IBM

Integrating AIX into Heterogeneous LDAP Environments

Describes the latest integration methods based on AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance Package

Provides detailed guidance for planning and implementation

Includes integration scenarios for IBM Tivoli Directory Server, Sun ONE Directory Server, Microsoft Active Directory Server, and OpenLDAP

Ed Geraghty
Philip Hsieh
Martin Loewer
Bill Moraca
John Tesch

Redbooks

ibm.com/redbooks

International Technical Support Organization

Integrating AIX into Heterogenous LDAP Environments

May 2006

SG24-7165-00

Note: Before using this information and the product it supports, read the information in "Notices" on page xi.

First Edition (May 2006)

This edition applies to AIX 5L Version 5.3.

Contents

Notices

Trademarks

Preface

This IBM® Redbook is a technical planning reference for IT organizations that are adding AIX® 5L™ clients to an existing LDAP authentication and user management environment. It presents integration scenarios for the AIX 5L LDAP client with IBM Tivoli® Directory Server, the Sun™ ONE™ Directory Server, and Microsoft® Active Directory.

The sample integration scenarios can be used as a road map for administrators migrating AIX 5L users from traditional local file authentication to an LDAP server, or for adding new AIX 5L boxes to an environment where there are users already defined in the aforementioned directory server products.

- Part 1, “AIX and LDAP” on page 1: The chapters in Part 1 introduce the book, provide a history of AIX and LDAP integration, and provide a detailed discussion of LDAP migration planning topics.
- Part 2, “LDAP client integration” on page 99: This part starts with a detailed chapter on AIX 5L LDAP client installation, integration, and configuration topics that apply to all of the following specific integration scenarios. The rest of Part 2 includes chapters that describe integration scenarios for four LDAP server environments: Sun ONE, IBM Tivoli Directory Server, OpenLDAP, and Microsoft Windows® 2003 Active Directory Server.
- Part 3, “Appendixes” on page 271: This part provides background and technical reference information supporting the integration scenarios presented in this book. The appendixes include IBM Tivoli Directory Server V6.0 Installation steps, Microsoft Windows 2003 Active Directory Server configuration procedures, example certificate authority setup procedures, an overview of schemas and migration tools, and an AIX 5L LDAP quick reference.

The team that wrote this redbook

Production of this IBM Redbook was managed by:

Chris Almond, an ITSO Project Leader and IT Architect based at the ITSO Center in Austin, Texas. In his current role, he specializes in managing content development projects focused on Linux® and AIX 5L systems engineering. He has a total of 15 years of IT industry experience, including the last six with IBM.

This redbook was produced by the team of specialists listed below. They came from around the world to work together at IBM International Technical Support Organization in Austin, Texas, in September and October of 2005.

Ed Geraghty is a Senior Software Engineer in IBM Advanced Technology Group in Boston, MA. He was Technical Web Master on several high-profile Web sites such as the IBM Olympic Web site, Sydney Olympic Web Store, IBM Intellectual Patent Network, and other sports sites. He is currently working with IBM high-performance computing customers in the Boston area. His areas of expertise include high-volume Web infrastructures, IT security, networks, pSeries® and xSeries® systems, and Linux/AIX. He has written extensively on the Microsoft Active Directory and AIX integration in this book.

Philip Hsieh is a Senior System Administrator in a large financial company. He has over 10 years of experience in managing a large multivendor UNIX® environment. His area of expertise is in Sun Solaris™, NIS+, and LDAP naming service. He holds a Master's degree in Electrical Engineering from the University of Massachusetts at Amherst.

Martin Loewer (CISSP) an IT Specialist at IBM in Mainz, Germany. He has more than six years of experience in managing UNIX systems and has extensive background in Network Infrastructure and Information Security disciplines. Martin studied Communication Technology and graduated from the University of Applied Science in Mannheim, Germany. He has performed several UNIX projects for international companies ranging from fortune 500 to small and medium businesses to assess network and security issues and to provide customers with appropriate solutions.

Bill Moraca is a Senior Field Technical Sales Support specialist in the Southeast region of the USA. Bill has spent the last 16 of his 27 years in the IT business with IBM in a variety of jobs emphasizing multivendor connectivity, AIX operational services, and most recently pSeries hardware sales. While supporting many AIX customers, his focus is on introducing pSeries/AIX into competitively installed accounts.

John Tesch is a Certified IT Consultant working for IBM Advanced Technical Support, Americas. His primary job is providing second-level technical support in pSeries and AIX to the Americas Field Technical Support Organization. He has worked for IBM for 24 years and has 14 years of experience working with AIX. His specialties include Power5 logical partitioning and virtualization, Java™ performance, LDAP authentication, networking, and printing. John has co-authored an IBM Redbook about AIX printing and one about IBM Network Stations. Prior to working with AIX he taught vector programming on IBM System 390 and worked in the development organization for the IBM lab computer, the CS9000. He holds a Ph.D. in Analytical Chemistry from the University of Colorado in Boulder.

A complete and detailed book on a topic such as this would not be possible without generous support and guidance from key staff members in the AIX 5L development organization, as well as other IBMers. This IBM Redbook team would like to acknowledge the following people for their excellent contributions in support of this project:

Carl Burnett, IBM AIX Kernel Architecture Team, for his overall guidance in helping us develop a content strategy for this book.

Ufuk Celikkan, **Drew Walters, Ravi Shankar, Jeroen van Hoof, Sridhar Murthy, and Yantian (Tom) Lu**, IBM AIX Security Development, for providing technical support and review feedback to the team during book development.

Become a published author

Comments welcome

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Part 1

AIX and LDAP

1 Introduction

We begin this book by describing at a high level the business requirements that make it necessary to create a system that can centrally manage user information and authentication. Following that, we define what LDAP is, and how it defines a protocol for accessing and updating information in directory databases. LDAP-enabled directories have become a very popular choice for storing and managing user information. We discuss many of the reasons why LDAP has emerged as the dominant method managing directories. Finally, we conclude with a discussion of planning considerations for LDAP user migration and an overview of each of the AIX 5L integration scenarios presented in this book.

1.1 The goal of centralized user information

Many computer users today have access to more than one computer, and normally this means that they have multiple user names and passwords. As security at an organization tightens, users are asked to make passwords that are more complicated and to change these passwords on a periodic basis. This can mean a lot of work for users, and in fact can reduce security as users are forced to write down passwords for different systems.

1.1.1 User management tasks

System administrators can also have a considerable amount of work as users are added and deleted from the system. The administrator must manage the users. When this book talks about user management, the term means any actions that are required to add, remove, or change any user attribute. Let us review some of the tasks that a system administrator performs in user management.

Assigning the user a name

The user name is used by the user to log on to the system, and must be unique. Prior to AIX 5L Version 5.3, the length of the user name was eight characters or less, meaning that AIX users normally had a different user name on the AIX system than they have on other UNIX/Linux systems and for logging into Windows. Beginning with AIX 5L Version 5.3, the user name can now contain up to 254 characters.

Assigning the user an ID

The user ID is used to assign a number used to define the permissions this user has for reading and writing files and for running programs. When using network file systems, it is important the user IDs from multiple systems match to avoid the user not having access to files on the other system, or inadvertently giving access to their files to another user.

Assigning the user to groups

To make assigning permissions easier, users are bunched together in groups that have similar functions on the system. Each user must belong to a primary group, and can also be assigned to other groups that give them special permissions on the system.

Assigning the user a password

Each user must be assigned a password that they have the authority to change. In addition to assigning the initial password, the administrator can also assign rules pertaining to the user's password such as the number of alpha and numeric characters, whether special symbols must be used, how often the password must be changed, and how soon passwords can be reused.

Assigning the user a HOME disk space

When a user logs onto a system, the Present Working Directory (PWD) is normally set to a value specified by a HOME environment variable. There are usually configuration files in this directory that are used to customize the user experience on the system and to give the user some space to store files that belong to him. The file can be disk space that is local to the system the user is logged in to, or can be storage that is accessed over the network.

Assigning other user attributes

System administrators also manage other user attributes like the default SHELL, the default PATH where executable commands are found, and other environmental variables assigned at login time. Once again, the administrator can control these for each user when they set up the user on the system. Normally, most of these can be set to a default for all users, but in some cases, special restrictions or privileges can be given to a particular user.

Removing users

When a user leaves the company, the department, or for some reason no longer has a need to use a system, the assets on the system that belong to the user must be removed from the system. An optional task when removing a user is to back up the user's information in a safe archive, in case it needs to be recovered in the future.

1.1.2 Centralized user information

Work multiplies when new servers are brought online or when new employees are hired. When a new server is acquired, many of the same users that are on other systems will need access to it. In the absence of a central user repository some combination of file copy, scripting, and manual customization is necessary to duplicate user information on the new system. A new employee has to be added directly to each system she will access. A central user repository greatly simplifies the addition of users and systems. The new employee is added to one place with one password and a list of the systems she can use. The new system is added to one place along with a list of the users allowed to access it. As servers proliferate, the increased administrative workload for user management increases the need for a central repository.

New laws and security concerns as well as the proliferation of operating system images with mechanisms such as logical partitions (LPARs) have prompted a movement toward a secure centralized user management scenario.

Central user management can include all the functions of user management listed above, or can be limited to a single aspect such as user authentication.

Some of the solutions for central management include NIS, NIS+, CDE, Kerberos, and LDAP. Chapter 2, “History of AIX and LDAP” on page 19, contains a review of these methodologies.

This book focuses on using LDAP as the mechanism to support centralized user management.

1.2 LDAP definition

The term LDAP stands for Lightweight Directory Access Protocol. It was derived from X.500-based directory services, which is an industry-standard directory-access protocol. The X.500 specification, which was created in 1988, organizes directory entries in a hierarchal name space capable of accessing large amounts of information. As an application layer protocol, X.500 requires the entire OSI protocol stack to operate. The LDAP defines a standard method for accessing and updating information in a directory that requires less resources and is easier to maintain than X.500. LDAP is based on TCP/IP.

For a more detailed description of LDAP, see IBM Redbook *Understanding LDAP - Design and Implementation*, SG24-4986.

1.2.1 What is a directory

A directory is a listing of information that gives details about the information stored inside the directory. Common examples of directories include a telephone directory or a library card catalog.

In computer terms, a directory is a specialized data repository that stores the ordered information in an easily searchable and retrievable format.

A directory is often described as a specialized database. The main characteristic is that the database is searched and read much more often than it is updated, because the information in it rarely changes. This characteristic is applicable when you think about using a directory to store computer host or user information.

Directory service

A directory service adds the following features to a directory:

- A network protocol to access the directory
- A replication scheme for synchronizing master and slave servers
- A data distribution scheme

LDAP standard

The LDAP standard defines:

- A network protocol to access directory information
- An extensible information model defining the form and properties of the information
- A namespace defining how the information is referenced and organized
- The distributed operation model of how data is distributed and referenced
- The operations for retrieving, storing, and manipulating the data (for example, search, add, delete, modify, and change)

Schemas

Schemas specify what can be stored in the directory. The schema defines the information model for LDAP in terms of what containers hold the data, and what types of data can be stored in those containers. It also defines the namespace, which allows storage and retrieval of information. A schema is made up of a series of entries. An entry is simply a place where one stores attributes. Each attribute has one or more values. When adding or retrieving information, the data is formatted into text files called Lightweight Directory Information File (LDIF) files. The attribute is always the first entry on a line followed by a colon. The values for that attribute are listed after the colon and a space. Entries are named to identify the meaning of the value stored in a key attribute. This attribute and its value are called the Relative Distinguished Name (RDN™). A schema will define entries as to which attributes belong to an RDN. For each attribute defined, the schema will tell whether that attribute is required to be there or just optional. It also defines the type of data that can be stored in the value field of the attribute and where the data will be a single value or a list of values. In addition, it defines whether a single attribute of that name is allowed, or whether multiple values are allowed. For example, a record for people may contain multiple common names for that person, for example, cn=John, cn=Johnny, cn=Jon, and cn=Jack. This is important because it allows searching that may potentially match any name that they may commonly be called.

In Example 1-1 on page 8 the entry `dn: ou=People` is the base distinguished name (DN) for the entry. The value ou=people defines the organizational unit or object class that will be used to define people added to the database. The entry `objectClass: organizationalUnit` says that later entries defined by ou: People will contain attributes that are defined in a schema that is called organizationalUnit.

Example 1-1 Example LDAP container entry in LDIF format

```
dn: ou=People,
ou: People
objectClass: organizationalUnit
```

Some of the common attribute names that you will see are DN (distinguished name), ou (organizational unit), cn (common name), objectClass, uid, and userpassword. The objectClass is a container class that defines values in an entry. By combining multiple objectClasses in an entry, you will get a collection of all of the attributes from the different objectClasses. This is what is known as extensibility or the ability to add new types of information in an entry type. Later in this book you will see how this is valuable when defining users that have all of the attributes defined in a standard, plus additional attributes necessary to support requirements.

Example 1-1 defines a container for people. Example 1-2 shows how some of the data defined by that container entry would look in an LDIF file.

Example 1-2 Example LDAP user entry in LDIF format

```
dn: uid=guest,ou=People,ou=accounting,ou=xyz,ou=com
uid: guest
objectClass: aixauxaccount
objectClass: shadowaccount
objectClass: posixaccount
objectClass: account
objectClass: ibm-securityidentities
cn: guest
passwordchar: !
uidnumber: 100
gidnumber: 100
homedirectory: /home/guest
userpassword: {crypt}*
```

The RDN for user guest that makes this user entry unique is shown as the DN (distinguished name) in this case, and it illustrates the hierarchical nature of LDAP naming. The name is also called uid. In this case uid is defined as a character string, which would normally be used to store the user name, while uidnumber is defined as a numerical user ID. Notice that the information stored for user guest could be information defined in any of the objectClasses listed. This entry contains only a small amount of the possible information that could be entered for this guest, all of it defined by the schemas that define these object classes.

LDAP client commands

Most LDAP servers have some type of graphical management interface for adding schemas and data, but it is important to also have data access from the LDAP client. The LDAP client commands normally specify the LDAP server that they are going to talk to and attributes such as the base distinguished name where the action will take place on the server. For some commands, a user name on the LDAP server will be sent as a distinguished name (DN) and a password will also be sent to give access to information that requires special permission. This DN and password are used to *bind* to the server to give the permissions. If no DN is sent, then the bind is said to be anonymous and will be able to only retrieve information that the server allows an anonymous user to retrieve in much the same way that anonymous FTP works.

The most common LDAP client commands used in this book are defined here:

ldapsearch	The `ldapsearch` program is an LDAP command-line client that can be used to extract information from an LDAP directory by specifying a server, a base DN, and a search filter. The flags for the AIX 5L ldapsearch client are different from those from OpenLDAP. The information is returned in LDIF format.
ldapadd	The ldapadd program is an LDAP command-line client that can be used to import or add directory information from an LDIF file into an LDAP directory.
ldapdelete	The ldapdelete program is an LDAP command-line client that can be used to remove information from an LDAP directory.
ldapmodify	The ldapmodify program is a command-line interface to the ldap_modify, ldap_add, ldap_delete, and ldap_modrdn application programming interfaces that can be used to modify or add entries to the LDAP server from an LDIF file.

1.2.2 Directory client/server model

Directories in LDAP are accessed using the client/server model. An application that wants to read or write information in a directory does not access the directory directly, but uses a set of programs or APIs that cause a message to be sent from one process to another. The second process retrieves the information on behalf of the first (client) application and returns the requested information if the client has permission to see the information. LDAP defines a message protocol used between the LDAP clients and the LDAP directory servers. This includes methods to search for information, read information, and to update information based on permissions.

To increase the availability, you can have a single master LDAP server with multiple replica servers or multiple synchronized master servers.

1.2.3 Directory security

The security of the information stored in a directory is an important consideration. Before a client can access data from the LDAP server, there is an authentication procedure based on a client user name called a bind DN and a password. Once the user is authenticated, then the user's access to data is based on access control lists (ACLs). An ACL is simply a list of what type of access the user has to information in the directory to read or modify the data. It may be possible that a user can have permission to change the value of an object, but not be able to delete the object, or that the user can only read certain fields of the data, but not others.

1.3 Why use LDAP for user information

LDAP provides a centralized data store of user access that can be securely accessed over the network. It allows the system administrators to manage the users from multiple servers in one central directory.

Many readers are familiar with Sun's Network Information System (NIS), one of the first attempts at centralizing user administration. The following sections highlight LDAP's advantages over NIS in the areas of scalability, security, and availability. These advantages are a key impetus for IT shops to move to LDAP. Another key reason why people are moving to LDAP for user management is the proposed RFC2307 standard to replace NIS with LDAP. More information about NIS can be found 2.1.2, "NIS services" on page 21.

1.3.1 Security of user information

The security of user names and passwords is the most important consideration of any user administration facility. LDAP has the ability to restrict access to data on the server (directory security) and also to encrypt password information during authentication.

LDAP directory security is implemented via configurable ACLs. The ACL implementation and syntax varies across LDAP server products, but the general function of restricting add/change/delete functions to specific users is always provided. The ACL implementations are sufficiently granular so that it is possible to set some user information to read only, while other user information can be read or changed only by specific users.

In any authentication scenario information is exchanged between the directory server and the client where access is being sought. LDAP allows secure communications between the LDAP client and server with encrypted SSL connections. This allows a setup where passwords never appear on the wire in clear text.

The AIX 5L client solution provides additional security by using a LDAP security client daemon that does all of the communication between the AIX 5L client and the user management LDAP server.

1.3.2 Availability solutions for user authentication

Any user authentication solution must have very high availability. Multiple servers are required to reduce downtime vulnerability. This introduces data synchronization issues among the servers—something that LDAP addresses very well. It may also be desirable for a user to be able to authenticate via other means if no server can be reached. LDAP provides this capability also.

LDAP provides two options for having multiple servers: master-replica and master-master. In a master-replica implementation one of the servers is designated as the primary server. All changes occur there and from this server they are propagated to one or more replica servers. The replication can occur on a scheduled basis or upon completion of a change on the master. In a master-master implementation there is no primary server. Changes made to any server are immediately propagated to the other masters.

AIX 5L has the capability to define a fallback method of authentication should the network be down. You can accomplish this by setting the SYSTEM attribute in /etc/security/user to first use LDAP, and only if that is not available use the AIX 5L local system files. This ensures that system administrators can always have a fallback method to get into the system to do problem determination and provide any fix actions that are required.

The LDAP client is made aware of multiple servers when it is configured. Generally, the LDAP client binds randomly to a server in the list during boot initialization. As an alternative to the random binding, the AIX 5L Version 5.3 LDAP client allows the server list to be prioritized on the command line. The client attempts to bind to the first server, and if it fails, the second server, and so on. If the first server comes available later the client will rebind to it.

Finally, the AIX 5L Version 5.3 LDAP client supports alternative authentication methods if no LDAP server can be found. Specific entries in the /etc/security/user file allow authentication against the local system files if the bind to an LDAP server is not successful.

1.3.3 Scalability

NIS employs a flat file structure for user, group, and host definitions and thus can handle only a limited number of users. LDAP uses a hierarchical data storage solution that scales very well to the tens of thousands of users that a company might want to put in a central directory.

One LDAP server may have a referral entry for a user record that points to the LDAP server where the data for that user actually resides. One use of this might be as a transitional centralized server for two organizations with previously separated LDAP configurations (as in a merger of two companies) as they are consolidated onto a single LDAP infrastructure.

1.3.4 Extensibility

New object types and operations can be dynamically defined by adding new schema or adding to existing schema on the directory server. This allows a schema to be created that contains information that is applicable to all the operating systems supported, and also contain custom information that is useful only to a specific operating system. This allows vendors to maintain standards compatibility and still provide value add in their products. For example, the AIX 5L LDAP client and IBM Tivoli Directory Server (ITDS) cooperate to support AIX-specific security attributes like *failed login count* that are part of an extended RFC2307 schema named RFC2307AIX.

1.3.5 Attribute mapping

Applications that need to use existing schema, but refer to attributes with different names, can use a mapping file that equates the operating system attributes with the schema attributes. AIX 5L started supporting security attribute mapping in AIX 5L Version 5.2. The AIX 5L map files are contained in /etc/security/ldap.

1.3.6 Standard for user authentication - RFC2307

Although the LDAP mechanism provides opportunity for individual UNIX vendors to provide a robust and scalable mechanism for user management, perhaps the greatest strength is provided when open standards are applied to allow an administrator to have a single LDAP server mechanism for management of UNIX users from various platforms.

L. Howard published RFC-2307, "*An Approach for Using LDAP as a Network Information Service*," in 1998. The text of the RFC can be found at:

`http://www.ietf.org/rfc/rfc2307.txt`

As the RFC title suggests, this RFC defines a set of attributes to replace NIS as a naming service. This includes both host naming services as well as user naming services. In this book we focus on the user naming services portion of this RFC.

Some of the user attributes defined in this RFC include uidNumber, gidNumber, gecos, homeDirectory, loginShell, shadowLastChange, shadowMin, shadowMax, shadowWarning, shadowInactive, shadowExpire, shadowFlag, memberUid, memberNisNetgroup, nisNetgroupTriple, nisMapName, nisMapEntry, posixAccount, shadowAccount, posixGroup, nisNetgroup, nisMap, and nisObject. As you can see, this set of attributes provides information for user authentication and authorization as well as NIS netgroup support.

As long as these schema are loaded on an LDAP server, both AIX 5L Versions 5.2 and 5.3 can be configured to use that LDAP server for user authentication and authorization. These schema are also supported by various Solaris, HP-UX, and Linux versions.

1.4 Planning considerations

Many of the considerations for migrating users to LDAP are the same ones that one would have considered when moving users to NIS. These considerations are addressed in more detail in Chapter 3, "Planning for LDAP migration" on page 49", and sample integration chapters. The following section only introduces those issues that are common to all environments.

1.4.1 Which LDAP server you should use

Not all LDAP servers fully support RFC-2307. It is easier to add schema additions to some servers than it is to others.

If you are using the AIX 5L based schemas, then it is easiest to use IBM Tivoli Directory Servers, which already contain all the required schema files. It is possible to add these schema to a Sun ONE Directory Server or OpenLDAP server, but doing so requires extra configuration steps.

Third-party LDAP servers with RFC2307

If you are already using a Sun ONE Directory Server or an OpenLDAP server for managing other systems, then you can configure AIX 5L Versions 5.2 and 5.3 to use this server with the RFC-2307 schema. If your company has standardized on Microsoft Active Directory Server (ADS) for user authentication then AIX 5L Versions 5.2 and 5.3 can be configured to use this server using Kerberos for authentication. The Novell Edirectory Server does not store the password in crypt() format, so you can only use this as your LDAP server if you save a separate crypt() password or use AIX 5L Version 5.3 with server-based authentication.

Microsoft Active Directory as an LDAP server

Microsoft Active Directory Server is an LDAP server with the potential to be used in the enterprise authentication scheme. Microsoft addresses user management in a unique manner that does not conform directly to RFC2307. This poses some challenges for the UNIX administrator that is asked to use ADS as the user management server. In this book we look at potential solutions to this problem that include:

- Combining LDAP with Kerberos

  Combining an LDAP solution with Kerberos authentication or PAM modules allows a more flexible solution, and is available starting in AIX 5L Version 5.2.

- Microsoft UNIX Services for Windows

  With Microsoft Unix Services for Windows, there is a partial solution to the RFC2307 solution, although the standard is not fully compliant. Starting with AIX 5L Version 5.3, you can use server-side authentication to bind against the ADS for authentication.

  See 3.2, “Which schema you should use” on page 60, for an examination of solutions to this problem.

1.4.2 What levels of AIX 5L are in your environment

The LDAP authentication capabilities for AIX 5L versions have increased significantly from the introduction of this function in AIX 4.3.3 to the current state of LDAP integration provided in AIX 5L Version 5.3, and you will see additional function added in the future. If you only have AIX 5L Versions 5.2 and 5.3, then RFC2307 support allows you to function in a heterogeneous LDAP environment with other UNIX variants or with the Microsoft Active Directory Server. AIX 4.3.3 and AIX 5L Version 5.1 clients can only authenticate against an AIX-based LDAP server. If your install base includes both AIX 4.3/5.1 and 5.2/5.3 you may consider two separate LDAP servers with the plan for migrating the older clients to the newer environment under RFC2307 as the operating systems are upgraded.

See 2.6, “LDAP support in AIX versions” on page 46, and Table 2-2 on page 47 for more information about how AIX versions determine which LDAP options are supported.

1.4.3 Common user and group IDs

When users are defined on individual UNIX servers, the user and group IDs can be unique on each individual server, but when you combine the users from these servers to a common user repository the differences must be fixed before the migration to an LDAP solution. This is necessary to provide the proper security mechanisms for file and application ownership.

1.5 Sample LDAP integration scenarios

To provide a road map for IT planning, a number of sample scenarios are provided in this book. This section contains a brief overview of the scenarios that are presented.

1.5.1 AIX 5L Version 5.3 LDAP client configuration

A number of new features were added to the LDAP security client daemon in AIX 5L Version 5.3. For example, starting with AIX 5L Version 5.3 the client can use a server-based authentication mechanism and can prioritize authentication against a group of LDAP servers. These and other new functions are presented in Chapter 2, “History of AIX and LDAP” on page 19, and Chapter 4, “AIX 5L Version 5.3 LDAP client configuration” on page 101. Because the setup of the LDAP client is similar for all the supported LDAP servers, the client setup and configuration that is common to all LDAP server platforms is described in Chapter 4, “AIX 5L Version 5.3 LDAP client configuration” on page 101.

1.5.2 Integration with Microsoft Active Directory Server

The number of Microsoft Windows users in most companies today is larger than the number of AIX 5L users or any other UNIX users. Often these users authenticate against the Microsoft Active Directory Server. Often most of the UNIX users are also Windows users and their information already exists in this directory. For this reason, some administrators desire to combine the Windows and UNIX user authentication into a single directory server. There is currently no standard that includes both UNIX and Microsoft solutions. In Chapter 8, “Scenario: Microsoft Windows 2003 Active Directory integration” on page 217, we discuss some of the potential solutions that might be applied when integrating

AIX 5L Version 5.3 and Microsoft ADS. Three potential solutions are explored in this chapter:

- 8.1, “Scenario: Kerberos only” on page 219

  In this section AIX authenticates against the Active Directory using the Kerberos KRB5A load module and does user authorization and identification from local AIX files.

- 8.2, “Scenario: Kerberos and LDAP” on page 233

  In this section AIX 5L authenticates against the Active Directory using Kerberos and also retrieves user information from the Active Directory using Microsoft Services for UNIX schemas. This section contains information about how customer mapping files were used.

- 8.3, “Scenario: LDAP only” on page 259

  In this section AIX 5L authenticates against the Active Directory using a server-side authentication with the new AIX 5L Version 5.3 authtype=ldap_auth. The user information is also retrieved from the Active Directory using the mapping files described in the second scenario.

The final section in this chapter discusses troubleshooting techniques for problem determination.

1.5.3 Integration with Sun ONE Directory Server

Many customers who use NIS on AIX come from a Sun background and often have LDAP authentication set up for their Solaris systems using Sun ONE Directory Server, formerly known as Netscape Directory Server. Chapter 5, “Scenario: Sun ONE LDAP server integration” on page 157, is a case study of how one AIX customer migrated their AIX user management into an already existing Sun ONE Directory Server environment. The major sections in this chapter include:

- 5.2, “Environment” on page 158

  This section describes a typical client’s existing Sun LDAP environment and the desire to integrate AIX users into this environment.

- 5.3, “AIX 5L client configuration” on page 158

  This section describes how the AIX 5L LDAP client was configured to use the LDAP server.

- 5.4, “Restriction of user login with compat mode (netgroups)” on page 164

  This section describes how the login capabilities on AIX 5L clients were configured to use their netgroup.

- 5.5, “automount configuration” on page 173

  This section describes how the user’s home directory was configured to use LDAP-defined automounter files.

- 5.6, “SSL configuration” on page 178

  This section describes how a secure SSL connection was configured to go between the AIX 5L LDAP client and the Sun ONE Directory Server.

1.5.4 Integration with IBM Tivoli Directory Server 6.0

AIX 5L Version 5.3 is shipped with IBM Tivoli Directory Server (ITDS) Version 5.2, but the latest version of ITDS is now Version 6.0. For this reason, we show how to install ITDS Version 6.0 and to configure the AIX 5L Version 5.3 LDAP authentication client to use this directory server. Configuring ITDS Version 5.2 is very similar to configuring Version 6.0.

Chapter 6, “Scenario: IBM Tivoli Directory Server V6.0 integration” on page 187, describes how the ITDS Server is configured to support the AIX 5L LDAP client. Topics described in this chapter include:

- 6.1, “Existing environment” on page 188

  This section describes the AIX 5L environment before integration with LDAP.

- 6.2, “Planned environment” on page 188

  This section describes the desired environment with LDAP integration for user information.

- 6.3, “Migration steps” on page 193

  This section describes how to migrate the data from the existing AIX files format into the LDAP directory server as well as set up secure communications between the client and server.

- 6.4, “Working with the AIX 5L clients” on page 204

  This section describes how working with the administration of AIX 5L users is different when they are in the LDAP environment.

- Appendix A, “IBM Tivoli Directory Server V6.0 installation” on page 273

  This appendix describes how the ITDS Server was set up to support the AIX or other LDAP user clients as well as how to set up security, including generating the SSL key files required for secure communications between the AIX LDAP security client daemon and the ITDS Server.

2

History of AIX and LDAP

This chapter discusses the progression of changes that have occurred in AIX using an LDAP server from the time it was first introduced (in AIX 4.3.3) until now. (The most recent version at the time this book was published is AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance package.) The goal of this chapter is to introduce as many of the new features as possible, placing them in one place. This chapter is not intended to describe the features at the same level as the AIX 5L Differences and Security guides.

Note: In the remainder of this book we refer to *AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance package* as *AIX 5L Version 5.3 RML3*.

In this chapter the following topics are discussed:

2.1 Historical context for LDAP integration

As the number of systems that any user has access to increases, the number of user IDs and passwords that the user must keep track of also increases. As this number increases, the ability to remember and maintain this user information becomes more difficult. The user may try to maintain the same password on all systems, but when forced to change these on a regular basis, the task of making the changes can consume a considerable amount of time and organization. One potential solution to this problem is to create a central repository for this information to keep track of all the systems the user has access to.

The concept of defining users in a single repository for a UNIX operating system such as AIX is not completely new. Naming services such as NIS and NIS+ have been around for over ten years on AIX. Network authentication with Kerberos has been an integral part of Distributed Computing Environment (DCE) on AIX. Kerberos and an open source utility called supper have been used on IBM RS/6000® SP systems to synchronize user information across multiple nodes.

This chapter describes the evolution of AIX user management from local user file management to the LDAP server management capability in AIX 5L Version 5.3 ML3 as of September 2005.

2.1.1 Traditional file-based security in AIX

In AIX file-based management, there are a few files that provide all of the user and group information. These files include:

/etc/security/user	This file contains the default user information about the system as well as a list of all locally defined users and their password and login restrictions and the type of authentication that the user can use.
/etc/passwd	This file contains the user’s user ID, user name, home directory, login shell, primary group ID, and finger information.
/etc/security/passwd	This file contains encrypted passwords, password flags, and other security information.
/etc/security/login.cfg	This file contains default login information for different TTY ports, other system-wide security attributes such as the valid shells, and starting with AIX 5L Version 5.3 the auth_type.
/etc/group	This file contains a list of all the locally defined groups and a list of users assigned to each group.

/etc/security/lastlog	This file contains information about the last location that a user logged in from and the date and time of the last unsuccessful login and the unsuccessful login count.
/etc/security/failedlogin	This file contains a record of unsuccessful login attempts stored in utmp format and used by the accounting and audit subsystems.
/etc/security/environ	This file contains the environment attributes that can be defined for individual users. Global environment variables are defined in /etc/environment.
/etc/security/limits	This file contains the process resource limits for users.

When a user tries to log in to AIX 5L, the security subsystem first checks to see whether the user exists in /etc/security/user. If it does, then it uses crypt() to compare the entered password to the data stored in /etc/security/passwd. If the authentication succeeds, then the credentials are returned from /etc/passwd, /etc/group, and /etc/security/user.

2.1.2 NIS services

Network Information Service (NIS, also known as *yellow pages*) was introduced by Sun in 1985 to enable central administration of operating system data. NIS stores information in maps that are accessible by all computers in the NIS domain through remote procedure calls. NIS clients bind to the NIS server during system boot. If a particular server is down, the client will attempt to bind to another server. Client systems bound to an NIS server resolve TCP/IP names and user name and password information against a combination of local files and the NIS maps from the server.

NIS allows multiple servers in a management domain to prevent clients from being unable to function due to NIS server overload or unavailability. One of the servers in an NIS domain serves as the NIS master server. The local files on the NIS master server (like /etc/passwd, for instance) contain all of the definitions for the domain it is serving. The local files are used to build maps that NIS clients use to resolve various requests. The other NIS servers (called slave servers) in the domain periodically pull read-only copies of the maps from the master server (or the master server may push copies). Map replication has some shortcomings. Among these are a high amount of network traffic because complete copies are always shipped. More importantly, changes to the master are not automatically pushed to the slaves. So it is possible for the master server to operate with fresh maps after a change while slaves use older map data. An NIS client bound to an un-refreshed slave could have different results than a client bound to the master. NIS has security shortcomings as well. Early implementations lacked support for storing user passwords outside of the /etc/passwd file. Such *shadow password files* are the norm for UNIX systems now, but that was not the case 20 years ago. All of these deficiencies render NIS less than ideal for enterprise-wide solutions. Organizations still using NIS may consider moving to LDAP since it addresses these shortcomings and provides a more scalable and flexible administration model.

2.1.3 NIS+

NIS+ was a complete rewrite of NIS that enhanced both the scalability and security capabilities of NIS. Major new features of NIS+ include hierarchal name space, client authentication with encryption and secure RPC, and a more flexible data structure. For a number of reasons, NIS+ was widely adopted by Sun customers but not by those of other UNIX vendors. NIS+ is not a recommended strategy over the long term since both Sun and HP have announced plans to discontinue support for NIS+ in future releases in favor of LDAP. IBM AIX does not fully support NIS+ and has not announced plans to do so in the future.

2.1.4 DCE

Distributed Computing Environment (DCE) is middleware software consisting of development tools, servers, and commands. It is used to manage distributed client and server applications in a heterogeneous computing environment. The security core component of DCE provides services for securing a DCE application and provides authentication, authorization, delegation, login, and other services. The underlying authentication mechanism used Kerberos. DCE was widely used as a directory service as a part of the RS/6000 SP environment and with PSSP. One of the primary DCE applications was the Distributed File System (DFS™) application. As the IBM DEC product is phased out, other solutions including LDAP are being used to replace this functionality.

2.2 Original AIX 4.3.3 and AIX 5L Version 5.1 LDAP authentication

Significant changes in the way AIX authentication is done were introduced in AIX 5L Version 5.1 using something called the loadable authentication module. This was first described in the AIX Version 4.3.3 /usr/lpp/bos/README framework. The mechanism is more fully documented in the AIX 5.2 security guide and changes from earlier AIX versions are described in Chapter 9 of IBM Redbook *AIX 5L Differences Guide Version 5.2 Edition*, SG24-5765.

2.2.1 Loadable authentication module (LAM) framework

Prior to AIX 5L Version 5.1, system administrators could add their own authentication mechanisms using the auth1 and auth2 user attributes defined in /etc/security/user. This provided methods for customers and vendors to easily add their own security methods to AIX 5L, but also opened up some security issues if they were not careful.

With the introduction of a more formal method of providing authentication modules that could be applied to AIX 5L user management, a more secure user management framework was provided. Now all AIX 5L user management is done through this framework. Any module that implements this interface can be used to perform authentication and identification functions. Authentication functions included password verification and modification. Identification functions include storage, retrieval, and modification of user and group account information.

When AIX needs to interact with one of the mechanisms, it determines the correct module to use by looking at the SYSTEM and registry parameters defined in /etc/security/user. For example, user Naomi shown in Example 2-1 is defined to use the LDAP server by the following stanza in /etc/security/user.

Example 2-1 Defining individual user for LDAP

```
naomi:
        SYSTEM = "LDAP"
        registry = LDAP
```

These parameters point to a module definition stanza in /usr/lib/security/methods.cfg. This stanza defines the program or programs that will be used for the authentication and identification functions. The AIX 5L supplied program modules that do the work are supplied in /usr/lib/security. The first module supplied in AIX 4.1 was the DCE module. The LDAP module was first supplied in AIX 4.3.3. Example 2-2 shows the entry for an LDAP module definition in /usr/lib/security/methods.cfg that performs both the authentication and identification functions.

Example 2-2 LDAP authentication module definition in methods.cfg file

```
LDAP:
        program = /usr/lib/security/LDAP
        program_64 =/usr/lib/security/LDAP64
```

It is possible to define one program to perform the authentication and another to provide the identification function. In Example 2-3, KRB5A is used to authenticate against a Kerberos Key Distribution Center and the identification functions are done using an LDAP server. This definition is called a compound load module.

Example 2-3 KRB5A, LDAP, and KRB5ALDAP stanzas in methods.cfg

```
LDAP:
        program = /usr/lib/security/LDAP
        program_64 =/usr/lib/security/LDAP64
KRB5A:
        program = /usr/lib/security/KRB5A
        program_64 =/usr/lib/security/KRB5A_64
        options = authonly

KRB5ALDAP:
        options = auth=KRB5A,db=LDAP
```

2.2.2 IBM SecureWay Directory V 3.2.2 shipped with AIX 5L V 5.1

The first LDAP server code that was shipped with the AIX was SecureWay® Directory Server Version 3.2.2. This LDAP server contained AIX proprietary schemas for user management and name services. The directory information was stored in a DB2® 7.1 database and was installed automatically as you installed the SecureWay server, providing a complete LDAP server solution for AIX authentication at no additional cost to the client.

