

ATTRIB_INVALID

Attribute type “`<attribute name>`” must have the same usage (<usage> tag) value as its super-type. Edit the schema definition file to correct the usage value for this attribute or its super-type.

If the attribute type specifies a supertype, both this attribute type and its supertype must have the same <usage> tag value. This message indicates the <usage> tag value of the specified attribute type and the <usage> tag value of its supertype do not match. Edit the <schema> file to correct the discrepancy.

ATTRIB_INVALID

Attribute type “`<attribute name>`” is missing a syntax value. Edit the schema definition file to specify a syntax (<syntax> tag) value, or a valid super-type (<subTypeOf>) value.

Most LDAP directory servers require attribute type definitions to specify either the syntax value or a super-type value. This message indicates that the specified attribute type definition in the <schema> file does not specify either of these values. Edit the <schema> file to specify either the <syntax> tag and its value, or a <subTypeOf> tag and its value in the specified attribute type definition.

ATTRIB_INVALID

Attribute type “`<attribute name>`” cannot be labeled as obsolete (<obsolete> tag) if any other attribute types or object classes depend on it. Edit the schema definition file to remove the <obsolete> tag from this attribute type definition in order for it to be added to the LDAP server schema.

Obsolete attribute types cannot be added to the LDAP directory server schema if any other attribute types or object classes depend on them. This messages indicates the given attribute type cannot specify the <obsolete> tag in its definition if it is used as a super-type in any other attribute types, or if it is used as a mandatory or optional attribute in any object classes. Edit the <schema> file to correct this discrepancy.

ATTRIB_UNRESOLVED Super-type used in "<attribute name>” attribute type definition is not defined in any LDAP schema.

[This message indicates the super-type specified with the <subTypeOf> tag in the given attribute type definition is undefined. Edit the <schema> file to correct the name of the super- type in the attribute type definition. The super-type used in the attribute type definition must be defined either in the LDAP directory server schema or in the <schema> file before this attribute type can be installed.]

ATTRIB_UNRESOLVED Matching Rule "<matching rule name> " used in the <attribute name> attribute type definition cannot be mapped because "-m -" option is specified. This matching rule is not supported on the LDAP server.

[This message indicates the matching rule specified with the <equality>, <ordering> or <substr> tag in the given attribute type definition is not supported on the LDAP directory server. Option -m - disables matching rule substitution in attribute types. Edit the <schema> file to specify an alternate matching rule supported on the LDAP server, or execute the ldapschema utility without the -m - option to substitute this matching rule with an alternative matching rule supported on the LDAP server.]

ATTRIB_UNRESOLVED Matching Rule "<rule name> " used in the <attribute name> attribute type definition cannot be mapped. This matching rule is not supported on the LDAP server.

ATTRIB_UNRESOLVED LDAP syntax "<syntax oid>” used in “<attribute name>” attribute type definition cannot be mapped because "-s -" option is specified. This LDAP syntax is not supported on the LDAP server.

[This message indicates the LDAP syntax specified with the <syntax> tag in the given attribute type definition is not supported on the LDAP directory server. Option -s - disables syntax substitution in attribute types. Edit the <schema> file to specify an alternate syntax supported on the LDAP server, or execute the ldapschema utility without the -s - option to substitute this syntax with an alternative syntax supported on the LDAP server.]

ATTRIB_UNRESOLVED LDAP syntax "<syntax oid>” used in “<attribute name>” attribute type definition cannot be mapped. This LDAP syntax is not supported on the LDAP server.

[This message indicates the LDAP syntax specified with the <syntax> tag in the given attribute type definition is not supported on the LDAP directory server. The default substitution syntax specified in the /etc/opt/ldapux/schema/map-rules.xml file is not supported on the LDAP directory server either. Edit the <schema> file to specify an alternate syntax supported on the LDAP server, or edit the /etc/opt/ldapux/schema/map-rules.xml file to specify a default substitution syntax supported on the LDAP server.]

ATTRIB_FOUND Attribute type “<attribute name>” is already installed in the LDAP server schema.

[This message indicates the LDAP directory server schema already includes a definition of an attribute type definition with the same numeric oid or name. If the ldapschema utility is executed in the extend mode, the given attribute type will not be added to the LDAP directory server schema. This message is displayed in verbose mode only.]

ATTRIB_MISMATCH

Definition of attribute type "`<attribute name>`" is incompatible with the definition already installed in the LDAP server schema.

ATTRIB_REJECTED

attribute type "`<attribute name>`" will not be added to the LDAP server schema because it is already part of the LDAP schema.

[This message indicates the LDAP directory server schema already includes a definition of an attribute type definition with the same numeric oid or name.]

ATTRIB_REJECTED

attribute type "`<attribute name>`" will not be added to the LDAP server schema because its definition is invalid.

[This message indicates definition of the specified attribute type is invalid. If the `ldapschema` utility is executed in the extend mode, the given attribute type will not be added to the LDAP directory server schema. Check the messages containing ATTRIB_INVALID for details.]

Object Class Status Messages

OBJECT_INVALID

Object class definition is missing a numeric oid. Edit the schema definition file to specify one <oid> tag and its value for every `<objectClassDefiniton>` definition.

[This message indicates the <oid> tag and its value need to be specified in the `<objectClassDefinition>` definition in the <schema> file.]

OBJECT_INVALID

Object Class definition is missing a name. Edit the schema definition file to specify at least one <name> tag and its value for every `<ObjectClassDefiniton>` definition.

[This message indicates the <name> tag and its value need to be specified in the `<objectClassDefinition>` definition in the <schema> file.]

OBJECT_INVALID

Object class "`<object name>`" has an invalid object type value. Edit the schema definition file to modify the value specified with the <type> tag, which can be one of the following:

- `STRUCTURAL`
- `AUXILIARY`
- `ABSTRACT`

[This message indicates the <type> tag value needs to be corrected in the`<objectClassDefinition>` definition in the <schema> file. Possible object class type values are `STRUCTURAL`, `AUXILIARY` or `ABSTRACT`. Any other type values are rejected. If the <type> tag is not specified in the `<objectClassDefinition>` definition, the default object class type value is STRUCTURAL. See RFC 2252 for details.]

OBJECT_UNRESOLVED

Super-class used in "`<object name>`" object class definition is not defined in any LDAP schema.

[This message indicates the super-class specified with the <subClassOf> tag in the given object class definition is undefined. Edit the <schema> file to correct the name of the super-class in the object class definition. The super-class used in the object class definition must be defined either in the LDAP directory server schema or in the <schema> file before this object class can be installed.]

OBJECT_UNRESOLVED Mandatory attribute used in the `<object name>` object class definition is not defined in any LDAP server schema.

[This message indicates the mandatory attribute type specified with the <must> tag in the given object class definition is undefined. Edit the <schema> file to correct the name of the mandatory attribute in the object class definition. The mandatory attribute used in the object class definition must be defined either in the LDAP directory server schema or in the <schema> file before this object class can be installed.]

OBJECT_UNRESOLVED Optional attribute used in "`<object name>`" object class definition is not defined in any LDAP server schema.

[This message indicates the mandatory attribute type specified with the <may> tag in the given object class definition is undefined. Edit the <schema> file to correct the name of the optional attribute in the object class definition. The optional attribute used in the object class definition must be defined either in the LDAP directory server schema or in the <schema> file before this object class can be installed.]

OBJECT_FOUND Object class "`<object name>`" is already installed in the LDAP server schema.

[This message indicates the LDAP directory server schema already includes a definition of an object class definition with the same numeric oid or name. If the `ldapschema` utility is executed in the extend mode, the given object class will not be added to the LDAP directory server schema. This message is displayed in verbose mode only.]

OBJECT _MISMATCH Definition of object class "`<object name>`" is incompatible with the definition already installed in the LDAP server schema.

OBJECT_REJECTED Object class "`<object name>`" will not be added to the LDAP server schema because it is already part of the LDAP schema.

[This message indicates the LDAP directory server schema already includes a definition of an object class definition with the same numeric oid or name.]

OBJECT_REJECTED Object class "`<object name>`" will not be added to the LDAP server schema because its definition is invalid.

[This message indicates definition of the specified object class is invalid. If the ldapschema utility is executed in the extend mode, the given object class will not be added to the LDAP directory server schema. Check the messages containing `OBJECT_INVALID` for details.]

Matching Rules Status Messages

RULE_INVALID Matching rule is missing a numeric oid. Edit the schema definition file to specify one <oid> tag and its value for every `<matchingRuleDefiniton>` definition.

[This message indicates the <oid> tag and its value need to be specified in the `<matchingRuleDefinition>` definition in the `/etc/opt/ldapux/schema/schema-ds_type.xml` file, where ds_type corresponds to the same value specified with the -T option on the command line when executing the `ldapschema` utility.]

`RULE_INVALID` Matching rule is missing a name. Edit the schema definition file to specify at least one <name> tag and its value for every `<matchingRuleDefiniton>` definition.

[This message indicates the <name> tag and its value need to be specified in the `<matchingRuleDefinition>` definition in the `/etc/opt/ldapux/schema/schema-ds_type.xml file`, where ds_type corresponds to the same value specified with the -T option on the command line when executing the ldapschema utility.]

`RULE_INVALID` Matching rule is missing an LDAP syntax. Edit the schema definition file to specify one `<syntax>` tag and its value for every `<matchingRuleDefiniton>` definition.

[This message indicates the <syntax> tag and its value need to be specified in the `<matchingRuleDefinition>` definition in the `/etc/opt/ldapux/schema/schema-ds_type.xml file`, where ds_type corresponds to the same value specified with the -T option on the command line when executing the ldapschema utility.]

`RULE_INVALID` Matching rule "<rule name>" used in the "`<attribute name>`" attribute type definition is not supported on the LDAP server. Matching rule "`<substitute rule name>`" will be used instead.

[This message indicates the specified matching rule `<matching rule name>` is not supported on the LDAP directory server. However, it was successfully mapped with a higher level (less specific) matching rule supported by that server, `<substitute matching rule name>`, as specified in the `/etc/opt/ldapux/schema/map-rules.xml` file. The attribute types which uses this matching rule with the <substr>, <ordering>, <equality> tags will use be queried or extended on the LDAP directory server using `<substitute matching rule name>`].

LDAP Syntax Status Messages

`SYNTAX_INVALID` LDAP syntax is missing a numeric oid. Edit the schema definition file to specify one <oid> tag and its value for every `<syntaxDefiniton>` definition.

[This message indicates the <oid> tag and its value need to be specified in the `<syntaxDefinition>` definition in the `/etc/opt/ldapux/schema/schema-ds_type.xml` file, where ds_type corresponds to the same value specified with the -T option on the command line when executing the `ldapschema` utility.]

`SYNTAX_INVALID` LDAP syntax is missing an oMSyntax value. Edit the schema definition file to specify one `<oMSyntax>` tag and its value for every `<syntaxDefiniton>` definition.

[This message indicates the <oMSyntax> tag and its value need to be specified in the `<syntaxDefinition>` definition in the `/etc/opt/ldapux/schema/schema-ds_type.xml` file, where ds_type corresponds to the same value specified with the -T option on the command line when executing the `ldapschema` utility. The <oMSyntax> tag is required for LDAP syntax definitions supported by the Active Directory Server.]

SYNTAX_UNRESOLVED LDAP syntax "<syntax oid>” used in the “<attribute name>” attribute type definition is not supported on the LDAP server. LDAP syntax “<substitute syntax oid>” will be used instead

[This message indicates the specified syntax <syntax oid> is not supported on the LDAP directory server. However, it was successfully mapped with a higher level (more inclusive) syntax supported by that server, <substitute syntax oid>, as specified in the /etc/opt/ldapux/schema/map-rules.xml file. The attribute types which uses this syntax with the <syntax> tag will use be queried or extended on the LDAP directory server using the <substitute syntax oid>.]

Extending schema containing invalid or incompatible attribute types or object classes is not recommended. To install elements defined in a schema file containing invalid or incompatible definitions requires specifying the force option (-F).

Name Service Migration Scripts

This section describes the shell and perl scripts that can migrate your name service data either from source files or NIS maps to your LDAP directory. These scripts are found in /opt/ldapux/migrate. The two shell scripts `migrate_all_online.sh` and `migrate_all_nis_online.sh` migrate all your source files or NIS maps, while the perl scripts `migrate_passwd.pl`, `migrate_group.pl`, `migrate_hosts.pl`, and so forth, migrate individual maps. The shell scripts call the perl scripts.

The migration scripts require perl, version 5 or later, which is installed with the NIS/LDAP Gateway in /opt/ldapux/contrib/bin/perl.

Naming Context

The naming context specifies where in your directory your name service data will be, under the base DN. For example, if your base DN is "ou=unix,o=hp.com," the passwd map would be at "ou=People,ou=unix,o=hp.com". Default Naming Context (page 170) shows the default naming context for the supported services. The default will work in most cases.

Table 6-8 Default Naming Context

Map Name	Location in the Directory Tree
passwd	ou=People
group	ou=Groups
netgroup	ou=Netgroup
hosts	ou=Devices
networks	ou=Networks
protocols	ou=Protocols
rpc	ou=Rcp
services	ou=Services

If you change the default naming context, modify the file migrate_common.ph and change it to reflect your naming context.

Migrating All Your Files

The two shell scripts `migrate_all_online.sh` and `migrate_all_nis_online.sh` migrate all your name service data either to LDIF or into your directory. The `migrate_all_online.sh` shell script gets information from the appropriate source files, such as /etc/passwd, /etc/group, /etc/hosts, and so forth. The `migrate_all_nis_online.sh` script gets information from your NIS maps using the *ypcat*(1) command. The scripts take no parameters but prompt you for needed information. They also prompt you for whether to leave the output as LDIF or to add the entries to your directory. These scripts call the perl scripts described under Migrating Individual Files (page 171). You will need to modify these scripts to ensure that any calls to perl scripts not listed in Table 5-6 are commented out, you need to comment out the following scripts in the file:

- `$PERL /opt/ldapux/migrate/migrate_fstab.pl`
- `$PERL /opt/ldapux/migrate/migrate_netgroup_byuser.pl`
- `$PERL /opt/ldapux/migrate/migrate_netgroup_byhost.pl`

NOTE: The scripts use `ldapmodify` to add entries to your directory. If you are starting with an empty directory, it may be faster for you to use `ldif2db` or `ns-slapd ldif2db` with the LDIF file. See the *Netscape Directory Server Administrator's Guide* for details on `ldif2db` and `ns-slapd`.

Migrating Individual Files

The migration scripts shown below can be used to migrate the service data, groups, hosts, netgroup, services, protocols, rpc, passwd individually from each of your source files in /etc to LDIF. These scripts are called by the shell scripts described under Migrating All Your Files (page 170). These scripts get their information from the input source file and output LDIF.

Migration Scripts

The migration scripts are described in the table below.

Table 6-9 Migration Scripts

Script Name	Description
migrate_base.pl	creates base DN information.
migrate_group.pl	migrates groups in /etc/group.
migrate_hosts.pl [1]	migrates hosts in /etc/hosts.
migrate_netgroup.pl[2]	migrates netgroups in /etc/netgroup.
migrate_passwd.pl	migrates users in /etc/passwd.
migrate_protocols.pl	migrates protocols in /etc/protocols.
migrate_rpc.pl	migrates RPCs in /etc/rpc.
migrate_services.pl[3]	migrates services in /etc/services.

[1] systems have been configured with the same hostname, then the migration script migrate_host.pl will create multiple entries in its resulting LDIF file with the same distinguished name for the hostname for each of the IP addresses. Since distinguished names need to be unique in an LDAP directory, users need to first manually merge the IP addresses with one designated host record and delete the duplicated records in their LDIF file. A resulting merge might look as follows:

```
. . . .
dn: cn=machineA, ou=devices, ou=unix, o=hp.com
objectClass: top
objectClass: ipHost
objectClass: device
ipHostNumber: 15.13.130.72
ipHostNumber: 15.13.104.4
ipHostNumber: 15.13.95.92
cn: mymachine
cn: hpma01.cup.hp.com
. . . .
```

[2] Netgroup
- The NIS optimization maps 'byuser' and 'byhost' are not utilized.
-Each triple is stored as a single string.
-Each triple must be enclosed by parentheses, e.g "(machine, user, domain)" is a valid triple while "machine, user, domain" is not.

[3] When migrating services data into the LDAP directory, users should keep in mind that only multiple protocols can be associated with one service name, but *not* multiple service ports.

Environment Variables

When using the perl scripts to migrate individual files, you need to set the following environment variable:

LDAP_BASEDN The base distinguished name where you want to put data in the LDAP directory.

For example, the following command sets the base DN to "o=hp.com":

```
export LDAP_BASEDN="o=hp.com"
```

General Syntax for Perl Migration Scripts

All the perl migration scripts use the following general syntax:

```
scriptname inputfile [outputfile]
```

where

scriptname is the name of the particular script you are using. The scripts are listed below.

inputfile is the name of the appropriate name service source file corresponding to the script you are using.

outputfile is optional and is the name of the file where the LDIF is written. stdout is the default output.