2.2.3 IBM AIX proprietary schema

The first implementation of LDAP authentication on AIX was based on the AIX security model. The AIX proprietary schema supported users, groups, and hosts naming services. This schema can still be used today to provide an AIX-only solution for versions of AIX from 4.3.3 to AIX 5L Version 5.3 and beyond. This solution allows a system administrator to replace all of the information in the /etc/security/user, /etc/passwd, and /etc/security/passwd with information stored on the LDAP server. With this LDAP support, the system administrator could now manage the users of many AIX 5L systems through a central LDAP server.

2.2.4 LDAP security client: secldapclntd

To allow the root user on any of the client machines to be able to add users into the LDAP directory, change user information, and to change user passwords, a client daemon was created that allows special permissions from the AIX client on the LDAP server. As the client daemon started, it read the LDAP administrative bind DN (Distinguished Name) and password from a configuration file (ldap.cfg) that allowed it to gain authority on the LDAP server to create, change, and delete users as well as to change users passwords. All communications between the

AIX client and the LDAP server are done through the secldapclntd daemon. For additional security, a secure connection between the client and server can be set up using SSL encryption. Once again, the secldapclntd daemon gets the SSL key file and key password from the /etc/security/ldap/ldap.cfg file.

When a user authenticates using LDAP with the unix_auth authentication, the secldapclntd actually retrieves the password, which is stored in crypt() format, from the LDAP server, and the authentication is done on the AIX 5L client. Likewise, it is this daemon that has bind access to add and change users. More details about these mechanisms are in 2.4.1, "Server-side authentication support" on page 31, when ldap_auth is introduced.

2.2.5 AIX naming service

The new LDAP integration added support for various name services, that is, NIS, NIS+, DNS, and LDAP. Systems can be configured to resolve network information from any supported name services through settings specified in system configurations files. The system uses the following variable/configuration files to determine the search order for name service resolution:

- NSORDER environment variable: This variable is used to specify the name service used for host resolution only.
- /etc/netsvc.conf: This file is used to specify hosts and aliases information.
- /etc/irs.conf: This file contains the search order for most network-related data, including hosts, networks, services, protocols, netgroups, and automount.
- /etc/rpc.conf: This file is used to control the search order for RPC services.

The configuration required for using LDAP as the name service is listed in Table 2-1.

Table 2-1 LDAP naming service: controlling mechanisms[1]

Entity	Mechanism	Controlled by
Users	LDAP	User's registry attribute in /etc/security/user file
Groups	LDAP	Groups's registry in /etc/security/group file
Hosts	nis_ldap	NSORDER variable, /etc/netsvc.conf and /etc/irs.conf

[1] This is from IBM white paper "*Using LDAP for Naming Services in AIX*" by Yantian Tom Lu. http://www-03.ibm.com/servers/aix/whitepapers/ldap_naming.html

Entity	Mechanism	Controlled by
Networks	nis_ldap	/etc/netsvc.conf and /etc/irs.conf
Protocols	nis_ldap	/etc/irs.conf
Services	nis_ldap	/etc/irs.conf
Netgroups	nis_ldap	/etc/irs.conf
RPC	nis_ldap	/etc/irs.conf
Automount	nis_ldap	/etc/irs.conf

2.3 Improvements to LDAP authentication in AIX 5L Version 5.2

The features added to AIX 4.3.3 and 5.1 provided a solution for managing AIX users in a central manner, but customers were asking for a solution that provided more flexibility in a heterogeneous environment. IBM responded in AIX 5L Version 5.2. The Version 5.2 features are well documented in the *AIX 5L Differences Guide Version 5.2 Edition*, SG24-5765. This section briefly describes these new features.

2.3.1 Support for open standard for LDAP authentication: RFC2307

One of the main limitations with the AIX 4.3.3 and AIX 5L Version 5.1 was that the schema only supported definition of users on an AIX system. A standard for multivendor UNIX authentication had emerged under RFC2307, "*An Approach for using LDAP as a Network Information Service*," written by L. Howard and published in March of 1998. A copy of this document can be obtained from:

http://www.ietf.org/rfc/rfc2307.txt

The RFC category for 2307 is experimental, and it was designed to be a starting point for discussion, as can be seen from this quote from the RFC:

> The intention is to assist the deployment of LDAP as an organizational name service. No proposed solutions are intended as standards for the Internet. Rather, it is hoped that a general consensus will emerge as to the appropriate solution to such problems, leading eventually to the adoption of standards. The proposed mechanism has already been implemented with some success.

Tom Bialaski and Michael Haines of Sun published "*Solaris and LDAP Naming Services, Deploying LDAP in the Enterprise*" in 2001 as a part of the Sun

Blueprints series, which described using the RFC for replacing the NIS environment with LDAP.

A company named PADL (LDAP spelled backwards) developed a number of solutions for using the information from this RFC to provide user management services for a number of UNIX platforms including an authentication-only solution for AIX 4.3.3:

`http://www.padl.com`

AIX 5L Version 5.2's full exploitation of the RFC2307 schema allows users logging in to AIX clients to authenticate against many LDAP servers, including the Sun ONE Directory Server, the OpenLDAP server, IBM Tivoli Directory Server (ITDS), and Microsoft Windows 2003 Active Directory (using Microsoft Windows Services for UNIX). Scenarios describing integration of the AIX 5L client with these servers are provided in Part 2, "LDAP client integration" on page 99, of this book.

The schema support for RFC2307 is fully described in "RFC2307 schema" on page 302.

2.3.2 AIX additions to the RFC2307 schema

Since AIX security includes some features not defined in the RFC2307 schema, an extended schema called RFC2307AIX was also added in AIX 5L Version 5.2. This allows functions such as the saving of password history to be accomplished while retaining full compliance with the RFC2307 schema. The RFC2307AIX extensions are included in IBM Directory Server 4.1 and later, and can be added to other LDAP servers. The method for adding schema to a server is product-specific and beyond the scope of this document, but it is worth noting that the raw schema definition is included in file /etc/security/ldap/nisSchema.ldif on the AIX client system. When using this schema, the added attributes are only visible to the AIX clients and are simply ignored by other UNIX clients.

The additional attributes added to RFC2307 to create RFC2307AIX are described in "RFC2307AIX schema" on page 305.

2.3.3 IBM Directory Server Version 4.1

The SecureWay Directory Server was renamed IBM Directory Server and Version 4.1 was shipped on the AIX 5L Version 5.2 base operating system CD. From an AIX 5L Version 5.2 client perspective the important features included in IBM Directory Server Version 4.1 and later are the inclusion of the three schema that are supported for AIX user authentication and network naming services. These schema are AIX, RFC2307, and RFC2307AIX. Once again, this is a full-function LDAP server that is shipped free of charge on the AIX operating system media.

2.3.4 Object class and attribute name mapping files

Starting with AIX 5L Version 5.2, the LDAP client daemon can now map object class and attribute names defined on the LDAP server to names that the AIX security subsystem understands. The mapping files allow AIX 5L to work with schemas with different object class and attribute names. AIX 5L currently supports mapping files for the user and group entities for the AIX, RFC2307, and RFC2307AIX schemas.

The locations of the map files used by the AIX 5L LDAP client is defined in the /etc/security/ldap/ldap.cfg file by the attributes shown in Example 2-4.

Example 2-4 AIX-LDAP attribute map path

```
# AIX-LDAP attribute map path.
userattrmappath:/etc/security/ldap/2307user.map
groupattrmappath:/etc/security/ldap/2307group.map
idattrmappath:/etc/security/ldap/aixid.map
```

Each attribute to be mapped has four values on the same line. These values are the AIX attribute, the AIX attribute type, the LDAP attribute, and finally a flag indicating whether the attribute can have only a single value or a list of values. See Example 2-5.

Example 2-5 AIX-LDAP attributes

```
# Format:
# AIX_ATTR   AIX_ATTR_TYPE   LDAP_ATTR   LDAP_VALUE
#
# AIX_ATTR:     AIX attribute name
# AIX_ATTR_TYPE AIX attribute type - SEC_CHAR, SEC_INT, SEC_LIST, SEC_BOOL
# LDAP_ATTR     LDAP attribute name
# LDAP_VALUE    LDAP attribute type - "s" for single-valued attribues
#               or "m" for multi-valued attributes.
```

For more details on object class and attribute name mapping files please refer to “Object class and attribute name mappings” on page 310.

2.3.5 White papers on client and server setup

The AIX development group published three white papers in March 2003 to help document the new features in AIX 5L Version 5.2:

- *Using LDAP for Naming Services in AIX 5L Version*

  `http://www-03.ibm.com/servers/aix/whitepapers/ldap_naming.html`
- *Configuring an IBM Directory Server for User Authentication and Management in AIX*

  `http://www-03.ibm.com/servers/aix/whitepapers/ldap_server.html`
- *Configuring an AIX Client System for User Authentication and Management Through LDAP*

  `http://www-03.ibm.com/servers/aix/whitepapers/ldap_client.html`

In addition, the *AIX 5L Version 5.2 Security Guide* has a chapter on LDAP exploitation of the security subsystem and migrating from NIS to LDAP:

```
http://publib16.boulder.ibm.com/doc_link/en_US/a_doc_lib/aixbman/security/ldap_
exploitation.htm
http://publib16.boulder.ibm.com/doc_link/en_US/a_doc_lib/aixbman/nisplus/migrat
ing.htm
```

2.3.6 Authentication against Active Directory using Kerberos

The KRB5A authentication module is introduced in AIX 5L Version 5.2 for the specific purpose of authenticating against a Microsoft Active Directory Server (ADS). The mechanism provides authentication only, but does provide a powerful tool for administrators that want to have all user passwords stored in one location. This mechanism must be combined with local files or an LDAP server for user identification and retrieving other user credentials such as the user shell, home directory, and user and group IDs.

When doing KRB5A authentication against Microsoft ADS, you will need to use a compound load module as described in 2.2.1, "Loadable authentication module (LAM) framework" on page 23. A short description of how to set this up is included in the *AIX 5L Version 5.2 Security Guide* section titled "How do I Configure an AIX Kerberos Client that Authenticates Against an Active Directory Server KDC":

```
http://publib16.boulder.ibm.com/doc_link/en_US/a_doc_lib/aixbman/security/kerbe
ros_questions_troubleshooting.htm#kerberos_questions_troubleshooting
```

An example scenario using KRB5A for authentication with LDAP used for identification is shown in 8.2, "Scenario: Kerberos and LDAP" on page 233.

2.3.7 Introduction of PAM authentication modules

With AIX 5L Version 5.2, there are two mechanisms that use Pluggable Authentication Modules (PAM). The first PAM mechanism uses a PAM library and module called pam_aix that allows applications that use PAM-based authentication to actually use the AIX operating system authentication mechanisms such as LDAP and local files. You can set up your application that uses PAM authentication to use AIX authentication by creating /etc/pam.conf and adding a line such as this:

```
yourapp session required        /usr/lib/security/pam_aix
```

The second PAM mechanism is a PAM authentication module named /usr/lib/security/PAM. In theory this module should allow any AIX authentication application such as login, telnetd, or su to use a PAM-based authentication instead of standard AIX authentication.

/usr/lib/security/PAM ends up being a limited solution because it requires major modifications to the source code of any PAM module before the module can be used. The source code changes are needed because the LAM API uses iterative calls to the authenticate() function for application communication, while PAM uses an out-of-band conversation function to communicate with the application. The different methods of application communication require the PAM modules to be modified to use the pam_get_item() and pam_set_item() calls to communicate with the application in AIX 5L Version 5.2.

Since the native AIX binaries (for example, /bin/login and /bin/su) still call LAM directly, there is no reasonable way for them to use existing off-the-shelf PAM modules. The only alternative is to try to replace the native AIX binaries with open source alternatives that are PAM-aware—a complicated proposition. The authors are not aware of specific open source modules that have been used in this way.

2.3.8 System V printing on AIX and LDAP naming services

Beginning with AIX 5L Version 5.2, the System V print subsystem is directory-enabled, allowing the System V print subsystem to be managed using the information stored in the LDAP directory.

The AIX System V print subsystem's use of IBM Directory allows for centralized storage of print information. This functionality can be used to keep printers, print queues, and system information common in a client-server environment. The **mkprtldap** command configures IBM Directory as a server containing System V print information, and one or more clients that use IBM Directory (LDAP) for print information.

For more details about the **mkprtldap** command and LDAP-enabled printing in AIX 5L Version 5.2 see Configuring Directory-Enabled (LDAP) System V Print on AIX in the AIX printing guide at:

http://publib16.boulder.ibm.com/doc_link/en_US/a_doc_lib/aixbman/printrgd/PS_printers.htm#mont041800bgh1

2.4 Improvements to LDAP authentication in AIX 5L Version 5.3

Many new features were implemented for AIX 5L Version 5.3 RML3 clients. New features such as support for netgroups and automount are designed to make AIX user management more compatible with additions to RFC2307 exploited by other UNIX systems. Other enhancements increase the security and reliability of the LDAP user management solution.

In the remainder of this book use of the AIX 5L Version 5.3 RML3 client is assumed unless specifically noted otherwise.

The key LDAP authentication features introduced in the base release of AIX 5L Version 5.3 include:

- 2.4.1, “Server-side authentication support” on page 31
- 2.4.2, “Proxy admin users” on page 33
- 2.4.3, “Specifying LDAP server priority” on page 35
- 2.4.4, “Support for netgroups” on page 35
- 2.4.5, “Support for automount maps in LDAP” on page 37
- 2.4.6, “Default entry location” on page 37
- 2.4.7, “Schema conversion tool” on page 38

2.4.1 Server-side authentication support

Prior to AIX 5L Version 5.3, all user authentication was done locally on the AIX LDAP client by retrieving the password in UNIX crypt format from the LDAP server and then doing the comparison. Using client-side authentication imposed several restrictions on the LDAP server setup.

The most serious restriction was that the user’s password must always be stored using the UNIX crypt encryption algorithm on the LDAP server. Also, when using the UNIX crypt encryption, the users’s password is restricted to a maximum of eight characters long. Even though most LDAP servers support stronger encryption algorithms, if you used anything besides the UNIX crypt encryption, AIX clients would not be able to authenticate.

The AIX 5L Version 5.3 LDAP client now supports server-side authentication, where the LDAP client authenticates a user by binding to the LDAP server with the user's DN and password. This allows the AIX 5L LDAP client to authenticate users irrespective of the encryption algorithm used to store the password on the LDAP server. This additional flexibility means that now that the user's password is sent to the LDAP server in clear text. Therefore it is recommended that you use SSL to encrypt the communications between client and server.

New mksecldap option for authentication type

The **mksecldap** command has been enhanced by adding the -A option to allow you to choose the authentication type when configuring the LDAP client. The authentication type is set in the /etc/security/ldap/ldap.cfg file with the authtype setting. The possible values for the -A option are:

-A unix_auth	This configures the LDAP client for client-side authentication. This is the default authentication type.
-A ldap_auth	This configures the LDAP client for server-side authentication.

Example 2-6 shows the section of the AIX 5L LDAP client daemon's configuration file related to setting the authentication type.

Example 2-6 AIX 5L LDAP client authentication type settings

```
# Authentication type. Valid values are unix_auth and ldap_auth.
# Default is unix_auth.
# unix_auth - Retrieve user password and authenticate user locally.
# ldap_auth - Bind to LDAP server to authenticate user remotely through LDAP.
authtype:ldap_auth
```

For more planning considerations about server-side and client-side authentication refer to 3.5.3, "Client-side and server-side authentication" on page 86.

2.4.2 Proxy admin users

Prior to AIX 5L Version 5.3, the secldapclntd daemon did a bind to the LDAP server as the LDAP administrator account distinguished name (DN) on the LDAP server to allow the management of users (add users, remove users, change users, and change passwords). The DN and password for this user are stored in clear text in the ldap.cfg file, as shown in Example 2-7.

Example 2-7 DN and password storage in ldap.cfg file

```
# LDAP server bind distinguished name (DN)
#binddn:cn=admin
# LDAP server bind DN password
#bindpwd:secret
```

Note: AIX 5L Version 5.2 can also make use of proxy bind from the client side, although the **`mksecldap`** command in Version 5.2 does not support creating the proxy account during server setup.

One major improvement is that a proxy user is created on the LDAP server with restricted authority to required information on the LDAP server, and the secldapclntd daemon binds using the DN for this user. The mksecldap script that is used to configure both the ITDS Server and the client has new flags that support the proxy user.

Server flags for mksecldap for proxy users

Two new flags are supported when **`mksecldap`** is run with the -s flag. These are:

-x	proxyuserDN
-X	proxyuserPasswd

When **`mksecldap -s`** is run with these flags, a new proxy user is created on the ITDS Server and the proxy user DN and password are placed into the ldap.cfg file.

Default permissions for the proxy user

The default permissions (Access Control Lists, ACLs) are defined in the file /etc/security/ldap/proxy.ldif.template. By default, the proxy user is granted only minimum ACL authority on the server for user management to work.

The default ACL allows all users to write to their userPassword attribute. The proxy user is denied write access to user passwords. Additionally, the proxy user is granted read, search, and compare rights to all other attributes and granted write access to a limited set of attributes.

LDIF entries for the proxy user may be altered or added in the proxy.ldif.template file to customize the user definition on the server when the setup is done through the **mksecldap**. It is recommended that the use of defined keywords in the file be used since all instances of keywords will be replaced properly by the **mksecldap** command.

If manual setup of an LDAP server is being performed, the definitions in the template file can still be used to set up the proxy user and ACL. Copy this template to a separate file, replace the keywords (*baseDN*, *proxyDN*, *proxyUser*, and *proxyPWD*) with the desired values and use the **ldapmodify** or **ldapadd** command to add the LDIF entries to the server.

Sample proxy user LDIF file

Example 2-8 shows an example modification of the proxy.ldif.template file that could be used to add a new user to the LDAP directory. In this case, the proxyUser is *proxyu1*, the baseDN is dc=example,dc=com, the proxyDN is *cn=proxyu1,ou=profile,dc=example,dc=com*, and the proxyPWD is *secret*.

The steps to create the following example are described in 6.3.5, “Step 5: Installing a LDAP client with a proxy user ID” on page 198.

Example 2-8   Proxy user definition LDIF file

```
# Create the proxy user
dn: {cn=proxyu1,ou=profile,dc=example,dc=com}
changetype: add
cn: {proxyu1}
sn: {proxyu1}
userpassword: {secret}
objectclass: person
# Modify proxy user password (if previously existed create above will fail).
dn: {cn=proxyu1,ou=profile,dc=example,dc=com}
changetype: modify
replace: userpassword
userpassword: {secret}

# Set the baseDN ACL
dn: {dc=example,dc=com}
changetype: modify
replace: aclentry
aclentry: access-id:cn=this:at.userPassword:grant:w
aclentry: access-id:{cn=proxyu1,ou=profile,dc=example,dc=com}:at.userPassword:grant:r
aclentry: access-id:{cn=proxyu1,ou=profile,dc=example,dc=com}:at.passwordchar:grant:r
aclentry: access-id:{cn=proxyu1,ou=profile,dc=example,dc=com}:at.gecos:grant:rscw
aclentry:access-id:{cn=proxyu1,ou=profile,dc=example,dc=com}:at.hostlastlogin:grant:rscw
...
aclentry: access-id:{cn=proxyu1,ou=profile,dc=example,dc=com}:normal:grant:rsc
```

```
aclentry: access-id:{cn=proxyu1,ou=profile,dc=example,dc=com}:critical:grant:rsc
aclentry: access-id:{cn=proxyu1,ou=profile,dc=example,dc=com}:sensitive:grant:rsc
aclentry: access-id:{cn=proxyu1,ou=profile,dc=example,dc=com}:object:deny:ad

# Propagate the ACL from the baseDN
dn: {dc=example,dc=com}
changetype: modify
replace: aclpropagate
aclpropagate: TRUE
```

Client setup for proxy users

From a client perspective, the binddn and bindpwd are used to point to the proxy user and no other changes are required.

2.4.3 Specifying LDAP server priority

In the AIX Version 4.3.3 and AIX 5L Version 5.1 LDAP clients, the LDAP client allows specification of multiple LDAP servers to allow for server failover. The LDAP client will open a single connection to each server and use the first available server. If the current server were to become unavailable, it would randomly choose another available server.

Starting with AIX 5L Version 5.3, the AIX 5L LDAP client will connect to each of the servers specified in the ldapservers settings, but will initially use the first one in the list that is available. If the current server was to become unavailable, the LDAP client will choose the first available LDAP server in the list starting from the left. This feature was back ported to AIX 5L Version 5.2.

Example 2-9 shows the ldapservers setting in the AIX LDAP client daemon configuration file. There are three LDAP servers listed, with ldap1.example.com being the highest priority.

Example 2-9 LDAP server priority configuration

```
ldapservers:ldap1.example.com,ldap2.example.com,ldap3.example.com
```

2.4.4 Support for netgroups

A netgroup is a list of user names or host names used in the NIS environment under a specific netgroup name to provide or restrict access of resources such as files, applications, or login. Netgroups can be used to define the hosts a group of users are allowed to log in to, specify automount information, and limit the hosts allowed to mount an NFS file system. Netgroups can also be used in /etc/hosts.equiv or in a user’s .rhost file to grant or deny login permissions for a group of users or hosts.

Setting up netgroups in LDAP is similar to setting them up in NIS. Netgroup definitions are in the LDAP database rather than the /etc/netgroup file. The netgroup names are added to the end of the /etc/passwd (or other) file using the +@ prefix. For example, to add netgroup itso_users to /etc/passwd, add the following line:

```
+@itso_users
```

Netgroup support in AIX LDAP authentication also involved changes to the /usr/lib/security/methods.cfg file by adding the options variable *netgroup*. The Example 2-10 sample entry in methods.cfg will enable netgroup support for LDAP.

Example 2-10 Adding netgroup support for LDAP authentication

```
LDAP:
     program = /usr/lib/security/LDAP
     program_64 =/usr/lib/security/LDAP64
     options = netgroup
```

To further enable netgroup support, the user SYSTEM and registry values in /etc/security/user must be changed from LDAP to compat, as shown in Example 2-11.

Example 2-11 /etc/security/user entry for LDAP-based netgroup support

```
test02:
        SYSTEM = "compat"
        registry = files
```

When the LDAP module is NIS enabled, compat will include the following registries: files, NIS, and LDAP.

The user's AUTHSTATE environment variable will now show `compat`.

In the /etc/irs.conf file, specify netgroup map to use LDAP by setting the mechanism to nis_ldap, as shown here:

```
netgroup nis_ldap
```

The nis_ldap term designates that this will use RFC2307-based LDAP mapping of NIS files. The actual netgroup definition is now contained in the LDAP directory under the netgroupbasedn pointed to in /etc/security/ldap/ldap.cfg.

An example of setting and using netgroups is described in 5.4, "Restriction of user login with compat mode (netgroups)" on page 164.

Since a number of fixes and enhancements to netgroup support were added after the base AIX 5L Version 5.3 release, to get full support make sure that you update to at least AIX 5L Version 5.3 ML3.

For details on netgroup support in AIX 5L Version 5.3 see “Migrating from NIS and NIS+ to RFC2307-compliant LDAP services” in the AIX 5L Version 5.3 InfoCenter at:

http://publib.boulder.ibm.com/infocenter/pseries/topic/com.ibm.aix.doc/aixbman/nisplus/migrating.htm

The netgroup support function was also back ported to AIX 5L Version 5.2 ML4 and beyond. See:

http://www.redbooks.ibm.com/abstracts/tips0123.html

2.4.5 Support for automount maps in LDAP

Prior to AIX 5L Version 5.3, an LDAP user could have an automounted home directory, but the automount maps had to be located in either local files or on NIS/NIS+ servers. Starting with AIX 5L Version 5.3, the automount maps can now be stored in the LDAP server. To specify LDAP automount maps add the following entry to /etc/irs.conf:

```
automount nis_ldap
```

The automount maps are then stored on the LDAP server at the automountbaseDN location specified in /etc/security/ldap/ldap.cfg.

An example of setting up a user’s home directory to use an automounted directory from an NFS server can be found in 5.5, “automount configuration” on page 173.

For more details, see the automount command in the AIX 5L Version 5.3 *Commands Reference* at:

http://publib.boulder.ibm.com/infocenter/pseries/topic/com.ibm.aix.doc/cmds/aixcmds1/automount.htm

Also see Managing LDAP Automount Maps in AIX 5L Version 5.3 *Network Information Services (NIS and NIS+) Guide* located at:

http://publib.boulder.ibm.com/infocenter/pseries/topic/com.ibm.aix.doc/aixbman/nisplus/nis_automount.htm

2.4.6 Default entry location

Any attributes that are not specifically assigned to a user are picked up from the default user definition. For LDAP users, the default location for the default user is defined in LDAP, while for local users the default user is taken from /etc/security/user. Starting in AIX 5L Version 5.3, the location of this default user for LDAP users can be specified in the /etc/security/ldap/ldap.cfg file with the defaultentrylocation attribute, which can have values of LDAP or local. If no value is specified, the defaultentrylocation will default to LDAP. To use /etc/security/user for defaults use this entry in the ldap.cfg:

```
defaultentrylocation: local
```

Example 2-12 on page 38 shows an example default user definition in the LDIF format.

Example 2-12 LDIF to define the default user

```
default:
        admin = false
        login = true
        su = true
        daemon = true
        rlogin = true
        sugroups = ALL
        admgroups =
...
```

2.4.7 Schema conversion tool

Note: Appendix D, "Schemas and migration tools" on page 297, provides a reference for the different LDAP schemas, object class and attribute mapping tables, and the AIX data migration tools used in this book.

The **secldifconv** command was added to AIX 5L Version 5.3 to provide a migration path for AIX 4.3.3 and 5.1 users with the AIX schema to move to AIX 5L Version 5.3 with full RFC2307 support. The command can be used to convert from any of the supported AIX authentication schemas to any other. The usage for **secldifconf** is shown here:

```
Usage: secldifconv -R load_modele -S schemaType -i inputFile [-r]
```

The -S flag specifies the output schema type. The -i flag points at the input file, which is in the LDIF format. The -R flag is used to specify the load module from which to retrieve the user passwords. The -r flag specifies that any attributes that are not defined in the specified output schema should be removed.

The following example shows how you can generate the output from the LDAP database and convert the data to the RFC2307AIX schema.

Generate the LDIF file from the LDAP directory. For example, this may be done on the AIX 5L Version 5.1 LDAP server that is acting as the AIX schema repository:

```
# db2ldif -o aixschema.ldif
```

Convert the output to the RFC2307AIX schema:

```
# secldifconv -R LDAP -S RFC2307AIX -i aixchema.ldif > rfc2307aix.ldif
```

Add the converted data to the AIX 5L Version 5.3 ITDS Server:

```
# ldif2db -i rfc2307aix.ldif
```

Note: Care should be taking when adding groups using LDIF files from different systems since the **`ldapadd`** and **`ldif2db`** commands only check for the user or group names and not for the numeric ID when adding entries. Merging users and groups could result in a sharing of a numeric ID by multiple accounts, which is a security violation. IBM Tivoli Directory Server 5.2 and later supports a unique attribute feature that can be used to avoid this issue.

For more details see the secldifconv man pages in the AIX 5L Version 5.3 InfoCenter.

2.4.8 Review additional security enhancements at AIX 5L Version 5.3

A number of other significant security enhancements were also added at AIX 5L Version 5.3.

Long user name and group name support

Starting with AIX 5L Version 5.3, user and group names can be configured to use up to 255 characters. The password length for local files remains at 8 bytes, but for network-based naming services such as LDAP, passwords up to 32 bytes can now be used.

Support for passwd.adjunct NIS map

Starting with AIX 5L Version 5.3, if you have an NIS server that uses the passwd.adjunct map file, you are now able to be authenticated through this map file. This tightens the security of the shadow passwords and authentication maps.

Introduction of PAM-based authentication for AIX users

Starting with AIX 5L Version 5.3, the system administrator has two choices for the base user authentication. The first is the traditional loadable authentication module framework, while the second is a PAM mechanism that is similar to that found on other versions of UNIX. The authentication mechanism is selected for the entire AIX system and is changed by changing the auth_type attribute in /etc/security/login.cfg. The two allowed values are STD_AUTH for the loadable authentication mechanism and PAM_AUTH to use PAM-based authentication.

When PAM is used in this way, authentication and identification will be first handled by PAM, and LAM will only be used when the pam.conf file selects the pam_aix module for the application being used.

In AIX 5L Version 5.3, the /etc/pam.conf file comes already configured in a working manner as shown in this example text extracted from the file (Example 2-13).

Example 2-13 Default pam.conf entries

```
# Authentication
#
ftp     auth    required        /usr/lib/security/pam_aix
imap    auth    required        /usr/lib/security/pam_aix
login   auth    required        /usr/lib/security/pam_aix
rexec   auth    required        /usr/lib/security/pam_aix
rlogin  auth    sufficient      /usr/lib/security/pam_rhosts_auth
rlogin  auth    required        /usr/lib/security/pam_aix
rsh     auth    required        /usr/lib/security/pam_rhosts_auth
snapp   auth    required        /usr/lib/security/pam_aix
su      auth    sufficient      /usr/lib/security/pam_allowroot
su      auth    required        /usr/lib/security/pam_aix
telnet  auth    required        /usr/lib/security/pam_aix
OTHER   session required        /usr/lib/security/pam_prohibit
```

This also shows that most of the AIX applications that support user authentication have also been enabled to recognize auth_type and to be able to use PAM. Because you can stack PAM modules, you are now able to combine traditional AIX authentication with a PAM module.

This is mentioned in this chapter because PAM modules can be stacked with the LDAP authentication methods available in AIX 5L Version 5.3 to allow the administrator to add modules such as mkhomedir the first time a user logs onto a new AIX client.

Login control for r commands

A new user attribute called *rcmds* is introduced in AIX 5L Version 5.3.

The attribute accepts the following values:

allow	Allows the user to execute rcmds.
deny	Denies the user access to rcmds.

hostlogincontrol	Specifies that the user r-command execution is under the control of two new attributes that control which servers the user can use remote commands for. The attributes that control this access are hostsallowedlogin and hostsdeniedlogin.

The definitions of hostallowedlogin and hostdeniedlogin are:

hostallowedlogin	Specifies the hosts that permit the user to log in. This attribute is intended to be used in a network environment where user attributes are shared by multiple hosts.
hostsdeniedlogin	Specifies the hosts that do not permit the user to log in. This attribute is intended to be used in a network environment where user attributes are shared by multiple hosts.

Changing passwords from scripts with chpasswd

The **chpasswd** command introduced in AIX 5L Version 5.3 allows an administrator to change passwords in a non-interactive form. This means that a group of users can easily be added to a system from a script and then the passwords can be set. To allow user migration from another system, the **chpasswd** command can be executed with the -e flag to use encrypted passwords. Another key feature of the **chpasswd** flag is that the -c flag can be used to clear the flags that by default would be set to ADMCHG and require the user to change his password on the first login.

The input to the **chpasswd** is standard input and contains a list of lines that contain user:password. For example, if you have a file called user that contained the following:

```
user1:APd1tSrw/JSxM
user2:1PdYvSNisGUhY
...
```

you can add change the passwords for these users to the AIX 5L Version 5.3 system using this command:

```
cat user.in | chpasswd -e -c
```

For more details see the **chpasswd** command in AIX 5L Version 5.3 InfoCenter.

2.4.9 Search mode attribute added

The searchmode attribute can be set in /etc/security/ldap/ldap.cfg to specify whether the data retrieved from the directory should be all of the attributes included for an entry or if only attributes that are applicable to the operating system (in this case AIX) should be retrieved. The two allowed values are ALL and OS and the default is ALL.

ALL	Returns all attributes of an entry.
OS	Returns only the AIX-required attributes of an entry. Non-OS related attributes such as telephone number, images, and home address will not be returned.

Use OS only when the entry has many non-AIX required attributes or the attributes have a large amount of data. This can help reduce the sorting effort by the LDAP server. An example may be when using Active Directory as the LDAP server (refer to 3.5.10, “AIX LDAP client daemon tuning” on page 95, for more information about this).

2.5 Improvements to LDAP authentication in AIX 5L Version 5.3 RML3

Improvements to function introduced at AIX 5L Version 5.3 maintenance Release 3 are shown in this section. This maintenance release provides the following key new features, which are discussed in more detail:

- 2.5.1, “Use of netgroups to specify automount permissions” on page 42
- 2.5.2, “Bind DN password stored in encrypted format” on page 43
- 2.5.3, “Global unique ID” on page 43
- 2.5.4, “Minimum ID values” on page 43
- 2.5.5, “New lsldap command” on page 43

2.5.1 Use of netgroups to specify automount permissions

One of the new features of AIX 5.3 is that a user’s home directory can be mounted when the user logs into the client. When used together with the compat mode (also known as netgroup support) the control of the permissions on the automount can be controlled by the netgroup the user belongs to. This allows some users to mount their home directory from one server, while other users can mount from another directory. For more information about restricting users with netgroup support see 5.4, “Restriction of user login with compat mode (netgroups)” on page 164. Details about exporting the home directory with netgroup permissions are given in 5.5.4, “Export NFS file systems by LDAP netgroup” on page 176.

2.5.2 Bind DN password stored in encrypted format

The bind DN password is encrypted so that it can only be read by secldapclntd. The password is stored in a format shown in this example:

```
bindpwd:{DES}D2 CDF3D5881686584AA55 C70A92C64D04B8A938964 0A8
```

Both the bind password and SSL password can be encrypted. This happens when the **mksecldap** command is run on releases of AIX equal to or later than AIX 5L Version 5.3 ML3. The bind DN password in prior versions is stored in clear text. Although the password is encrypted, it cannot be moved to another system because it must be retrieved by the **secldapclntd** and converted to the text password before it is used. This means that the seed for the encryption is based on the AIX instance.

2.5.3 Global unique ID

When creating new users and groups, **mkuser** and **mkgroup** commands have been enhanced to check all the registry locations to make sure that a unique ID is created. This prevents the case where you have a local user with UID of 501 and an LDAP-based user with the same ID. Having multiple users with the same user ID is a security violation and should be avoided, and these commands help keep you from accidently creating this situation.

2.5.4 Minimum ID values

The minimum ID value provides a mechanism to specify the minimum ID values for the LDAP users when the **mkuser** command is executed without the -id flag. This helps to keep a separate ID range for LDAP users from the local users, making it easy to identify where the user registry values reside. The minimum ID is set and stored in LDAP and is effective to all the managed client systems. You can override the minimum ID value by specifying an ID value from the command line, as seen in this example:

```
# mkuser -R LDAP id=225 SYSTEM=LDAP registry=LDAP lydia
```

See 3.3.7, “Manual versus mksecldap configuration” on page 73, for instructions on how this minimum ID can be set for both normal and administrative users.

2.5.5 New lsldap command

A new command called **lsldap** has been added to query LDAP entries. The supported queries include users, groups, hosts, automount maps, hosts, protocols, networks, netgroup, RPC, services, and aliases. The **lsldap** command uses the **secldapclntd** daemon, which must be running for the command to work. By default, the **lsldap** command displays on the distinguished name (DN) of the returned object. By adding the -a flag, all of the associated attributes can be viewed.

Example 2-14 lsldap syntax

```
The syntax of the command is shown here:
Usage: lsldap [-a] [ entity [entry_name | filter] ]
        entity: aliases, automount, bootparams, ethers, group, hosts
                netgroup, networks, passwd, protocols, rpc, services
```

If no entity name of flags are supplied from the command line, then **lsldap** displays container entries of the entities as in Example 2-15.

Example 2-15 Default lsldap output with no flags

```
# lsldap
dn: cn=Directory Administrators, dc=example,dc=com
dn: ou=Groups, dc=example,dc=com
dn: ou=Special Users,dc=example,dc=com
dn: ou=rpc,dc=example,dc=com
dn: ou=protocols,dc=example,dc=com
dn: ou=networks,dc=example,dc=com
dn: ou=netgroup,dc=example,dc=com
dn: ou=aliases,dc=example,dc=com
dn: ou=hosts,dc=example,dc=com
dn: ou=services,dc=example,dc=com
dn: ou=ethers,dc=example,dc=com
dn: ou=profile,dc=example,dc=com
dn: nismapname=auto_home,dc=example,dc=com
dn: nismapname=auto_appl,dc=example,dc=com
dn: nismapname=auto_master,dc=example,dc=com
dn: nismapname=auto_apps,dc=example,dc=com
dn: nismapname=auto_next_apps,dc=example,dc=com
dn: nismapname=auto_opt,dc=example,dc=com
dn: ou=group,dc=example,dc=com
dn: ou=people,dc=example,dc=com
#
```

Example 2-16 shows how to use the **lsldap** command with the passwd option to list the distinguished names of all users.

Example 2-16 Using lsldap to show user entries

```
# lsldap passwd
dn: uid=user1,ou=people,dc=example,dc=com
dn: uid=user2,ou=people,dc=example,dc=com
dn: uid=user3,ou=people,dc=example,dc=com
#
```

To retrieve all of the information for the user named user3 run the **lsldap** command with the -a passwd option and the user3 name as shown in Example 2-17.

Example 2-17 Using lsldap by root to show entry for user3

```
# lsldap -a passwd user3
dn: uid=user3,ou=people,dc=example,dc=com
uidNumber: 20003
uid: user3
gidNumber: 20000
gecos: ITSO  user3
homeDirectory: /home/user3
loginShell: /bin/ksh
cn: ITSO user3
shadowLastChange: 12996
shadowInactive: -1
shadowMax: -1
shadowFlag: 0
shadowWarning: -1
shadowMin: 0
objectClass: posixAccount
objectClass: shadowAccount
objectClass: account
objectClass: top
userPassword: {crypt}hTQBG25y7pz6Q
```

All users can run the **lsldap** command, but when a normal user runs this command she will only be able to see public information. For example, using the same command as a normal user you see the information shown in Example 2-18.

Example 2-18 Normal user using lsldap to view user3

```
$ lsldap -a passwd user3
dn: uid=user3,ou=people,dc=example,dc=com
uidNumber: 20003
uid: user3
gidNumber: 20000
gecos: ITSO  user3
homeDirectory: /home/user3
loginShell: /bin/ksh
cn: ITSO user3
objectClass: posixAccount
objectClass: shadowAccount
objectClass: account
objectClass: top
```

More examples of using **lsldap** for the netgroup and other information are shown in 5.4.5, “Step 5: Configure /etc/passwd to control login access” on page 167.

For more details about the command see the lsldap man page in the AIX 5L Version 5.3 InfoCenter commands.

2.5.6 Advanced accounting and LDAP

Although not associated with authentication, the ability to store advanced accounting data in the LDAP directory is indicative of a development strategy in AIX 5L toward making greater use of directory services. The **mkprojldap** command can be used to convert the client and server for handling advanced accounting in LDAP. This allows consolidation of accounting from multiple servers to a common server for use in things like billing. AIX LDAP advanced accounting is functionally dependent on the secldapclntd daemon.

2.6 LDAP support in AIX versions

In AIX Version 4.3.3, IBM introduced their first AIX-based user authentication method using an LDAP server as the repository for user information. This early implementation, which was based on a proprietary schema, was also available in AIX 5L Version 5.1 in about the same time frame. Responding to client requests for a more open environment that would allow AIX authentication to LDAP in a heterogeneous environment, IBM added support for an RFC2307 NIS-based schema in AIX 5L Version 5.2. At this time, IBM also introduced a Kerberos load module to allow Kerberos authentication to Microsoft Windows servers, while allowing the user information to still be contained in an RFC2307 schema-based LDAP server. Since that introduction numerous improvements to this environment have been added to the AIX 5L Version 5.3 operating system. Some of the primary new features include server-side authentication, restricting clients for users based on NIS netgroups, and LDAP-based automount support. Also, a new PAM-based user authentication mechanism was added to AIX that allows users to use off-the-shelf PAM modules for the first time for AIX user authentication.

Table 2-2 on page 47 provides an overview of LDAP integration features available with each release.

Table 2-2 LDAP integration features at different AIX levels

Feature	**AIX Version 4.3.3/5.1**	**AIX 5L Version 5.2**	**AIX 5L Version 5.3**
LDAP client daemon			
Prioritized LDAP server list	No	Yes	Yes
Proxy bind DN	No	No[3]	Yes[1]
Encrypted bind DN password	No	No	Yes[1]
Client-side authentication (UNIX_AUTH)	Yes	Yes	Yes
Server-side authentication (LDAP_AUTH)	No	No	Yes
Default user definition location can be selected (files or LDAP)	No	No	Yes
Kerberos bind	No	No	Yes
SSL connection	Yes	Yes	Yes
Searchmode	No	No	Yes
LDAP schema support			
AIX	Yes	Yes	Yes
RFC2307	No	Yes	Yes
RFC2307AIX	No	Yes	Yes
Object class and attribute name mapping	No	Yes	Yes
AIX security features			
Long user name support	No	No	Yes
Kerberos 5 authentication-only module (KRB5A)	No	Yes	Yes
PAM authentication module (PAM_AUTH)	No	No	Yes
NIS support for Netgroups	No	Yes[2]	Yes
NIS support for passwd.adjunct map	No	No	Yes

Feature	AIX Version 4.3.3/5.1	AIX 5L Version 5.2	AIX 5L Version 5.3
Login restrictions using rcmds attribute	No	Yes (Version 5.2H and later)	Yes
LDAP enabled sub-systems			
LDAP-enabled System V printing subsystem	No	Yes	Yes
Automount maps stored in LDAP	No	No	Yes[1]
Netgroups maps stored in LDAP	No	No	Yes[1]
LDAP-based advanced accounting	No	No	Yes[1]
LDAP-related utilities			
`chpasswd` command	No	Yes (Version 5.2H and later)	Yes
`lsldap` command	No	No	Yes[1]
secldifconv schema conversion	No	No	Yes
mkuser/mkgroup verifies unique UID/GID	No	Yes (Version 5.2I and later)	Yes[1]
mkuser has separate minimum UID value for LDAP	No	No	Yes[1]

[1] This feature was added with AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance Package.
[2] This feature was added with AIX 5L Version 5.2 with the 5200-04 Recommended Maintenance Package.
[3] For proxy bind DN, the difference between Version 5.2 and Version 5.3 is that Version 5.3 supports creation of the proxy account during server setup, but AIX 5.2 does not.

The focus on LDAP has also gone beyond the user-based information storage and recover to support for directory-based attributes for AIX System V based printing, and the ability to store accounting information in LDAP by the new AIX 5L Version 5.3 advanced accounting subsystem.

3

Planning for LDAP migration

Before making major changes to any subsystem, it is important that you understand the consequences of the decisions that you make. This is especially true in the security subsystem, where design errors will not only affect the security of system and user files and weaken overall login security, but can actually prevent users from logging into the system. This chapter discusses some of the considerations that you should understand when migrating to a system that integrates AIX user management with an LDAP server. We describe some of the things that you must do prior to moving an LDAP-based user management system, as well as explain choices that you can make during the integration process and the effects that they will have on your system.

In this chapter the following topics are discussed:

3.1 General planning considerations

This section describes general planning considerations that you must think about before you can migrate to any network-based user management system. These are the same steps you would take whether you are moving to NIS or LDAP-based authentication.

3.1.1 Collecting user and group information from multiple systems

Before user information can be added to the LDAP server, it must first be extracted from its current location and placed in an LDAP input formatted file called an LDIF file. AIX has a number of tools shown below for transferring this data from the security system to the LDIF file.

mksecldap

The **`mksecldap`** command is used to configure the LDAP server and client on AIX. When the server is configured, unless you specify *-u none*, the user information from local security database files will be exported into the LDAP directory on that system. The base DN for storing the user and group information is specified with the -d flag of mksecldap. When looking at this in more detail, the LDIF file is actually created by the **`sectoldif`** command described in the next section.

If this is a new LDAP server, and you specify no users, then the default containers will not get built and you will have to manually add information before you can add users. See 6.3.2, “Step 2: Importing the data into the LDAP directory” on page 194, for more details about adding user information into a new server.

sectoldif

The **`sectoldif`** command can be run by root to extract security information from the local security database files and store it into an LDIF file, which can subsequently be added to the LDAP directory. When you run this command specify the base DN with the -d flag, the schema to use with the -S flag, and the users to migrate with the -u flag. If no user is specified, all users in the security database will be added to the LDIF file.

nistoldif

The **`nistoldif`** command converts the data from passwd, group, hosts, services, protocols, rpc, networks, netgroup, and automount into LDIF output compliant with RFC2307. It will first attempt to read data from NIS, and if it cannot find a NIS map it will fall back to the flat files.

If the server information (the -a, -h, and -p flags) is given on the command line, data will be written directly to the server. If any data conflicts with an entry already on the server, either because the entry already exists or because the UID or GID already exists, a warning will be printed. If the server information is not given, the data will be written to standard output in LDIF format. In either case, nistoldif does not add an entry for the suffix itself. If that entry does not exist, attempts to add data to the server will fail. Once again the -d flag can be used to specify the base DN and the -S flag can be used to specify the schema to be used.

If you are migrating from a NIS environment and plan to use netgroups and automount support, then this is the preferred command for extracting data.

secldifconv

The secldifconv utility can be used to migrate your users from the AIX schema at the earlier AIX versions to use with one of the RFC2307 schemas.

Method for migrating users

Once you have extracted the information from the existing security directory, there are a number of ways to add the information that is now in LDIF format into the LDAP directory. Some of these methods like **`ldapadd`** and **`ldapmodify`** are standard LDAP command-line clients that take a small LDIF file and add it to a running server. Others like **`mksecldap`** and **`nistoldif`** are AIX security clients that can take the information directly from the existing directory to the LDAP directory. When using IBM Tivoli Directory Server, there are other clients to add larger files more effectively. Two of these LDAP clients on AIX are ldif2db and bulkload.

It is important to understand how the client LDAP tool will react to information in the LDIF file that duplicates information in the LDAP directory. For example, **`ldapadd`** will recognize duplicate user names and will not add the entry, but if you have two users with the same UID, both of these users will be added.

For more information and examples of using these data migration utilities please refer to "Information about AIX Data Migration Tools" on page 316.

3.1.2 Consolidating user and group ID

When consolidating users from multiple systems, it is possible that the users have different names and user IDs on the different system and may belong to different groups. When you move the user identification to a single server, you must make sure that each use has a single user name and UID on all of the systems before the migration. This also means that you have to also change the file permissions on the files that these users own.

Duplicate user names and user IDs

When consolidating users from multiple systems, it is also possible that different users on two or more systems could have the same user name, user ID, or both. If this is not fixed before the consolidation, a number of problems may occur depending on the way the users are added and the LDAP server. Some of the problems that may occur include:

- The second user to add may not get added.
- Two users may be added with different names and the same UID.
- Two users may be added with the same user name and different user IDs.

This problem may not only exist with normal users, but can also exist with administrative and special purpose users. AIX normally has reserved UIDs that are below 200 for users such as root, daemon, bin, sys, adm, uucp, guest, nobody, ldp, and sometimes lp, invscout, and nuucp. Often when applications such as LDAP and OpenSSH are installed, they will have taken the next available UID above 200. This means that you may have to change UIDs for these users and change the permissions of corresponding files. For example, on one of my systems user sshd has the UID of 202, while on the second system this is at UID of 205. In this case there is not a problem because no files are programs owned by sshd, but instead by users root and bin. For this reason, you can simply change the UID for sshd on one of the systems with no problems, except that on both systems there are other users that are defined with those corresponding UIDs, and each of them owns a significant number of files, but they were all in a single directory structure. Here are the steps to take:

1. Collect the user information on all systems with the **lsuser** command, as shown here:

   ```
   # lsuser -a id ALL > /tmp/system1.userids
   ```

2. Change the user ID to 502 with:

   ```
   # chuser id=502 moveit
   ```

3. Looked at the permissions on the /home/moveit directory, which showed the files were now owned by user 205:

   ```
   # ls -lR /home/moveit
   ```

4. Use the **find** command to change the owner of all files and directories on the machine to be owned by moveit only if it was owned by user ID 205:

   ```
   # find / -user 205 -exec chown moveit {} \;
   ```

5. After finding no files owned by sshd, you can now just change the UID of the sshd user to 205:

   ```
   # chuser id=205 sshd
   ```

6. Finally, you check to make sure you can log on with sshd.

Two things that you can do during the planning and data collection phase are to write scripts to check for duplicate information and to plan to use a different range of UIDs for LDAP users versus local file-based users. The client configuration section below describes how you can create a user range for new users in AIX. See 3.3.7, “Manual versus mksecldap configuration” on page 73.

Duplicate group names and GIDs

Duplicate group names or group IDs can also create similar problems. AIX has certain group names that are standard such as system, staff, bin, sys, adm, uucp, mail, security, cron, and printq that use group IDs 0 through 9. Although you may not see any immediate problems by changing the names of these groups, you may see problems when you try to apply updates later.

For example, at one company, the group name security was used on a Sun system as a special group that had a GID other than 7, which is the standard GID for security on AIX. Since the security group was widely used on the systems already attached to the LDAP server, the administrators decided to test making a change to the group name of GID 7 by changing it to AIXsecur. This seemed to work well, until a Program Temporary Fix (PTF) was added to the AIX 5.3 system. At this point, the AIX 5.3 PTF installation failed.

One potential solution to this problem is to have local administrative users that use the local group security defined in /etc/security/group, and have LDAP users not be a part of this group. By default, this means that you would define the root user as a local user.

3.1.3 Local and LDAP user and group coexistence

With a UNIX system with user IDs on a network service the question that arises is how they are going to work together. There are a couple of issues that should be considered when you are planning the LDAP service.

Mixing local and LDAP-based users

It is possible to mix local and LDAP-based users on the same AIX client. This is done by setting the SYSTEM and registry attributes for different users to either files or compat for local users and LDAP for LDAP-based users. Some of the reasons for doing this might be to maintain local administrative user access if the LDAP server goes down or there are network problems. It is also possible in AIX to set up the SYSTEM attribute to fall back to files if the LDAP server is not available. When you do this, you must manually maintain the local password because there is no automatic synchronization mechanism.

Note: On AIX 5L, local file-based users must belong to groups defined in /etc/group, while LDAP users must belong to groups defined on the LDAP server. LDAP users with local groups or local users with LDAP groups cannot be used.

Which user IDs should be on the LDAP server

Generally speaking we can distinguish three different types of users:

System user IDs	These are user IDs the system needs to run. For example, the init process is running as root.
Application user IDs	These are user IDs a specific application needs to run. For example, an apache Web server may run as **`httpd`**.
User user IDs	These are user IDs that a person uses to access the system. For example, John Smith may have a user ID jsmith.

The group IDs can be grouped in the same way as the user IDs. Now the question arises as to which type of user IDs should be on an LDAP server and which should be local or on both. We discuss this below for each group. One of the common solutions is to make certain arrangements, for example, everything below 500 belongs to the system, 500–2000 is used for applications, and above 2000 the normal user accounts start.

User user IDs

This type of user ID is normally the reason for using a centralized LDAP server, so you want to have them on the LDAP server. But there are some special users (the system administrators, for example). You may consider that they need to log in even when the LDAP server is down or the network connection is lost for whatever reason, so you may want to have them on local or on both local and the LDAP server. Or you may consider another solution like allowing direct root login from a serial console that is remote accessible.

Application user IDs

The application user IDs depend on your environment and of course on your applications, but there are general guidelines:

- If you have business critical applications you might not want to rely on your LDAP server and your network to have them operational.
- On the other hand, if you have a load balanced cluster of Web servers that cannot do anything when your network may experience problems, it is OK to have these IDs on an LDAP server.

- When you have a couple of systems that interact with each other based on the user ID of systems (NFS can be example here), it is a good idea to have a single consistent point of administration for your user IDs, the LDAP server.

System user IDs

The system user IDs have to be local at least. Otherwise you may encounter problems during system boot. But you could have a copy of them at your LDAP server. The copy of the user IDs will give you the following advantages and disadvantages:

- You can give rights to special groups on your LDAP server. For example, you can add somebody to the security group at the LDAP server and therefore give him rights that belong to this group.
- There may be confusion between the users and groups if they are different on the LDAP server and on the LDAP client. This may not be a problem during normal operation, but for troubleshooting it makes things more difficult.
- In a heterogeneous environment you encounter the problem that different UNIX operating systems are using different user ID and group ID numbers for different names. For example, on an AIX system UID 2 is bin and UID 1 is daemon. On a SUSE Linux Enterprise Server 9 (SLES 9) Linux system it is the reverse: UID 1 is bin and UID 2 is daemon. For more details see Example 3-1.

Example 3-1 First 10 lines of /etc/passwd of different UNIX systems

```
AIX 5L Version 5.3:
root:!:0:0::/home/root:/usr/bin/ksh
daemon:!:1:1::/etc:
bin:!:2:2::/bin:
sys:!:3:3::/usr/sys:
adm:!:4:4::/var/adm:
uucp:!:5:5::/usr/lib/uucp:
guest:!:100:100::/home/guest:
nobody:!:4294967294:4294967294::/:
lpd:!:9:4294967294::/:
lp:*:11:11::/var/spool/lp:/bin/false

SUSE Linux Enterprise Server 9:
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
daemon:x:2:2:Daemon:/sbin:/bin/bash
lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
mail:x:8:12:Mailer daemon:/var/spool/clientmqueue:/bin/false
news:x:9:13:News system:/etc/news:/bin/bash
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
games:x:12:100:Games account:/var/games:/bin/bash
man:x:13:62:Manual pages viewer:/var/cache/man:/bin/bash
```

```
at:x:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash

SUN Solaris 8:
root:x:0:1:Super-User:/root:/usr/bin/ksh
daemon:x:1:1::/:
bin:x:2:2::/usr/bin:
sys:x:3:3::/:
adm:x:4:4:Admin:/var/adm:
lp:x:71:8:Line Printer Admin:/usr/spool/lp:
uucp:x:5:5:uucp Admin:/usr/lib/uucp:
nuucp:x:9:9:uucp Admin:/var/spool/uucppublic:/usr/lib/uucp/uucico
listen:x:37:4:Network Admin:/usr/net/nls:
nobody:x:60001:60001:Nobody:/:
```

3.1.4 Combining users from heterogeneous environments

Considering that every type of UNIX-like system has small differences to each other, it is also clear that if you want to integrate them on a single LDAP server you run into problems. Some of these problems are discussed here.

Different shells

The default user shell on AIX is the KornShell shell or ksh, but on other systems it may be the bash shell or a csh. The RFC2307-based LDAP directory stores the user's shell in the loginshell attribute. Through mapping files, the value is moved to the AIX user attribute named shell*,* which is subsequently set to the user's SHELL environment variable. To get to a solution there are the following possibilities:

- All users can use the same shell on all systems. That may be possible if the systems are very similar, for example, the /bin/sh.
- The LDAP server can contain a default shell that is on the system or that is linked on the system to the default shell of the system.
- Also, the systems may have installed the shell that is defined on the LDAP server as a additional shell when is not the default shell of the system.

The user can still decide to change the shell with the different startup scripts, or just when she logs in to the system directly.

Home directory location

When users are managed on each local UNIX server, the user normally has a home directory on that server that is maintained on local disks. When the environment changes to one where users are managed by a directory server, then decisions need to be made about how the user's home directory will be managed. Some of the options include:

- Maintain the user directory on each server's local disks.

  This option makes sense when a user has a totally different function on each server that he logs into and wants to see a different environment. The disadvantages are that it can be hard to remember where you place files, which means that you have to make changes to a lot of different configuration files when you want to make global changes, and it makes extra work for administrators when creating and removing users.

- Mount a /home directory from an NFS server for all users.

  Advantages of this are ease of maintenance because you can simply create a large directory on a file server and do a single mount when you start up the client. The disadvantage is the security concern with a global export of the file system.

- Automount user's home directories as required.

  Advantages of this are that the user's home directory is only mounted when that user is logged into the system. This means that you will have a mount for each user logged on. You can control the automount through NIS maps in AIX 5L Version 5.3. This scenario is explored in Example 4.3.5 on page 127.

Password encryption methods

The only supported method for AIX password encryption for passwords stored in /etc/security/passwd is the crypt() subroutine. If the LDAP server stores the passwords in some other format, then you will not be able to log in using those passwords in AIX versions prior to AIX 5L Version 5.3. Starting with AIX 5L Version 5.3, you can specify ldap_auth as your method of doing LDAP authentication. This provides server-based authentication and can use any encryption method provided by the server. This is described in more detail in 2.4.1, "Server-side authentication support" on page 31.

Non-standard LDAP attribute name mapping

When the LDAP server does not support the RFC2307 schema, it is still possible to map attributes stored on the server to the attributes required by AIX by creating custom mapping files. One example of this is the mapping of attributes that are available in the Microsoft Windows Services for UNIX to the required AIX attributes, as described in "Microsoft Windows Services for UNIX mappings" on page 314.

3.1.5 Supporting long user names and passwords

When using file-based user management in AIX prior to AIX 5L Version 5.3, both the user name and the password are restricted to a length of eight characters. This section looks at why you might want longer values and how this can be accomplished at AIX 5L Version 5.3.

Long user names

In the Internet age, most users have a corporate Internet mail account with an associated user name. Many companies use an e-mail address as the user identification for logging into many services in the company. For this reason, it may be desirable to be able to use this e-mail address as your AIX user login. This is not possible prior to AIX 5.3, but starting with AIX 5.3, you can assign user names that are up to 254 characters.

By default, the length of user names and passwords in AIX is eight characters. To change the user name length you can use SMIT or the command line. From SMIT choose **System Environments** → **Change / Show Characteristics of Operating System**. Once you are at this point, scroll down until you find `Maximum login name length at boot time`, and set this to the number of characters to which you want to limit the user name length. To change this value from the command line, use the **`chdev`** command, as shown in this example that changes the name length to 124 characters:

```
chdev -l sys0 -a max_logname='125'
```

Once you have made these changes, you will have to reboot the operating system before they will take effect.

Note: In SMIT, it will show the default value as nine characters. This includes room for the null character, and the actual user name can only be eight characters. If you set the max_logname to 125 characters, the name you can use will only be 124 characters.

Long passwords

The password length of eight characters is actually restricted by the crypt() subroutine used to store the password in /etc/security/passwd, or the crypt() function used to store the function on the LDAP server. Starting with AIX 5L Version 5.3, you can use LDAP server-side authentication as described in 3.5.3, "Client-side and server-side authentication" on page 86. If you are using server-side authentication, the password length depends on the authentication method used on the server, and from an AIX client perspective can be up to 32 bytes long.

3.1.6 What levels of AIX you need to support

The AIX levels of the servers that you want to integrate will have a major effect on the solution that you can implement. For example, with AIX 4.3.3 and AIX 5.1, your LDAP integration is limited to an AIX-only solution. Starting with AIX 5.2 you can participate in a heterogeneous LDAP solution and can use Kerberos to provide an authentication-only solution to integrate with Microsoft Active Directory. A more detailed discussion and reference table on this topic can be found in 2.6, “LDAP support in AIX versions” on page 46. Also see 3.2, “Which schema you should use” on page 60, for more details about the appropriate schema based on your operating system base.

3.1.7 Locations of the default user stanza

The default user is defined by a stanza in /etc/security/user or on the LDAP server under the DN: uid=default, ou=... This is not a normal user that you can log in as, but instead defines the attributes that are used if that attribute is not specifically defined for a particular user. For example, if you want all AIX users to log in using LDAP unless you specify differently for that user, you could set the default user attribute SYSTEM = LDAP. Likewise, if you wanted to set the number of login retries that fail before the account is locked, you could specify this in the default user stanza with loginretries=5.

By default, when a user is defined as an LDAP user, the default attributes will come from the default user defined in the LDAP directory, while local users will get their default attributes from the default user defined in /etc/security/user. Starting with AIX 5L Version 5.3, you can choose to use the local default user values with LDAP users. This may be valuable in heterogeneous environments. To change this location of the default user for LDAP users to the local default, make this change in the /etc/security/ldap/ldap.cfg file:

```
defaultentrylocation:local
```

An example of defaults that you might want to set for different systems are password restrictions, rlogin permissions, sugroups, and so on.

3.1.8 Using SSL to secure LDAP connection

When LDAP clients connect to an LDAP servers using the LDAP protocol, you will face the following security exposures:

- All traffic between the client and the server can be monitored by malicious users for sensitive information. If an LDAP client binds non-anonymously to an LDAP server, the password is sent in the clear.
- The LDAP server and client cannot detect if data has been modified by malicious users in flight.

- The LDAP client has no way to confirm the identity of the LDAP server it is connecting to. A malicious user could set up a rogue LDAP server to collect authentication information and other sensitive information.
- The LDAP server has no way to confirm the identity of the client initiating the connection to the server.

These security exposures can be mitigated by using TLS/SSL to secure the communication between your LDAP client and servers. TLS/SSL provides methods for establishing identity using X.509 certificates and ensuring message privacy and integrity using encryption. In order to create a SSL connection the LDAP server must have a digital certificate signed by a trusted certificate authority (CA). Companies have the choice of creating their own certificate authority or using a trusted third-party CA. In this scenario, the example.com company uses its own CA. For more details on the setup of the example.com CA used in the scenarios in this book see Appendix C, “Example.com certificate authority setup” on page 289.

The procedure to create signed digital certificates for the client and server is very similar. You must first create a certificate request (CER) and send it to the certificate authority to have it signed. The CA will then sign the CER and return the signed certificate to the requestor. The certificates can then be imported into the local keystore on the client and server. Since the certificates are signed by example.com’s own CA and not trusted third-party CAs, the certificates will not be trusted by the local computer. You must import the CA certificate for example.com into the client and server’s keystore as a trusted root certificate authority.

Tip: During initial LDAP client setup or while troubleshooting, it is often easier not to use SSL. The use of network sniffing tools such as tcpdump and ethereal allows you to debug the communications between the LDAP client and server. When using SSL (that is, secure sockets layer), by definition eavesdropping on client and server communications is not possible.

When preparing or returning to production mode, securing your client to server communications with SSL is highly recommended.

For more details about configuring the AIX 5L LDAP client to use SSL, refer to 4.3.1, “Configuring SSL” on page 113.

3.2 Which schema you should use

This section discusses the different schemas that exist for UNIX LDAP integration. The AIX, RFC2307, and RFC2307AIX schemas are the only schemas that IBM supports officially. Even though IBM does not support the RFC2307bis schema, IBM automounter client does support the new schema for automount maps defined in RFC2307bis. IBM does not currently support the Microsoft Windows Services for UNIX schemas, but this book covers Active Directory integration with AIX in detail in Chapter 8, “Scenario: Microsoft Windows 2003 Active Directory integration” on page 217.

The following is a list of the schemas that this section discusses:

- AIX
- RFC2307
- RFC2307AIX
- Microsoft Windows Services for UNIX
- Microsoft Windows 2003 Server R2

In your environment, it is possible that you will come to point when you realize that one schema will not be able to support all of your machines. The end of this section discusses methods of supporting multiple schemas in different trees and using IBM Tivoli Directory Integrator to keep them synchronized.

3.2.1 AIX schema

The AIX schema is a unique schema to the AIX environment. It is supported at all levels of AIX starting at AIX 4.3.3. It was developed based on the standard features of the AIX file-based security database. It is the only schema supported on AIX 4.3.3 and AIX 5L Version 5.1, so if your plans include these two versions of AIX, then you must choose this schema.

One limitation of the AIX schema is that you cannot combine the AIX users in the same directory structure as other UNIX users. One of the decisions that you have to make in planning is whether it is worth migrating these users to the AIX schema LDAP directory, and then later when you upgrade to at least AIX 5L Version 5.2, converting them to one of the other schemas that can be used in with other UNIX flavors.

If you later need to convert users from the AIX schema to one of the other schemas, this can be done easily with the secldifconv tool. At that point, you will have to extract the data from the directory into LDIF files and then add it to the server in the new schema format. The **secldifconv** command can create LDIF files that are in either RFC2307 or RFC2307AIX format.

More details about the AIX schema are given in “AIX schema” on page 298.

3.2.2 RFC2307 schema

RFC2307 was proposed as an LDAP naming service replacement for NIS. It is defined in the RFC written by L. Howard as an experimental protocol and is titled “An Approach for Using LDAP as a Network Information Service.” The text of the RFC can be found at:

`http://www.ietf.org/rfc/rfc2307.txt`

Support for RFC was first introduced in AIX 5L Version 5.2. AIX supports both the host naming and the user and group naming functions of the RFC, but in this book we only address those features related to AIX user management and security.

The RFC2307 schema on AIX also supports two attributes not defined in the RFC. These are hostsdeniedlogin and hostsallowedlogin, which can be used to restrict access of users to certain hosts. These are defined under ibmattributetypes in /etc/security/ldap/nisSchema.ldif.

RFC2307-defined password restrictions

The RFC2307 schema defines a number of attributes to determine how often the password must be changed. The RFC2307 attributes associated with passwords include userpassword, shadlowlastchange, shadowmax, shadowmin, shadowexpire, shadowwarning, shadowflag, and shadowinactive. The shadowflag attribute is a reserved field and shadowinactive is not supported by AIX. The rest of the attributes have the following meanings on AIX:

userpassword	The attribute contains the encrypted password.
shadlowlastchange	The time that the password was last changed given in seconds past epoch, which maps to the AIX lastupdate parameter normally in /etc/security/passwd. If password aging is in effect, the shadowlastupdate attribute forces a password change when the time limit expires.
shadowmax	This defines the age in weeks after the lastchange after which the user is forced to change her password. This corresponds to the AIX maxage parameter.
shadowmin	This defines the minimum age in weeks a password must be before it can be changed, which corresponds to the AIX minage parameter.
shadowexpire	This defines the maximum time in weeks beyond the shadowmax value that a user can change an expired password. After this time, only an administrative user can change the password. This attribute corresponds to the AIX maxexpired parameter.

shadowwarning	This defines the number of days before the password is set to expire (defined with shadowmax) that the user will be warned to change his password. This attribute corresponds to the AIX pwdwarntime attribute.
shadowflag	This is a reserved field and is different from the AIX flags attribute.
shadowinactive	This attribute is not used by the AIX client.

If you want to take advantage of more sophisticated password restrictions, such as minimum number of alphanumeric characters or maximum number of repeated characters, you should use the RFC2307AIX schema. Non-AIX machines will not recognize these extended attributes.

More details about the RFC2307 schema are given in "RFC2307 schema" on page 302.

3.2.3 RFC2307AIX schema

The RFC2307AIX schema includes all the attributes from the RFC2307 schema as well as a number defined to add extra functions in AIX. The resulting schema is a combination of the AIX schema capabilities with RFC2307 capability. Because the extra attributes are not used by other UNIX servers, the compatibility still remains, while adding the advantages that are there with AIX security. The RFC2307AIX schema includes the attributes added in the aixAuxAccount and aixAuxGroup object classes and their attributes.

More details about the RFC2307AIX schemas are given in "RFC2307AIX schema" on page 305.

RFC2307AIX-defined password restrictions

The RFC2307AIX schema includes all the password restrictions of the RFC2307AIX schema as discussed in "RFC2307-defined password restrictions" on page 62. This list shows the additional password restriction capability that is added by extending the schema to the RFC2307AIX schema.

- isaccountenabled

  This attribute is a boolean. It corresponds to the AIX security attribute account_locked.

- passworddictfiles

  This attribute contains a comma-separated list of dictionary files used to restrict passwords. This corresponds to the dictionlist attribute from /etc/security/user on AIX.

- timeexpirelockout

  This identifies the expiration date of the account. The value is a 10-character string in the MMDDhhmmyy format. This corresponds to the expires AIX attribute.

- passwordflags

  This specifies the password restrictions for login, passwd, and su. This value can be empty or contain ADMIN, ADMCHG, or NOCHECK. When a new user is created with mkuser, this value is normally set to ADMCHG, indicating that the user must change the password when he first logs in. This corresponds to the flags attribute in the /etc/security/passwd file on AIX.

- passwordhistexpire

  This designates the time period in weeks after which a user cannot reuse the same password. This corresponds to the histexpire in /etc/security/user on AIX.

- passwordhistsize

  This designates the number of previous passwords that the user cannot reuse. This maps to AIX /etc/security/user attribute histsize.

- passwordhistlist

  This contains a list of previously used passwords. It corresponds to the AIX attribute histlist.

- passwordmaxrepeatedchars

  This defines the maximum number of times a character can be repeated in a new password. This maps to AIX /etc/security/user attribute maxrepeats.

- passwordminimalphachars

  This defines the minimum number of alphabetic characters that must be in a new password. This maps to the AIX /etc/security/user attribute minalpha.

- passwordmindiffchars

  This defines the minimum number of characters requied in a new password that were not in the old password. This maps to the AIX /etc/security/user attribute mindiff.

- passwordminlength

  This defines the minimum length of a password. This maps to the AIX /etc/security/user attribute minlen.

- passwordminotherchars

  This defines the minimum number of non-alphabetic characters that must be in a new password. This maps to the AIX /etc/security/user attribute minother.

- passwordcheckmethods

  This gives a list of loadable modules that must be executd to validate the password. This lets you implement things like the minimum number of capital letters. This maps to the AIX /etc/security/user attribute pwdchecks.

The extended schema also gives additional security information such as recording the last host logged in from and other parameters that AIX administrators are used to.

3.2.4 RFC2307bis schema

The RFC2307bis schema, which was a follow-on the original RFC2307 schema, was an Internet Engineering Task Force (IETF) working draft that expired without becoming an official standard. Even though RFC2307bis is expired, vendors have already implemented some of the newer features. For example, the automount daemon on AIX 5L Version 5.3 and later supports the new automount and automountMap object classes. At the time this book was written, a copy of this schema could be obtained from the PADL Web site at:

http://www.padl.com/~lukeh/rfc2307bis.txt

More details including the object classes for RFC2307bis are described in “RFC2307bis schema” on page 308.

3.2.5 Microsoft Windows Services for UNIX (SFU) schemas

Microsoft released the Microsoft Windows Services for UNIX in order to improve Windows interoperability with UNIX. SFU included a NIS and NFS server and a customized LDAP schema to hold the POSIX attributes needed for UNIX users and groups. The SFU schemas are not RFC2307 complaint, as they use custom object class and attribute names.

There are two different versions of the SFU schemas one for SFU 2.0 and the other SFU 3.0/3.5. They can be distinguished by the object class and attribute names. SFU 3.0/3.5 names are normally prefixed by msSFU30, while the SFU 2.0 names use msSFU or no prefix at all. The SFU 2.0 schema adds msSFUPosixAccount and msSFUShadowAccount as auxiliary classes to the user class, and the msSFUPosixGroup object class is added as an auxiliary class to the group class.The SFU 3.0/3.5 schema adds msSFU30PosixAccount and msSFU30ShadowAccount as auxiliary classes to the user class, and the msSFU30PosixGroup is added as an auxiliary class to this class.

Important: As of the writing of this book, `mksecldap` does not support configuring AIX against Microsoft Active Directory (MSAD), and AIX 5L does not provide MSAD attribute mappings. However, through manual configuration steps and customized attribute mappings, AIX 5L can be made to work with MSAD. This integration scenario is described in detail in Chapter 8, “Scenario: Microsoft Windows 2003 Active Directory integration” on page 217.

For more details and customized map files for SFU 2.0 and 3.0/3.5 refer to “Microsoft Windows Services for UNIX schemas” on page 308.

3.2.6 Microsoft Windows 2003 R2 schema

Microsoft announced that the upcoming Windows 2003 Server R2 will now support a RFC2307-compliant schema. Since Windows 2003 Server R2 is not released, this book does not cover this topic.

3.2.7 Schema selection

To decide which schema to use, you can break the decision into two parts. If you have an all-AIX environment then your choices are the AIX schema or the RFC2307 AIX schema. If you want all users to be in a single directory, then choose the AIX schema. But if your plans include a future where you might want to share authentication with other operating systems, then moving the AIX 5L Version 5.2 and 5.3 users to RFC2307AIX is your best choice.

In the mixed environment, realize that there is no supported way to integrate your AIX 4.3.3 and AIX 5L 5.1 users, so focus on integrating your AIX 5L Version 5.2 and 5.3 users with the other UNIX variants and then integrate your older operating systems into the LDAP environment as you upgrade.

Do you need to support AIX 4.3.3 or 5.1 systems?

- Yes: You can only use the AIX schema for these systems.

  Do you also have non-AIX systems?

  – Yes: Separate the AIX 4.3.3 and 5.1 systems on one LDAP directory. Place AIX 5.2 and later systems with other UNIX on an RFC2307-based LDAP.

  – No: Place all AIX systems in one directory with AIX schema.

- No: Use one of the RFC2307-based schemas.

  Are you using IBM Tivoli Directory Server?

  – Yes: Use RFC2307AIX for AIX systems and RFC2307 for others.

– No: Add the AIX extension schemas if possible and user RFC2307AIX. If you cannot add AIX extension schemas use RFC2307.

Summary

In summary, if you have AIX 4.3.3 or AIX 5L Version 5.1 systems to support, you can only do this with the AIX schema. Moving to an RFC2307 schema allows you to place users in a common directory for heterogeneous servers, but you lose some of the password control that AIX administrators are used to. By using an RFC2307AIX schema you can regain control of some of this security information and password restrictions. The AIX extensions are contained in IBM Tivoli Directory Server, but can also be added to other LDAP servers.

3.2.8 Object class and attribute name mapping

Starting with AIX 5L Version 5.2, AIX now supports the mapping of object class and attribute names for both user and group entities. The mapping files allows AIX 5L to work with schemas with different object class and attribute names. AIX currently supports mapping files for the user, group, and the AIX ID attribute entities.

Table 3-1 lists the supported AIX schemas and the file name of the mapping files used for the user and group entities. The mapping files are included in the bos.rte.security LPP and are installed in the /etc/security/ldap directory.

Table 3-1 User and group entities mapping files shipped with AIX 5L

AIX 5L schema name	User entity mapping file	Group entity mapping file
AIX	aixuser.map	aixgroup.map
RFC2307	2307user.map	2307group.map
RFC2307AIX	2307aixuser.map	2307aixgroup.map

Although IBM only officially supports the AIX, RFC2307, and RFC2307AIX schemas, with the user and group map files you should be able to support any custom schema. In Chapter 8, “Scenario: Microsoft Windows 2003 Active Directory integration” on page 217, you will find information about the custom map files used to integrate the AIX 5L LDAP client with Microsoft Active Directory. For more information about object class and attribute mapping for different schemas please refer to “Object class and attribute name mappings” on page 310.

3.2.9 The possibilities if none of the schemas fit

In today's enterprise environments a single solution or schema might not fit all needs that must be satisfied. To get around a couple of common problem there are tools and utilities on the market today. For example, there are meta-directories, virtual directories, and data synchronization tools. A discussion of these tools and their features is not in the scope of this book, but the following example illustrates them.

In a pure AIX environment the AIX schema could be used because every AIX version since 4.3.3 supports it. But none of the other UNIX systems may be able to support it. On the other hand, with the RFC2307AIX schema the AIX 5L Version 5.2 and AIX 5L Version 5.3 can operate with other UNIX systems but not with older versions of AIX.