Examples

The following command converts all name service files in /etc to LDIF:

```
$ migrate_all_online.sh
```

The following commands convert /etc/passwd into LDIF and output it to stdout:

```
$ export LDAP_BASEDN="dc=hp,dc=com"
$ migrate_passwd.pl /etc/passwd

dn: uid=jbloggs,ou=People,dc=hp,dc=com
uid: jbloggs
cn: Joe Bloggs
objectclass: top
objectclass: posixAccount
objectclass: account
userPassword: {crypt}daCXgaxahRNkg
loginShell: /bin/ksh
uidNumber: 20
gidNumber: 20
homeDirectory: /home/jbloggs
gecos: Joe Bloggs,42U-C3,555-1212
```

The following commands convert /etc/group into LDIF and place the result in /tmp/group.ldif:

```
$ export LDAP_BASEDN="o=hp.com"
$ migrate_group.pl /etc/group /tmp/group.ldif

dn: cn=mira.hp.com,ou=Groups,o=hp.com
objectclass: posixGroup
objectclass: top
cn: mira
 cn: mira.hp.com
userPassword: {crypt}*
gidNumber: 325
```

The following command migrates /etc/hosts into LDIP and place the result in /tmp/host.ldif:

```
export LDAP_BASEDN="o=hp.com"
migrate_hosts.pl /etc/hosts /tmp/host.ldif
dn: cn=hostA.hp.com,ou=Hosts,o=hp.com
objectclass: ipHost
objectclass: device
objectclass: top
ipHostNumber: 10.1.2.5
cn: HostA
cn: HostA.hp.com
```

Unsupported Contributed Tools and Scripts

This section describes contributed tools and scripts which are not officially supported by HP at the present time.

beq Search Tool

The new beq tool expands the search capability beyond that currently offered by nsquery, which is limited to hosts, passwd, and group. This search utility bypasses the name service switch and queries the backend directly based on the specified library. The search will include the following services: pwd, grp, shd, srv, prt, rpc, hst, net, ngp, and grm.

The syntax for this tool, along with example output, is shown below.

Syntax

`beq -k [n|d] -s <service> (-l <library>) (-h | -H <#>) <id1> (id1> (<id2> (...))`

where

k [n|d] Required. The search key may be either n for name string or d for digit (a numeral search).

-s <service> Required. Indicates what backends are to be searched for information.

-l <library> Query the backend directly. Bypass the APIs and skip the name service switch.

-h Provides Help on this command.

-H <#> Specifies Help level (0-5). Larger numbers provide more information. If you specify -h or -H, no other parameters are needed.

Service	Description
pwd	Password
grp	Group
shd	Shadow Password
srv	Service
prt	Protocol
rpc	RPC
hst	Host
net	Network
ngp	Netgroup
grm	Group Membership

Examples

1. An example beq command using igrp1 (group name) as the search key, grp (group) as the service, and ldap as the library is shown below:

   ```
   ./beq -k n -s grp -l /usr/lib/libnss_ldap.1 igrp1
   ```

   ```
   nss_status ............. NSS_SUCCESS
   pw_name...........(iuser1)
   pw_passwd.........(*)
   pw_uid............(101)
   pw_gid............(21)
   pw_age............()
   pw_comment........()
   pw_gecos..........(gecos data in files)
   pw_dir............(/home/iuser1)
   pw_shell..........(/usr/bin/sh)
   ```

```
pw_audid.........(0)
pw_audflg........(0)
```

2. An example beq command using user name adm as the search key, pwd (password) as the service, and files as the library is shown below:

   ```
   ./beq -k n -s pwd -l /usr/lib/libnss_files.1 adm
   ```

   ```
   nss_status ............. NSS_SUCCESS
   pw_name..........(adm)
   pw_passwd........(*)
   pw_uid...........(4)
   pw_gid...........(4)
   pw_age...........()
   pw_comment.......()
   pw_gecos.........()
   pw_dir...........(/var/adm)
   pw_shell.........(sbin/sh)
   pw_audid.........(0)
   pw_audflg........(0)
   ```

3. An example beq command using uid number 102 as the search key, pwd (password) as the service and ldap as the library is shown below:

   ```
   ./beq -k d -s pwd -l /usr/lib/libnss_ldap.1102
   ```

   ```
   nss_status ............. NSS_SUCCESS
   pw_name..........(user2)
   pw_passwd........(*)
   pw_uid...........(102)
   pw_gid...........(21)
   pw_age...........()
   pw_comment.......()
   pw_gecos.........(gecos data in files)
   pw_dir...........(/home/user2)
   pw_shell.........(/usr/bin/sh)
   pw_audid.........(0)
   pw_audflg........(0)
   ```

4. An example beq command using group name igrp1 as the search key, grp (group) as the service, and ldap as the library is shown below:

   ```
   ./beq -k n -s grp -l /usr/lib/libnss_ldap.1 igrp1
   ```

   ```
   nss_status ............. NSS_SUCCESS
   gr_name..........(igrp1)
   gr_passwd........(*)
   gr_gid...........(21)
   pw_age...........()
   gr_mem
   (iuser1)
   (iuser2)
   (iuser3)
   ```

5. An example beq command using a gid number as the search key, grp (group) as the service, and ldap as the library is shown below:

   ```
   ./beq -k d -s grp -l /usr/libnss_ldap.l 22
   ```

   ```
   nss_status ............. NSS_SUCCESS
   gr_name..........(igrp2)
   gr_passwd........(*)
   gr_gid...........(22)
   pw_age...........()
   ```

```
gr_mem
(iuser1)
```

certutil — Certificate Database Tool

You can use the `certutil` command-line utility to create and modify the Netscape Communicator *cert7.db* and *key3.db* database files. This tool can also list, generate, modify, or delete certificates within the *cert7.db* file. You can also use this tool to create, change the password, generate new public and private key pairs, display the contents of the key database, or delete key pairs within the *key3.db* file. For detailed command options and their arguments, refer to *Using the Certificate Database Tool* available at *http://www.mozilla.org/projects/security/pki/nss/tools/certutil.html*.

uid2dn — Display User's Distinguished Name Tool

This tool, found in `/opt/ldapux/contrib/bin`, displays user's Distinguish Name (DN) information for a given UID.

Syntax

```
uid2dn [UID]
```

where ***UID*** is a user's UID information.

Examples

The following command displays the user's DN information for a given user's UID `john`:

```
./uid2dn john
```

The output shows below after you run the above command:

```
CN=john lee,CN=Users,DC=usa,DC=example,DC=hp,DC=com
```

get_attr_map.pl — Get Attributemap from Profile Tool

This tool, found in `/opt/ldapux/contrib/bin`, gets the attributemap information for a given name service from the profile file `/etc/opt/ldapux/ldapux_profile.ldif`.

Syntax

```
get_attr_map.pl [<service>.<attribute>]
```

where ***services*** is the name of the supported service, ***attribute*** is the name of an attribute.

Examples

The following command gets the `homedirectory` attribute information for the `passwd` service:

```
./get_attr_map.pl passwd homedirectory
```

The following command gets the `uidnumber` attribute information for the `passwd` service:

```
./get_attr_map.pl passwd uidnumber
```

7 User Tasks

This chapter describes the following tasks your users will need to do:

- To Change Passwords (page 177)
- To Change Personal Information (page 178)

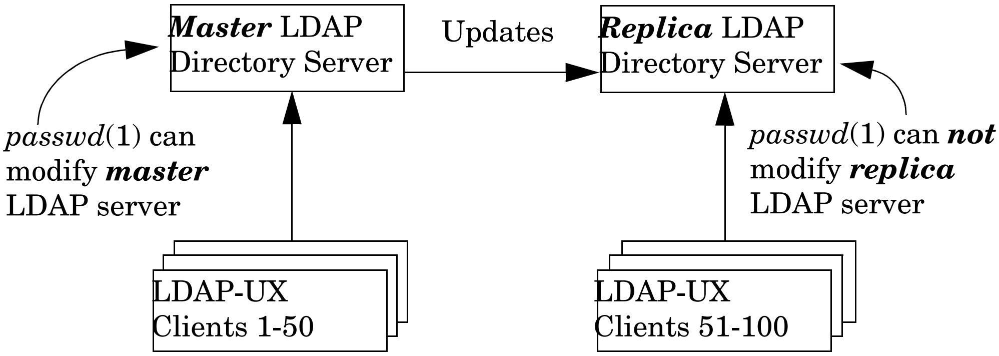
To Change Passwords

With LDAP-UX Client Services, users change their password with the *passwd*(1) command. Depending on how you have PAM configured and depending on where the user's information is, in the directory or in /etc/passwd, users may get prompted for their password twice as PAM looks in the configured locations for the user's information.