One solution is to create two separated directory trees and to synchronize them with IBM Tivoli Directory Integrator (ITDI). Any change will be synchronized from one tree to the other. The synchronization can be done immediately with plug-ins to the LDAP server or on a scheduled basis.

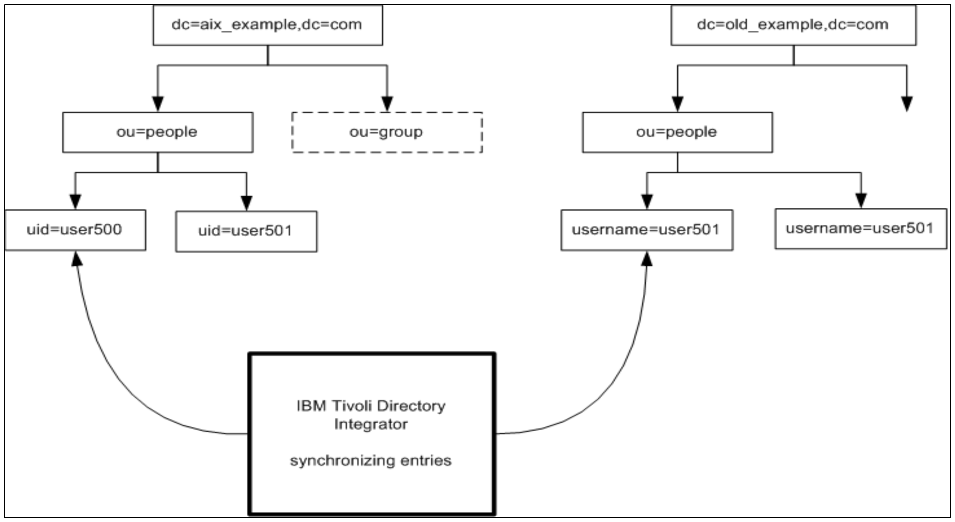


Figure 3-1 Synchronizing entries

3.3 Server-related considerations

This section describes decisions that you need to make related to the LDAP server that you are using. Topics in this section include:

- 3.3.1, "Which LDAP server you should use" on page 69

- 3.3.2, “What base distinguished name (DN) you should use” on page 70
- 3.4, “Microsoft Active Directory considerations” on page 73
- 3.3.5, “Server availability and failover considerations” on page 72
- 3.3.6, “Scalability and performance” on page 73
- 3.3.7, “Manual versus mksecldap configuration” on page 73

3.3.1 Which LDAP server you should use

This section helps you make the decision about which LDAP server to use. The supported LDAP servers for AIX 5L Version 5.3 include:

- IBM Tivoli Directory Server Version 5.2 and 6.0
- Sun ONE Directory Server
- OpenLDAP
- Any other LDAP server that fully supports RFC2307

Attention: The Microsoft Active Directory server is not mentioned in the list above because it is not RFC2307 compliant. Therefore it is not an officially supported solution for AIX client LDAP integration. Integration with Active Directory is still possible under the right circumstances, and this book covers that in detail in Chapter 8, “Scenario: Microsoft Windows 2003 Active Directory integration” on page 217.

The Novel eDirectory server supports an RFC2307-like schema, but the password is stored with MD5 encryption. This means that if you want to use eDirectory as your user database, you will have to do server-side authentication as described in 3.5.3, “Client-side and server-side authentication” on page 86.

This section is designed to help you ask the questions that will help you decide which LDAP server to use.

What levels of AIX you need to support

If you use AIX 4.3.3 or AIX 5L Version 5.1 with the AIX schema, then it is easiest to use IBM Tivoli Directory server. The recommendation we give is to use the latest version of ITDS that comes with the highest level of AIX that you are going to be supporting. This will be the version that has the most testing. In this book, we describe using a later version of ITDS 6.0 and show how that is used with AIX 5.3, but this would work just as well with ITDS 5.2 that comes with AIX 5L Version 5.2 and 5.3 and it is likely that more testing was done by development with the ITDS 5.2 than with ITDS 5. The advantage of starting out with ITDS 6.0 is that you will not have to make that upgrade later.

What other UNIX systems use the same LDAP server

If you plan to use the same directory server for AIX 5L Version 5.2 and 5.3 and other UNIX variants such as Solaris, HP/UX, and Linux, then the choice is less clear. By using IBM ITDS Server you will be assured that the AIX extensions are already installed to the RFC2307 schema, but if the environment contains a larger number of one of the other variants then it is likely that you will be better off using the directory server that is most compatible with that UNIX variant.

What your current environment is

If you are adding AIX systems to an already existing LDAP server, then there is no reason to install the ITDS Server to support your AIX systems. If your current environment is NIS, then you should plan to use the nistoldif tool to help you convert the files to LDIF format, as this can be added to any LDAP server that supports the RFC2307 schema.

What LDAP skills you have

The current skills of the customer are always an important consideration when choosing any platform. If you have skills running a particular directory server, then it may be a wise chose to standardize on this server because your skilled people will best be able to customize and tune the server as well as to provide the best high-availability solution.

For an AIX shop that has very little LDAP skill, the mksecldap configuration script makes your setup installation quite easy when using IBM Tivoli Directory Server solution that comes on the AIX Expansion Pack. You will need to develop more skills to add multiple servers or to use a master server with replica servers.

3.3.2 What base distinguished name (DN) you should use

Before you can place users into the directory, there must be a suffix that acts as a container for the user and group information. If you add users with the **`mksecldap -s`** command, these will get added to the directory automatically. If you do not specify a base distinguished name with the -d flag, then the data will go into an organizational unit called people, as shown in Example 3-2.

Example 3-2 Default DN for adding users in LDIF format

```
dn: ou=People,cn=aixdata
ou: People
objectClass: organizationalUnit
```

If you are adding to an already existing directory, or if you want to place this into a location in the directory that more closely indicates the organization of the servers that you will support, you can specify this with the -d flag. To see the output before adding it, you can use the **sectoldif** command with the -d flag and the -S flag to specify the schema.

Example 3-3 on page 71 shows the output of the **sectoldif** command when you specify the base DN of dc=example,dc=com with the -d option. The user account information would be created in the ou=People,dc=example,dc=com container.

Example 3-3 Specifying a base DN with the sectoldif command

```
# sectoldif -d dc=example,dc=com -S RFC2307 > /tmp/RFC23071.ldif
#
# cat > /tmp/RFC23071.ldif
dn: ou=People,dc=example,dc=com
ou: People
objectClass: organizationalUnit
<excess output removed>
#
```

When planning your server configuration, it is important to use a suffix or distinguished name container that is appropriate and that you will not want to change. In this case both the group and user information would be stored under the organizational unit domain described by ou=xyzcomp,ou=myorg. Because of the hierarchical nature of LDAP, it is actually possible to support different client domains on the same LDAP server located in different base DNs.

When choosing the bind distinguished name of the administrator you need to choose between a direct bind, binding with SSL, a Kerberos bind, and using Proxy users to bind.

For more information about setting up the bind distinguished name see 4.3.2, “LDAP client daemon bind distinguished name (DN) options” on page 120.

3.3.3 Access rights when binding to the LDAP server

Prior to AIX 5L Version 5.2, the AIX client binds to the LDAP server using an administrative distinguished name. Starting with AIX 5L Version 5.2, the secldapclntd LDAP security client daemon can bind to the LDAP server with a proxy user distinguished name. However, in Version 5.2 the **mksecldap** command does not support creating a proxy account during setup. In Version 5.3 is does. If you choose to use this access method, then you need to decide what permission ACLs you assign to that DN. You can change the default proxy user permissions when setting up the ITDS Server on AIX with the **mksecldap -s** command by modifying the ACLs described in /etc/security/ldap/proxy.ldif.template. See 6.3.5, “Step 5: Installing a LDAP client with a proxy user ID” on page 198.

Some of the decisions that need to be made are what properties of a user should the proxy be able to change. For example, do you want the proxy user for a particular machine to be able to change the password for the user that will be effective on all servers? One of the clear advantages of proxy users is that you can have a proxy user for each departmental group of clients.

There is an example of setting up a proxy DN in 6.3.5, “Step 5: Installing a LDAP client with a proxy user ID” on page 198.

3.3.4 Adding initial users to the LDAP server

When configuring IBM Tivoli Directory Server on AIX, it may be tempting to add all of the users on the LDAP server during the setup. This is the default action for **mksecldap** unless you specify the -u NONE flag, which will not add any users. The disadvantage of adding all of the users is that this will place all the local users in the directory, even ones that you may not want to use LDAP. The advantage is that this is the easiest way to get a working system because all of the auxiliary schema information will get added, which will allow you to add new users with the **mkuser -R LDAP** command.

The other way to add the initial users is to create an LDIF file with sectoldif or nistoldif, and then manually remove the files that you do not want to add to the server prior to adding the users to the directory with an LDAP client command such as **ldapadd**. This give you a more controlled approach to populating the directory.

3.3.5 Server availability and failover considerations

For a highly available LDAP solution, you can set up one of two scenarios. The first is to use a single master LDAP server with replica servers. The second scenario is to use multiple master servers. IBM Tivoli Directory server provides single-master multiple replication and multiple-master replication as well as cascaded, gateway, and partial replication. This allows configuration of servers to match the topography of the network to ensure data availability and to maximize server response time. The term *master* is used for a server that accepts client updates for a replicated sub-tree. The term *replica* is used for a server that only accepts updates from other servers designated as a supplier for the replicated subtree.

Master LDAP server with multiple replica servers

In this scenario clients can authenticate against either the master or any of the replica servers. You can request updates on a replica server, but the update is actually forwarded to the master server by returning a referral to the client. If the update is successful, the master server then sends the update to the replicas.

Until the master has completed replication of the update, the change is not reflected on the replica server where it was originally requested. If the replication fails, it is repeated even if the master is restarted. Changes are replicated in the order in which they are made on the master.

Multiple LDAP masters

Multiple master servers is also known as peer-to-peer replication. In this scenario updates can be made to any of the masters and the masters stay in synchronization with a peer-to-peer relationship. If you are implementing a solution that has data centers that are located in different geographic locations this gives you some significant advantages in performance and reliability. Performance is improved by providing a local server to handle updates in a widely distributed network. Reliability is improved by proving a backup master server ready to take over immediately if the primary master fails.

3.3.6 Scalability and performance

Scalability can be accomplished by using the same scenario as availability by having multiple replica or master servers that the clients can authenticate against. By having a master or replica server located in the same geographical location you will have the least amount of network traffic over the slower network links.

For client tuning see 4.3.3, “AIX 5L LDAP client tuning” on page 122.

3.3.7 Manual versus mksecldap configuration

If you are using IBM Tivoli Directory server (ITDS) on an AIX-based system, you have two options for configuring the server and initially populating it from that server. These are to use the mksecldap script or to manually make the changes. While using mksecldap is a simple way to get started quickly, it does not give you the full picture that you can get by configuring everything manually. For details about how ITDS was downloaded and configured in this project see Appendix A, “IBM Tivoli Directory Server V6.0 installation” on page 273.

3.4 Microsoft Active Directory considerations

When planning to integrate your AIX 5L clients into a Microsoft Active Directory (MSAD) environment, it is very important to understand the differences between AD and a standard UNIX LDAP deployment. For that reason, we have chosen to take this section out of the standard LDAP server considerations and make a separate section dedicated to Active Directory. The following sections describe some of the challenges integrating UNIX into Active Directory environments and several limitations involving the AIX 5L LDAP client and MSAD.

As of the writing of this book, IBM does not officially support Microsoft Active Directory integration with AIX. There are three scenarios in this book covering AIX 5L and MSAD integration that have been shown to work with certain limitations. These scenarios are described Chapter 8, “Scenario: Microsoft Windows 2003 Active Directory integration” on page 217.

IBM is aware of customer demand for AIX 5L to support Microsoft Windows Active Directory.

3.4.1 AIX level and MSAD considerations

In this section we discuss the AIX level and MSAD considerations.

AIX 4.3.3 and AIX 5L Version 5.1

Prior to AIX 5L Version 5.2, there are no IBM-supported methods for integrating users with a Microsoft Active Directory. Some customers have implemented solutions based on the PADL pam_ldap module, which is an authentication-only module and is beyond the scope of this book.

AIX 5L Version 5.2

Starting at AIX 5L Version 5.2, IBM introduced a Kerberos authentication-only module supported in the standard LAM methodology described in 2.2.1, “Loadable authentication module (LAM) framework” on page 23. The KRB5A authentication module allows AIX users to authenticate against a Microsoft Windows Kerberos Key Distribution Center (KDC) server or any KDC that does not support the kadmin interface. When using KRB5A, the user information other than the password must be stored either locally on the AIX client or in an LDAP directory. AIX 5L Version 5.2 also introduced the concept of mapping files that allow nonstandard LDAP schemas to be used to store the AIX user information. Although the scenarios in this book use AIX 5L Version 5.3, the same basic information should work for AIX 5L Version 5.2 as well. The only exception is that the AIX 5L Version 5.2 LDAP client cannot bind to the Active Directory using Kerberos—you must either bind anonymously or use a specific bind DN and password.

AIX 5L Version 5.3

Starting with AIX 5L Version 5.3, the concept of server-side authentication was introduced. This added another ability to the integration of AIX and Microsoft Active Directory. The LDAP client daemon can now bind to the Active Directory using Kerberos. This allows the LDAP client to securely bind to the Active Directory without SSL.

3.4.2 Challenges for integrating AIX into MSAD

The major challenges you will find when integrating UNIX into an MSAD environment are listed below. These topics are described in more detail in this section:

- Multiple containers are used to store users and groups, and users and groups are not separated into separate containers.
- Windows supports long user and group names with incompatible characters, such as spaces.
- Between Microsoft Windows Services for UNIX (SFU) and Windows 2003 Server R2, there are multiple schemas that can support UNIX clients:
  - SFU uses object class and attribute names that are not compliant with RFC2307 standards. There are also multiple versions of the SFU schema.
  - Microsoft announced that the upcoming Windows 2003 Server R2 will now support a RFC2307-compliant schema. Since Windows 2003 Server R2 is not released, this book does not covering this topic.
- Microsoft SFU determines group membership with two attributes. One attribute uses user login names and the other uses distinguished names (DN).

Multiple containers used to store users and groups together

One major difference between Active Directory and UNIX LDAP directory deployments is the location of the user and group entries. Active Directory stores the user and group entries together in the same container. AD also uses multiple containers to store user and group entries. The following figure shows the Active Directory Hierarchy for this scenario. The Mainz and Cambridge containers were added to show other possibilities. Figure 3-2 on page 76 is a list of all the possible places user and group entries could be stored.

- cn=Builtin,dc=example,dc=com
- cn=Users,dc=example,dc=com
- cn=cambridge,cn=Users,dc=example,dc=com
- cn=mainz,cn=Users,dc=example,dc=com

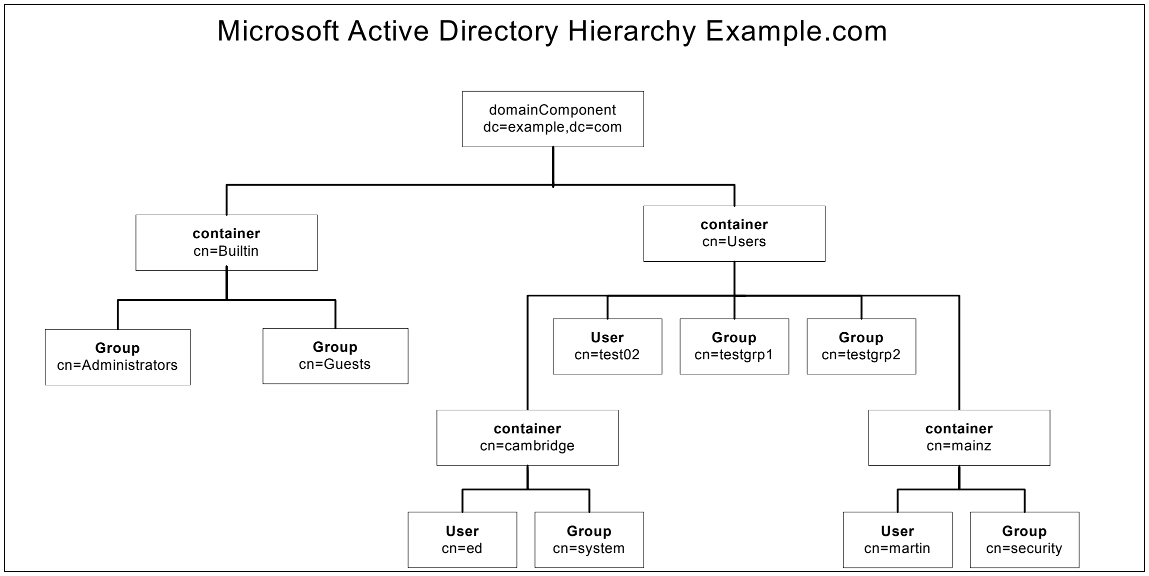


Figure 3-2 Microsoft Active Directory hierarchy for example.com

A typical UNIX LDAP deployment, especially one that was migrated from Network Information Service (NIS), would only use a single container for user entries and another container for group entries. While this statement might not describe all environments, most LDAP clients will use a single container for users and another for groups. Figure 3-3 on page 77 shows a typical UNIX LDAP hierarchy with a container for users, one for groups, and one for the NIS maps. In this sample hierarchy, users and groups are only stored in the following containers:

- ou=People,dc=example,dc=com
- ou=Groups,dc=example,dc=com

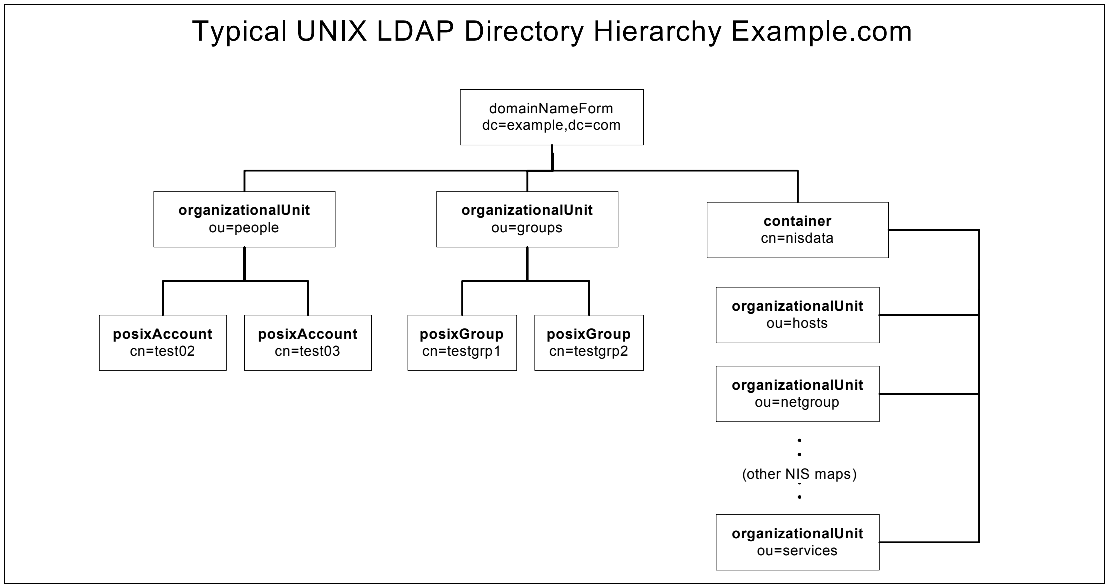


Figure 3-3 Typical UNIX LDAP Directory hierarchy with NIS maps

The AIX 5L LDAP client will only support a single container for user entries and a single container for group entries. The container specified for user and group entries can be the same.

Long user and group names with unsupported characters

Versions of Windows 2000 and later support user and group logon names up to 104 characters. However, it is not practical to use logon names that are longer then 64 characters, as the display name attribute is limited to 64 characters. Windows also supports the use characters that are illegal to use in UNIX user names, such as spaces.

In AIX 5L the default maximum user name and group names is eight characters. Starting with AIX 5L Version 5.3 user and group names can be up to 254 characters. For more information about changing this limit please see 3.1.5, “Supporting long user names and passwords” on page 58. AIX will not support any user logon name that includes a space. If possible replace the spaces in the user name with an underscore (_).

Active Directory schemas for UNIX integration

In this section we discuss Active Directory schemas for UNIX integration.

Microsoft 2003 Server R2 schema

The Microsoft upcoming release of Windows 2003 Server R2 will now support object classes and attribute names that are RFC2307 compliant. Since this is not officially released and subject to change, it is not covered in this book.

For more information about Windows 2003 Server R2, please visit the following Web site:

`http://www.microsoft.com/windowsserver2003/r2/default.mspx`

Microsoft Windows Services for UNIX (SFU) schemas

Depending on the version of SFU that you are using, SFU defines two different schemas for UNIX client integration. Microsoft's SFU defines object classes and attribute names to store the different NIS maps but does not use the RFC2307-compliant name. This requires you to use the LDAP attribute and object class mapping feature in the AIX LDAP client. This book will only cover the user and group maps, as AIX 5L Version 5.2 and later only support mapping of the user and groups maps.

The SFU 2.0 schema adds msSFUPosixAccount and msSFUShadowAccount as auxiliary classes to the user class and the msSFUPosixGroup object class is added as an auxiliary class to the group class.

The SFU 3.0/3.5 schema adds msSFU30PosixAccount and msSFU30ShadowAccount as auxiliary classes to the user class. and the msSFU30PosixGroup is added as an auxiliary class to this class.

One way to distinguish between SFU 2.0 and 3.0/3.5 versions of object classes and attributes is by the name. SFU 3.0/3.5 names are normally prefixed by msSFU30, while the SFU 2.0 names use msSFU or no prefix at all.

Microsoft uses different password encryption methods

By default, Microsoft user passwords are stored in Unicode format on the directory server in the unicodePwd attribute. When SFU is installed, an additional password attribute is added to support a UNIX encrypted password. When a user account is added to a Windows domain, the user's password is stored in the unicodePwd attribute, in a Unicode format. When UNIX attributes are added to the user, the msSFU30Password containing the user's password is given the default value of ABCD!efgh12345$67890. The user or the administrator must change the password again and Windows will synchronize the unicodePwd and msSFU30Password passwords in their respective formats.

The SFU attribute msSFU30Password, which contains the user's password in UNIX crypt format, is only necessary if you are using client-side authentication with LDAP. When using Kerberos or server-side authentication, the unicodePwd is used to authenticate users.

Multiple attributes to specify group memberships

The proposed RFC2307bis standard suggests using both the memberUid and uniqueMember attributes to define a group's membership list.

The uniqueMember attribute is defined by the RFC2256 standard as a multi-valued list of distinguished names (DN), in the groupOfUniqueNames object class. The uniqueMember attribute allows you define to a group membership that extends beyond a single container. It would also be possible to have group members with the same user login names from different containers. Example 3-4 illustrates a group named testgrp2, which has a user named test03 from the cn=groups,dc=example,dc=com and cn=mainz,cn=users,dc=example,dc=com containers.

Example 3-4 Using uniqueMember attribute

```
dn: CN=testgrp2,CN=Users,DC=example,DC=com
objectClass: top
objectClass: posixGroup
objectClass: groupOfUniqueNames
cn: testgrp2
gidNumber: 10001
uniqueMember: cn=test02,cn=groups,dc=example,dc=com
uniqueMember: cn=test03,cn=groups,dc=example,dc=com
uniqueMember: cn=test03,cn=mainz,cn=users,dc=example,dc=com
```

The memberUid attribute is defined by the RFC2307 standard as a multi-valued list containing a list of user login names in the posixGroup object class. Since the group member names are not uniquely identifiable, the memberUid attribute values are only valid within a single container. In Example 3-5 there would be no way to distinguish between users from different containers with the same user login name.

Example 3-5 Using memberUid attribute

```
dn: CN=testgrp2,CN=Users,DC=example,DC=com
objectClass: top
objectClass: posixGroup
cn: testgrp2
gidNumber: 10001
memberUid: test02
memberUid: test03
```

The Microsoft Windows Services for UNIX (SFU) uses two attributes for specifying group membership. The object class and attribute names are not compliant with the RFC2307 and RFC2256 standards and even differ between SFU versions. The following table shows the attribute names, values of type, and examples.

Table 3-2 Microsoft SFU 2.0 and 3.X attributes used to specify user group membership

MS SFU 2.0	MS SFU 3.0/3.5	Values of type/example values
memberUid	msSFU30MemberUid	User login names test02 test03

posixMember	msSFU30PosixMember	Distinguished names
		
		cn=test02,cn=groups,dc=example,dc=com
		cn=test03,cn=groups,dc=example,dc=com
		cn=test03,cn=mainz,cn=users,dc=example,dc=com

Microsoft SFU will populate both the user name and DN valued attributes at different times. The following list describes which attributes are used in different situations:

- MS SFU will populate the msSFU30PosixMember (or posixMember) attribute if it is able to resolve the user's DN, by looking for the user in the NIS passwd map. Users are added to the passwd map when their UNIX attributes are defined.
  - The Active Directory Users and Computers management tool will only allow you to add group members that are available in the passwd map.
  - When using the **nismap** command to add group members and the user's DN is able to be resolved.
- MS SFU populates the msSFU30MemberUid (or memberUid) attribute when if it is unable to resolve the user's DN, by looking for the user in the passwd map.
  - When using the **nismap** command to add members to a group and the user's DN is unable to be resolved.
  - If you import a NIS group map using the **nismap** command before you import the corresponding passwd map. The group member's DN would not be able to be resolved as the passwd map was not yet populated.

While the ability to have group membership lists that are uniquely identifiable and cross multiple containers is very useful, the UNIX LDAP client support is not very widespread. Currently AIX 5L does not support defining group memberships using DNs. The easiest method to resolve this problem is by synchronizing the DN and user login name membership lists with an external program. This allows you to support clients who use user login names and DNs for group membership. One problem with this solution is that you would be unable to distinguish between users with the same user name in multiple containers.

The following example shows the synchronization of the msSFU30PosixMember and msSFU30MemberUid attributes. The test02 and test03 users are both in the same container.

Example 3-6 lsldap output showing memberUid and posixMember group membership

```
# lsldap -a group testgrp2
dn: CN=testgrp2,CN=Users,DC=example,DC=com
objectClass: top
objectClass: group
cn: testgrp2
distinguishedName: CN=testgrp2,CN=Users,DC=example,DC=com
instanceType: 4
whenCreated: 20050915223328.0Z
whenChanged: 20050919053249.0Z
uSNCreated: 20494
uSNChanged: 20720
name: testgrp2
objectGUID: ;&-H^Ê[O–lÛ
objectSid:
sAMAccountName: testgrp2
sAMAccountType: 268435456
groupType: -2147483646
objectCategory: CN=Group,CN=Schema,CN=Configuration,DC=example,DC=com
msSFU30Name: testgrp2
msSFU30GidNumber: 10001
msSFU30MemberUid: test03
msSFU30MemberUid: test02
msSFU30NisDomain: example
msSFU30PosixMember: CN=test03,CN=Users,DC=example,DC=com
msSFU30PosixMember: CN=test02,CN=Users,DC=example,DC=com
```

3.4.3 Whether you should use Kerberos for user authentication

Starting with AIX 5L Version 5.2, support was added to allow authentication only to a Microsoft Active Directory Server using Kerberos with the KRB5A authentication module. This was first documented in the *AIX 5L Version 5.2 Differences Guide* and in the *AIX 5L Version 5.2 Security Guide*, which can be found at the following Web site:

http://publib16.boulder.ibm.com/doc_link/en_US/a_doc_lib/aixbman/security/kerberos_questions_troubleshooting.htm

Using this method you can combine Kerberos authentication against Microsoft's Kerberos Key Distribution Center (KDC) with user authorization and identification using local files or LDAP. An example of this type of supported integration is described in 8.1, "Scenario: Kerberos only" on page 219.

Limits to PAM-based integration solutions

One of the solutions that administrators often try to put together for Microsoft Active Directory server integration from other UNIX variants is to use a custom PAM module such as the ones from PADL (`http://www.padl.com`). Although PADL offers an authentication-only module for AIX 4.3.3, there is none that we are aware of for AIX 5L Version 5.1 or 5.2. Because AIX 5L Version 5.1 does not offer PAM support for login or other authentication methods, there is simply no way to use PAM at AIX 5L Version 5.1 to do Microsoft ADS integration. Support for PAM authentication was enhanced at AIX 5.2, but requires that applications be recompiled after mollifications to remove some of the standard library modules that are not supported. The experience of this author in talking to people that have attempted this is that no one has been successful in accomplishing this task. Part of the issue is related to required feedback from the PAM custom module back through the PAM authentication module through to the security.

Starting with AIX 5L Version 5.2, if you set auth_type=PAM_AUTH in /etc/security/login.cfg, this replaces the authentication mechanism with a true PAM mechanism and is compatible with standard modules that you can find. Although this may be an attractive solution for those who are using the modules on other UNIX variants, it is probably a better solution to go with the supported AIX solution of KRB5A authentication. If you want to avoid Kerberos in an AIX 5L Version 5.3 based solution, setting the authtype to LDAP_AUTH in /etc/security/ldap/ldap.cfg may provide a method of doing direct LDAP server-based authentication with ADS.

3.5 Client-related considerations

After deciding on the LDAP server and how it should be configured, then you need to understand the decisions that you can make when setting up the client configurations. Most of the topics discussed in this section are elaborated on in Chapter 4, "AIX 5L Version 5.3 LDAP client configuration" on page 101, where sample configurations for different types of clients are discussed.

This section describes some of the decisions that you will make on the client side when configuring your LDAP environment. The topics discussed in this section include:

- "Which accounts should remain in local files" on page 83
- "Authentication failover" on page 85
- "Client-side and server-side authentication" on page 86
- "Proxy users" on page 89
- "User login restriction" on page 91
- "User addition and removal" on page 91
- "Password management" on page 94
- "Specifying LDAP server priority" on page 85
- "AIX LDAP client daemon tuning" on page 95

3.5.1 Which accounts should remain in local files

One of the most important decisions you need to make is which user accounts to leave with local file authentication and which accounts to place on the LDAP server. You can put all users by default in LDAP, but if you do and the network goes down, you may not be able to log in to the server to fix things. Also, you need to decide whether you want the same root password on all of your UNIX servers. It is easy to designate any user such as root to use the local files by simply specifying SYSTEM=files in the user's /etc/security/user stanza. During the planning phase, you need to decide which users to leave local. Another thing that you may want to do in the early stages of testing is to only make a single user or group of users on the client be LDAP users. In this case, leave the default user to use the *compat* authentication and specify SYSTEM=LDAP and registry=LDAP for the test users.

LDAP users with default user set to local or compat

During early testing, the following setup would allow a single user called test001 to use the LDAP directory, while the rest of the users used the default authentication configuration method previously used. It is a good practice to always specify the root with a SYSTEM = "compat" or "files" and registry = files so that if any problems are encountered with the network, you will be able to gain access with your system. If you specify that you cannot directly log in using the root user ID, you need to have a user ID that is also in local files.

Example 3-7 shows the stanzas from /etc/security/user for LDAP testing of a single user, while users such as normu continue to use the AIX files and password from /etc/security/passwd.

Example 3-7 Default local users with test001 LDAP user

```
default:
        SYSTEM = "compat"
        admin = false
        login = true
        su = true
        daemon = true
        rlogin = true
        sugroups = ALL
...
root:
        admin = true
        SYSTEM = "compat"
        registry = files
        loginretries = 0
        account_locked = false
...
normu:
        admin = false
        expires = 0101000070

test001:
        admin = false
        SYSTEM = "LDAP"
        registry = LDAP
```

LDAP users with default user set to LDAP

Once you have completed tests with your limited user set shown by test001 here, then the default user can be changed to use LDAP. In this case (as shown in Example 3-8) the default user is set to LDAP, so user test001 no longer needs to be included in the /etc/security/user file since the LDAP authentication module will check for the users on LDAP. Make sure to leave root local but login may be set to false with the sugroups set to admin. Now user admin1, who is in group admin, should also be included as a local user so that if the network or LDAP server is down you will be able to log in using the local user admin1 and still have access to the system as root through su from admin1.

Example 3-8 Default AIX user set to LDAP authentication

```
default:
        SYSTEM = "LDAP"
        registry = LDAP
        login = true
        su = true
        daemon = true
        rlogin = true
        sugroups = ALL
...
root:
        admin = true
        registry = LDAP
        login = false
        su = true
        daemon = true
        rlogin = false
        sugroups = admin
        SYSTEM = "compat"
        registry = files
        loginretries = 0
        account_locked = false
...
admin1:
        admin = true
        SYSTEM = "compat"
        registry = files
```

3.5.2 Authentication failover

AIX supports two different failover capabilities to handle server failures. The AIX LDAP clients can have multiple servers configured to allow the LDAP client to fail over to another server if the current server is unavailable. The AIX security subsystem is also able to fall back to an alternative authentication methods if the primary authentication methods fails.

Specifying LDAP server priority

In the AIX Version 4.3.3 and AIX 5L Version 5.1 LDAP clients, the LDAP client allows specification of multiple LDAP servers to allow for server failover. The LDAP client will open a single connection to each server and use the first available server. If the current server was to become unavailable, it would randomly choose another available server.

Starting with AIX 5L Version 5.3, the AIX 5L LDAP client will connect to each of the servers specified in the ldapservers settings but will initially use the first one in the list that is available. If the current server was to become unavailable, the LDAP client will choose the first available LDAP server in the list starting from the left. This feature was back ported to AIX 5L Version 5.2.

Example 3-9 shows the ldapservers setting in the AIX LDAP client daemon configuration file. There are three LDAP servers listed, with ldap1.example.com being the highest priority.

Example 3-9 LDAP server priority configuration

```
ldapservers:ldap1.example.com,ldap2.example.com,ldap3.example.com
```

Note: The LDAP client daemon will open connections to every server listed in the ldapservers setting. The AIX Version 4.3.3 and AIX 5L Version 5.1 LDAP client will only open a single connection to each LDAP server.

Starting with AIX 5L Version 5.2, the LDAP client will open the number of connections specified by the connectionsperserver setting, which defaults to 10.

Fallback to alternative authentication method

The second alternative is to set up administrative users to be files only, as described in 3.5.1, "Which accounts should remain in local files" on page 83.

The third alternative is to set up users to use LDAP as the primary authentication method, and to use files as the backup. You can set this up for users who need this fallback by properly configuring the SYSTEM attribute in the /etc/security/user file. The following example shows one method of doing this for user admin02. If LDAP is available, then admin02 will be authenticated using LDAP, but if LDAP is unavailable then authentication will be done with local files.

Example 3-10 Primary LDAP authentication with failover to files

```
admin02:
        admin = true
        SYSTEM = "LDAP or (LDAP UNAVAIL AND files)"
        registry = LDAP
```

This means that if only LDAP fails, then the user will need a password in the /etc/security/passwd file to authenticate.

3.5.3 Client-side and server-side authentication

The AIX LDAP client supports two different authentication types, client-side and server-side authentication. Client-side authentication, also called unix_auth, is supported on all versions of AIX starting with AIX 4.3.3. Server-side authentication, also called ldap_auth, is only supported on AIX 5L Version 5.3.

Client-side authentication is when the AIX 5L LDAP client retrieves the encrypted password from the LDAP server and compares it to the locally encrypted password received from the user. This authentication type is called client-side authentication because the local LDAP client is authenticating the user.

Server-side authentication is when the AIX 5L LDAP client authenticates a user by binding to the LDAP server using the user's DN and password. If the LDAP bind succeeds, the user will be successfully authenticated and the user will be allowed to log on.

Figure 3-4 illustrates the differences between client-side and server-side authentication.

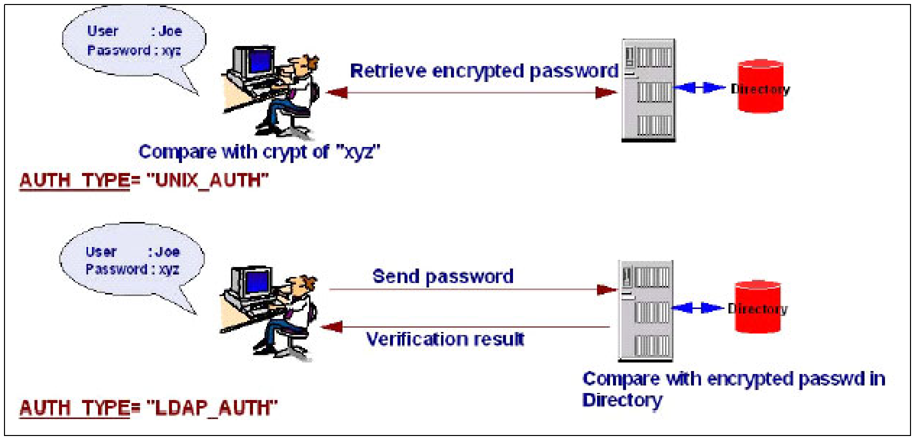


Figure 3-4 Differences between client-side and server-side authentication

Client-side authentication

The following list discusses the advantages and disadvantages when using client-side authentication with the AIX LDAP client:

- The AIX LDAP client only supports the UNIX crypt encryption algorithm. If your LDAP server stores the user's passwords in another format, AIX clients will not be unable to authenticate. For example, IBM Tivoli Directory Server V6.0 can store user passwords in crypt, SHA-1, AES128, AES192, AES256, or clear text.
- The traditional UNIX crypt encryption algorithm is limited to a maximum password length of eight characters.
- When a user authenticates, no passwords are sent in clear text across the network, as the LDAP server sends the encrypted user's password to the client for comparison. Depending on the strength of the user passwords and

your security policy, you might not choose to use SSL to secure the LDAP connections.

- Using the client-side authentication, the AIX LDAP client daemon can use the existing connections to the LDAP server and does not need to open an additional connection for each user authentication. On a busy system, this could allow better performance.
- When using client-side authentication, invalid logon attempts are not centrally recorded, as the LDAP server is unaware of when a request for a user's password entry is for authentication or for information lookup. Invalid logon attempts will only be tracked on your local AIX machines.

Server-side authentication

The following list discusses the advantages and disadvantages of using server-side authentication with the AIX 5L LDAP client:

- Since the users's password is sent to the LDAP server in clear text, the LDAP server can do the password encryption and comparison using the encryption algorithm used to store the passwords on the server. You are no longer required to use the UNIX crypt encryption algorithm to support AIX LDAP clients.
- With server-side authentication, we highly recommend using SSL to protect communications between client and server, as the user's password is now sent as clear text across the network.
- One advantage of not using the UNIX crypt encryption algorithm is that you are no longer limited to a maximum password length of eight characters. When using server-side authentication, AIX 5L supports passwords up to 32 characters long.
- With server-side authentication, every user authentication requires a new connection to the LDAP server. On an AIX 5L LDAP client with many user authentications, there will be an increased number of connections to the LDAP servers, and the cost per connection increases when using SSL.
- If all users must authenticate against the LDAP servers, you can enforce security and password policies centrally on the LDAP servers. For example, the LDAP servers can track invalid login attempts centrally and lock out users for too many invalid logins. When a user changes his password, the password is sent to the LDAP server in clear text and the LDAP server can ensure the password complies with the password policy before accepting the password.

Choosing between client-side and server-side authentication will depend on the levels of AIX LDAP clients and how your environment and directory server are set up. If you are supporting AIX clients other then AIX 5L Version 5.3 you will obviously need to do client-side authentication. If all of your clients are AIX 5L Version 5.3 you can choose either method.

3.5.4 Proxy users

Information about creating proxy users on the LDAP server to have specific permission was described in 3.5.4, “Proxy users” on page 89. If this proxy user is set up on the LDAP server, then the proxy user should be set up on the client to use this proxy user. This is done inside the /etc/security/ldap/ldap.cfg file in the basedn value, as shown in 4.3.2, “LDAP client daemon bind distinguished name (DN) options” on page 120.

3.5.5 LDAP or compat mode use for SYSTEM attribute

In the stanza of every user is the SYSTEM entry. This entry defines with which method a user should be authenticated. For the use with LDAP this could be LDAP or compat, as explained below:

LDAP mode

In LDAP mode the SYSTEM stanza in the /etc/security/user file is set to LDAP, as shown in Example 3-11. Therefore the system checks for an LDAP server when a user tries to authenticate directly. All this is done through the loadable authentication modules (LAM) mechanism using the LDAP module.

Example 3-11 User entry for LDAP mode

```
user500:
        admin = false
        SYSTEM = "compat"
```

Compat mode

In compat mode the system checks the /etc/passwd file, and if it finds a plus sign (+ or +@netgroup or + user) in there it will check for network authentication services through NIS. This service for authentication is offered by the LAMs. So if the LDAP module is configured with *options = netgroup*, the LDAP LAM will offer this to the system. So now the system is going to use the NIS-like authentication to an LDAP server with a RFC2307 schema. See Example 3-12.

Example 3-12 User entry for compat mode

```
user501:
        admin = false
        SYSTEM = LDAP
```

Figure 3-5 describes the decision process for determining whether user information will come from local files, the NIS server, or LDAP when SYSTEM and REGISTRY are set to compat.

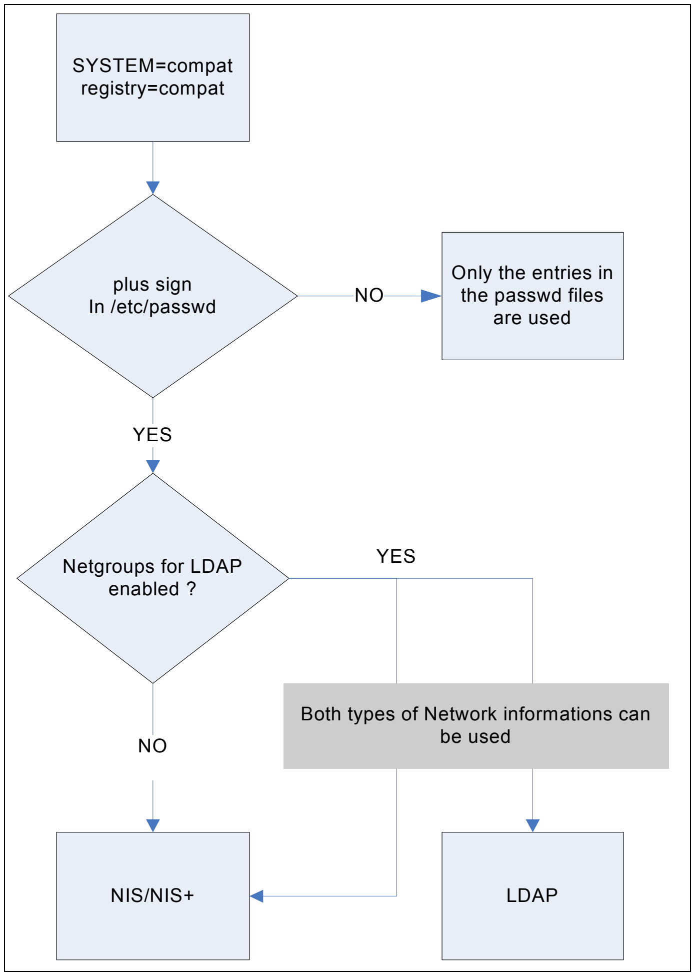

SYSTEM=compat
registry=compat
plus sign
In /etc/passwd
NO
Only the entries in the passwd files are used
YES
Netgroups for LDAP enabled ?
YES
Both types of Network informations can be used
NO
NIS/NIS+
LDAP

Figure 3-5 Compat authentication with LDAP

Which mode should be chosen

The main decision point for determining which configuration to use is the question of whether netgroups must be used to restrict user access to the system. If netgroups have to be used for restricting the users from accessing the system, the only possible way is using the compat mode.

In any other case the LDAP mode can be used for any implementation.

3.5.6 User login restriction

In AIX 5L there are three different ways of using LDAP for restricting user login:

- Restricting users in /etc/security/user
- Restricting users with hostsallowedlogins and hostsdeniedlogin
- Restricting users with netgroups in compat mode

Restricting users in /etc/security/user

The first method restricts users setting the SYSTEM attribute for the default user to files or compat, and listing each allowed LDAP user in the /etc/security/user file with the individual user stanza SYSTEM attribute set to LDAP.

Restricting users with hostsallowedlogins and hostsdeniedlogin

The second method of restricting LDAP users sets the SYSTEM attribute for the default user to LDAP. All users that will use local files must have a stanza in /etc/security/user that has the SYSTEM=files set. The rest of the users do not have to be listed in any of the local security files, only on the LDAP server. Restriction of these users can be done for the AIX or the RFC2307AIX schemas by using the hostsallowedlogin and hostsdeniedlogin. If neither of these attributes exists for a user, then the user is allowed to log in to any LDAP client system by default. The attributes are user-specific attributes that can be changed with the **chuser** command, as shown here:

```
# chuser -R LDAP hostsallowedlogin=host1, host2, host3 test01
```

In this case, the user test01 is only allowed to log in to hosts1, host2, and host3. The hostsdeniedlogin is normally used when the hostsallowedlogin lists a domain of clients and you want to restrict the user from a particular client.

Restricting users with netgroups

The restriction with netgroups has the advantage that it can be used across different platforms. The same netgroups can be used for AIX 5L, Linux, and Sun Solaris. The login restrictions are based on groups that are defined on the LDAP server. This groups can then be specified to be allowed to log in or not to allow login. See also “Compat mode” on page 89.

3.5.7 User addition and removal

The goal of user management is to easily manage the creation of, changes to, and deletion of users from a central source.

LDAP UID minimum

Starting with AIX 5L Version 5.3, you can set the minimum ID for LDAP users and groups created with the **mkuser** and **mkgroup** commands. This helps isolate local users and LDAP users into to user ID ranges. Normally in AIX, the next user or group numeric ID for both normal and administrative users is determined by the /etc/security/.ids file. We recommend that you do not edit this file. With LDAP, the users are determined by the aixuserid, aixgroupid, aixadminuserid, and aixadmingroupid parameters, which are a part of the aixbaseid container. For information about changing these see 3.3.7, “Manual versus mksecldap configuration” on page 73.

For more details about the system security implications of using the **mkuser** and **mkgroup** commands see the *IBM AIX 5L Security Guide*, found at the following location:

`http://publib.boulder.ibm.com/infocenter/pseries/v5r3/topic/com.ibm.aix.doc/aixbman/security/security.pdf`

Adding users

There are several ways of adding users to an LDAP directory:

- Users can be added when configuring the LDAP server with **mksecldap** command.
- Users can be generated using the **sectoldif** or **nistoldif** commands.
- Users can be added from the AIX client with the AIX **mkuser -R LDAP** command, or by adding them with the **ldapadd** command after creating an LDIF file.

Removing users

Users can be removed from the LDAP directory from the AIX client with LDAP client commands such as **ldapdelete** or with the AIX security **rmuser -R LDAP** command. If the user exists in /etc/security/user, then they will need to be removed with **rmuser**.

When removing users, consideration should be given to what happens to files owned by that user. If the user is staying in the company, but simply being removed from one of the clients, then the user should be responsible for moving files if they are stored locally. Of course, if you are using a centralized home directory with automount, then this is not a problem. If the user is leaving the company, the files should be archived for security reasons prior to removing the user and the directory.

3.5.8 Default values aixid and aixbaseid

During the migration with **`sectoldif`** or **`nistoldif`**, three entries are created in the LDAP directory that are not directly related to a user or a group. These three are:

default	The default entry specifies all values for a user that are not set in the entry of the user itself. It is corresponding to the default entry in the /etc/security/user file. The entry is stored at cn=default,ou=People,dc=example,dc=com. This value can be changed with normal AIX commands.
aixid	The aixid specifies the values to use when a new user is created. The values for the user ID increase every time a user is created. This is equivalent to the /etc/security/.ids file. To change it a **`ldapmodify`** command has to be used with a valid LDIF file based on /etc/security/ldap/ldapid.ldif.template.
aixbaseid	This entry specifies the base values when creating new users and groups. All user IDs and group IDs that will be created in the directory are above these values.

Example 3-13 shows how the LDIF for this looks.

Example 3-13 LDIF file to set aixbaseid: minimum ID value for users

```
dn: ou=System,
ou: System
objectClass: organizationalUnit

dn: cn=aixid,ou=System,
cn: aixid
objectClass: aixadmin
aixadminuserid: 8
aixuserid: 205
aixadmingroupid: 14
aixgroupid: 204

dn: cn=aixbaseid,ou=System,
cn: aixbaseid
objectClass: aixadmin
aixadmingroupid: 1
aixadminuserid: 1
aixuserid: 200
aixgroupid: 200
```

The numeric value of the normal user and group ID for new users and groups created with the **`mkuser`** or **`mkgroup`** command will have values greater than or equal to the value specified for aixuserid and aixgroupid. The admin users and groups follow the same rules but use the aixadminuserid and aixadmingroupid parameters.

If you do not do the setup with the AIX conversion commands, these attributes can still be added using the /etc/security/ldap/ldap.ldifid.template as your starting point. The LDIF formatted data in this file is shown in Example 3-14.

Example 3-14 Sample minimum user ID for normal users of 10000, admin users 2000

```
dn: cn=aixbaseid,<ou=system,cn=aixdata>
objectClass: aixadmin
aixadmingroupid: 2000
aixadminuserid: 2000
aixgroupid: 10000
aixuserid: 10000
```

The steps to modify and add this file to your LDAP server are:

1. Copy this file to ldapid.ldif.
2. Change the value <ou=system,cn=aixdata> to match the actual DN from your LDAP server. You can find the DN by querying the ou=system entry with the **`ldapsearch`** command.
3. Change the aixadmingroupid and aixadminuserid to the minimum numeric ID values you want for administrative users and groups.
4. Change the aixgroupid and aixuserid to the minimum numeric ID values you want for normal users and groups.
5. Add the ldapid.ldif file to your LDAP server with **`ldapadd`**.

Once added, test the command from AIX with **`mkuser -R LDAP`** testuser and then check the UID for that user with **`lsuser -R LDAP`**.

3.5.9 Password management

Managing passwords consists of setting the rules for how many and what characters make up the password, and how often they need to be changed, as well as who can change the password.

Password rules

By default AIX has very few restrictions on the password except that local passwords must be eight characters or less. This eight-character limit will be in effect whenever the password is encrypted by the crypt() subroutine. When dealing with LDAP users, the restrictions should be understood by all participating clients.

The password restrictions for the various schemas are described in the 3.2, “Which schema you should use” on page 60.

Password and user ID rules with RFC2307AIX

A normal AIX 5L system deals with restrictions on passwords and user IDs in the /etc/security/user file. For example, the minlen attribute defines the minimum length of a password. With an LDAP server with the RFC2307AIX schema all values that are used in the /etc/security/user file can be used at the LDAP server also.

Changing passwords

When setting up the LDAP server and deciding on the authentication type and the permissions of the proxy user, you need to decide exactly who needs the authority to change user passwords and how those password changes will be made. In addition to this, you will need to understand what restrictions you will put on the user’s password and how often they will need to be changed. In some cases, such as Microsoft Active Directory, you may need to use server-based tools to change the password.

When allowed, the root user can change passwords on the LDAP server by using **`passwd -R LDAP`** ***`username`***. Normally the user will be able to change her own password with the **`passwd`** command without having to add the -R LDAP field. The database to be change will be determined by the user’s SYSTEM attribute defined either in /etc/security/user or on the LDAP directory.

3.5.10 AIX LDAP client daemon tuning

The secldapclntd daemon not only acts as the interface between users on the client and the LDAP server, it also takes on the role of caching the user and group information. The types of things that you can control with the caching daemon are how long items will be cached before they are refreshed and how large the user and group caches will be.

The advantage of having the cache is that you avoid network traffic for applications that do frequent authentication and normally you will get better login performance with cached user and group data. The disadvantage is that when you change the user password from one client, that password may not be recognized immediately at the other clients if the data is still cached. The parameters in /etc/security/ldap/ldap.cfg that are related to this include cachetimeout, usercachesize, groupcache, heartbeatinterval, and ldaptimeout.

Other parameters that will affect the client performance are the number of client daemon threads as well as the number of simultaneous connections between the client daemon and the LDAP server. The attributes that you would tune in

/etc/security/ldap.ldap.cfg include numberofthread, connectionsperserver, searchmode, and ldaptimeout. The searchmode will affect performance positively on an LDAP server where the object classes contain a lot of information that is not related to AIX security user information.

Also, parameters that will affect the behavior of the client if there is a problem with the server are the heart beat interval that the daemon uses to see if the server is still alive, and the time-out value for how long the daemon will wait for the server to respond.

Details on setting up the caching daemon, the values that can be set with the **mksecldap** command, and the parameters in /etc/security/ldap/ldap.cfg are described in 4 4.2.3, “AIX 5L LDAP client daemon” on page 108.

3.5.11 Default group must exist on LDAP server for mkuser command

The **mkuser** command checks the /usr/lib/security/mkuser.default file to determine the default values for the process group (pgrp) and groups that will be assigned to new users. These same values are used for creating local users and users in the database specified by the -R flag. If the group does not exist on that database then the user will not exist and the **mkuser** command will return an error that the group listed in the mkuser.default file does not exist. This means that for the **mkuser** to work properly for both local and LDAP users, the default pgrp must exist both in local files as well as on the LDAP directory. For example, in the default mkuser.default file shown in the Example 3-15, the staff user must exist both in /etc/group and on the LDAP server.

Example 3-15 Default /usr/lib/security/mkuser.default file

```
user:
        pgrp = staff
        groups = staff
        shell = /usr/bin/ksh
        home = /home/$USER

admin:
        pgrp = system
        groups = system
        shell = /usr/bin/ksh
        home = /home/$USER
```

There is no easy way to get around this problem if you do not want to have the group exist in both places or if you want a different default user in the LDAP directory from the local directory when using **mkuser**. To make things worse, the problem exists even if you try to override the default group with the -a pgrp parameter for **mkuser**. The group in mkuser.default must still exist or you will get the error.

3.6 Planning summary

No matter what your existing environment is, when you plan for moving to LDAP, you must decide the following things before beginning the migration:

1. Which LDAP server are you going to use?
2. Which schema makes sense in your environment?
3. Resolve any conflicting information in a heterogeneous environment.
4. Install and configure the server.
5. Install and configure the client.
6. Add users.
7. Decide on failover mechanisms.

When moving to LDAP from a system that previously used local files, there are a number of activities that you will complete during the planning phase. The following list summarizes those planning activities:

1. Collect information from all of the clients using **sectoldif**.
2. Compare the information for duplication of user and group names and IDs.
3. Resolve conflicting information.
4. Add the information to the LDAP server using **ldapadd**.

When moving to LDAP from a system that previously used NIS or NIS+, there are other considerations. The following lists summarizes the planning steps in this area:

1. Collect information from the NIS server using **nistoldif**.
2. Add the information to the LDAP server using **ldapadd**.
3. Add any other containers and information.

Part 2

LDAP client integration

4

AIX 5L Version 5.3 LDAP client configuration

This chapter describes the basic AIX LDAP client installation and configuration steps for using the client with all of the various LDAP servers described later in this book. More detailed setup steps for configuration with individual LDAP servers are described in the individual chapters.

The topics discussed in this chapter include the following:

4.1 Steps for client installation and configuration

This chapter takes you through the basic setup steps that are applicable to all LDAP servers. The following steps include the things that you need to do to have a working client server:

1. Plan for the installation and configuration.

   Prior to starting the installation, we recommend that you read the planning chapter and this chapter completely to understand what features you can implement and what restrictions may be caused by your choice of things like the LDAP server, base DN, and schema.

2. Install the AIX 5L LDAP client software.

   Install the AIX 5L LDAP client as described in 4.2.1, “AIX 5L LDAP client software installation” on page 105. The LDAP client file set, ldap.client, and subsequent software are available on the AIX 5L product media. Make sure that you have installed the file sets necessary to have all of the security that your plan calls for.

3. Make sure that you have a working LDAP server.

   Before installing the client make sure that you have a working LDAP server and that you have a bind DN with the appropriate permissions configured on that server. You will need the bind DN of an administrative account or a proxy account on the LDAP server, the password for that account when setting up the client, and the base DN for the user and group information stored on that server.

4. Perform basic client configuration with **`mksecldap`**.

   Do the initial configuration using the **`mksecldap`** command, as described in 4.2, “Installation and basic client configuration” on page 105. This creates the configuration file and starts the AIX 5L LDAP client daemon secldapclntd that controls all communications from this client to and from the LDAP server. This will also automatically add the LDAP LAM entry to the /usr/lib/security/methods.cfg, as described in 4.2.5, “LDAP authentication module enablement” on page 111.

5. Test the client connection with the LDAP server.

   Before setting up users, encryption, or customizing the configuration, make sure that your LDAP client can talk to the server. This will make sure that you can retrieve information from the LDAP server and that you have correctly configured the DN and password. This is easily done with the **`lsldap`** and **`ldapsearch`** commands.

6. Enable a test user for LDAP authentication.

   If possible use a user that is already in the LDAP directory with a known password. To enable a test user such as user test1, change the SYSTEM and registry attributes for that user to LDAP in the /etc/security/user. If the user previously exists on the client, this can be done with the **chuser** command, as shown here:

   ```
   # chuser -R LDAP SYSTEM=LDAP registry=LDAP test1
   ```

   This will create the user stanza in /etc/security/user, but will not create a HOME directory for the user, so you can add that with:

   ```
   # mkdir /home/test1
   # chown test1 /home/test1
   # chmod 755 /home/test1
   ```

   Test to make sure this user can log in using the password stored on the LDAP server. The SYSTEM=LDAP and registry=LDAP will make sure that LDAP is used instead of local files.

7. Advanced configuration.

   Once you have confirmed that you have communications and can authenticate a single user you know that you have a working LDAP server. If it does not work, then you may need to do some advanced configuration of the /etc/security/ldap/ldap.cfg file using the information described in 4.2.3, “AIX 5L LDAP client daemon” on page 108. If you are using a nonstandard schema, you will also have to create custom map files, as described in 4.3.6, “Object class and attribute name mapping” on page 131.

8. Configure root and other administrative users for local file authentication.

   Before setting the default user stanza to use LDAP, set specific users to use local files for authentication. Make sure to include the root user in this list, and if you do not allow root login, make sure to include at least one user that can **su** to root. Once again, you can make this change with the **chuser** command, as shown here:

   ```
   # chuser registry=files SYSTEM=compat root
   ```

9. Make sure that users are populated in the LDAP directory.

   Before setting the default user stanza to use LDAP, make sure that you have users populated in LDAP. You can use **ldapsearch** to do this, but perhaps the easiest way is to simply use **lsuser**, as shown here.

   ```
   # lsuser -R LDAP -a id pgrp SYSTEM registry ALL
   ```

10. Set the default user stanza in /etc/security/user to use LDAP.

    Once you are sure that LDAP is configured properly and that you have populated users into the LDAP directory, then set the default user to use LDAP, as described in 4.2.6, “Update default stanza in /etc/security/user” on page 112. This will insure that you can log in to the AIX client with any user in the LDAP directory that is not restricted.

11. Test that you can log in using the users defined in LDAP.

    Once again, confirm that you can log in to the AIX client using one of the LDAP users. If that user was previously on the AIX client, you may need to change the user's SYSTEM and registry attribute to LDAP, or remove that user from the local files. If that user is not on the local system, you will get an error that no HOME directory exists.

12. Test that the AIX client user management commands work.

    If you have set up the proper permissions, you should be able to add users to the LDAP directory using the **mkuser** command, change user information with the **chuser** command, and change the user's password with the AIX client **passwd** command. This is described in 6.4, "Working with the AIX 5L clients" on page 204.

13. Configure SSL for security between the client and server.

    Once you have a working configuration, it is important that you make it secure. The easiest way to do this is by setting up SSL between the client and the server. This is described in 4.3.1, "Configuring SSL" on page 113. One other method of security that you may want to configure is an initial bind by the client to the server using Kerberos.

14. Set up user access restriction.

    Without restrictions, any user in the LDAP server can log in to the AIX client. For this reason, it is important to set up some type of user restrictions, as described in 3.5.6, "User login restriction" on page 91. Details for restricting using NIS netgroups are described in 4.3.4, "Restricting user access using netgroups" on page 123.

15. Set up methods to handle the user's HOME directories.

    When the default user is set to LDAP, there is not an automatic method to create users the first time someone logs in. There are a number of ways to create the user's home directory, and in this chapter we describe using automount in Example 4.3.5 on page 127

16. Tune the LDAP client.

    Once you have everything working, tune the secldapclntd daemon caching parameters, as described in 4.3.3, "AIX 5L LDAP client tuning" on page 122.

Follow the steps in the rest of this chapter to set up your first client.

4.2 Installation and basic client configuration

In this section we discuss the required configuration for an AIX 5L client to use the LDAP directory for name service resolution. The topics we discuss include installation of the software and initial configuration of the AIX 5L LDAP client, including the options available in the configuration file and available to the secldapclntd daemon.

4.2.1 AIX 5L LDAP client software installation

In order to use the AIX 5L LDAP client, you must install the required Licensed Product Packages (LPPs) from the product media. The base LDAP client is packaged in the ldap.client file sets located on the AIX 5L product media, and if you require SSL to connect to the LDAP server, you must install the gskta.rte and ldap.max_crypto_client file sets located on the AIX 5L Expansion Pack. Use the **installp** command, SMIT, or Web-based System Manager to install the required LPPs.

Example 4-1 shows the installation of the LDAP client with SSL support. Replace the LPPSOURCE tag in the following commands with the correct location of your LPPs in your environment.

Example 4-1 Installation of LDAP client with SSL support

```
# installp -acgXd LPPSOURCE ldap.client
<excess output removed>
# installp -acgXd LPPSOURCE gskta ldap.max_crypto_client
<excess output removed>
# lslpp -l "gsk*" "ldap*"
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  gskta.rte                 7.0.1.16  COMMITTED  AIX Certificate and SSL Base
                                                 Runtime ACME Toolkit
  ldap.client.adt            5.2.0.0  COMMITTED  Directory Client SDK
  ldap.client.rte            5.2.0.0  COMMITTED  Directory Client Runtime (No
                                                 SSL)
  ldap.max_crypto_client.adt
                             5.2.0.0  COMMITTED  Directory Client SDK
  ldap.max_crypto_client.rte
                             5.2.0.0  COMMITTED  Directory Client Runtime (SSL)

Path: /etc/objrepos
  ldap.client.rte            5.2.0.0  COMMITTED  Directory Client Runtime (No
                                                 SSL)
```

4.2.2 AIX 5L LDAP client configuration

The **mksecldap** command with the -c option is used to set up the AIX 5L LDAP client. The following is a sequential summary of what the command can accomplish when used to set up a client:

1. Sets up the configuration in the /etc/security/ldap/ldap.cfg file:
   - Host names or IP addresses of the LDAP servers
   - Base DNs to use from the LDAP servers for users, groups, and (if they exist) hosts, networks, services, netgroups, protocols, and rpc
   - Mappings to use to map the system's required attributes to the schema on the LDAP server
   - SSL setup
   - Stores the encrypted bind password in the LDAP configuration file
   - Saves the LDAP server bind DN and password
   - Saves the caching daemons and other options given on the command line to the configuration file
2. If requested, it modifies the /etc/security/user file to enable a list of users to log in.
3. Starts the client daemon process.
4. Adds the client daemon process to /etc/inittab to have this daemon start after a reboot.
5. Adds an entry for LDAP to the /usr/lib/security/methods.cfg.

In this chapter there are tables that list the ldap.cfg options. If this option is able to be configured using the **mksecldap** command, then the name of the command-line option is listed in parentheses.

The **mksecldap** command has the following syntax:

```
Usage: mksecldap -c -h serverlist -a bindDN -p bindPasswd
        [-d baseDN] [-n serverport] [-k SSLkeypath] [-w SSLkeypasswd]
        [-t cachetimeout] [-C cachesize] [-P NumberOfThread] [-u userlist]
        [-T heartBeat] [-M searchMode] [-D defaultEntry] [-A AuthType] [-U]
```

An explanation of the minimum command parameter options required for setup of the client (for the options not mentioned see the tables below) is:

-c	Specifies to set up a LDAP client and not the server
-h	Specifies the list of servers
-a	Specifies the bind DN to use
-p	Specifies the bind password
-d	Base DN to use for the client

Note: The -U switch is the undo operation. During the configuration with **mksecldap** a backup file of the /etc/security/ldap/ldap.cfg file will be made. With the following command that file will be restored:

```
mksecldap -c -U
```

These are minimum parameters to use for **mksecldap**. This chapter explains all parameters that are adjustable for the AIX 5L client. Not all of them are able to be configured though the **mksecldap** command. Therefore these parameters are mentioned in parentheses, if there is a command-line option flag for the **mksecldap** command.

Example 4-2 shows the use of **mksecldap** with the minimum set of parameters to configure a client.

Example 4-2 mksecldap -c

```
# mksecldap -c -h bs01b02 -a cn=admin -p its0g00d -d "dc=example,dc=com"
# ls-secldapclntd
ldapservers=bs01b02
ldapport=389
ldapversion=3
userbasedn=ou=People,dc=example,dc=com
groupbasedn=ou=Groups,dc=example,dc=com
idbasedn=cn=aixid,ou=System,dc=example,dc=com
usercachesize=1000
usercacheused=0
groupcachesize=100
groupcacheused=0
cachetimeout=300
heartbeatT=300
numberofthread=10
connectionsperserver=10
alwaysmaster=no
authtype=UNIX_AUTH
searchmode=ALL
defaultentrylocation=LDAP
ldaptimeout=60
userobjectclass=account,posixaccount,shadowaccount,aixauxaccount,ibm-securityId
entities
groupobjectclass=posixgroup,aixauxgroup
```

4.2.3 AIX 5L LDAP client daemon

The secldapclntd daemon is required for the client. It should be started during system boot.

The **mksecldap** command will update the /etc/inittab automatically to contain the following entry:

```
ldapclntd:2:once: /usr/sbin/secldapclntd > /dev/console 2>&1
```

The settings can be grouped in three sections:

- Table 4-1 shows the settings that are affecting the caching behavior.
- Table 4-2 on page 109 shows the setting that defined the server and all bind-related information.
- Table 4-3 on page 110 shows the settings that are related to the LDAP schemas and the attribute mapping.

The caching options can be left as they are if no special requirements have to be met. For more information about changing the options see the 4.3.3, "AIX 5L LDAP client tuning" on page 122.

Table 4-1 Caching options and other settings for the LDAP client daemon

Setting	Default	Range	Description
cachetimeout (-t)	300	60–3600	This defines the time for which a cache entry is valid.
usercachesize (-C)	1000	100–10000	The user cache size specifies the number of user entries to cache.
groupcache	100	10–1000	The group cache defines the number of entries of the group cache. With **mksecldap** it will be 10% of the value specified for the usercache.
numberofthread (-P)	10	1–1000	This is the number of threads the secldapclntd daemon uses to process jobs.
connectionsperserver	10	1–100	This specifies the number of connections to the LDAP server. If multiple servers are used, then this number of connections will be established to each of the servers.
heartbeatinterval (-T)	300	60–3600	This is the time interval in seconds that the secldapclntd daemon contacts the LDAP server for server status.

Setting	Default	Range	Description
searchmode (-M)	ALL	ALL or OS	This defines whether the daemon only asks the server for all attributes of a certain user (ALL) or if it only asks for the values that are needed for the AIX 5L operating system (OS).
defaultentrylocation (-D)	LDAP	LDAP or local	This defines which default user entry will be used. If local is specified, the default entry defined in /etc/security/user will be used.
ldaptimeout	60	0–3600	This is the time period the client will wait for a server response. The value 0 means no time out.

Multiple LDAP servers should be used to provide failover capability. If the primary LDAP server becomes unavailable, the client would automatically fail over to the secondary server. In an enterprise environment, the LDAP clients should be distributed among different LDAP servers to distribute LDAP workload among the LDAP servers.

The bind options are explained in more detail in 4.3.2, “LDAP client daemon bind distinguished name (DN) options” on page 120.

Table 4-2 LDAP server options for secldapclntd

Setting	Default	Description
ldapservers (-h)	N/A	This specifies the list of servers to use. The list is prioritized, which means that if the first server in the list is available it will be used.
binddn (-a)	N/A	This specifies the bind DN to connect to the LDAP server.
bindpw (-p)	N/A	This specifies the password that is used with the bind DN.
auth_type (-A)	unix_auth	The authentication type specifies whether the secldapclntd will retrieve the encrypted password from the server (unix_auth) or if it will send the password to the server for comparison (ldap_auth).
ldapport (-n)	389	This specifies the LDAP server port to use.
ldapsslport (-n)	636	This specifies the LDAP server port to use with SSL.
useSSL	No	This specifies whether SSL should be used.
ldapsslkeyf	N/A	This specifies the key database to use with SSL. Note that the key database must have been created with GSKit. Also note that even if you use server-side only certificates you must have a key database that contains the CA key for your server.

Setting	Default	Description
ldapsslkeypwd	N/A	This specifies the password to use for the SSL key file.
useKRB5	N/A	This specifies whether Kerberos should be used for binding to the directory.
krbprincipal	N/A	This is the Kerberos principal used to bind to the server.
krbkeypath	/etc/security/ldap/krb5.keytab	This is the path to the Kerberos keytab.
krbcmddir	/usr/krb5/bin/	This is the directory that contains the Kerberos **kinit** command.
ldapversion	3	This specifies which version of the LDAP protocol should be used. Possible values are 2 and 3.
followaliase	NEVER	This defines in which way aliases (referrals) should be treated. Possible values are NEVER, SEARCHING, FINDING, and ALWAYS.

To define which part of the LDAP directory is used to store the user and group information, the userbasedn and the groupbasedn is used. The other DN-related entries define all other RFC2307 schema-related information.

The userattrmappath, groupattrmappath, and idattrmappath define which attribute mapping file will be used according to the schema that is used on the LDAP server.

Table 4-3 LDAP configuration options for secldapclntd

Setting	Default	Description
userattrmappath	/etc/security/ldap/aixuser.map	These three values specify the locations of the files for attribute mapping.
groupattrmappath	/etc/security/ldap/aixgroup.map	
idattrmappath	/etc/security/ldap/aixid.map	

Setting	Default	Description
userbasedn groupbasedn idbasedn hostbasedn servicebasedn protocolbasedn networkbasedn netgroupbasedn rpcbasedn automountbasedn aliasbasedn bootparambasedn etherbasedn	N/A	These entries specify the location on the LDAP server for certain information. These DNs represent the maps used with the NIS schemas. For a minimum operation of an AIX 5L system only the userbasedn and the groupbasedn must be specified.
userclasses	N/A	The userclasses specifies which object class a newly created LDAP object will belong to. These values will be used by the **`mkuser`** command.
groupclasses	N/A	The groupclasses specifies which object class a newly created LDAP object will belong to. These values will be used by the **`mkgroup`** command.

4.2.4 LDAP authentication

Loadable authentication module (LAM) is the security mechanism that is used in AIX 5L to control user authentication. The LAM framework enables AIX systems to use various name services for user and group resolution.

The support of authentication based on RFC2307 LDAP schema is available since AIX 5L Version 5.2. The system administrator can configure AIX 5L systems to use LDAP for user and group resolution. This section describes the required configuration to enable LDAP authentication.

4.2.5 LDAP authentication module enablement

Add the LDAP stanza to /usr/lib/security/methods.cfg. The methods.cfg file contains stanzas defining available authentication module information. Each stanza is identified by a module name followed by a colon (:) and contains attributes in the form attribute=value. Example 4-3 shows the stanza for the LDAP authentication module.

Example 4-3 LDAP stanza in methods.cfg

```
LDAP:
        program = /usr/lib/security/LDAP
        program_64 =/usr/lib/security/LDAP64
```

Note: If compat mode is used, then “options = netgroup” should also be added to the LDAP stanza in the methods.cfg file. More details about compat are available in 4.3.4, “Restricting user access using netgroups” on page 123.

4.2.6 Update default stanza in /etc/security/user

The SYSTEM and registry attributes in the /etc/security/user are used to specify user the authentication method and the database used for user management. To enable LDAP authentication and user management, set the SYSTEM and registry attributes in the default stanza to LDAP. This can be done with the following commands:

```
chsec -f /etc/security/user -s default -a "SYSTEM=LDAP or files"
chsec -f /etc/security/user -s default -a "registry=LDAP"
```

Example 4-4 shows the default stanza in the /etc/security/user file with the SYSTEM and registry attributes set to LDAP.

Example 4-4 Default stanza in /etc/security/user to enable LDAP

```
default:
        admin = false
        login = true
        su = true
        daemon = true
        rlogin = true
        sugroups = ALL
        admgroups =
        ttys = ALL
        auth1 = SYSTEM
        auth2 = NONE
        tpath = nosak
        umask = 022
        expires = 0
        SYSTEM = LDAP
        registry = LDAP
        logintimes =
        pwdwarntime = 0
        account_locked = false
```

```
loginretries = 0
histexpire = 0
histsize = 0
minage = 0
maxage = 0
maxexpired = -1
minalpha = 0
minother = 0
minlen = 0
mindiff = 0
maxrepeats = 8
dictionlist =
pwdchecks =
```

4.3 Advanced LDAP client configuration

The following sections discuss several advanced LDAP client configuration topics:

- SSL connection configuration between client and server
- Kerberos bind
- User access restriction with netgroups
- Base DN bind configuration
- Automount configuration to use LDAP
- LDAP client tuning
- Object class and attribute name mapping

4.3.1 Configuring SSL

This section discusses the AIX 5L LDAP client configuration to use SSL to secure the connection to the server. For more information about why you would use SSL please refer to 3.1.8, “Using SSL to secure LDAP connection” on page 59.

Each of the three scenarios covered in this book has a section discussing the steps to configure SSL in that scenario:

- For more information about SSL in the Sun ONE Directory scenario refer to 5.6, “SSL configuration” on page 178.
- For more information about using SSL in an IBM Tivoli Directory Server environment refer to 6.3.3, “Step 3: Creating a key database to use with SSL clients” on page 195.
- For more information about SSL in the Microsoft Active Directory scenario refer to 8.2.4, “Configuring LDAP to use SSL” on page 247.

Software installation requirements

In order to use SSL with the AIX 5L LDAP client, you must install the SSL-enabled LDAP client packages certificates. Below we list the required packages SSL supports for the AIX 5L LDAP client:

- ldap.max_crypto_client
- gsksa.rte
- gskta.rte

For more information about installing the required packages refer to 4.2.1, “AIX 5L LDAP client software installation” on page 105.

Configuring LDAP client to work with server-side certificates

To get AIX 5L to work with an SSL-enabled server the client must have the ability to verify the correctness of the certificate of the server. Therefore the client must have the CA certificate installed in his key ring and it must be flagged as a trusted certificate.

The examples in this section only cover the command-line tool. A detailed explanation of the usage of the GUI can be found in 5.6, “SSL configuration” on page 178.

For more information about IBM Global Security Toolkit, please visit the following location for the product documentation:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itame2.doc_5.1/ss7aumst.htm

Generating a key database

To generate a key database, the command line tool gsk7cmd will be used with the following parameters:

-keydb	This defines the type of operation.
-create	The action is to create a new key database.
-db filename	This is the file name of the key database (key.kdb).
-pw password	This specifies the password to encrypt the key ring file (its0g00d).
-type type	This defines the type of the key database (CMS).