Since LDAP directory replicas may not be modifiable, the *passwd*(1) command may not work on clients configured to use a directory replica. In this case you could use the *ldappasswd*(8) command. You might wrap an ldappasswd command in a passwd wrapper, similar to the *yppasswd*(1) command. The wrapper would ask the user for the old password, call ldapsearch to find the current user's DN, then call *ldappasswd*(8) and specify the master LDAP directory server. See Sample passwd Command Wrapper (page 178) for an example you can modify and use.

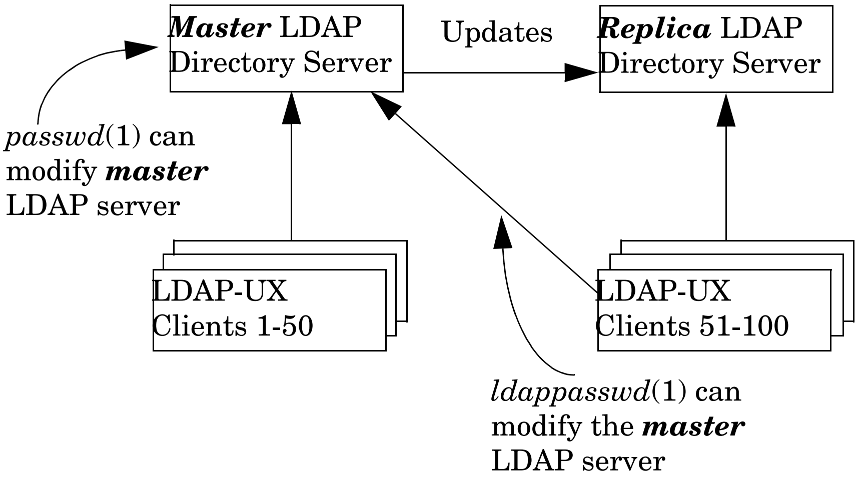
For example, say clients 1-50 use the master directory server on sys001 and clients 51-100 use the replica directory server on sys002. The *passwd*(1) command on clients 1-50 can modify passwords in the master directory on sys001. However, the *passwd*(1) command on clients 51-100 will fail because the replica server on sys002 cannot be modified. See the diagram below.

Figure 7-1 Cannot Change Passwords on Replica Servers


Master LDAP Directory Server
Updates
Replica LDAP Directory Server
passwd(1) can modify ***master*** LDAP server
passwd(1) can ***not*** modify ***replica*** LDAP server
LDAP-UX Clients 1-50
LDAP-UX Clients 51-100

One way to allow clients 51-100 to change their passwords is to create a new *passwd*(1) command wrapper on these clients that calls *ldappasswd*(1), which modifies the master directory. When the replica server is updated depends on how you have configured the replication. All other LDAP requests continue to go to the replica server through PAM and NSS. See Changing Passwords on Master Server with ldappasswd (page 178) below. See also Sample passwd Command Wrapper (page 178) for a sample passwd wrapper command.

Figure 7-2 Changing Passwords on Master Server with ldappasswd


Master LDAP Directory Server
Updates
Replica LDAP Directory Server
passwd(1) can modify ***master*** LDAP server
LDAP-UX Clients 1-50
LDAP-UX Clients 51-100
ldappasswd(1) can modify the ***master*** LDAP server

See ldappasswd (page 137) for details of this command.

Figure 7-3 Sample passwd Command Wrapper

```
#!/usr/bin/ksh
#
# You can put a default master LDAP server host name
# here.  Otherwise the local host is the default.
#
#LDAP_MASTER="masterHostName"

if [[ "$1" != "" ]]
  then
  LDAP_MASTER="$1"
fi

if [[ "$LDAP_MASTER" = "" ]]
  then
  eval "$(sed -e "1,/Service: NSS/d" /etc/opt/ldapux/ldapux_client.conf | \
    grep "^LDAP_HOSTPORT")"
  LDAP_MASTER="$(echo $LDAP_HOSTPORT | cut -d" " -f 1)"
fi

LDAP_BASEDN="$(grep -i "^defaultsearchbase:" \
  /etc/opt/ldapux/ldapux_profile.ldif | cut -d" " -f 2-99)"

/opt/ldapux/bin/ldappasswd -b "$LDAP_BASEDN" -h $LDAP_MASTER
```

Alternatively, your users can use a simple LDAP gateway through a web browser connected to the directory to change their password. The advantage to this method is that your users can also change their other personal information as described below.

To Change Personal Information

On HP-UX, users change their personal information (sometimes called "gecos" information) such as full name, phone number, and location with the *chfn*(1) command which changes /etc/passwd. HP-UX users change their login shell with the *chsh*(1) command, which also changes /etc/passwd. See the *LDAP-UX Integration B.04.10 Release Notes* for whether or not these commands change entries in the directory with this release.

If you have Netscape/Red Hat Directory Server for HP-UX, you can use the Directory Console or the `ldapmodify` command to change personal information. Or you can use a simple LDAP gateway through a web browser to display and change this information.

8 Mozilla LDAP C SDK

This chapter describes the Mozilla LDAP SDK for C and the SDK file components. This chapter contains the following sections:

- Overview (page 179).
- The Mozilla LDAP C SDK File Components (page 179) briefly describes many of files that comprise the LDAP C SDK.

Overview

The LDAP-UX Client Services provides the Mozilla LDAP C SDK 5.17.1 support. The LDAP C SDK is a Software Development Kit that contains a set of LDAP Application Programming Interfaces (API) to allow you to build LDAP-enabled clients. The functionality implemented in the SDK closely follows the interface outlined in RFC 2251. Using the functionality provided with the SDK, you can enable your clients to connect to LDAP v3-compliant servers and perform the LDAP functions.

The API functions provided by the Netscape LDAP C SDK allow you to perform the following major LDAP operations:

- Search for retrieving a list of entries
- Add new entries to the directory
- Update existing entries
- Delete entries
- Rename entries

NOTE: For the detailed information on how to use the LDAP API functions contained in the Mozilla SDK for C, and how to enable your client applications to connect to the LDAP servers, refer to *Mozilla LDAP C SDK Programmer's Guide* at *http://www.mozilla.org/directory/csdk-docs/*.

The Mozilla LDAP C SDK File Components

Table 7-1 shows the Mozilla LDAP C SDK 5.14.1file components on the HP-UX 32 or 64 bit PA machine:

Table 8-1 Mozilla LDAP C SDK File Components on the PA machine

Files	Description
/usr/lib/libldap.sl (32-bit) /usr/lib/pa20_64/libldap.sl (64-bit)	Main LDAP C SDK API libraries that link to the */opt/ldapux/lib* libraries.
/opt/ldapux/lib/libnspr4.sl (32-bit) /opt/ldapux/lib/libnss3.sl (32-bit) /opt/ldapux/lib/libsoftokn3.sl (32-bit) /opt/ldapux/lib/libssl3.sl (32-bit) /opt/ldapux/lib/libfreebl_hybrid_3.sl (32-bit) /opt/ldapux/lib/libfreebl_pure32_3.sl (32-bit) /opt/ldapux/lib/libplc4.sl (32-bit) /opt/ldapux/lib/pa20_64/libnspr4.sl (64-bit) /opt/ldapux/lib/pa20_64/libnss3.sl (64-bit) /opt/ldapux/lib/pa20_64/libplc4.sl (64-bit) /opt/ldapux/lib/pa20_64/libsoftokn3. sl (64-bit) /opt/ldapux/lib/pa20_64/libssl3.sl (64-bit) /opt/ldapux/lib/pa20_64/libplds4.sl (64-bit)	LDAP C SDK dependency libraries.

Table 8-1 Mozilla LDAP C SDK File Components on the PA machine *(continued)*

Files	Description
/usr/include/*	Include files from LDAP C SDK
/opt/ldapux/contrib/bin/certutil	Unsupported command tool that creates and modifies the certificate database files, *cert8.db* and *key3.db*.
/opt/ldapux/contrib/ldapsdk/examples	Unsupported Netscape LDAP C SDK examples.
/opt/ldapux/contrib/ldapsdk/source.tar.gz	Mozilla LDAP C SDK source (for license compliance).
/opt/ldapux/bin/ldapdelete /opt/ldapux/bin/ldapmodify /opt/ldapux/bin/ldapsearch /opt/ldapux/bin/ldapcmp /opt/ldapux/bin/ldapcompare	Tools to delete, modify, and search for entries in a directory. See the *Netscape Directory Server Administrator's Guide* for details.