The command will return without any response. To verify that the file is there and usable, list the CA certificates that are included by default with the following command:

```
gsk7cmd -cert -list CA -db <key ring> -pw <password>
```

This step is shown in Example 4-5.

Example 4-5 Creating a key ring file

```
# gsk7cmd -keydb -create -db /etc/security/ldap/key.kdb -pw its0g00d -type cms
# gsk7cmd -cert -list CA -db /etc/security/ldap/key.kdb -pw its0g00d
Certificates in database: /home/root/key.kdb
   Entrust.net Global Secure Server Certification Authority
   Entrust.net Global Client Certification Authority
   Entrust.net Client Certification Authority
   Entrust.net Certification Authority (2048)
   Entrust.net Secure Server Certification Authority
   VeriSign Class 3 Public Primary Certification Authority
   VeriSign Class 2 Public Primary Certification Authority
   VeriSign Class 1 Public Primary Certification Authority
   VeriSign Class 4 Public Primary Certification Authority - G2
   VeriSign Class 3 Public Primary Certification Authority - G2
   VeriSign Class 2 Public Primary Certification Authority - G2
   VeriSign Class 1 Public Primary Certification Authority - G2
   VeriSign Class 4 Public Primary Certification Authority - G3
   VeriSign Class 3 Public Primary Certification Authority - G3
   VeriSign Class 2 Public Primary Certification Authority - G3
   VeriSign Class 1 Public Primary Certification Authority - G3
   Thawte Personal Premium CA
   Thawte Personal Freemail CA
   Thawte Personal Basic CA
   Thawte Premium Server CA
   Thawte Server CA
   RSA Secure Server Certification Authority
#
```

Adding a CA certificate to the key database

In order to add the CA's certificate to the key database, GSKit requires that all human-readable text be removed from the certificate file. The final certificate file should only contain the cipher text between the begin and end certificate markers.

- -cert

  This specifies to perform a certificate operation.
- -add

  This specifies to add something.
- -db <filename>

  This is the file name of the key ring file (key.kdb).
- -file <filename>

  This is the file containing the CA certificate (cacert.pem).

- -format <ascii | binary>

  This is the format of the CA certificate. Here the ASCII format is used.

- -label <label>

  This specifies a label that is used for that certificate (example.com CA Certificate).

- -pw <password>

  This specifies the password for the key ring (its0g00d).

- -trust <enable | disable>

  This specifies whether the certificate should be trusted. Enable must be used here to get a working key ring file (enable).

This command will also return with no output. To verify the CA certificate use the following commands (as shown in Example 4-6):

```
gsk7cmd -cert -list CA -db <key ring> -pw <password>
gsk7cmd -cert -details -showOID -db <key ring> -pw <password> -label <LABEL>
```

Example 4-6 Adding a CA certificate to a key database

```
# gsk7cmd -cert -add -db /etc/security/ldap/key.kdb -file cacert.pem -format
ascii -label "Example.com CA Certificate" -pw its0g00d -trust enable
#  gsk7cmd  -cert -list CA  -db /etc/security/ldap/key.kdb -pw its0g00d
Certificates in database: /home/root/key.kdb
   Example.com CA Certificate
   Entrust.net Global Secure Server Certification Authority
   .
   .
   .
   RSA Secure Server Certification Authority
# gsk7cmd  -cert -details -showOID   -db /home/root/key.kdb -pw its0g00d -label
"Example.com CA Certificate"



Key Label: Example.com CA Certificate
  TBS Certificate
    Version: X509 V3
    Serial Number: 00
    Issuer
      Country or region
        Type: 2.5.4.6
        Value: us
      State/Province
        Type: 2.5.4.8
```

```
.
.
.
```

Enable SSL in ldap.cfg

In the ldap.cfg file the configuration for the use of SSL and for the key database file must be made.

The configuration for ldap.cfg can be done by the following options of **`mksecldap`**:

-k SSLkeypath	Specifies the file name with the complete path for the key ring file.
-w SSLkeypasswd	Specifies the password for the key ring file.
-n serverport	Specifies the server port to use. This is optional. The default is 636 for SSL.

The options affecting the SSL configuration are listed in Table 4-2 on page 109. The configuration and the resulting part of the ldap.cfg file are shown in Example 4-7.

Example 4-7 SSL configuration

```
# mksecldap -c -h bs01b02 -a "cn=admin" -p its0g00d -d dc=example,dc=com  -k
/etc/security/ldap/key.kdb -w its0g00d
# cat /etc/security/ldap/ldap.cfg
   ...
useSSL:yes
ldapsslkeyf:/etc/security/ldap/key.kdb
ldapsslkeypwd:{DES}78FDE8A9C458261715 F78C82555B9903247B1 9637669A0
ldapport:389
ldapsslport:636
   ...
```

Configuring LDAP client to work with client-side certificates

Up to now only the client is able to verify the identity of the server. This is the normal implementation and the most widely used.

For certain applications it might be necessary for the server to be able to verify the identity of the client. Therefore we can deploy a client-side certificate that allows the server to verify the identity of the client.

Note: There is no change necessary on the LDAP client itself. The only difference is that the key ring has a key and a certificate that can be used.

To do this we have to:

1. Create a public/private key pair and a certificate signing request.
2. Get the certificate signed by our CA (not explained here).
3. Import our certificate and set it as the default certificate.

Create a public/private key pair and a certificate signing request

To create the key pair and the certificate signing request, **`gsk7cmd`** with the following parameters is used:

-certreq	This specifies to perform a certificate request operation.
-create	This specifies to create a new certificate signing request.
-db <filename>	This is the file name of the key ring file (key.kdb).
-pw <password>	This specifies the password for the key ring (its0g00d).
-file <filename>	This is the file containing the CA certificate (cacert.pem).
-label <label>	This specifies a label that is used for the certificate signing request now and for the certificate later (bs01b13).
-dn <DN>	This specifies the distinguished name for the certificate and the request ("C=us, ST=TX, L=Austin, O=example.com, CN=bs01b13").
-size <key size>	This specifies the size of the key to use (1024).
-file <filename>	This specifies the file name to store the output for the certificate request (bs01b13.req).

This command does not produce usable output on the command line. To verify whether the command was successful, the following commands can be issued:

```
gsk7cmd -certreq -list -db <filename> -pw <password>
gsk7cmd -certreq -details -label <label> -db <filename> -pw <password>
```

Also, the output file can be examined. The output of these two commands is shown in Example 4-8.

Example 4-8 Creating a certificate signing request

```
# gsk7cmd -certreq -create  -db key.kdb -pw its0g00d -label "bs01b13" -dn
"C=us, ST=TX, L=Austin, O=example.com, CN=bs01b13" -size 1024 -file bs01b13.req
gsk7cmd -certreq -list -db key.kdb -pw its0g00d
Certificate requests in database: key.kdb
bs01b13
# gsk7cmd -certreq -details -label "bs01b13" -db key.kdb -pw its0g00d

Label: bs01b13
```

```
Key Size: 1024
Subject: bs01b13
example.com
Austin, TX, us
Fingerprint: 2A:F9:68:2F:ED:57:AA:76:13:2A:0D:27:07:22:3F:BD
Signature Algorithm: 1.2.840.113549.1.1.4

#
```

Afterwards the certification request in the file has to be sent to a CA for signing. The CA will return a certificate that can be imported to the key ring.

Import our certificate and set it as default certificate

The certificate that the CA returned has to be in a format without additional text. The import will be done with the **gsk7cmd** with the following parameters:

- -cert

  This specifies to perform a certificate operation.
- -receive

  This specifies that a certificate for a existing key will be received.
- -db <filename>

  This is the file name of the key ring file (key.kdb).
- -pw <password>

  This specifies the password for the key ring (its0g00d).
- -file <filename>

  This is the file containing the CA certificate (cacert.pem).
- -format <ascii | binary>

  This is the format of the file in which our certificate is stored. Here the ASCII format is used.
- -default_cert <yes | no>

  This specifies whether the certificate and the key especially should be used as the default. Yes must be used here to get a working key ring file for LDAP client.

This command does not produce usable output on the command line to verify whether the command was successful. The following commands can be issued:

```
gsk7cmd -cert -getdefault -db <filename> -pw <password>
```

The result and therefore the information from the certificate are shown in Example 4-9.

Example 4-9 Importing a certificate to the key database

```
# gsk7cmd -cert -receive -file bs01b13.cert  -db key.kdb -pw its0g00d -format
ascii  -default_cert yes
# gsk7cmd -cert -getdefault -db key.kdb -pw its0g00d

Label: bs01b13
Key Size: 1024
Version: X509 V3
Serial Number: 06
Issued By: CA
example.com
Austin, TX, us
Subject: bs01b13
example.com
Austin, TX, us
Valid From: Monday, October 3, 2005 5:07:41 PM CDT To: Tuesday, October 3, 2006
5:07:41 PM CDT
Fingerprint: 5D:D3:35:74:5C:38:96:A4:D1:D5:26:03:A1:AF:2A:97:BC:79:6B:20
Signature Algorithm: 1.2.840.113549.1.1.4
Trust Status: enabled
```

The installation of the client certificate is now complete.

4.3.2 LDAP client daemon bind distinguished name (DN) options

LDAP clients are required to bind to the LDAP server in order to retrieve directory information. The LDAP bind operation is similar to a user logging into a host. The LDAP server would respond to the client request based on the access right that is defined for the user.

The bind distinguished name (bind DN) is a LDAP user that has specific access rights to the LDAP database. Normally, the bind DN that is used in an AIX 5L client should have read access to all RFC2307 LDAP data. In some cases, the bind DN might also require write access to certain attributes in the directory, for example, user password. If RFC2307AIX schema is used, the client may also require write access to AIX-specific attributes. Some careful planning is required to determine the proper access rights of the bind DN.

The AIX 5L LDAP client supports the following bind operation:

- ► Anonymous bind
- ► Specific DN bind
- ► Kerberos bind

Simple bind

The client binds to the LDAP server as a particular DN. The bind DN as well as the clear text password is required to send to the server for authentication. In the AIX 5L client, the bind DN and password are recorded in the /etc/security/ldap/ldap.cfg file. The **mksecldap** command creates the binddn and bindpwd fields in the ldap.cfg.

Prior to AIX 5L Version 5.3 RML3, the client stores the LDAP bind DN password in clear text, as shown in Example 4-10 on page 121. In AIX 5L, the **mksecldap** command would store the bind DN password in encrypted format. See Example 4-11. The AIX 5L client can also read the clear text password to provide backward compatibility.

Example 4-10 Bind DN and password in clear text

```
# LDAP server bind distinguished name (DN)
binddn:cn=proxy,ou=profile,dc=example,dc=com

# LDAP server bind DN password
bindpwd:Password
```

Example 4-11 Binddn and DES encrypted bind DN password

```
# LDAP server bind distinguished name (DN)
binddn:cn=proxy,ou=profile,dc=example,dc=com

# LDAP server bind DN password
bindpwd:{DES}EBFA4D3D3D41F9 653B9F71B58CBEA103DFF78FE348B7EDF
```

In an open network environment in which transmitting a clear text password is a concern, the LDAP bind operation should be encrypted. The AIX 5L native LDAP client supports SSL and can be used to secure the LDAP server/client communication. For more details about configuring SSL, refer to 5.6, “SSL configuration” on page 178.

Kerberos bind

Starting with AIX 5L Version 5.3, you can now configure the LDAP client daemon to bind to a LDAP server using Kerberos. The Kerberos bind allows the LDAP client daemon to authenticate to the KDC instead of binding with a DN and password. Since all user and LDAP client daemon authentication is handled by Kerberos, no passwords are sent in clear text across network. Depending on your data confidentiality policy, this might remove the need to deploy LDAP over SSL. If you are already using Kerberos and LDAP, there is no reason you should not use this feature, as you already have a host principal and keytab generated for each Kerberos client.

Example 4-12 on page 122 shows the configuration to be added to the LDAP client daemon configuration file to enable Kerberos bind. The settings are as follows:

useKRB5	This setting makes the LDAP client daemon bind to the Active Directory using Kerberos. You must set this to yes to enable Kerberos bind.
krbprincipal	This setting specifies the Kerberos principal to authenticate as when accessing the LDAP directory.
krbkeypath	This setting specifies the location of the keytab file for the krbprincipal.
krbcmddir	This setting specifies the location of the **kinit** command. The default value is correct and should not be modified.

Example 4-12 Kerberos bind configuration in ldap.cfg

```
# Whether to use Kerberos for initial bind to the server.
# useKRB5       - valid value is either "yes" or "no". Default is "no".
# krbprincipal  - Kerberos principal used to bind to the server.
# krbkeypath    - Path to the kerberos keytab, default to
#                 /etc/security/ldap/krb5.keytab
# krbcmddir     - Directory that contains the Kerberos kinit command.
#                 Default to /usr/krb5/bin/.
useKRB5:yes
krbprincipal:principal
krbkeypath:/etc/security/ldap/krb5.keytab
krbcmddir:/usr/krb5/bin/
```

Anonymous bind

The client does not bind as a particular user, nor is the password supplied in the bind operation. Normally, the LDAP server would allow read access to some data in the directory for anonymous bind. The client should have no write access to the LDAP directory and should not have any read access to sensitive information on the directory.

4.3.3 AIX 5L LDAP client tuning

The secldapclntd daemon controls how clients process LDAP requests and also manages a cache for recently used LDAP data. Tuning parameters are provided to adjust the overall performance impact of these activities on the client system. In an environment that consists of many LDAP clients, it is important to tune these parameters in order to ensure optimal performance. Below is a list of a few of the key tunable variables that merit consideration:

- connectionsperserver

  This parameter controls the number of static server connections that the client maintains. The default is 10 connections per server. In most cases, the client does not fully utilize all 10 connections, so it may be desirable to reduce the number of connections so the client will not occupy unnecessary server connections.

  Normally, the connections per server settings should be less than or equal to the number of thread. In most environments, two or three server connections should be sufficient.

- usercachesize

  This setting determines the number of LDAP user entries that can be cached. The LDAP cache improves client performance since it reduces the number of LDAP lookups made to the server. The default setting for the user cache is 1000 entries. Increase the user cache size if you have more than one thousand LDAP users in your directory that might log in to the host. The maximum is 10000.

- groupcachesize

  This setting determines the number of LDAP groups that can be cached. The default is 100. The maximum group cache size is 1000.

- cachetimeout

  This setting allows users to control the frequency at which secldapclntd flushes the data cache. A long cachetimeout value means that the client does not have to make LDAP requests as often. On the other hand, a long time-out value also means that the client might use old data until the cached entry expires. The user should tune the time-out setting to find the balance between performance and integrity of data.

4.3.4 Restricting user access using netgroups

The password compatibility mode can be used to restrict user access based on the configuration defined in the /etc/passwd file. When compat mode is enabled, only users that are defined locally in the passwd file have access to the system. System administrators can specify LDAP users, LDAP netgroups, or both in the passwd file to allow LDAP users logins to the system. The syntax “+USER” would allow a LDAP user “USER” login access, while “+@NETGROUP” would allow users defined in a LDAP netgroup “NETGROUP” login access to the system

The compat mode also supports “-USER” and “-@NETGROUP” syntax. In those cases, the specified user or netgroup would not be allowed to log in to the system.

The following should be done to enable passwd compatibility mode. More details on configuring passwd compat are given in 5.4, “Restriction of user login with compat mode (netgroups)” on page 164.

Step 1: Update the default stanza in /etc/security/user

Change the SYSTEM and registry default settings in the /etc/security/user file to compat, as shown in Example 4-13.

Example 4-13 Default stanza in /etc/security/user for compat

```
default:
        admin = false
        login = true
        su = true
        daemon = true
        rlogin = true
        sugroups = ALL
        admgroups =
        ttys = ALL
        auth1 = SYSTEM
        auth2 = NONE
        tpath = nosak
        umask = 022
        expires = 0
        SYSTEM = "compat"
        registry = compat
        logintimes =
        pwdwarntime = 0
        account_locked = false
        loginretries = 0
        histexpire = 0
        histsize = 0
        minage = 0
        maxage = 0
        maxexpired = -1
        minalpha = 0
        minother = 0
        minlen = 0
        mindiff = 0
        maxrepeats = 8
        dictionlist =
        pwdchecks =
```

Step 2: Enable LDAP netgroup support

To enable netgroup support, you must add “options = netgroup” to the LDAP stanza in the /usr/lib/security/methods.cfg file. Example 4-14 shows the modified LDAP stanza.

Example 4-14 Enabling netgroup support in LDAP stanza in methods.cfg

```
LDAP:
        program = /usr/lib/security/LDAP
        program_64 =/usr/lib/security/LDAP64
        options = netgroup
```

Step 3: Enable compat mode resolution for LDAP groups

Append a plus sign (+) (on a line by itself) to the end of the /etc/group file. The plus sign (+) line enables the client to use the LDAP directory for group resolution. Otherwise, the client would only use and local group file and any LDAP group resolution would fail.

Example 4-15 shows the /etc/group file with the LDAP group enabled.

Example 4-15 Modified /etc/group for compat mode

```
system:!:0:root
staff:!:1:ipsec,sshd,ldap
bin:!:2:root,bin
sys:!:3:root,bin,sys
adm:!:4:bin,adm
uucp:!:5:uucp,nuucp
mail:!:6:
security:!:7:root
cron:!:8:root
printq:!:9:lp
audit:!:10:root
ecs:!:28:
nobody:!:4294967294:nobody,lpd
usr:!:100:guest
perf:!:20:
shutdown:!:21:
lp:!:11:root,lp,printq
snapp:!:12:snapp
invscout:!:13:invscout
ipsec:!:200:
sshd:!:201:sshd
ldap:!:202:ldap
+
```

Step 4: Modify /etc/irs.conf

Append `netgroup nis_ldap` to the /etc/irs.conf file to enable the AIX 5L client to use LDAP to lookup netgroup information.

Example 4-16 /etc/irs.conf with netgroup lookup from the LDAP server

```
# grep netgroup /etc/irs.conf
netgroup nis_ldap
```

Step 5: Configure /etc/passwd to control login access

Modify the /etc/passwd file by adding desired LDAP users and netgroups to the end. Add authorized LDAP users with the syntax +username. Add authorized netgroups with the syntax +@netgroupname.

Example 4-17 Modified /etc/passwd with +@netgroup and +user

```
root:!:0:0::/home/root:/usr/bin/ksh
daemon:!:1:1::/etc:
bin:!:2:2::/bin:
sys:!:3:3::/usr/sys:
adm:!:4:4::/var/adm:
uucp:!:5:5::/usr/lib/uucp:
guest:!:100:100::/home/guest:
nobody:!:4294967294:4294967294::/:
lpd:!:9:4294967294::/:
lp:*:11:11::/var/spool/lp:/bin/false
snapp:*:200:12:snapp login user:/usr/sbin/snapp:/usr/sbin/snappd
nuucp:*:6:5:uucp login user:/var/spool/uucppublic:/usr/sbin/uucp/uucico
invscout:*:7:13::/var/adm/invscout:/usr/bin/ksh
ipsec:*:201:1::/etc/ipsec:/usr/bin/ksh
sshd:*:202:201::/var/empty:/usr/bin/ksh
ldap:*:203:1::/home/ldap:/usr/bin/ksh
fred:*:204:1::/home/fred:/usr/bin/ksh
someone:*:205:1::/home/someone:/usr/bin/ksh
+@itso_users
+user9
```

The /etc/passwd shown in Example 4-17 has two plus sign (+) entries. The "+@itso_users" allows all LDAP users defined in "itso_users" netgroup login access. The "+user9" allows LDAP user "user9" login.

The passwd compatibility also supports "-" syntax. The -username and -@netgroupname can be added to the passwd file to deny login access for particular LDAP users defined in an LDAP netgroup. In Example 4-18 on page 127, the system would deny logins for "user9" and users defined in the “itso_users” netgroup. The plus sign (+) line at the end of the passwd file allows all other LDAP users to log in.

Example 4-18 Modified /etc/passwd file with -@netgroup and -user

```
# cat /etc/passwd
root:!:0:0::/root:/usr/bin/ksh
daemon:!:1:1::/etc:
bin:!:2:2::/bin:
sys:!:3:3::/usr/sys:
adm:!:4:4::/var/adm:
uucp:!:5:5::/usr/lib/uucp:
guest:!:100:100::/home/guest:
nobody:!:4294967294:4294967294::/:
lpd:!:9:4294967294::/:
lp:*:11:11::/var/spool/lp:/bin/false
snapp:*:200:12:snapp login user:/usr/sbin/snapp:/usr/sbin/snappd
nuucp:*:6:5:uucp login user:/var/spool/uucppublic:/usr/sbin/uucp/uucico
invscout:*:7:13::/var/adm/invscout:/usr/bin/ksh
ipsec:*:201:1::/etc/ipsec:/usr/bin/ksh
sshd:*:202:201::/var/empty:/usr/bin/ksh
ldap:*:203:1::/home/ldap:/usr/bin/ksh
-@itso_users
-user9
+::0:0:::
```

4.3.5 Configuring LDAP automounter

The automount daemon automatically mounts specific file systems when accessed and unmounts them when they are no longer needed. For example, if a user logs into a machine, the automounter could automatically mount his home directory. When the user logs off, his home directory will be automatically unmounted after a predetermined inactivity time. The main advantage of using automounter is that the automounter maps can be managed centrally either in NIS or in LDAP. Starting in AIX 5L Version 5.3, the automount daemon can retrieve its automount maps from LDAP. The automounter supports both the RFC2307 and RFC2307bis automount schemas.

Automounter map LDAP schemas

The AIX 5L automounter supports two different automount map schemas, the RFC2307 and RFC2307bis schemas.

RFC2307 did not have specific object classes for automounter maps, but did define object classes to handle generic NIS map data. The automounter maps can be stored in the nisMap and nisObject object classes.

The cn attribute stores the key of the automounter maps and the nisMapEntry stores the actual mount location for that particular key. Comma-separated mount options can also be included in the nisMapEntry attribute. For replicated file systems, multiple comma-separated servers can be specified:

```
nismapentry: [mount-options]srv1,srv2,srv3:pathname
```

Example 4-19 shows sample automount maps, using the nisMap and nisObject object classes defined in RFC2307.

Example 4-19 automount entries using nisMap and nisObject object classes

```
# automount maps with nismap/nisobject objectclass
dn: nismapname=auto_home,dc=example,dc=com
nismapname: auto_home
objectClass: top
objectClass: nisMap

dn: cn=userid,nismapname=auto_home,dc=example,dc=com
objectClass: nisObject
objectClass: top
nisMapName: auto_home
nisMapEntry: server:/homedirectory/userid
cn=userid
```

RFC2307bis defined two new object classes, automount and automountMap, to store automount maps. It is no longer necessary to use the generic nisMap and nisObject object classes.

Example 4-20 shows the same automount map data in the previous example in the automount and automountMap schema.

Example 4-20 automount entries in automountMap and automount object classes

```
# automount maps with automountMaps/automount objectclass
dn: automountMapName=auto_home,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto_home

dn: automountKey=userid,automountMapName=auto_home,dc=example,dc=com
objectClass: automount
objectClass: top
automountKey: userid
automountInformation: server:/homedirectory/userid
```

The AIX 5L Version 5.3 automountd daemon recognizes both nisMap/nisObject and automountMap/automount schemas. However, it expects that you will use one schema or the other, or both. You should not user one schema for some information and the other schema for different information. If you want to use both schemas make sure you populate all the data into both formats.

Automount configuration

The automount daemon is started by the /etc/rc.nfs script during system reboot, if the /etc/auto_master file exists. The automount daemon then searches for its mount points in the /etc/auto_master file and in the auto_mount map in NIS and LDAP. The automount daemon will then create the required autoFS mount points and start up **`automountd`**.

There are a few options that are configurable:

-i Interval	Specifies an interval, in seconds, that an automounted directory lives. automountd would automatically unmount the file system if the system is idle for the specified period of time.
-m	Specifies not to search NIS for automount maps.
-n	Specifies the nobrowse option.
-t Duration	Specifies a duration, in seconds, that an autofs unmount thread sleeps before it starts to work again.
-v	Displays on standard output verbose status and warning messages.

In order for automount to use LDAP to resolve automount maps, a automount entry should be created in /etc/irs.conf:

```
automount nis_ldap
```

The automount daemon also supports multiple name services specified in the irs.conf file. In that case, the search order is important. For example, if the automount setting is set to *files nis_ldap*, automount would search from local auto maps prior to searching from the LDAP directory:

```
automount files nis_ldap
```

In addition, automounter supports client-specific variables within the automount entry. For example, if $HOST is specified within the path name of a automount map, automount would expand the $HOST variable with the host name and use it as the mount location.

Example 4-21   Automount entry with HOST variable

```
# lsldap -a auto_home user8
dn: automountKey=user8,automountMapName=auto_home,dc=example,dc=com
objectClass: automount
```

```
objectClass: top
automountKey: user8
automountInformation: localhost:/var/tmp/$HOST/
#
# hostname
bs01b11.itsc.austin.ibm.com
#
# cd /home/user8
#
# df -k .
Filesystem    1024-blocks      Free %Used    Iused %Iused Mounted on
/var/tmp/bs01b11/      524288    514692    2%      354     1% /home/user8
#
```

In Example 4-21 on page 129, the path name of the auto_home map contains a automount variable $HOST, which would resolve to the host name when automount is initiated.

Other supported automount variables are:

HOST	The host name; output of **`uname -n`**.
OSNAME	OS name, the output of **`uname -s`**.
OSREL	OS release, the output of **`uname -r`**.
OSVERS	OS version, the output of **`uname -v`**.

Example 4-22 illustrates how to use variables in the automount maps.

Example 4-22 Automount entry with variables

```
# lsldap -a auto_home user7
dn: automountKey=user7,automountMapName=auto_home,dc=example,dc=com
objectClass: automount
objectClass: top
automountKey: user7
automountInformation: localhost:/var/tmp/${HOST}/${OSNAME}/${OSVERS}/${OSREL}
#
# hostname
bs01b11.itsc.austin.ibm.com
#
# uname -s
AIX
#
# uname -v
5
# uname -r
3
# cd /home/user7
#
# pwd
```

```
/home/user7
# df -k .
Filesystem    1024-blocks      Free %Used    Iused %Iused Mounted on
/var/tmp/bs01b11/AIX/5/3      524288    514664    2%      357     1%
/home/user7
```

Export file system by LDAP netgroups

In AIX 5L Version 5.3, host type netgroups can be used in the /etc/exports file to export file systems. The syntax is:

```
filesystem [options],access=netgroup1,netgroup2,netgroup3...
```

Example 4-23 illustrates exporting a local file system /test/home to netgroup itso_blades.

Example 4-23 Export a file system by netgroup

```
# cat /etc/exports
/test/home -rw,access=itso_blades
#
# exportfs -a
# exportfs
/test/home    -rw,access=itso_blades
#
# lsldap -a netgroup itso_blades
dn: cn=itso_blades,ou=netgroup,dc=example,dc=com
cn: itso_blades
objectClass: top
objectClass: nisNetGroup
nisNetgroupTriple: (bs01b01.itsc.austin.ibm.com,-,)
nisNetgroupTriple: (bs01b02.itsc.austin.ibm.com,-,)
nisNetgroupTriple: (bs01b11.itsc.austin.ibm.com,-,)
nisNetgroupTriple: (bs01b07.itsc.austin.ibm.com,-,)
```

4.3.6 Object class and attribute name mapping

Starting with AIX 5L Version 5.2, AIX now supports the mapping of object class and attribute names for both user and group entities. The mapping files allow AIX 5L to work with schemas with different object class and attribute names. AIX currently supports mapping files for the user, group, and AIX ID entities.

Table 4-4 on page 132 lists the supported AIX schemas and the file names of the mapping files used for the user and group entities. The mapping files are included in the bos.rte.security LPP and are installed in the /etc/security/ldap directory.

Table 4-4 User and group entity mapping files shipped with AIX 5L

AIX 5L schema name	User entity mapping file	Group entity mapping file
AIX	aixuser.map	aixgroup.map
RFC2307	2307user.map	2307group.map
RFC2307AIX	2307aixuser.map	2307aixgroup.map

Table 4-5 lists the object classes that make up each of the user and group entries for each schema type. These object class names would be used by the **mkuser** and **mkgroup** commands when accounts or groups were created. For example, if you create a user in the RFC2307 schema, that new user would be defined with the account, posixAccount, and shadowAccount object classes.

Table 4-5 User and group entity object class definitions

AIX 5L schema name	User entity class definition	Group entity class definition
AIX	aixAccount, ibm-securityidentities	aixAccessGroup
RFC2307	account, posixAccount, shadowAccount	posixGroup
RFC2307AIX	account, posixAccount, shadowAccount, aixAuxAccount, ibm-securityIdentities	posixGroup,aixAuxGroup
Microsoft SFU 2.0 & 3.X	User	Group

Example 4-24 shows the configuration options that configure the AIX 5L LDAP client to use the mapping files and object class definitions. The userclassess and groupclasses settings are not directly related to object class and attribute name mapping, but those settings are there to support different schemas just like the map files.

Example 4-24 Object class and attribute namemapping settings in ldap.cfg

```
# LDAP class definitions.
userclasses:account,posixAccount,shadowAccount
groupclasses:posixGroup

# AIX-LDAP attribute map path.
```

```
userattrmappath:/etc/security/ldap/2307user.map
groupattrmappath:/etc/security/ldap/2307group.map
idattrmappath:/etc/security/ldap/aixid.map
```

For more information about how to use the map files refer to Chapter 8, “Scenario: Microsoft Windows 2003 Active Directory integration” on page 217. For more information about object class and attribute mapping for different schemas refer to “Object class and attribute name mappings” on page 310.

4.4 Problem determination

This section discusses methods and tools used to diagnose problems with the AIX 5L LDAP client. The following is a list of common troubleshooting topics:

- SSL errors
- Kerberos bind
- Compat mode problems
- Automounter problems
- Generic LDAP debugging

4.4.1 SSL problem determination

This section discusses several of the problems you may have using SSL with LDAP and methods to determine where the problem lies.

ldap_ssl_client_init failed or Unknown SSL error

If you are running an LDAP command, such as **ldapsearch**, that is using the SSL libraries and you get the error shown in Example 4-25, you should verify that ldap.max_crypto packages are installed properly, using the **lslpp** command.

Example 4-25 ldapsearch command error because ldap.max_cypto packages missing.

```
# ldapsearch -b 'cn=Users,dc=Example,dc=Com' -h bs01b06.example.com -s base -Z
-p 636 -K /usr/ldap/etc/bs01b08_keydb.kdb  -P 'passw0rd!'-D
'cn=test02,cn=Users,dc=example,dc=com' -w 'passw0rd!'  '(objectclass=*)'
Error “ldap_ssl_client_init failed! rc == -1, failureReasonCode == 804399993
Unknown SSL error”
#
```

Example 4-26 on page 134 shows how to use the **lslpp** command to verify that the required packages for LDAP SSL support are installed. If you are missing the required packages refer to 4.2.1, “AIX 5L LDAP client software installation” on page 105 for more details about installing them.

Example 4-26 lslpp command showing required LDAP SSL packages

```
# lslpp -h "ldap.max_crypto*" "gsk*"
  Fileset         Level     Action       Status       Date         Time
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  gskta.rte
                 7.0.1.16   COMMIT       COMPLETE     09/22/05     15:07:16

  ldap.max_crypto_client.adt
                  5.2.0.0   COMMIT       COMPLETE     09/22/05     15:08:18

  ldap.max_crypto_client.rte
                  5.2.0.0   COMMIT       COMPLETE     09/22/05     15:08:18
#
```

SSL problem determination

When using SSL to secure another network protocol such as LDAP, you will often receive misleading and incorrect error messages from the application. The problem is that applications might not receive detailed diagnostics of low-level SSL failures or might just ignore failure information. For example, you might receive an application error saying `unable to contact LDAP server`, when your actual problem is that your LDAP server certificate failed to verify. While the error might be technically correct, the diagnostic message is not very helpful. The best way to determine where the problem is located is to start at the bottom and work your way up the application stack. When using LDAP over SSL, you need to test the following layers:

- Test the low-level SSL protocol by using the s_client function of the **`openssl`** command.
- Test whether your local AIX 5L key database and LDAP server is working properly with SSL by using the **`ldapsearch`** command.

Low-level SSL validation using the openssl command

The easiest way to test the low-level SSL connection to the LDAP server is by using the **`openssl s_client`** command with the -showcerts option. This command will connect to the specified host and list the server certificate, the certificate authority chain, supported ciphers, SSL session information, and verify return code. If the SSL connection worked, the **`openssl s_client`** command results in the *verify return code* will be 0 (Ok).

Example 4-27 shows the output of the **openssl s_client** command connecting the AIX 5L machine (bs01b08.example.com) to the Active Directory server (bs01b06.example.com). The parameters used are as follows:

- -CAfile /usr/ldap/etc/cacert.pem

  This option specifies the PEM file containing example.com's CA certificate. This allows the SSL client to verify that the server is using a certificate signed by the trusted CA.

- -connect bs01b06.example.com:636

  This option connects to the host name of the Active Directory server using the secure LDAP port (636).

- -showcerts

  This option specifies that you want to display the SSL certificates of the remote server and the CA that signed it.

- -ssl2

  This option specifies that you want to connect using SSL Version 2 protocol.

Example 4-27 Low-level SSL validation using the openssl s_client command

```
# openssl s_client -ssl2 -connect bs01b06.example.com:636 -CAFile
/usr/ldap/etc/cacert.pem -showcerts
CONNECTED(00000003)
depth=1 /C=us/ST=TX/L=Austin/O=example.com/CN=CA/emailAddress=ca@example.com
verify return:1
depth=0 /CN=bs01b06.example.com
verify return:1
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIDnzCCAwigAwIBAgIBBzANBgkqhkiG9w0BAQQFADBtMQswCQYDVQQGEwJ1czEL
MAkGA1UECBMCVFgxDzANBgNVBAcTBkF1c3RpbjEUMBIGA1UEChMLZXhhbXBsZS5j
b20xCzAJBgNVBAMTAkNBMR0wGwYJKoZIhvcNAQkBFg5jYUBleGFtcGxlLmNvbTAe
Fw0wNTA5MjExNjU2NDJaFw0wNjA5MjExNjU2NDJaMB4xHDAaBgNVBAMTE2JzMDFi
MDYuZXhhbXBsZS5jb20wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDq
nwZsBwamzyaE/qIeVChBvMtysmuEK4QnD8eWZGvOFEsctFmXfaj5/Lw146TkWDGJ
UkQ8dJbFnbcZDN4TntHCWS2S57ZBL+ygD9ckML6f75o+eLG1DF8mDfq+AfBB1d5u
cnz8/IVtVp52MnRo3B2DvR3u8RyJ1bWZeYeAvezdeZjiqPRc9sdLgvgpMn8La+JU
5mCPjjyhX+O49m5o7wJxIRPGcC++IgGmktf0d2jZ4TY/o4FghcUOg4SRPky/tu2p
myTLs3X3OhwuJCvaMGAYRznNk8wByOoANNAPlJ4eKM6oW51L36o8NyAZrU6yjH5A
yKuuuYD8Gf4L9PhmvfyNAgMBAAGjggEYMIIBFDAJBgNVHRMEAjAAMCwGCWCGSAGG
+EIBDQQfFh1PcGVuU1NMIEdlbmVyYXRlZCBDZXJ0aWZpY2F0ZTAdBgNVHQ4EFgQU
f16kZry6799UsEMZcIBC5cSnsS4wgZcGA1UdIwSBjzCBjIAUT/s0fmJWd9dsPQzH
+0ACLX7fQrihcaRvMG0xCzAJBgNVBAYTAnVzMQswCQYDVQQIEwJUWDEPMA0GA1UE
BxMGQXVzdGluMRQwEgYDVQQKEwtleGFtcGxlLmNvbTELMAkGA1UEAxMCQ0ExHTAb
BgkqhkiG9w0BCQEWDmNhQGV4YW1wbGUuY29tggEAMBMGA1UdJQQMMAoGCCsGAQUF
```

```
BwMBMAsGA1UdDwQEAwIFoDANBgkqhkiG9w0BAQQFAAOBgQCzRmVqOqOA3WeQ//al
vsUiRYul/ZzZFMk4ZnIq/NNgB9w1BEcEXDPLeRkCY7Qm4CIg4GFxojjZ468VDy6D
+kAEE5TemLCCQcdQju6lj8BmlCLhgT/CD9V5v79yQcdpFhwj14nafu3H3HMnUNxX
r9OujGOspd2eqShWlfI9oEYnIA==
-----END CERTIFICATE-----
subject=/CN=bs01b06.example.com
issuer=/C=us/ST=TX/L=Austin/O=example.com/CN=CA/emailAddress=ca@example.com
---
No client certificate CA names sent
---
Ciphers common between both SSL endpoints:
RC4-MD5         EXP-RC4-MD5     RC2-CBC-MD5
EXP-RC2-CBC-MD5 DES-CBC-MD5     DES-CBC3-MD5
---
SSL handshake has read 1064 bytes and written 367 bytes
---
New, SSLv2, Cipher is DES-CBC3-MD5
Server public key is 2048 bit
SSL-Session:
    Protocol  : SSLv2
    Cipher    : DES-CBC3-MD5
    Session-ID: 7A230000C1C03D58BA5DD89C1D0C6DA5
    Session-ID-ctx:
    Master-Key: 2FE71F0B199339A2B5F239FC72966C5B94917371131A88F4
    Key-Arg   : 6842E42B41C3D5BF
    Start Time: 1127489298
    Timeout   : 300 (sec)
    Verify return code: 0 (ok)
---
#
```

Basic secure LDAP validation using the ldapsearch command

After you have confirmed that the SSL connection is working properly, you should then confirm that you are able to search your LDAP directory using LDAP on a secure port. This will confirm that the LDAP server can communicate with SSL and that your local AIX 5L certificate keystore was correctly created. Example 4-28 on page 137 shows the use of the **ldapsearch** command to test the secure LDAP search. The following is a partial list of **ldapsearch** options used to perform the test:

- -D 'cn=host_bs01b08,cn=Users,dc=example,dc=com'

  This option specifies the DN used to bind to the directory. If your directory allows anonymous bind, you can skip the -D and -w options.

- -h bs01b06.example.com

  This option specifies the host name of the LDAP server to connect to.

- -K /usr/ldap/etc/bs01b08_keydb.kdb

  This option specifies the location of the local keystore where the example.com CA certificate was imported.
- -P 'passw0rd!'

  This option specifies the password needed to open the keystore.
- -p 636

  This option specifies the port number to connect to. The default port for secure LDAP is 636.
- -w 'passw0rd!'

  This option specifies the password to bind to the LDAP server with.
- -Z

  This option specifies that the **`ldapsearch`** command should use SSL to connect to the LDAP server.

Example 4-28 Testing LDAP over SSL using ldapsearch command

```
# ldapsearch  -b 'cn=Users,dc=Example,dc=Com' -h bs01b06.example.com -s base -Z
-p 636 -K /usr/ldap/etc/bs01b08_keydb.kdb  -P 'passw0rd!' -D
'cn=host_bs01b08,cn=Users,dc=example,dc=com' -w 'passw0rd!'  '(objectclass=*)'
objectClass=top
objectClass=container
cn=Users
description=Default container for upgraded user accounts
distinguishedName=CN=Users,DC=example,DC=com
instanceType=4
whenCreated=20050913201915.0Z
whenChanged=20050913201915.0Z
uSNCreated=4304
uSNChanged=4304
showInAdvancedViewOnly=FALSE
name=Users
objectGUID=NOT ASCII
systemFlags=-1946157056
objectCategory=CN=Container,CN=Schema,CN=Configuration,DC=example,DC=com
isCriticalSystemObject=TRUE
#
```

4.4.2 Kerberos bind to LDAP server

This section discusses steps to diagnose problems with the AIX 5L LDAP client daemon, binding to the LDAP server using Kerberos. For more information about using the AIX Network Authorization Service (NAS) client refer to the *AIX 5L Version 5.3 Security Guide* or Chapter 8, “Scenario: Microsoft Windows 2003 Active Directory integration” on page 217. The Microsoft Active Directory chapter covers the NAS client configuration to authenticate against a Windows Kerberos Key Distribution Center (KDC). Figure 4-1 shows a flow chart of the suggested decision path for resolving Kerberos bind problems.

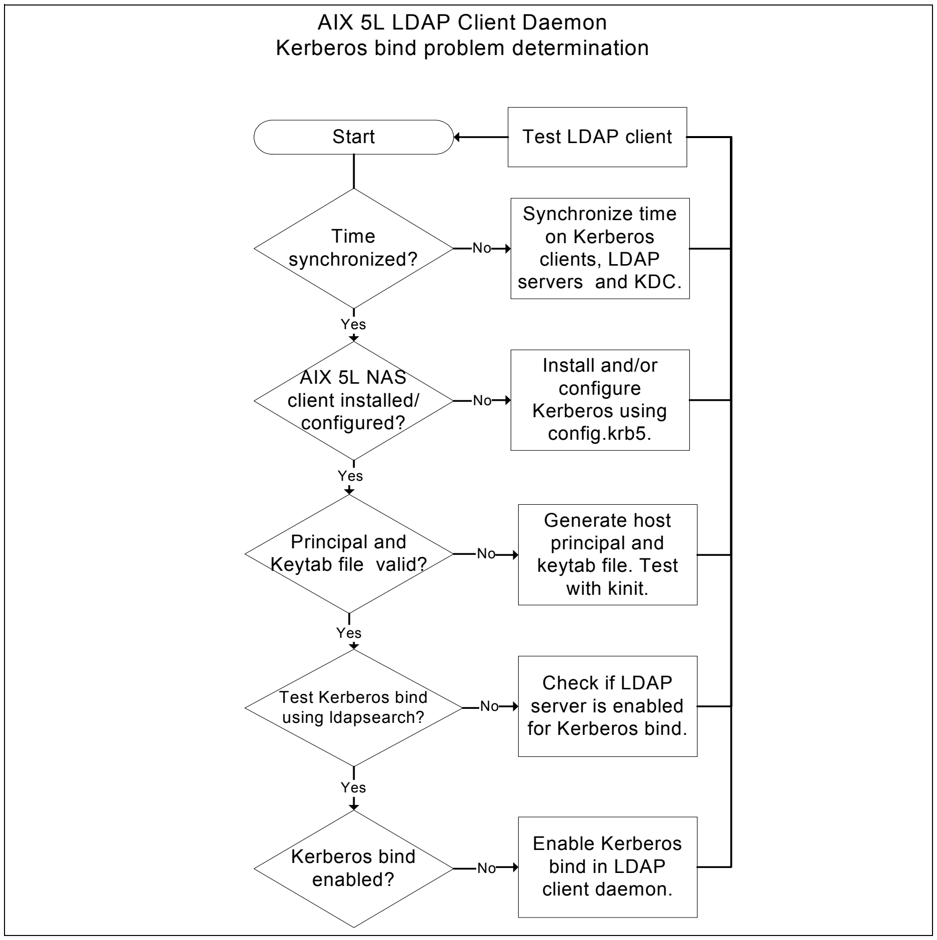

AIX 5L LDAP Client Daemon
Kerberos bind problem determination
Start
Test LDAP client
Time synchronized?
No
Synchronize time on Kerberos clients, LDAP servers and KDC.
Yes
AIX 5L NAS client installed/ configured?
No
Install and/or configure Kerberos using config.krb5.
Yes
Principal and Keytab file valid?
No
Generate host principal and keytab file. Test with kinit.
Yes
Test Kerberos bind using ldapsearch?
No
Check if LDAP server is enabled for Kerberos bind.
Yes
Kerberos bind enabled?
No
Enable Kerberos bind in LDAP client daemon.

Figure 4-1 LDAP client daemon Kerberos bind problem determination

Attention: There is a known problem with the AIX 5L Version 5.3 LDAP client binding with Kerberos to an Active Directory server. IBM is aware of the problem and has released APAR IY79120 to resolve this problem. The full APAR description can be found at the following Web site:

http://www-1.ibm.com/support/docview.wss?rs=111&context=SWG10&q1=IY79120&uid=isg1IY79120&loc=en_US&cs=utf-8&lang=en

When the fix is publicly released, it can be downloaded at the following Web site:

http://www.ibm.com/servers/eserver/support/unixservers/aixfixes.html

In the meantime, if you want a subscription service to notify you by e-mail when the PTF for the APAR is available, you can do so at the following Web site:

http://www14.software.ibm.com/webapp/set2/aparsubscriptions/bjyqvcmjd

Verify time is synchronized

Whenever you have problems with Kerberos, the first thing you should always check is that the system clocks on the LDAP clients and servers and the KDC are synchronized.

In order to prevent malicious users from replaying network traffic to authenticate to the KDC as legitimate users, Kerberos servers will automatically discard all authentication requests over the allowable clock skew of 300 seconds. If your clocks are not synchronized, you can synchronize your clocks and use Network Time Protocol (NTP) to maintain clock synchronization. For more information about setting up the AIX 5L NTP client refer to “Set up time synchronization” on page 220.

Verify AIX 5L NAS client is installed and configured

In order for any Kerberos authentication to work the AIX 5L NAS client must be installed and configured. Example 4-29 shows how to use the **lslpp** command to verify that the AIX 5L NAS client file sets are installed. If the NAS client is not installed, then refer to the *AIX 5L Security Guide* for installation and configuration instructions.

Example 4-29 Verifying that the AIX 5L NAS client is installed

```
# lslpp -l "krb5*"
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  krb5.client.rte            1.4.0.3  COMMITTED  Network Authentication Service
                                                 Client
  krb5.client.samples        1.4.0.3  COMMITTED  Network Authentication Service
                                                 Samples
  krb5.doc.en_US.html        1.4.0.3  COMMITTED  Network Auth Service HTML
                                                 Documentation - U.S. English
  krb5.doc.en_US.pdf         1.4.0.3  COMMITTED  Network Auth Service PDF
                                                 Documentation - U.S. English
  krb5.lic                   1.4.0.3  COMMITTED  Network Authentication Service
                                                 License

Path: /etc/objrepos
  krb5.client.rte            1.4.0.3  COMMITTED  Network Authentication Service
                                                 Client
```

After confirming that the AIX 5L NAS client is installed, you must now make sure that the NAS client is configured and working correctly. If the NAS client is configured, the file /etc/krb5/krb5.conf will exist and the keytab for the host principal will also exist. Example 4-30 shows the krb5.conf file generated by the **config.krb5** command in the MSAD scenario. This configuration file will differ slightly depending on your environment. If this file does not exist you must use the **config.krb5** command to configure the NAS client.

The next step is to confirm that the Kerberos keytab file for the host principal, located in /etc/krb5/krb5.keytab by default, exists and is valid. If the keytab file is missing you must ensure that the host principal is created and a valid keytab is generated.

You should then use the **ktutil** command to read the keytab file and list the keys it contains. You should then be able to use the **kinit** command to authenticate as the host principal and then use the **klist** command to confirm receiving a TGT and other service tickets. If you are unable to authenticate with the keytab file, you should ensure that the password and principal name used to generate the keytab file is still valid.

Example 4-30 Verifying AIX 5L NAS client is configured and host principal created

```
# cat /etc/krb5/krb5.conf
[libdefaults]
        default_realm = EXAMPLE.COM
        default_keytab_name = FILE:/etc/krb5/krb5.keytab
        default_tkt_enctypes = des-cbc-md5 des-cbc-crc
        default_tgs_enctypes = des-cbc-md5 des-cbc-crc

[realms]
        EXAMPLE.COM = {
                kdc = bs01b06.example.com:88
                admin_server = bs01b06.example.com:749
                default_domain = example.com
        }

[domain_realm]
        .example.com = EXAMPLE.COM
        bs01b06.example.com = EXAMPLE.COM

[logging]
        kdc = FILE:/var/krb5/log/krb5kdc.log
        admin_server = FILE:/var/krb5/log/kadmin.log
        default = FILE:/var/krb5/log/krb5lib.log
#
# ktutil
ktutil:  rkt /etc/krb5/krb5.keytab
ktutil:  list
slot   KVNO   Principal
------ ------ ------------------------------------------------------
     1      3     host/bs01b08.example.com@EXAMPLE.COM
ktutil:  exit
#
# kinit -kt /etc/krb5/krb5.keytab
# klist
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_0
Default principal:  host/bs01b08.example.com@EXAMPLE.COM

Valid starting     Expires            Service principal
10/23/05 11:29:48  10/23/05 21:29:48  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        Renew until 10/24/05 11:29:48
#
```

Verify Kerberos bind with ldapsearch

After verifying that the AIX 5L NAS client is working properly, you must now ensure that you can bind to the LDAP server using Kerberos. The following example uses the **ldapsearch** command with the SASL/GSSAPI mechanism to bind to the LDAP server. The following list describes the steps involved:

1. Use the **kinit** command to authenticate as the host principal using /etc/krb5/krb5.keytab.
2. Use the **klist** command to verify that you have received a TGT from the KDC.
3. Use the **ldapsearch** command with the -m GSSAPI mechanism to connect to the LDAP server using the current Kerberos credentials. If you are unable to bind to the LDAP server with Kerberos, your LDAP server might not have Kerberos bind enabled.

4. After binding to the LDAP server with Kerberos, you can run the **klist** command again and you will have an additional LDAP service ticket. The LDAP service ticket has been highlighted for easy recognition.

Example 4-31 Testing Kerberos bind with LDAP server with the ldapsearch command

```
# kinit -kt /etc/krb5/krb5.keytab
# klist
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_0
Default principal:  host/bs01b08.example.com@EXAMPLE.COM

Valid starting     Expires            Service principal
10/23/05 19:45:55  10/24/05 05:45:55  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        Renew until 10/24/05 19:45:55
#
# ldapsearch -b dc=example,dc=com -h bs01b06 -m GSSAPI -s one objectclass=* dn
CN=Builtin,DC=example,DC=com
CN=Computers,DC=example,DC=com
CN=DefaultMigrationContainer30,DC=example,DC=com
OU=Domain Controllers,DC=example,DC=com
CN=ForeignSecurityPrincipals,DC=example,DC=com
CN=Infrastructure,DC=example,DC=com
CN=LostAndFound,DC=example,DC=com
CN=NTDS Quotas,DC=example,DC=com
CN=Program Data,DC=example,DC=com
CN=System,DC=example,DC=com
CN=Users,DC=example,DC=com
DC=DomainDnsZones,DC=example,DC=com
CN=Configuration,DC=example,DC=com
DC=ForestDnsZones,DC=example,DC=com
#
# klist
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_0
Default principal:  host/bs01b08.example.com@EXAMPLE.COM

Valid starting     Expires            Service principal
10/23/05 19:45:55  10/24/05 05:45:55  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        Renew until 10/24/05 19:45:55
10/23/05 19:46:12  10/24/05 05:45:55  ldap/bs01b06.example.com@EXAMPLE.COM
        Renew until 10/24/05 19:45:55
#
```

Verify AIX 5L LDAP client has Kerberos bind enabled

You must now ensure that the AIX 5L LDAP client daemon is configured to use Kerberos to bind to the LDAP server. Example 4-32 on page 143 shows the Kerberos bind section of the LDAP client daemon configuration file

/etc/security/ldap/ldap.cfg. For more information about configuring the LDAP client daemon for Kerberos bind refer to “Kerberos bind” on page 121.

Example 4-32 Verify LDAP client daemon has Kerberos bind configured

```
# Whether to use Kerberos for initial bind to the server.
# useKRB5       - valid value is either "yes" or "no". Default is "no".
# krbprincipal  - Kerberos principal used to bind to the server.
# krbkeypath    - Path to the kerberos keytab, default to
#                 /etc/security/ldap/krb5.keytab
# krbcmddir     - Directory that contains the Kerberos kinit command.
#                 Default to /usr/krb5/bin/.
useKRB5:yes
krbprincipal:host/bs01b08.example.com
krbkeypath:/etc/krb5/krb5.keytab
krbcmddir:/usr/krb5/bin/
```

Verify LDAP client daemon’s Kerberos ticket

When the LDAP client daemon starts, it will authenticate with the LDAP server using the Kerberos principal and keytab specified in the configuration file. If there are any problems authenticating, the LDAP client daemon will log errors to the syslog daemon. When the LDAP client daemon authenticates against the KDC successfully, it will store its Kerberos credentials in /etc/security/ldap/krb5cc_secldapclntd.

Example 4-33 shows how to verify that the LDAP client daemon received a TGT and LDAP service ticket from the KDC. You must first set the KRB5CCNAME environment variable to the location of the LDAP client daemon’s credential cache and then use the **klist** command to list the tickets in the credential cache.

Example 4-33 Verifying LDAP client daemon’s Kerberos ticket with the klist command

```
# export KRB5CCNAME=/etc/security/ldap/krb5cc_secldapclntd
# klist
Ticket cache:  FILE:/etc/security/ldap/krb5cc_secldapclntd
Default principal:  host/bs01b08.example.com@EXAMPLE.COM

Valid starting     Expires            Service principal
10/17/05 17:23:05  10/18/05 03:23:05  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        Renew until 10/18/05 17:23:05
10/17/05 17:23:05  10/18/05 03:23:05  ldap/bs01b06.example.com@EXAMPLE.COM
        Renew until 10/18/05 17:23:05
#
```

4.4.3 Generic LDAP client and server debugging

This section discusses how to debug the AIX 5L LDAP client using the LDAP_DEBUG and LDAP_DEBUG_FILE environment variables. These variables allow detailed tracing and debugging of any application that uses the AIX 5L LDAP libraries, such as **`ldapsearch`** or the **`secldapclntd`**. The LDAP_DEBUG_FILE variable specifies the file name the debug output is sent to and the LDAP_DEBUG specifies the debug level for the LDAP program. A complete list of debugging levels is included in Table 4-6. Note that some of these levels might not be used for LDAP client software. Multiple levels can be specified by adding the decimal values of the debug flags.

Tip: If you set the LDAP_DEBUG_FILE variable to /dev/tty, the debugging information will be sent to the terminal that executed the LDAP command. This allows you to debug LDAP programs without having to watch a separate file.

Table 4-6 LDAP_DEBUG debugging levels

Hex	Decimal	Value	Description
0x0000	0	LDAP_DEBUG_OFF	Turns debugging off
0x0001	1	LDAP_DEBUG_TRACE	Entry and exit from routines
0x0002	2	LDAP_DEBUG_PACKETS	Packet activity
0x0004	4	LDAP_DEBUG_ARGS	Data arguments from requests
0x0008	8	LDAP_DEBUG_CONNS	Connection activity
0x0010	16	LDAP_DEBUG_BER	Encoding and decoding of data
0x0020	32	LDAP_DEBUG_FILTER	Search filters
0x0040	64	LDAP_DEBUG_MESSAGE	Messaging subsystem activities and events
0x0080	128	LDAP_DEBUG_ACL	Access control list activities
0x0100	256	LDAP_DEBUG_STATS	Operational statistics
0x0200	512	LDAP_DEBUG_THREAD	Threading statistics
0x0400	1024	LDAP_DEBUG_REPL	Replication statistics
0x0800	2048	LDAP_DEBUG_PARSE	Parsing activities
0x1000	4096	LDAP_DEBUG_PERFORMANCE	Relational back-end performance statistics
0x2000	8192	LDAP_DEBUG_RDBM	Relational back-end activities (RDBM)

Hex	Decimal	Value	Description
0x4000	16384	LDAP_DEBUG_REFERRAL	Referral activities
0x8000	32768	LDAP_DEBUG_ERROR	Error conditions
0xffff	65535	LDAP_DEBUG_ANY	All levels of debug

Example 4-34 shows how to debug the LDAP calls used in the **ldapsearch** command. The debug output will be sent to the file /tmp/ldapdebug.out. The LDAP_DEBUG level was set to 25, which means that the LDAP_DEBUG_BER, LDAP_DEBUG_CONNS, and LDAP_DEBUG_TRACE levels are turned on.

Example 4-34 Using LDAP_DEBUG for ldapseach command

```
# export LDAP_DEBUG_FILE=/tmp/ldapdebug.out
# export LDAP_DEBUG=25
# ldapsearch -b dc=example,dc=com -h bs01b06 -m GSSAPI -s base objectclass=* dn
dc=example,dc=com
#
```

Example 4-35 shows abbreviated output from the **ldapsearch** command from Example 4-34. The content was truncated to reduce its size.

Example 4-35 Debugging output of ldapsearch command using LDAP_DEBUG

```
290:05:38:00 T614565 ldap_init: defhost=bs01b06, defport=389
290:05:38:00 T614565 ldap_sasl_bind_s
290:05:38:00 T614565 ldap_sasl_bind_s_call_plugin
290:05:38:00 T614565 ldap_register_plugin()
type=sasl  subtype=GSSAPI  path=ldap_plugin_ibm_gsskrb
290:05:38:00 T614565 ldap_read_conf_file()
290:05:38:00 T614565 ldap_load_plugin(): ldap_plugin_ibm_gsskrb
290:05:38:00 T614565 ldap_gpt_eval()
290:05:38:00 T614565 ldap_plugin_sasl_bind_s_prepare: gss_import_name()
maj_stat=0 min_stat=0
290:05:38:00 T614565 ldap_plugin_sasl_bind_s_prepare: gss_init_sec_context()
maj_stat=1 min_stat=0
290:05:38:00 T614565 ldap_plugin_sasl_bind_s_prepare: gss_init_sec_context()
returns with Delgation OFF
290:05:38:00 T614565 send_initial_request
290:05:38:00 T614565 open_default_connection
290:05:38:00 T614565 new_connection: connect=1
290:05:38:00 T614565 open_ldap_connection
290:05:38:00 T614565 connect_to_host: bs01b06:389
290:05:38:00 T614565 sd 4 connected to: 9.3.5.143
290:05:38:00 T614565 new_connection: successful - return(lc=200134e8)
290:05:38:00 T614565 ber_flush: 1203 bytes to sd=4
```

```
T614565:       +---------------------------------------------------------+
T614565:       |OSet| Address = 2002BFF8  Length = 04B3 |     ASCII      |
T614565:       +---------------------------------------------------------+
T614565:       |0000|308204AF 02010160 82048602 01030400|0......`........|
T614565:       |0010|A382047D 04064753 53415049 04820471|...}..GSSAPI...q|
T614565:       |0020|6082046D 06092A86 4886F712 01020201|`..m..*.H.......|
T614565:       |0030|006E8204 5C308204 58A00302 0105A103|.n..\0..X.......|
T614565:       |0040|02010EA2 07030500 20000000 A3820388|........ .......|
T614565:       |0050|61820384 30820380 A0030201 05A10D1B|a...0...........|
T614565:       |0060|0B455841 4D504C45 2E434F4D A2263024|.EXAMPLE.COM.&0$|
T614565:       |0070|A0030201 03A11D30 1B1B046C 6461701B|.......0...ldap.|
T614565:       |0080|13627330 31623036 2E657861 6D706C65|.bs01b06.example|
T614565:       |0090|2E636F6D A3820340 3082033C A0030201|.com...@0..<....|
T614565:       |00A0|17A10302 0104A282 032E0482 032A0AE0|.............*..|
T614565:       |00B0|187E0909 D9FB3B5B 3619CCC2 98050903|.~....;[6.......|
T614565:       |00C0|78D453CC AEF4FE4B DF30C2FB BEBBA409|x.S....K.0......|
T614565:       |00D0|343B32CA 683C433F 340EAB5A 6BFF465C|4;2.h<C?4..Zk.F\|
<output truncated>
```

4.4.4 Network protocol analysis

Attention: Before using a network packet sniffer, ensure that the use of this tool is not restricted in your organization or illegal in your locality. While packet sniffers are legitimate and useful tools, they can also be used by malicious hackers to capture sensitive information in the network.

This section discusses the use of network protocol analyzers or packet sniffers to debug communications between client and server applications. When you are trying to debug a network application, you often do not have access to the source code, making it like a black box. When you eavesdrop on the communications between these black boxes, you can often find information or see errors that are not normally exposed to you. For example, you might find application errors or authentication failures in the network trace, but the application will not give any indications of failures. A common method for problem determination is to take a network trace from a known working operation and compare it to the trace from a a non-working operation. This allows you to compare the two operations and determine at what point the operation failed.

Attention: When using SSL to encapsulate the LDAP protocol, all details of the network conversation are encrypted and not useful for debugging. The only nonencrypted packet will be the initial SSL handshake. One suggestion is to disable the use of SSL until your problem is solved.

In order to capture a packet trace, you must run the **tcpdump** command and either capture and display the packets in real-time or write the captured packets to a file for later analysis. When using the **tcpdump** command to view a packet trace, information about the captured packets can be displayed with varying levels of detail. The following examples do not use the real-time display of captured packets, as **tcpdump** can only display the captured data and not decode network protocols such as LDAP.

The biggest limitation with the **tcpdump** command is extremely limited support for decoding network protocols. The best option is to take a network trace with **tcpdump** and then use either an open source or commercial network protocol analyzer. Ethereal is an open source network protocol analyzer that is licensed under the GNU General Public License (GPL). For more information about and to download the Ethereal package please use the following Web site:

http://www.ethereal.com/

Example 4-36 shows how to use the **tcpdump** command to capture a network trace for later analysis by a network protocol analyzer. The options specified in the following **tcpdump** command are as follows:

-c 40	This option specifies that **tcpdump** should exit after capturing 40 packets. If no packet count is specified, **tcpdump** will continue to capture packets until it is interrupted.
-s 1600	This option specifies that **tcpdump** should capture the first 1600 bytes of the packet rather than the default of 68 bytes. While the default snap length is sufficient to capture most IP, TCP, UDP, and ICMP headers, protocol information might be truncated if the snap length is too small.
-w /tmp/tcpdump.out	This option specifies the file name to write the captured network packets into.
port ldap	This option specifies the expression used to restrict capturing packets that match certain requirements. You are often interested only in conversations between specific machines or using certain protocols. The expression *port ldap* will capture all packets to and from the local machine that have the LDAP port in the source or destination port number.

Example 4-36 Using the tcpdump command to capture packets for later analysis

```
# tcpdump -c 40 -s 1600 -w /tmp/tcpdump.out port ldap
tcpdump: listening on en1, link-type 1, capture size 1600 bytes
111 packets received by filter
```

```
0 packets dropped by kernel
#
#  ls -l /tmp/tcpdump.out
-rw-rw-r--   1 root     system         1530 Oct 20 02:07 /tmp/tcpdump.out
#
```

Example 4-37, Example 4-38 on page 149, and Figure 4-2 on page 150 all illustrate different views of the network trace captured in the previous example. The LDAP packets were generated by telneting as the test03 user into the local machine.

Example 4-37 shows the basic output from the **tcpdump** command, displaying the list of captured packets and their capture time, protocol, source and destination host name, source and destination port numbers, and other flags. The packets on lines 7 and 8 are highlighted, as we will limit our future packet analysis to only those two packets to conserve space.

Example 4-37   Displaying summary of captured packets using the tcpdump command

```
# tcpdump -r /tmp/tcpdump.out
reading from file /tmp/tcpdump.out, link-type 1
02:06:48.196130 IP bs01b08.example.com.1023 > bs01b06.example.com.ldap: P 
1889799466:1889799564(98) ack 4039034172 win 65535
02:06:48.196558 IP bs01b06.example.com.ldap > bs01b08.example.com.1023: P 1:23(22) ack 98 win 
64954
02:06:48.344932 IP bs01b08.example.com.1023 > bs01b06.example.com.ldap: . ack 23 win 65535
02:06:50.747088 IP bs01b08.example.com.34038 > bs01b06.example.com.ldap: S 
654831232:654831232(0) win 65535 <mss 1460>
02:06:50.747176 IP bs01b06.example.com.ldap > bs01b08.example.com.34038: S 
4146056382:4146056382(0) ack 654831233 win 16384 <mss 1460>
02:06:50.747209 IP bs01b08.example.com.34038 > bs01b06.example.com.ldap: . ack 1 win 65535
02:06:50.747331 IP bs01b08.example.com.34038 > bs01b06.example.com.ldap: P 1:60(59) ack 1 win 
65535
02:06:50.748948 IP bs01b06.example.com.ldap > bs01b08.example.com.34038: P 1:23(22) ack 60 win 
65476
02:06:50.749065 IP bs01b08.example.com.34038 > bs01b06.example.com.ldap: P 60:67(7) ack 23 win 
65535
02:06:50.749085 IP bs01b08.example.com.34038 > bs01b06.example.com.ldap: F 67:67(0) ack 23 win 
65535
02:06:50.749170 IP bs01b06.example.com.ldap > bs01b08.example.com.34038: . ack 68 win 65469
02:06:50.749272 IP bs01b06.example.com.ldap > bs01b08.example.com.34038: F 23:23(0) ack 68 win 
65469
02:06:50.749295 IP bs01b08.example.com.34038 > bs01b06.example.com.ldap: . ack 24 win 65535
02:06:53.055463 IP bs01b08.example.com.1023 > bs01b06.example.com.ldap: P 98:153(55) ack 23 win 
65535
```

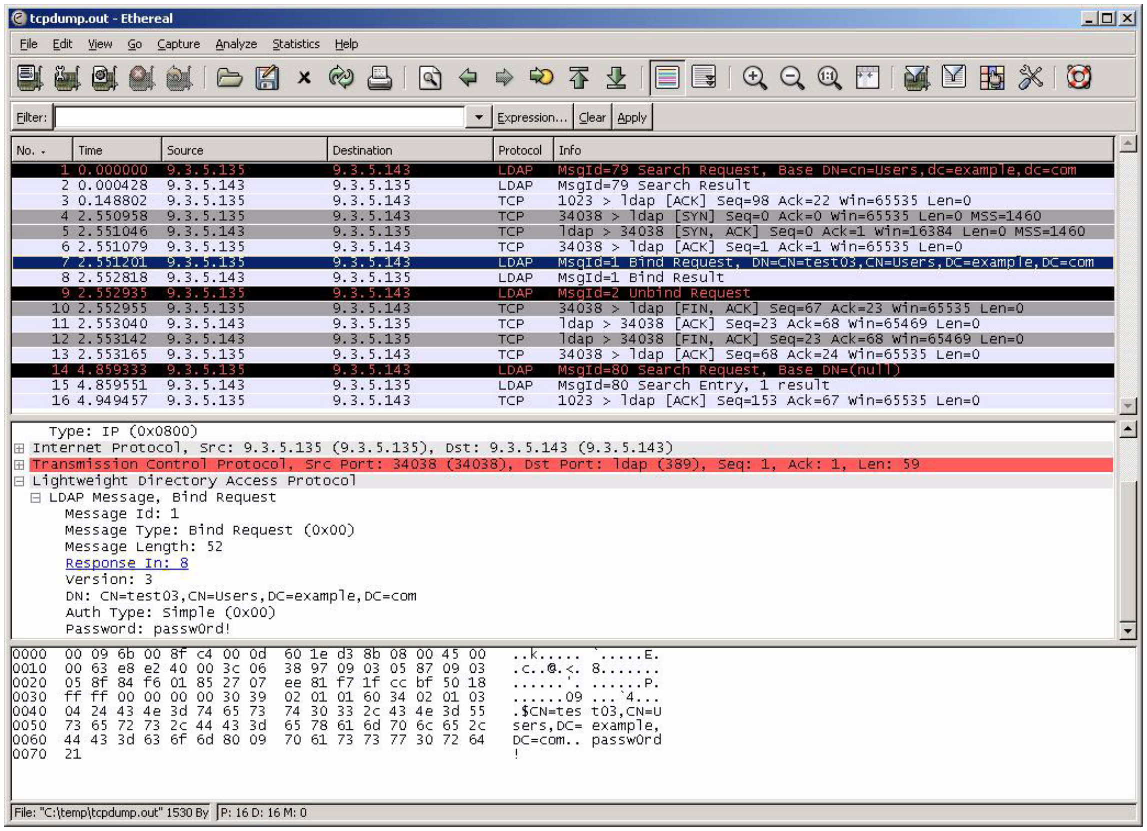
```
02:06:53.055681 IP bs01b06.example.com.ldap > bs01b08.example.com.1023: P 23:68(45) ack 153 win
64899
02:06:53.145587 IP bs01b08.example.com.1023 > bs01b06.example.com.ldap: . ack 68 win 65535
#
```

The first highlighted packet is the LDAP client attempting to bind to the LDAP server using a DN and password of CN=test03,CN=Users,DC=example,DC=com and passw0rd!. The second highlighted packet is the LDAP server's reply to the bind request. Example 4-38 shows the **tcpdump** command's output using the -X command-line option to display the packet information in both hexadecimal and in ASCII. The bind DN and password were transmitted in clear text and are visible in the captured packet. Since the LDAP protocol elements are encoded before transmission, visualizing most of the LDAP protocol using ASCII or hexadecimal is not very useful.

Example 4-38 Displaying ASCII and hexadecimal packet dump using tcpdump command

```
# tcpdump -r  /tmp/tcpdump.out -X
reading from file /tmp/tcpdump.out, link-type 1
<output removed>
02:06:50.747331 IP bs01b08.example.com.34038 > bs01b06.example.com.ldap: P 1:60(59) ack 1 win
65535
0x0000   4500 0063 e8e2 4000 3c06 3897 0903 0587        E..c..@.<.8.....
0x0010   0903 058f 84f6 0185 2707 ee81 f71f ccbf        ........'.......
0x0020   5018 ffff 0000 0000 3039 0201 0160 3402        P.......09...`4.
0x0030   0103 0424 434e 3d74 6573 7430 332c 434e        ...$CN=test03,CN
0x0040   3d55 7365 7273 2c44 433d 6578 616d 706c        =Users,DC=exampl
0x0050   652c 4443 3d63 6f6d 8009 7061 7373 7730        e,DC=com..passw0
0x0060   7264 21                                        rd!
02:06:50.748948 IP bs01b06.example.com.ldap > bs01b08.example.com.34038: P 1:23(22) ack 60 win
65476
0x0000   4500 003e 2b43 4000 8006 b25b 0903 058f        E..>+C@....[....
0x0010   0903 0587 0185 84f6 f71f ccbf 2707 eebc        ............'...
0x0020   5018 ffc4 6ab6 0000 3084 0000 0010 0201        P...j...0.......
0x0030   0161 8400 0000 070a 0100 0400 0400 af37        .a.............7
0x0040   11e6
<output removed>
#
```

Figure 4-2 on page 150 shows the use of the Ethereal package to decode packet number 7, the LDAP bind operation with the server. The default screen for Ethereal is split up into three sections: the packet list, details, and bytes. The packet list section displays the list of all of the packets in the packet trace. The packet detail section displays the detailed protocol information about the current packet. The packet bytes section displays the hexadecimal and ASCII bytes, similar to the -X option on **tcpdump**.


tcpdump.out - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No. -  Time      Source      Destination  Protocol Info
     1 0.000000  9.3.5.135   9.3.5.143    LDAP     MsgId=79 Search Request, Base DN=cn=Users,dc=example,dc=com
     2 0.000428  9.3.5.143   9.3.5.135    LDAP     MsgId=79 Search Result
     3 0.148802  9.3.5.135   9.3.5.143    TCP      1023 > ldap [ACK] Seq=98 Ack=22 Win=65535 Len=0
     4 2.550958  9.3.5.135   9.3.5.143    TCP      34038 > ldap [SYN] Seq=0 Ack=0 Win=65535 Len=0 MSS=1460
     5 2.551046  9.3.5.143   9.3.5.135    TCP      ldap > 34038 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0 MSS=1460
     6 2.551079  9.3.5.135   9.3.5.143    TCP      34038 > ldap [ACK] Seq=1 Ack=1 Win=65535 Len=0
     7 2.551201  9.3.5.135   9.3.5.143    LDAP     MsgId=1 Bind Request, DN=CN=test03,CN=Users,DC=example,DC=com
     8 2.552818  9.3.5.143   9.3.5.135    LDAP     MsgId=1 Bind Result
     9 2.552935  9.3.5.135   9.3.5.143    LDAP     MsgId=2 Unbind Request
    10 2.552955  9.3.5.135   9.3.5.143    TCP      34038 > ldap [FIN, ACK] Seq=67 Ack=23 Win=65535 Len=0
    11 2.553040  9.3.5.143   9.3.5.135    TCP      ldap > 34038 [ACK] Seq=23 Ack=68 Win=65469 Len=0
    12 2.553142  9.3.5.143   9.3.5.135    TCP      ldap > 34038 [FIN, ACK] Seq=23 Ack=68 Win=65469 Len=0
    13 2.553165  9.3.5.135   9.3.5.143    TCP      34038 > ldap [ACK] Seq=68 Ack=24 Win=65535 Len=0
    14 4.859333  9.3.5.135   9.3.5.143    LDAP     MsgId=80 Search Request, Base DN=(null)
    15 4.859551  9.3.5.143   9.3.5.135    LDAP     MsgId=80 Search Entry, 1 result
    16 4.949457  9.3.5.135   9.3.5.143    TCP      1023 > ldap [ACK] Seq=153 Ack=67 Win=65535 Len=0
```

```
    Type: IP (0x0800)
⊞ Internet Protocol, Src: 9.3.5.135 (9.3.5.135), Dst: 9.3.5.143 (9.3.5.143)
⊞ Transmission Control Protocol, Src Port: 34038 (34038), Dst Port: ldap (389), Seq: 1, Ack: 1, Len: 59
⊟ Lightweight Directory Access Protocol
  ⊟ LDAP Message, Bind Request
      Message Id: 1
      Message Type: Bind Request (0x00)
      Message Length: 52
      Response In: 8
      Version: 3
      DN: CN=test03,CN=Users,DC=example,DC=com
      Auth Type: Simple (0x00)
      Password: passw0rd!
```

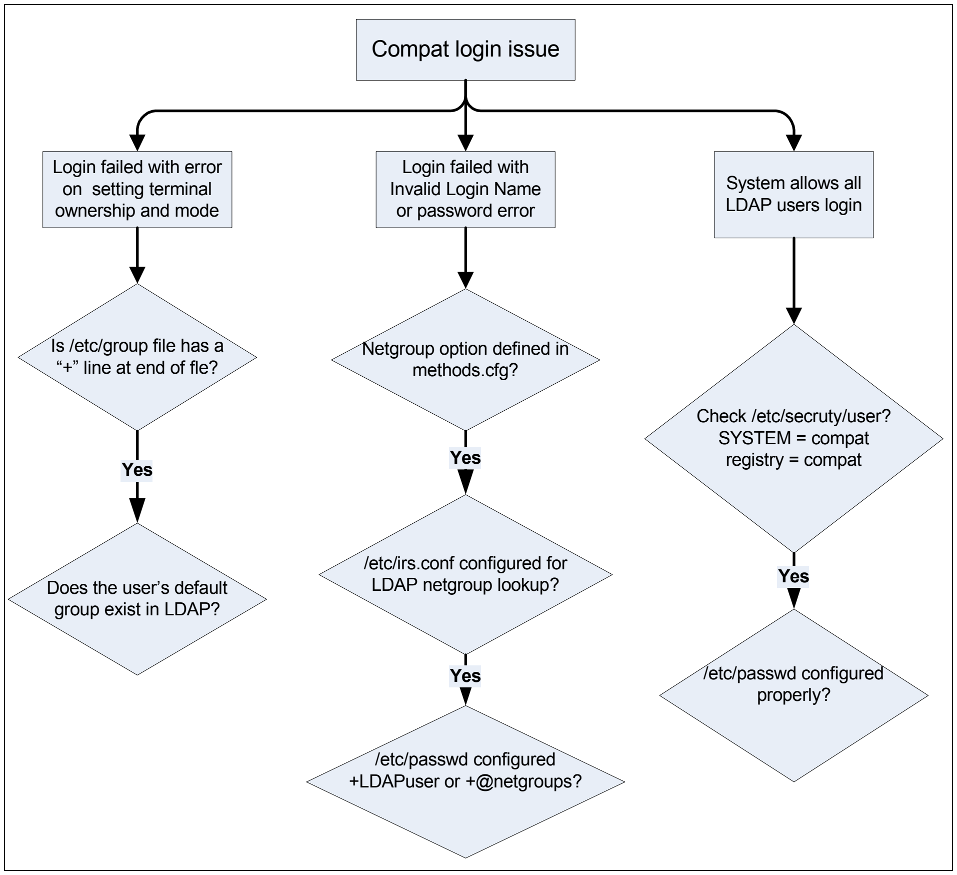
```
0000  00 09 6b 00 8f c4 00 0d  60 1e d3 8b 08 00 45 00   ..k..... `.....E.
0010  00 63 e8 e2 40 00 3c 06  38 97 09 03 05 87 09 03   .c..@.<. 8.......
0020  05 8f 84 f6 01 85 27 07  ee 81 f7 1f cc bf 50 18   ......'. ......P.
0030  ff ff 00 00 00 00 30 39  02 01 01 60 34 02 01 03   ......09 ...`4...
0040  04 24 43 4e 3d 74 65 73  74 30 33 2c 43 4e 3d 55   .$CN=tes t03,CN=U
0050  73 65 72 73 2c 44 43 3d  65 78 61 6d 70 6c 65 2c   sers,DC= example,
0060  44 43 3d 63 6f 6d 80 09  70 61 73 73 77 30 72 64   DC=com.. passw0rd
0070  21                                                 !
```

```
File: "C:\temp\tcpdump.out" 1530 By   P: 16 D: 16 M: 0
```

Figure 4-2 Decoding LDAP operations using the Ethereal Network Protocol Analyzer

4.4.5 Compat mode problems

Figure 4-3 is a flow chart describing common problems in compat mode.



Compat login issue
Login failed with error on setting terminal ownership and mode
Login failed with Invalid Login Name or password error
System allows all LDAP users login
Is /etc/group file has a "+" line at end of fle?
Netgroup option defined in methods.cfg?
Check /etc/secruty/user? SYSTEM = compat registry = compat
Yes
Yes
Yes
Does the user's default group exist in LDAP?
/etc/irs.conf configured for LDAP netgroup lookup?
/etc/passwd configured properly?
Yes
/etc/passwd configured +LDAPuser or +@netgroups?

Figure 4-3 Troubleshoot compat issues

Login failure: Failed setting terminal ownership and mode

Do the following to troubleshoot if there is such an error (shown in Example 4-39 on page 152):

1. Check the /etc/group file to verify that LDAP is enabled for group resolution. Refer to step 3 in 4.3.4, "Restricting user access using netgroups" on page 123.
2. Check the user's default group existence in LDAP. The default group of a user must be resolvable. If it does not exist in LDAP, the system would deny the login with an error. Add the user's default group to LDAP if it does not exist.

Example 4-39 Login failed with terminal ownership and mode setting error

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2005.
login: user1
user1's Password:
Failed setting terminal ownership and mode.
```

Login failure: Invalid login name or password

Do the followings to troubleshoot:

1. Check the netgroup option setting in the /usr/lib/security/methods.cfg file. It should be defined in the LDAP stanza. If it does not exist, please follow step 2 in 4.3.4, “Restricting user access using netgroups” on page 123.
2. Verify that the *netgroup nis_ldap* set is in the /etc/irs.conf file.
3. Verify that /etc/passwd has the proper +LDAPuser and +@LDAPnetgroup specified in the file.