Table 7-2 shows the Mozilla LDAP C SDK 5.17.1 file components on the HP-UX 32 or 64 bit IA machine:

Table 8-2 Mozilla LDAP C SDK File Components on the IA machine

Files	Description
/usr/lib/hpux32/libldap.so (32-bit) /usr/lib/hpux64/libldap.so (64-bit)	Main LDAP C SDK API libraries that link to the */opt/ldapux/lib* libraries.
/opt/ldapux/lib/hpux32/libnspr4.so (32-bit) /opt/ldapux/lib/hpux32/libnss3.so (32-bit) /opt/ldapux/lib/hpux32/libplc4.so (32-bit) /opt/ldapux/lib/hpux32/libsoftokn3.so (32-bit) /opt/ldapux/lib/hpux32/libssl3.so (32-bit) /opt/ldapux/lib/hpux32/libfreebl_pure32_3.so /opt/ldapux/lib/hpux32/libplds4.so (32-bit) /opt/ldapux/lib/hpux64/libnspr4.so (64-bit) /opt/ldapux/lib/hpux64/libnss3.so (64-bit) /opt/ldapux/lib/hpux64/libplc4.so (64-bit) /opt/ldapux/lib/hpux64/libsoftokn3.so (64-bit) /opt/ldapux/lib/hpux64/libssl3.so (64-bit) /opt/ldapux/lib/hpux64/libplds4.so (64-bit) /opt/ldapux/lib/libnspr4.sl (32-bit) /opt/ldapux/lib/libnss3.sl (32-bit) /opt/ldapux/lib/libplc4.sl (32-bit) /opt/ldapux/lib/libsoftokn3.sl (32-bit) /opt/ldapux/lib/libssl3.sl (32-bit) /opt/ldapux/lib/freebl_pure32_3.sl (32-bit) /opt/ldapux/lib/libplds4.sl(32-bit) /opt/ldapux/lib/pa20_64/libnspr4.sl (64-bit) /opt/ldapux/lib/pa20_64/libnss3.sl (64-bit) /opt/ldapux/lib/pa20_64/libplc4.sl (64-bit) /opt/ldapux/lib/pa20_64/libsoftokn3.sl (64-bit) /opt/ldapux/lib/pa20_64/libssl3.sl (64-bit) /opt/ldapux/lib/pa20_64/libplds4.sl (64-bit)	LDAP C SDK dependency libraries.
/usr/include/*	Include files from LDAP C SDK
/opt/ldapux/contrib/bin/certutil	Unsupported command tool that creates and modifies the certificate database files, *cert8.db* and *key3.db*.
/opt/ldapux/contrib/ldapsdk/examples	Unsupported Mozilla LDAP C SDK examples.
/opt/ldapux/contrib/ldapsdk/source.tar.gz	Mozilla LDAP C SDK source (for license compliance).
/opt/ldapux/bin/ldapdelete /opt/ldapux/bin/ldapmodify /opt/ldapux/bin/ldapsearch /opt/ldapux/bin/ldapcmp /opt/ldapux/bin/ldapcompare	Tools to delete, modify, and search for entries in a directory. See the *Netscape Directory Server Administrator's Guide* for details.

Table 7-3 shows header files that support the LDAP libraries existing under /usr/include, except where noted:

Table 8-3 Mozilla LDAP C SDK API Header Files

Header Files	Description
/usr/include/ldap.h	Main LDAP functions, structures and defines.
/usr/include/ldap-extension.h	Support for LDAP v3 extended operations, controls and other server specific features. This file must be included in source code that uses LDAP v3 extended operations or controls.
/usr/include/ldap_ssl.h	Support for creation of SSL connections. This file must be included in source code that requires SSL connections.
/usr/include/srchpref.h	Support for LDAP search preferences configuration files (ldapsearchprefs.conf). A common method used by applications that use the OpenLDAP API to define organizational search preferences.
/usr/include/disptmpl.h	Support for LDAP display templates. Allows applications to convert LDAP entries into displayable text strings and HTML.
/usr/include/lber.h	Support for creating messages that follow the Basic Encoding Rules syntax. These APIs are used when building extended LDAP operations or controls. This file is a support file for ldap.h and does not need to included in source code.
/usr/include/ldap-standard.h	Contains basic LDAP defines. This file is a support file for ldap.h and does not need to be included in source code.
/usr/include/ldap-platform.h	Contains platform specific information for compiling on a variety of platforms. This file is a support file for ldap.h and does not need to be included in source code.
/opt/ldapux/include/ldap-to-be-deprecated.h	LDAP APIs that will not be available in the future. Do not use this header file for newly created LDAP-enabled applications.
/opt/ldapux/include/ldap-deprecated.h	LDAP APIs that have been deprecated. Do not use.

NOTE: If you attempt to use the LDAP C SDK in your code , you only need to put in "`#include <ldap.h>`" in the code and compile with the `-lldap` parameter to load the LDAP C SDK library.

Unsupported Contributed Tools and Scripts

This section describes contributed tools and scripts which are not officially supported by HP.

A Configuration Worksheet

Use this worksheet to help you configure LDAP-UX Client Services. See Installing And Configuring LDAP-UX Client Services (page 21) for details.

Table A-1 LDAP-UX Client Services Configuration Worksheet

LDAP-UX Client Services Configuration Worksheet	
Directory administrator DN:	
Directory server host:	
Directory server port:	
Configuration profile DN:	
Base DN of name service data:	
Credential type:	
Proxy user DN:	
Source of user, group data:	
Migration method:	

See the next page for an explanation and sample table. For installation and configuration details, see Installing And Configuring LDAP-UX Client Services (page 21).

Table A-2 LDAP-UX Client Services Configuration Worksheet Explanation

LDAP-UX Client Services Configuration Worksheet	
Directory administrator DN:	The distinguished name of a directory administrator allowed to modify the directory. Example: cn=directory manager
Directory server host:	The host name or IP address where your directory server is running. Example: sys001.hp.com (12.34.56.78)
Directory server port:	The TCP port number your directory server is using. Example: 389
Configuration profile DN:	The distinguished name where your configuration profile is. Example: cn=profile1, o=hp.com
Base DN of name service data:	The distinguished name where your name service data is. Example: ou=People, o=hp.com
Credential type:	The method clients use to access the directory. Can be "anonymous," "proxy," or "proxy anonymous." Example: anonymous Default: anonymous
Proxy user DN:	The distinguished name of the proxy user, if needed. Example: cn=proxyuser,ou=special users, o=hp.com
Source of user, group data:	Where you get your user and group data from to migrate into the directory. Example: /etc/passwd and /etc/group on sys001
Migration method:	How you will migrate your user and group data into the directory, for example, using the migration scripts. Example: migrate_all_online.sh edited to remove all but migrate_passwd.pl, migrate_group.pl, and migrate_base.pl

B LDAP-UX Client Services Object Classes

This Appendix describes the object classes LDAP-UX Client Services uses for configuration profiles.

In release B.02.00, LDAP-UX Client Services used two object classes for configuration profiles:

1. posixDUAProfile
2. posixNamingProfile

With release B.03.00, the posixDUAProfile and posixNamingProfile objectlcasses have been replaced by a single STRUCTURAL objectclass DUAConfigProfile.

In addition, four new attributes are added. These changes are to reflect the definition shown in the most current IETF draft "A Configuration Schema for LDAP Based Directory User Agents" (in the document file titled, draft-joslin-config-schema-07.txt). This allows LDAP-UX to integrate with configuration profiles that are supported by other vendors.

The object class `DUAConfigProfile` is defined as follows:

```
objectclass DUAConfigProfile
     superior top
     requires
          cn
     allows
          authenticationMethod,
          attributeMap,
          bindTimeLimit,
          credentialLevel,
          defaultSearchBase,
          defaultSearchScope,
          defaultServerList,
          followReferrals,
          objectclassMap,
          preferredServerList,
          profileTTL,
          searchTimeLimit,
          serviceAuthenticationMethod,
          serviceCredentialLevel,
          servicesearchDescriptor
```

Profile Attributes

The attributes of DUAConfigProfile is defined as follows:

`cn` is the common name of the profile entry.

`attributeMap` is a mapping from RFC 2307 attributes to alternate attributes. Use this if your entries do not conform to RFC 2307. Each entry consists of: *Service*:*Attribute*=*Altattribute* where *Service* is one of the supported services: passwd, group, shadow, pam, networks, hosts, protocols, services, rpc, or netgroup. *Attribute* is an attribute of the service as defined by RFC 2307. *Altattribute* is the attribute that should be used instead of the standard attribute.

For example, pam:userPassword=ntUserPassword maps the userPassword attribute to ntUserPassword for the pam service. passwd:uidnumber=employeeNumber maps the uidnumber attribute to employeeNumber for the passwd service.

NOTE: The userPassword attribute is mapped to *NULL* to prevent passwords from being returned for increased security and to prevent PAM_UNIX from authenticating users in the LDAP directory. Mapping to *NULL* or any other nonexistent attribute means do not return anything.

`authenticationMethod` is how the client binds to the directory. The value can be "simple" indicating bind using a user name and password. If this attribute has no value, "simple" is the default.

`bindTimeLimit` is how long, in seconds, the client should wait to bind before aborting. 0 (zero) means no time limit. If this attribute has no value, the default is no time limit.

`credentialLevel` is the identity clients use when binding to the directory. The value must be one of the following: "proxy", "anonymous", or "proxy anonymous". "proxy" means use the configured proxy user. "anonymous" means use anonymous access. "proxy anonymous" means use the configured proxy user and if that fails, bind anonymously. If this attribute has no value, "anonymous" is the default.

`defaultSearchBase` is the base DN where clients can find name service information, for example `ou=hpusers,o=hp.com`. This attribute must have a value.

`defaultServerList` is the same as preferredServerList except the order in which the specified hosts is tried can be interpreted, and defaultServerList is used only after preferredServerList. If neither defaultServerList nor preferredServerList specifies a host, the client tries the host where the profile is. See preferredServerList below.

`followReferrals` specifies whether or not referrals should be followed. If the entry is 0 (zero) or FALSE, referrals will not be followed. If the attribute has no value, any other numeric value, or TRUE referrals will be followed.

`preferredServerList` is a list of one or more host IP addresses and optional port numbers where LDAP directory servers are running. Each host is searched in the order given. If this attribute has no value, or if none of the specified servers satisfies the client's request, the defaultServerList is used. See defaultServerList above.

For example, `15.13.128.145:250` is the host at IP address 15.13.128.145 using port number 250. When specifying multiple hosts, each host:port entry must be separated by a space.

`profileTTL` is the recommended time interval before refreshing the cached configuration profile.

`searchTimeLimit` is how long, in seconds, a client should wait for directory searches before aborting. 0 (zero) means no time limit. If this attribute has no value, the default is no time limit.

`serviceSearchDescriptor` is one to three custom search descriptors for each service. The format is *`Service`*:*BaseDN*?*`Scope`*?(*`Filter`*) where *`Service`* is one of the supported services passwd, group, shadow, or pam. *BaseDN* is the base DN at which to start searches. *`Scope`* is the search scope and can be one of the following: one, base, sub. *`Filter`* is an LDAP search filter, typically the object class. Each service can have up to three custom search descriptors.

For example, the following defines a search descriptor for the passwd service specifying a baseDN of `ou=people,ou=unix,o=hp.com`, a search scope of sub, and a search filter of the posixAccount object class.

```
passwd:ou=people,ou=unix,o=hp.com?sub?(objectclass=posixAccount)
```

C Sample /etc/pam.ldap.trusted file

This Appendix provides the sample PAM configuration file, `/etc/pam.ldap.trusted`, used as the `/etc/pam.conf` file to support the coexistence of LDAP-UX and Trusted Mode. This `/etc/pam.ldap.trusted` file must be used as the `/etc/pam.conf` file if your directory server is the Netscape/Red Hat Directory Server and your LDAP client is in the Trusted Mode. If your system is in a standard mode, you still need to use the`/etc/pam.ldap`file as the `/etc/pam.conf`file.

The following is a sample PAM configuration file, `/etc/pam.ldap.trusted`, used on the HP-UX 11.0 or 11i v1 system:

```
#
# PAM configuration
#
# This pam.conf file is intended as an example only.
#
#

#################################################################
# This configuration file has only been modified for default   #
# services. Other services can be added or modified as needed  #
# or desired. If a service is not listed, it will use the      #
# OTHER classification.                                        #
#                                                               #
# the format for a entry is                                    #
# <service> <module_type> <control> <module path> <options>    #
#                                                               #
# see pam.conf(4) for more details                             #
#                                                               #
# NOTE: This pam.conf file is recommended only if you convert  #
# your system to a Trusted System. If your system is in the    #
# Standard Mode, use the pam.ldap file as an example.          #
#                                                               #
#                                                               #
#################################################################
#
# Authentication management
#
login      auth sufficient   /usr/lib/security/libpam_ldap.1
login      auth required     /usr/lib/security/libpam_unix.1 try_first_pass
su         auth sufficient   /usr/lib/security/libpam_ldap.1
su         auth required     /usr/lib/security/libpam_unix.1 try_first_pass
dtlogin    auth sufficient   /usr/lib/security/libpam_ldap.1
dtlogin    auth required     /usr/lib/security/libpam_unix.1 try_first_pass
dtaction   auth sufficient   /usr/lib/security/libpam_ldap.1
dtaction   auth required     /usr/lib/security/libpam_unix.1 try_first_pass
ftp        auth sufficient   /usr/lib/security/libpam_ldap.1
ftp        auth required     /usr/lib/security/libpam_unix.1 try_first_pass
OTHER      auth sufficient   /usr/lib/security/libpam_ldap.1
OTHER      auth required     /usr/lib/security/libpam_unix.1 try_first_pass
# Account management
#
login      account sufficient  /usr/lib/security/libpam_ldap.1
login      account required    /usr/lib/security/libpam_unix.1
su         account sufficient  /usr/lib/security/libpam_ldap.1
su         account required    /usr/lib/security/libpam_unix.1
dtlogin    account sufficient  /usr/lib/security/libpam_ldap.1
dtlogin    account required    /usr/lib/security/libpam_unix.1
dtaction   account sufficient  /usr/lib/security/libpam_ldap.1
dtaction   account required    /usr/lib/security/libpam_unix.1
ftp        account sufficient  /usr/lib/security/libpam_ldap.1
ftp        account required    /usr/lib/security/libpam_unix.1
OTHER      account sufficient  /usr/lib/security/libpam_ldap.1
OTHER      account required    /usr/lib/security/libpam_unix.1
# Session management
#
login      session requried    /usr/lib/security/libpam_ldap.1
login      session required    /usr/lib/security/libpam_unix.1
```

```
dtlogin    session required    /usr/lib/security/libpam_ldap.1
dtlogin    session required    /usr/lib/security/libpam_unix.1
dtaction   session required    /usr/lib/security/libpam_ldap.1
dtaction   session required    /usr/lib/security/libpam_unix.1
OTHER      session required    /usr/lib/security/libpam_ldap.1
OTHER      session required    /usr/lib/security/libpam_unix.1
# Password management #
login      password.sufficient    /usr/lib/security/libpam_ldap.1
login      password required      /usr/lib/security/libpam_unix.1 try_first_pass
passwd     password sufficient    /usr/lib/security/libpam_ldap.1
passwd     password required      /usr/lib/security/libpam_unix.1 try_first_pass
dtlogin    password sufficient    /usr/lib/security/libpam_ldap.1
dtlogin    password required      /usr/lib/security/libpam_unix.1 try_first_pass
dtaction   password sufficient    /usr/lib/security/libpam_ldap.1
dtaction   password required      /usr/lib/security/libpam_unix.1 try_first_pass
OTHER      password sufficient    /usr/lib/security/libpam_ldap.1
OTHER      password required      /usr/lib/security/libpam_unix.1 try_first_pass
```

The following is a sample PAM configuration file, `/etc/pam.ldap.trusted`, used for the HP-UX 11i v2 system:

```
#
# PAM configuration
#
# This pam.conf file is intended as an example only.
#
#

################################################################
# This configuration file has only been modified for default   #
# services. Other services can be added or modified as needed  #
# or desired. If a service is not listed, it will use the      #
# OTHER classification.                                        #
#                                                              #
# the format for a entry is                                    #
# <service> <module_type> <control> <module path> <options>    #
#                                                              #
# see pam.conf(4) for more details                             #
#                                                              #
# NOTE: This pam.conf file is recommended only if you convert  #
# your system to a Trusted System. If your system is in the    #
# Standard Mode, use the pam.ldap file as an example.          #
#                                                              #
# NOTE: If the path to a library is not absolute, it is assumed#
# to be relative to the directory /usr/lib/security/$ISA.      #
# The "$ISA (i.e Instruction Set Architecture) token is        #
# replaced by the PAM engine (libpam) with "hpux64" for IA     #
# 64-bit modules, or with "hpux32" for IA 32-bit modules, or   #
# with "pa20_64" for PA 64-bit modules, or with NULL for PA    #
# 32-bit modules.                                              #
# For PA applications, library name ending with "so.1" is a    #
# symbolic link that points to the corresponding PA (32 or 64  #
# bit) backend library.                                        #
################################################################
#
# Authentication management
#
login      auth required       libpam_hpsec.so.1
login      auth sufficient     libpam_ldap.so.1
login      auth required       libpam_unix.so.1 try_first_pass
su         auth required       libpam_hpsec.so.1
su         auth sufficient     libpam_ldap.so.1
su         auth required       libpam_unix.so.1 try_first_pass
dtlogin    auth required       libpam_hpsec.so.1
dtlogin    auth sufficient     libpam_ldap.so.1
dtlogin    auth required       libpam_unix.so.1 try_first_pass
```

```
dtaction   auth required       libpam_hpsec.so.1
dtaction   auth sufficient     libpam_ldap.so.1
dtaction   auth required       libpam_unix.so.1 try_first_pass
ftp        auth required       libpam_hpsec.so.1
ftp        auth sufficient     libpam_ldap.so.1
ftp        auth required       libpam_unix.so.1 try_first_pass
rcomds     auth required       libpam_hpsec.so.1
rcomds     auth sufficient     libpam_ldap.so.1
rcomds     auth required       libpam_unix.so.1 try_first_pass
sshd       auth required       libpam_hpsec.so.1
sshd       auth sufficient     libpam_ldap.so.1
sshd       auth required       libpam_unix.so.1 try_first_pass
OTHER      auth sufficient     libpam_ldap.so.1
OTHER      auth required       libpam_unix.so.1 try_first_pass
# Account management
#
login      account required    libpam_hpsec.so.1
login      account sufficient  libpam_ldap.so.1
login      account required    libpam_unix.so.1
su         account required    libpam_hpsec.so.1
su         account sufficient  libpam_ldap.so.1
su         account required    libpam_unix.so.1
dtlogin    account required    libpam_hpsec.so.1
dtlogin    account sufficient  libpam_ldap.so.1
dtlogin    account required    libpam_unix.so.1
dtaction   account required    libpam_hpsec.so.1
dtaction   account sufficient  libpam_ldap.so.1
dtaction   account required    libpam_unix.so.1
ftp        account required    libpam_hpsec.so.1
ftp        account sufficient  libpam_ldap.so.1
ftp        account required    libpam_unix.so.1
rcomds     account required    libpam_hpsec.so.1
rcomds     account sufficient  libpam_ldap.so.1
rcomds     account required    libpam_unix.so.1
sshd       account required    libpam_hpsec.so.1
sshd       account sufficient  libpam_ldap.so.1
sshd       account required    libpam_unix.so.1
ftp        account required    libpam_unix.so.1
OTHER      account sufficient  libpam_ldap.so.1
OTHER      account required    libpam_unix.so.1
# Session management
#
login      session required    libpam_hpsec.so.1
login      session requried    libpam_ldap.so.1
login      session required    libpam_unix.so.1
dtlogin    session required    libpam_hpsec.so.1
dtlogin    session required    libpam_ldap.so.1
dtlogin    session required    libpam_unix.so.1
dtaction   session required    libpam_hpsec.so.1
dtaction   session required    libpam_ldap.so.1
dtaction   session required    libpam_unix.so.1
ftp        session  required   libpam_hpsec.so.1 bypass_limit_login
                                      bypass_umask bypass_nologin
ftp        session requried    libpam_ldap.so.1
ftp        session required    libpam_unix.so.1
rcomds     session required    libpam_hpsec.so.1 bypass_limit_login
rcomds     session required    libpam_ldap.so.1
rcomds     session required    libpam_unix.so.1
sshd       session required    libpam_hpsec.so.1
sshd       session required    libpam_ldap.so.1
sshd       session required    libpam_unix.so.1
OTHER      session required    libpam_ldap.so.1
OTHER      session required    libpam_unix.so.1
# Password management
#
```

```
login      password required    libpam_hpsec.so.1
login      password sufficient  libpam_ldap.so.1
login      password required    libpam_unix.so.1 try_first_pass
passwd     password required    libpam_hpsec.so.1
passwd     password sufficient  libpam_ldap.1
passwd     password required    libpam_unix.so.1 try_first_pass
dtlogin    password required    libpam_hpsec.so.1
dtlogin    password sufficient  libpam_ldap.so.1
dtlogin    password required    libpam_unix.so.1 try_first_pass
sshd       password required    libpam_hpsec.so.1
sshd       password sufficient  libpam_ldap.so.1
sshd       password required    libpam_unix.so.1 try_first_pass
OTHER      password sufficient  libpam_ldap.so.1
OTHER      password required    libpam_unix.so.1 try_first_pass
```

D Sample /etc/pam.conf File for Security Policy Enforcement

This Appendix provides the sample PAM configuration file, `/etc/pam.conf` file to support account and password policy enforcement for Secure Shell (SSH) key-pair or r-commands. In the `/etc/pam.conf` file, the pam_authz library must be configured for the `sshd` and `rcommds` services under `account management` role.

The following is a sample PAM configuration file, `/etc/pam.conf`, used on the HP-UX 11i v1 system:

```
#
# PAM configuration
#
# This pam.conf file is intended as an example only.
#
#

################################################################
# This configuration file has only been modified for default   #
# services. Other services can be added or modified as needed  #
# or desired. If a service is not listed, it will use the      #
# OTHER classification.                                        #
#                                                              #
# the format for a entry is                                    #
# <service> <module_type> <control> <module path> <options>    #
#                                                              #
# see pam.conf(4) for mor details                              #
#                                                              #
#                                                              #
################################################################
#
# Authentication management
#
login      auth sufficient   /usr/lib/security/libpam_unix.1
login      auth required     /usr/lib/security/libpam_ldap.1 try_first_pass
su         auth sufficient   /usr/lib/security/libpam_unix.1
su         auth required     /usr/lib/security/libpam_ldap.1 try_first_pass
dtlogin    auth sufficient   /usr/lib/security/libpam_unix.1
dtlogin    auth required     /usr/lib/security/libpam_ldap.1 try_first_pass
dtaction   auth sufficient   /usr/lib/security/libpam_unix.1
dtaction   auth required     /usr/lib/security/libpam_ldap.1 try_first_pass
ftp        auth sufficient   /usr/lib/security/libpam_unix.1
ftp        auth required     /usr/lib/security/libpam_ldap.1 try_first_pass
sshd       auth sufficient   /usr/lib/security/libpam_unix.1
sshd       auth required     /usr/lib/security/libpam_ldap.1 try_first_pass
OTHER      auth sufficient   /usr/lib/security/libpam_unix.1
OTHER      auth required     /usr/lib/security/libpam_ldap.1 try_first_pass
# Account management
#
login      account sufficient  /usr/lib/security/libpam_unix.1
login      account required    /usr/lib/security/libpam_ldap.1
su         account sufficient  /usr/lib/security/libpam_unix.1
su         account required    /usr/lib/security/libpam_ldap.1
dtlogin    account sufficient  /usr/lib/security/libpam_unix.1
dtlogin    account required    /usr/lib/security/libpam_ldap.1
dtaction   account sufficient  /usr/lib/security/libpam_unix.1
dtaction   account required    /usr/lib/security/libpam_ldap.1
ftp        account sufficient  /usr/lib/security/libpam_unix.1
ftp        account required    /usr/lib/security/libpam_ldap.1
rcomds     account required    /usr/lib/security/libpam_authz.1
rcomds     account sufficient  /usr/lib/security/libpam_unix.1
rcomds     account required    /usr/lib/security/libpam_ldap.1 rcommand
sshd       account required    /usr/lib/security/libpam_authz.1
sshd       account sufficient  /usr/lib/security/libpam_unix.1
sshd       account required    /usr/lib/security/libpam_ldap.1 rcommand
OTHER      account sufficient  /usr/lib/security/libpam_unix.1
OTHER      account required    /usr/lib/security/libpam_ldap.1
# Session management
#
login      session sufficient  /usr/lib/security/libpam_unix.1
login      session required    /usr/lib/security/libpam_ldap.1
dtlogin    session sufficient  /usr/lib/security/libpam_unix.1
dtlogin    session required    /usr/lib/security/libpam_ldap.1
dtaction   session sufficient  /usr/lib/security/libpam_unix.1
dtaction   session required    /usr/lib/security/libpam_ldap.1
sshd       session sufficient  /usr/lib/security/libpam_unix.1
sshd       session required    /usr/lib/security/libpam_ldap.1
OTHER      session sufficient  /usr/lib/security/libpam_unix.1
OTHER      session required    /usr/lib/security/libpam_ldap.1
# Password management #
login      password.sufficient    /usr/lib/security/libpam_unix.1
login      password required      /usr/lib/security/libpam_ldap.1 try_first_pass
passwd     password sufficient    /usr/lib/security/libpam_unix.1
passwd     password required      /usr/lib/security/libpam_ldap.1 try_first_pass
dtlogin    password sufficient    /usr/lib/security/libpam_unix.1
dtlogin    password required      /usr/lib/security/libpam_ldap.1 try_first_pass
dtaction   password sufficient    /usr/lib/security/libpam_unix.1
dtaction   password required      /usr/lib/security/libpam_ldap.1 try_first_pass
OTHER      password sufficient    /usr/lib/security/libpam_unix.1
OTHER      password required      /usr/lib/security/libpam_ldap.1 try_first_pass
```

The following is a sample PAM configuration file, `/etc/pam.conf`, used on the HP-UX 11i v2 system:

```
#
# PAM configuration
#
# This pam.conf file is intended as an example only.
#
#

################################################################
# This configuration file has only been modified for default   #
# services. Other services can be added or modified as needed  #
# or desired. If a service is not listed, it will use the      #
# OTHER classification.                                        #
#                                                              #
# the format for a entry is                                    #
# <service> <module_type> <control> <module path> <options>    #
#                                                              #
# see pam.conf (4) for more details                            #
#                                                              #
################################################################
#
# Authentication management
#
login      auth required        libpam_hpsec.so.1
login      auth sufficient      libpam_unix.so.1
login      auth required        libpam_ldap.so.1 try_first_pass
su         auth required        libpam_hpsec.so.1
su         auth sufficient      libpam_unix.so.1
su         auth required        libpam_ldap.so.1 try_first_pass
dtlogin    auth required        libpam_hpsec.so.1
dtlogin    auth sufficient      libpam_unix.so.1
dtlogin    auth required        libpam_ldap.so.1 try_first_pass
dtaction   auth required        libpam_hpsec.so.1
dtaction   auth sufficient      libpam_unix.so.1
dtaction   auth required        libpam_ldap.so.1 try_first_pass
ftp        auth required        libpam_hpsec.so.1
ftp        auth sufficient      libpam_unix.so.1
ftp        auth required        libpam_ldap.so.1 try_first_pass
rcomds     auth required        libpam_hpsec.so.1
rcomds     auth sufficient      libpam_unix.so.1
rcomds     auth required        libpam_ldap.so.1 try_first_pass
sshd       auth required        libpam_hpsec.so.1
sshd       auth sufficient      libpam_unix.so.1
sshd       auth required        libpam_ldap.so.1 try_first_pass
```

```
OTHER      auth sufficient      libpam_unix.so.1
OTHER      auth required        libpam_ldap.so.1 try_first_pass
# Account management
#
login      account required     libpam_hpsec.so.1
login      account required     libpam_authz.so.1
login      account sufficient   libpam_unix.so.1
login      account required     libpam_ldap.so.1
su         account required     libpam_hpsec.so.1
su         account sufficient   libpam_unix.so.1
su         account required     libpam_ldap.so.1
dtlogin    account required     libpam_hpsec.so.1
dtlogin    account sufficient   libpam_unix.so.1
dtlogin    account required     libpam_ldap.so.1
dtaction   account required     libpam_hpsec.so.1
dtaction   account sufficient   libpam_unix.so.1
dtaction   account required     libpam_ldap.so.1
ftp        account required     libpam_hpsec.so.1
ftp        account sufficient   libpam_ldap.so.1
ftp        account required     libpam_unix.so.1
rcomds     account required     libpam_hpsec.so.1
rcomds     account required     libpam_authz.so.1
rcomds     account sufficient   libpam_unix.so.1
rcomds     account required     libpam_ldap.so.1 rcommand
sshd       account required     libpam_hpsec.so.1
sshd       account required     libpam_authz.so.1
sshd       account sufficient   libpam_unix.so.1
sshd       account required     libpam_ldap.so.1 rcommand
OTHER      account sufficient   libpam_unix.so.1
OTHER      account required     libpam_ldap.so.1
# Session management
#
login      session required     libpam_hpsec.so.1
login      session sufficient   libpam_unix.so.1
login      session required     libpam_ldap.so.1
dtlogin    session required     libpam_hpsec.so.1
dtlogin    session sufficient   libpam_unix.so.1
dtlogin    session required     libpam_ldap.so.1
dtaction   session required     libpam_hpsec.so.1
dtaction   session sufficient   libpam_unix.so.1
dtaction   session required     libpam_ldap.so.1
ftp        session required     libpam_hpsec.so.1 bypass_limit_login
                                bypass_umask bypass_nologin
ftp        session sufficient   libpam_unix.so.1
ftp        session required     libpam_ldap.so.1
rcomds     session required     libpam_hpsec.so.1 bypass_limit_login
rcomds     session sufficient   libpam_unix.so.1
rcomds     session required     libpam_ldap.so.1
sshd       session required     libpam_hpsec.so.1
sshd       session sufficient   libpam_unix.so.1
sshd       session required     libpam_ldap.so.1
OTHER      session sufficient   libpam_unix.so.1
OTHER      session required     libpam_ldap.so.1
# Password management #
login      password required    libpam_hpsec.so.1
login      password sufficient  libpam_unix.so.1
login      password required    libpam_ldap.so..1 try_first_pass
passwd     password required    libpam_hpsec.so.1
passwd     password sufficient  libpam_unix.so.1
passwd     password required    libpam_ldap.so.1 try_first_pass
dtlogin    password required    libpam_hpsec.so.1
dtlogin    password sufficient  libpam_unix.so.1
dtlogin    password required    libpam_ldap.so.1 try_first_pass
sshd       password required    libpam_hpsec.so.1
sshd       password sufficient  libpam_unix.so.1
sshd        password required    libpam_ldap.so.1 try_first_pass
OTHER       password sufficient  libpam_unix.so.1
OTHER       password required    libpam_ldap.so.1 try_first_pass
```

Glossary

See also the Glossary in the *Netscape Directory Server for HP-UX Administrator's Guide* available at http://docs.hp.com/hpux/internet.

Access Control Instruction A specification controlling access to entries in a directory.

Access Control List One or more ACIs.

ACI *See* See Access Control Instruction.

Configuration profile An entry in an LDAP directory containing information common to many clients, that allows clients to access user, group and other information in the directory. Clients download the profile from the directory.
See also See also Client Configuration File..

DIGEST-MD5 Message Digest version 5. It is a one-way hash function and always generates 20 bytes of output from text data.

IETF Internet Engineering Task Force; the organization that defines the LDAP specification. See http://www.ietf.org.

LDAP *See* See Lightweight Directory Access Protocol.

LDAP Data Interchange Format (LDIF) The format used to represent directory server entries in text form.

LDIF *See* See LDAP Data Interchange Format.

Lightweight Directory Access Protocol (LDAP) A standard, extensible set of conventions specifying communication between clients and servers across TCP/IP network connections.
See also See also SLAPD..

Name Service Switch (NSS) A framework that allows a host to get name information from various sources such as local files in /etc, NIS, NIS+, or an LDAP directory without modifying applications. See *switch*(4) for more information.

Network Information Service (NIS) A distributed database system providing centralized management of common configuration files, such as /etc/passwd and /etc/hosts.

NIS *See* See Network Information Service.

NSS *See* See Name Service Switch.

PAM *See* See Pluggable Authentication Mechanism.

PAM Authorization Service Module *See* The PAM Authorization Service Module allows the administrator to control which user subgroups of a large repository can login to the system pam_authz(5)..

Pluggable Authentication Module (PAM) A framework that allows different authentication service modules to be made available without modifying applications. See *pam_ldap*(5), *pam*(3), and *pam.conf*(4) for more information.

Profile *See* See Configuration profile.

RFC Request for Comments; a document and process of standardization from the IETF.

RFC 2307 The IETF specification for using LDAP as a Network Information Service. See http://www.ietf.org/rfc/rfc2307.txt.

SLAPD The University of Michigan's stand-alone implementation of LDAP, without the need for an X.500 directory.

Start-up file A text file containing information the client needs to access an LDAP directory and download a configuration profile.
See also See also Configuration profile.configurationstart-up file ldapux_client.confstart-up file ldapux_client.confldapux_client.conf start-up fileclient start-up file ldapux_client.conf.

ypldapd The NIS/LDAP Gateway daemon, part of the NIS/LDAP Gateway subproduct. ypldapd replaces the NIS ypserv daemon by accepting NIS client requests and getting the requested information from an LDAP directory rather than from NIS maps.

See *Installing and Administering NIS/LDAP Gateway* at http://docs.hp.com/hpux/internet

Index