Example 4-40 Login error when netgroup is not enabled in methods.cfg

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2005.
login: user1
user1's Password:
You entered an invalid login name or password.
```

If the system allows all LDAP users to log in to the system instead of allowing only those users defined in /etc/passwd, then verify that the SYSTEM and registry settings are in the /etc/security/user file. Both attributes should be set to compat. Please refer to step 1 in 4.3.4, “Restricting user access using netgroups” on page 123.

4.4.6 Automount problems

Figure 4-4 shows common automount errors and troubleshooting tips.

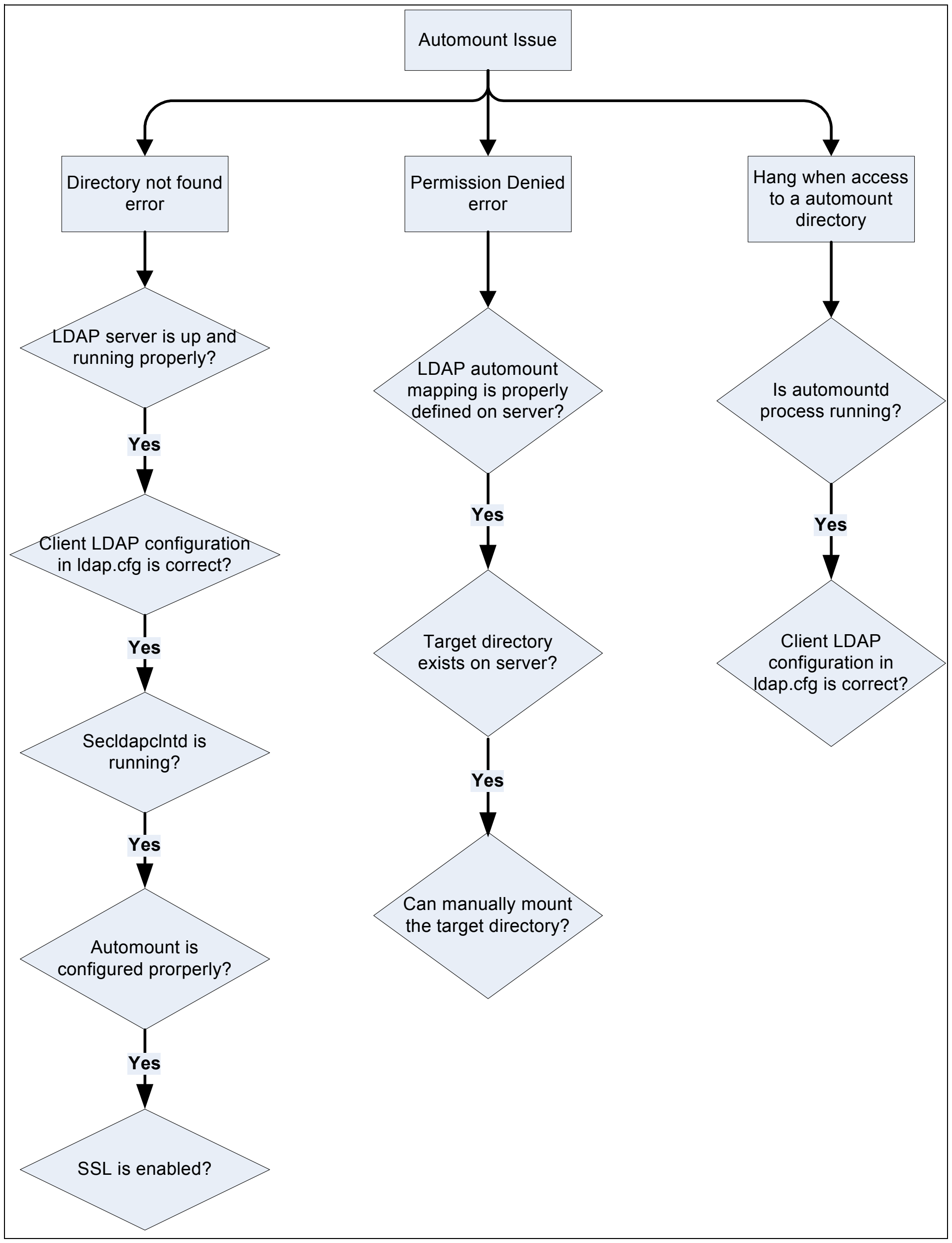

Automount Issue
Directory not found error
Permission Denied error
Hang when access to a automount directory
LDAP server is up and running properly?
Yes
Client LDAP configuration in ldap.cfg is correct?
Yes
Secldapclntd is running?
Yes
Automount is configured prorperly?
Yes
SSL is enabled?
LDAP automount mapping is properly defined on server?
Yes
Target directory exists on server?
Yes
Can manually mount the target directory?
Is automountd process running?
Yes
Client LDAP configuration in ldap.cfg is correct?

Figure 4-4 Troubleshoot automount issue

Client receives automount directory not found error

The automount daemon needs to resolve automount maps information from LDAP. If such information is n0t available or is incorrect, automount would fail. Example 4-41 shows the error message when the system failed on automount on a user's home directory.

Example 4-41 automount directory not-found error

```
# cd /home/user1
ksh: /home/user1:  not found
```

When such and error occurs, the user should do the following to troubleshoot the problem:

1. Check the LDAP server to confirm that the server is functioning properly. Use the **ldapsearch** command to check for the server's availability.
2. If the issue is not on the server, then check the client to make sure LDAP is configured properly. Check the ldapservers, binddn, binddnpwd, and automountbasedn settings in /etc/security/ldap/ldap.cfg. Use the **ldapsearch** command to bind to the LDAP server as the bind DN.
3. Run **ls-secldapclntd** to verify that the secldapclntd daemon is running properly.

   Note: The automount daemon does not use **secldapclntd** to query data. It opens LDAP connections based on the settings on /etc/security/ldap/ldap.cfg. The automount would work even if **secldapclntd** is not working.

4. Verify that automount is configured correctly. See Example 4.3.5 on page 127.
5. Check whether SSL is enabled. automount will not work if SSL is enabled.

Client receives permission denied error

Example 4-42 shows the permission denied error when a automount directory is referenced. Normally this error indicates that the remote NFS server denies client access to mount the requested directory.

Example 4-42 automount permission error

```
# cd /home/user1
ksh: /home/user1: permission denied
```

Do the following to troubleshoot the permission error:

1. Run **`lsldap`** to check the automount mapping in LDAP. If the automount mapping stored in LDAP is incorrect, modify the LDAP automount entry to the correct server and path setting.
2. Check the NFS server to verify that the target directory exists on the server.
3. Use the **`mount`** command to manually mount the NFS directory. If the system responds with the same permission denied error, check the NFS server to verify that the server is exporting out the file system to the client. If the NFS server uses netgroups to export the directory, make sure the client is a member of the netgroup.

Client hangs when access a automount directory

If this is the case, then:

1. Check whether automountd is running on the client. If it is not running, manually run **`/usr/sbin/automount`** to start automountd.
2. If the automountd daemon does not start, check the LDAP configuration in /etc/security/ldap/ldap.cfg.

5

Scenario: Sun ONE LDAP server integration

This chapter describes an existing Sun ONE Directory Server environment using LDAP and the integration of an AIX 5L client that environment. This chapter describes the following steps in the process:

- 5.1, “Overview” on page 158
- 5.2, “Environment” on page 158
- 5.3, “AIX 5L client configuration” on page 158
- 5.4, “Restriction of user login with compat mode (netgroups)” on page 164
- 5.5, “automount configuration” on page 173
- 5.6, “SSL configuration” on page 178

5.1 Overview

This chapter describes generic integration of an AIX 5L Version 5.3 Maintenance Release 3 LDAP client into an existing LDAP environment. In this case we assume that the existing environment evolved from NIS and NIS+, so considerations for netgroups and automount usage are described. The server and client setups conform to RFC2307 specifications to enable interoperability in a multi-OS environment.

5.2 Environment

The existing environment has many hundreds of Sun servers running Solaris Version 8 and later. As the install base has grown over the years the problems of user authentication, authorization, and home directory management have been addressed with Sun's management tools. Network Information Services (NIS) was first used to manage users. It was later replaced by NIS+. More recently NIS+ is being replaced by LDAP. The Sun ONE Directory Server is used as the LDAP server. The NIS and NIS+ implementations rely heavily on netgroups to specify which Solaris instances a user may log in to. Each user has a single home directory that resides on a server in the enterprise. Automount is used to mount the user's home directory automatically when the user logs in. SSL is enabled to provide secure connections between the LDAP clients and servers.

The Sun ONE Directory Server installation/configuration is beyond the scope of this book and will not be discussed in this chapter. The assumption is that the LDAP directory server has been properly installed and configured according to RFC2307 schema.

5.3 AIX 5L client configuration

The AIX 5L LDAP client supports schema defined in RFC2307 as well as RFC2307AIX. Most UNIX vendors support RFC2307. Compliance with RFC2307 is required if you want to successfully integrate various UNIX clients in a heterogeneous environment. Refer to 3.2.2, “RFC2307 schema” on page 62 for more detail about RFC2307. The current chapter discusses the configuration of AIX 5L clients in a heterogeneous environment, which conforms to RFC2307 (Figure 5-1 on page 159). RFC2307AIX is an AIX 5L-specific schema and is discussed in 3.2.3, “RFC2307AIX schema” on page 63.

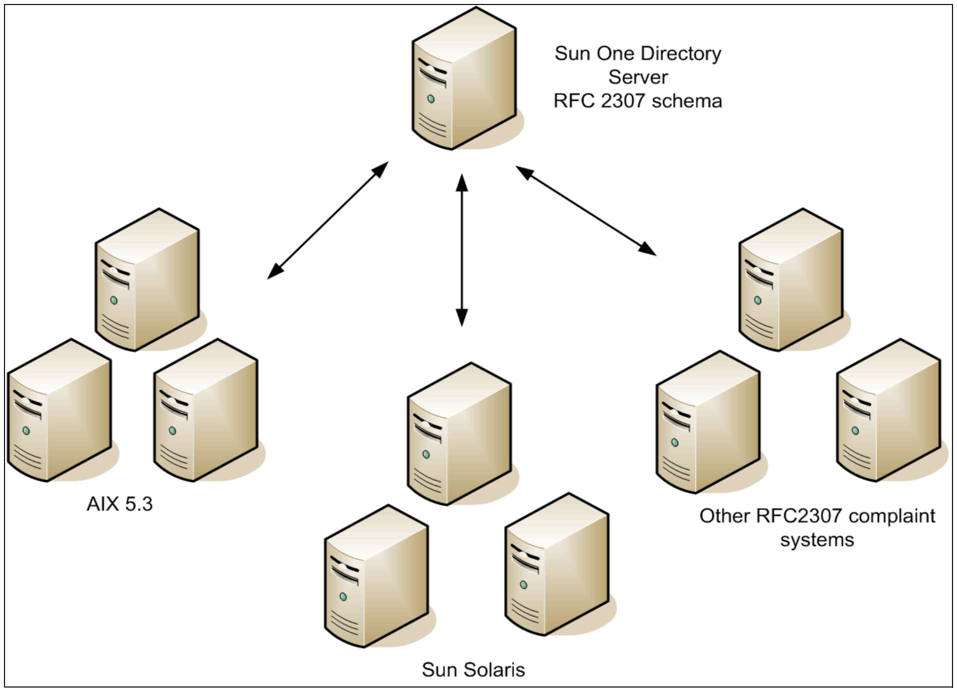


Figure 5-1 Sun ONE Directory Server and AIX 5L LDAP integration

Figure 5-2 on page 160 summarizes client configuration options covered in this chapter. This section discusses authentication of user information against the LDAP server, thereby allowing login by all users in the LDAP database. 5.4, “Restriction of user login with compat mode (netgroups)” on page 164, describes the use of netgroups to restrict login to groups of users named in the LDAP netgroup database.

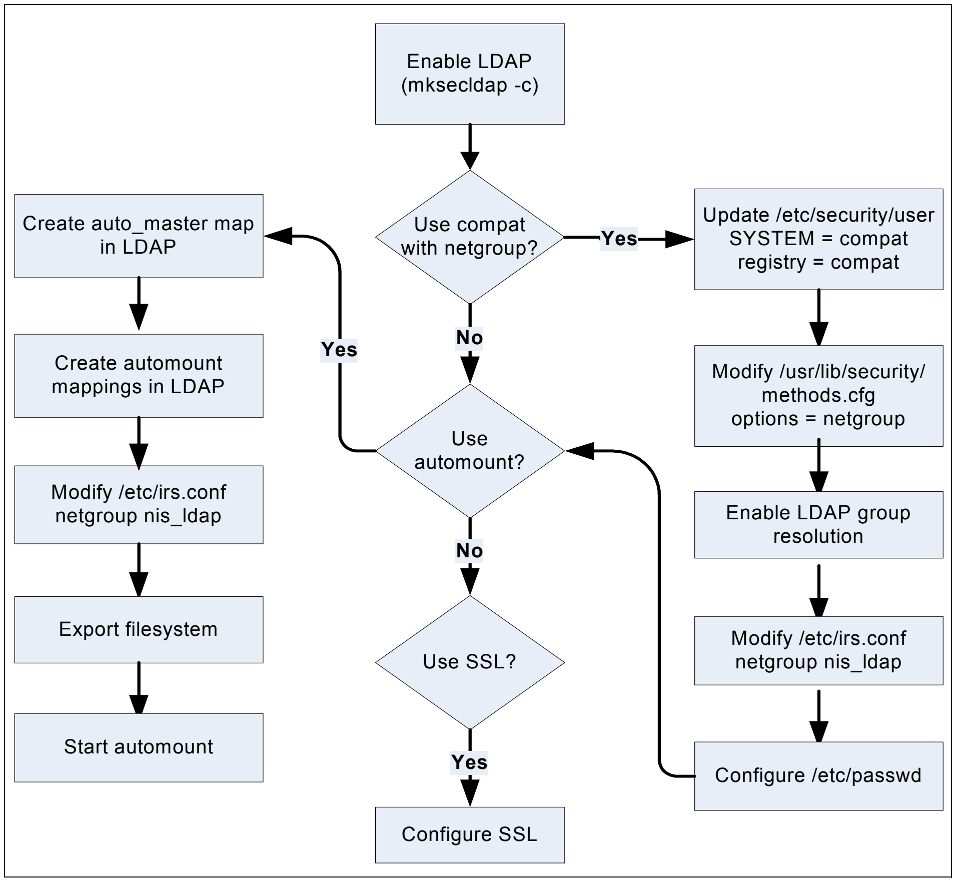

Enable LDAP (mksecldap -c)
Use compat with netgroup?
Yes
Update /etc/security/user SYSTEM = compat registry = compat
Modify /usr/lib/security/ methods.cfg options = netgroup
Enable LDAP group resolution
Modify /etc/irs.conf netgroup nis_ldap
Configure /etc/passwd
No
Use automount?
Yes
Create auto_master map in LDAP
Create automount mappings in LDAP
Modify /etc/irs.conf netgroup nis_ldap
Export filesystem
Start automount
No
Use SSL?
Yes
Configure SSL

Figure 5-2 LDAP client configuration

5.3.1 Client initialization

The **mksecldap** command is used to initialize the LDAP client. The -c option should be used to enable LDAP on the client. The following options are required to enable the LDAP client:

```
mksecldap -c -h <server> -a <binddn> -w <binddn password> -d <basedn>
```

Where:

server	LDAP server (or list of servers separated by commas).
binddn	Distinguished name (DN) that the client uses to bind to the LDAP server to retrieve data. binddn identifies a user with specific access to the LDAP database.

binddn password	Password of DN.
basedn	Base directory on the LDAP server where the client searches for data. Note that the basedn must be the same on all servers in the above server list.

Example 5-1 mksecldap to initialize an AIX 5L client

```
# mksecldap -c -h sunldapserver -a "cn=proxy,ou=profile,dc=example,dc=com" -p
password -d "dc=example,dc=com"
```

Note: The binddn is used by the client to retrieve LDAP data. The binddn needs to have read access to all RFC2307 data. If unix_auth is used as the authentication mechanism then it also needs read access to the userpassword attribute.

The above **`mksecldap`** command configures the /etc/security/ldap/ldap.cfg file, based on the specified options as well as data stored on the LDAP server. The LDAP client daemon secldapclntd reads the ldap.cfg file during startup.

`mksecldap` also appends the following entry to /etc/inittab to ensure that the security LDAP daemon starts automatically during system boot up:

```
ldapclntd:2:once: /usr/sbin/secldapclntd > /dev/console 2>&1
```

To verify the LDAP client is configured properly:

1. Check the AIX-LDAP attribute mapping settings in /etc/security/ldap/ldap.cfg. Since we are using rfc2307 schema, the mapping should be set to rfc2307user.map and rfc2307group.map, as shown in Example 5-2.

Example 5-2 Partial contents of /etc/security/ldap/ldap.cfg file

```
# AIX-LDAP attribute map path.
userattrmappath:/etc/security/ldap/2307user.map
groupattrmappath:/etc/security/ldap/2307group.map
```

2. Check the DN settings in /etc/security/ldap/ldap.cfg to confirm that the correct LDAP containers are used. The commented lines in Example 5-3 mean that the LDAP container does not exist on the LDAP server.

Example 5-3 Partial contents of /etc/security/ldap/ldap.cfg file

```
# Base DN where the user and group data are stored in the LDAP server.
# e.g., if user foo's DN is: uid=foo,ou=people,cn=aixdata
# then the user base DN is: ou=people,cn=aixdata
userbasedn:ou=people,dc=example,dc=com
groupbasedn:ou=group,dc=example,dc=com
servicebasedn:ou=services,dc=example,dc=com
```

```
protocolbasedn:ou=protocols,dc=example,dc=com
networkbasedn:ou=networks,dc=example,dc=com
netgroupbasedn:ou=netgroup,dc=example,dc=com
rpcbasedn:ou=rpc,dc=example,dc=com
automountbasedn:dc=example,dc=com
aliasbasedn:ou=aliases,dc=example,dc=com
bootparambasedn:ou=ethers,dc=example,dc=com
etherbasedn:ou=ethers,dc=example,dc=com
```

Note: The **mksecldap** command sets up the DN mappings based on the existence of the LDAP containers. If the LDAP data did not exist at the time that **mksecldap** was run, the DN mappings would not be set up properly. In that case, rerun **mksecldap** or manually modify /etc/security/ldap/ldap.cfg and restart secldapclntd after the LDAP containers are created.

3. Run **ls-secldapclntd** to verify the LDAP settings.

Example 5-4 Output of ls-secldapclntd command

```
# ls-secldapclntd
ldapservers=sunldapserver
ldapport=389
ldapversion=3
userbasedn=ou=people,dc=example,dc=com
groupbasedn=ou=group,dc=example,dc=com
idbasedn=
usercachesize=1000
usercacheused=0
groupcachesize=100
groupcacheused=0
cachetimeout=300
heartbeatT=300
numberofthread=10
connectionsperserver=5
alwaysmaster=no
authtype=UNIX_AUTH
searchmode=ALL
defaultentrylocation=LDAP
ldaptimeout=60
userobjectclass=account,posixaccount,shadowaccount
groupobjectclass=posixgroup
```

4. Run **lsldap** to make sure that the client can retrieve data from the server. If this command works the LDAP environment is operating correctly.

Example 5-5 Sample lsldap output

```
# lsldap -a passwd user1
dn: uid=user1,ou=people,dc=example,dc=com
objectClass: top
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
uid: user1
cn: ITSO user1
gecos: ITSO user1
loginShell: /bin/csh
uidNumber: 20000
gidNumber: 20000
homeDirectory: /home/user1
userPassword: {crypt}hTQBG25y7pz6Q
```

Note: AIX 5L Version 5.3 ML3 introduces a password encryption feature. The binddn password can now be stored encrypted in the /etc/security/ldap/ldap.cfg file. This is a security enhancement to protect the password in the client's LDAP configuration file. The LDAP client daemon can read either clear text or the encrypted password to provide backward compatibility.

Important: Prior to the release of APAR IY77803, if automount is used in an AIX 5L Version 5.3 ML03 system, the binddn would need to be set in clear text in the /etc/security/ldap/ldap.cfg file. This was due to the fact that the AIX 5L Version 5.3 ML03 automounter would not support encrypted passwords in ldap.cfg.

You can apply APAR IY7803: “AutoFS Support for BIND Passwd Encryption” to support encrypted passwords in ldap.cfg. The full APAR description can be found at the following Web site:

```
http://www-1.ibm.com/support/docview.wss?rs=111&context=SWG10&q1=IY77803&uid
=isg1IY77803&loc=en_US&cs=utf-8&lang=en
```

When the fix is publicly released, it can be downloaded from the following Web site:

```
http://www.ibm.com/servers/eserver/support/unixservers/aixfixes.html
```

In the meantime, if you want a subscription service to notify you by e-mail when the PTF for the APAR is available, you can do so at the following Web site:

```
http://www14.software.ibm.com/webapp/set2/aparsubscriptions/bjyqvcmjd
```

5.4 Restriction of user login with compat mode (netgroups)

Note: LDAP netgroup support is only supported in AIX 5L Version 5.3 and later.

The password compatibility mode provides the capability to restrict user login based on the configuration in the /etc/passwd file, rather than allowing all LDAP users to log in to the system. The syntax “+user” in /etc/passwd allows the system administrator to specify which LDAP users have access to a system. The syntax “+@netgroup” in /etc/passwd identifies a user type netgroup defined on the LDAP server. In either case, the LDAP user needs to authenticate against the LDAP server in order to log in successfully. For example, a particular system allows login for user USER1 if any of the following is true:

- USER1 is defined locally in /etc/passwd.
- The line “+USER1” appears in /etc/passwd and USER1 is a user defined on the LDAP server.

- The line “+@NETGROUP1” is specified in the passwd file and USER1 is an LDAP user who is a member of NETGROUP1, a netgroup defined on the LDAP server.

The rest of this section shows how to enable compatibility mode on the AIX 5L LDAP client to restrict login access for LDAP users.

5.4.1 Step 1: Update the default stanza in /etc/security/user

Change the SYSTEM and registry lines in the default stanza in the /etc/security/user file from LDAP to compat, as shown in Example 5-6.

Example 5-6 Modified default stanza in /etc/security/user

```
default:
        admin = false
        login = true
        su = true
        daemon = true
        rlogin = true
        sugroups = ALL
        admgroups =
        ttys = ALL
        auth1 = SYSTEM
        auth2 = NONE
        tpath = nosak
        umask = 022
        expires = 0
        SYSTEM = compat
        registry = compat
        logintimes =
        pwdwarntime = 0
        account_locked = false
        loginretries = 0
        histexpire = 0
        histsize = 0
        minage = 0
        maxage = 0
        maxexpired = -1
        minalpha = 0
        minother = 0
        minlen = 0
        mindiff = 0
        maxrepeats = 8
        dictionlist =
        pwdchecks =
```

Tip: For compat mode *do not* use quotation marks (“ “) on the registry line. The text registry = “compat” causes login problems.

5.4.2 Step 2: Enable LDAP netgroup support

Add the netgroup option to the LDAP stanza in /usr/lib/security/methods.cfg, as shown in Example 5-7. This enables netgroup support in the LDAP environment.

Example 5-7 Modified LDAP stanza in /usr/lib/security/methods.cfg

```
LDAP:
        program = /usr/lib/security/LDAP
        program_64 =/usr/lib/security/LDAP64
        options = netgroup
```

If compat mode is to be disabled in the future the “options = netgroup” line should be removed from this file to avoid login problems.

5.4.3 Step 3: Enable compat mode resolution for LDAP groups

Append a plus sign (+) (on a line by itself) to the end of the /etc/group file, as shown in Example 5-8 on page 166. The plus sign (+) is necessary to enable the LDAP client to use the LDAP directory for group resolution. Otherwise, the client would only use the local /etc/group file.

Example 5-8 Modified /etc/group for compat mode

```
system:!:0:root
staff:!:1:ipsec,sshd,ldap
bin:!:2:root,bin
sys:!:3:root,bin,sys
adm:!:4:bin,adm
uucp:!:5:uucp,nuucp
mail:!:6:
security:!:7:root
cron:!:8:root
printq:!:9:lp
audit:!:10:root
ecs:!:28:
nobody:!:4294967294:nobody,lpd
usr:!:100:guest
perf:!:20:
shutdown:!:21:
lp:!:11:root,lp,printq
snapp:!:12:snapp
invscout:!:13:invscout
```

```
ipsec:!:200:
sshd:!:201:sshd
ldap:!:202:ldap
+
```

5.4.4 Step 4: Modify /etc/irs.conf

In general, the /etc/irs.conf file is used to specify the type of name service and search order that the client should use for name resolution. In this case, append `netgroup nis_ldap` to the /etc/irs.conf file, as shown in Example 5-9, to configure the client to use LDAP to look up netgroup information.

Example 5-9 /etc/irs.conf with netgroup lookup from the LDAP server

```
# cat /etc/irs.conf
netgroup nis_ldap
```

5.4.5 Step 5: Configure /etc/passwd to control login access

Modify the /etc/passwd file, as shown in Example 5-10, by adding the desired LDAP users and netgroups to the end. Add authorized LDAP users with the syntax `+username`. Add authorized netgroups with the syntax `+@netgroupname`.

Example 5-10 Modified /etc/passwd with +@netgroup and +user

```
root:!:0:0::/home/root:/usr/bin/ksh
daemon:!:1:1::/etc:
bin:!:2:2::/bin:
sys:!:3:3::/usr/sys:
adm:!:4:4::/var/adm:
uucp:!:5:5::/usr/lib/uucp:
guest:!:100:100::/home/guest:
nobody:!:4294967294:4294967294::/:
lpd:!:9:4294967294::/:
lp:*:11:11::/var/spool/lp:/bin/false
snapp:*:200:12:snapp login user:/usr/sbin/snapp:/usr/sbin/snappd
nuucp:*:6:5:uucp login user:/var/spool/uucppublic:/usr/sbin/uucp/uucico
invscout:*:7:13::/var/adm/invscout:/usr/bin/ksh
ipsec:*:201:1::/etc/ipsec:/usr/bin/ksh
sshd:*:202:201::/var/empty:/usr/bin/ksh
ldap:*:203:1::/home/ldap:/usr/bin/ksh
fred:*:204:1::/home/fred:/usr/bin/ksh
someone:*:205:1::/home/someone:/usr/bin/ksh
+@itso_users
+user9
```

The **lsldap** command can display user information from the LDAP server (Example 5-11).

Example 5-11 lsldap output for an LDAP user in compat mode

```
# lsldap -a passwd user9
dn: uid=user9,ou=people,dc=example,dc=com
objectClass: top
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
uid: user9
cn: ITSO user9
gecos: ITSO user9
loginShell: /bin/csh
uidNumber: 20000
gidNumber: 20000
homeDirectory: /home/user9
userPassword: {crypt}hTQBG25y7pz6Q
```

Since user9 is listed in /etc/passwd, the system allows user9 to log in (Example 5-12).

Example 5-12 Compat mode login of an LDAP user

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2005.
login: user9
user9's Password:
$ id
uid=20009(user9) gid=20000(itso)
$
```

Lsldap can also display information about LDAP netgroups, as shown in Example 5-13.

Example 5-13 lsldap output for an LDAP user netgroup

```
# lsldap -a netgroup itso_users
   dn: cn=itso_users,ou=netgroup,dc=example,dc=com
   cn: itso_users
   objectClass: top
   objectClass: nisNetGroup
   nisNetgroupTriple: (-,user1,)
   nisNetgroupTriple: (-,user2,)
   nisNetgroupTriple: (-,user3,)
   nisNetgroupTriple: (-,user4,)
   nisNetgroupTriple: (-,user5,istoNISdomain)
```

The nisNetgroupTriple field in the above netgroup entry has the syntax (hostname, username, NISdomainname). Various combinations of these fields can be used to restrict login in the netgroup definition, based on local needs. The above triple for user5 only allows that user to log in to systems having the NIS domain set to itsoNISdomain.

With the addition of `+@itso_users` in the passwd file (see then end of the file in Example 5-10 on page 167), all of the members of the itso_users netgroup have login access to the system (demonstrated in Example 5-14), assuming the proper NIS domain for user5 noted in the previous paragraph.

Example 5-14 Compat mode login of a netgroup member

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2005.
login: user2
user2's Password:
$ id
uid=20002(user2) gid=20000(itso) groups=20001(redbook)
```

The system denies user login in the normal way if the user is not listed in /etc/passwd, and the user is not identified as an LDAP user, and the user is not a member of an LDAP netgroup specified in /etc/passwd. See Example 5-15.

Example 5-15 Failed compat mode login

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2005.
login: user6
user6's Password:
You entered an invalid login name or password.
login:
```

In AIX 5L Version 5.3 ML03, if more than one netgroup is referenced in the /etc/passwd file then the client must perform multiple netgroup lookups. To minimize the impact of this process, we propose a solution where you use only one netgroup in the /etc/passwd file. If there are multiple netgroups that need access to the system, create a nested netgroup by using the memberNisNetgroup attribute to put multiple netgroups as members to a netgroup, and specify the nested netgroup in the passwd file.

Example 5-16 shows a nested netgroup. *itso2* is a nested netgroup member of netgroup *itso_users*.

Example 5-16 Nested netgroup method for limiting client lookups

```
# lsldap -a netgroup itso_users
dn: cn=itso_users,ou=netgroup,dc=example,dc=com
cn: itso_users
objectClass: top
objectClass: nisNetGroup
nisNetgroupTriple: (-,user1,)
nisNetgroupTriple: (-,user2,)
nisNetgroupTriple: (-,user3,)
memberNisNetgroup: itso2

# lsldap -a netgroup itso2
dn: cn=itso2,ou=netgroup,dc=example,dc=com
cn: itso2
objectClass: top
objectClass: nisNetGroup
nisNetgroupTriple: (-,user7,)
```

By using a nested netgroup, you reference only one netgroup in /etc/passwd.

Important: Example 5-16 on page 170 shows the use of nested netgroups to limit the system impact of performing multiple client lookups and should not be necessary if you apply that APAR IY76428: “Logins Fail when Multiple Netgroups are Specified.” The full APAR description can be found at the following Web site:

http://www-1.ibm.com/support/docview.wss?rs=111&context=SWG10&q1=IY76428&uid=isg1IY76428&loc=en_US&cs=utf-8&lang=en

When the fix is publicly released, it can be downloaded at the following Web site:

http://www.ibm.com/servers/eserver/support/unixservers/aixfixes.html

In the meantime, if you want a subscription service to notify you by e-mail when the PTF for the APAR is available, you can do so at the following Web site:

http://www14.software.ibm.com/webapp/set2/aparsubscriptions/bjyqvcmjd

5.4.6 Step 6: Configure .rhosts and /etc/hosts.equiv to use netgroups

The AIX 5L client supports LDAP host netgroup references in the $HOMEDIR/.rhosts and /etc/hosts.equiv files. Add the text `+@netgroup` to those files to enable host netgroup members to remotely execute commands on the host. This section provides configuration examples for both the .rhosts and the /etc/hosts.equiv files.

.rhosts configuration

In Example 5-17, netgroup itso_blades is a host netgroup referenced in the root user's .rhosts file.

Example 5-17 .rhosts with ldap hosts netgroup

```
# hostname
bs01b07
# cat /.rhosts
+@itso_blades
#
# lsldap -a netgroup itso_blades
dn: cn=itso_blades,ou=netgroup,dc=example,dc=com
cn: itso_blades
objectClass: top
objectClass: nisNetGroup
nisNetgroupTriple: (bs01b01.itsc.austin.ibm.com,-,)
nisNetgroupTriple: (bs01b02.itsc.austin.ibm.com,-,)
nisNetgroupTriple: (bs01b11.itsc.austin.ibm.com,-,)
nisNetgroupTriple: (bs01b07.itsc.austin.ibm.com,-,)
```

As a result (Example 5-18), root can remotely execute a command on any host defined in the netgroup.

Example 5-18 Netgroup member remotely executes a command on another host

```
# hostname
bs01b11.itsc.austin.ibm.com
# id
uid=0(root) gid=0(system)
groups=2(bin),3(sys),7(security),8(cron),10(audit),11(lp)
#
# rsh bs01b07 hostname
bs01b07
```

/etc/hosts.equiv configuration

In a similar way (Example 5-19) LDAP netgroups can be used in the /etc/hosts.equiv file to control hosts and users that can remotely execute commands.

Example 5-19 Sample hosts.equiv file with user type netgroup

```
# hostname
bs01b11.itsc.austin.ibm.com
# cat /etc/hosts.equiv
bs01b07 +@itso_users
# lsldap -a netgroup itso_users
dn: cn=itso_users,ou=netgroup,dc=example,dc=com
cn: itso_users
objectClass: top
objectClass: nisNetGroup
nisNetgroupTriple: (-,user1,)
nisNetgroupTriple: (-,user2,)
nisNetgroupTriple: (-,user3,)
```

With the configuration shown in Example 5-19, any user in the itso_users netgroup on host bs01b07 can remotely execute a command on host bs01b11, as shown in Example 5-20.

Example 5-20 remotely execute a command

```
% hostname
bs01b07
%
% id
uid=20000(user1) gid=20000(itso) groups=20001(redbook)
%
% rsh bs01b11 hostname
bs01b11.itsc.austin.ibm.com
%
```

Both host type and user type netgroups (Example 5-21) can be used in the /etc/hosts.equiv file to control which remote hosts and users can execute a command.

Example 5-21 Sample /etc/hosts.equiv file with hosts and user type netgroups

```
# hostname
bs01b11.itsc.austin.ibm.com
#
# cat /etc/hosts.equiv
+@itso_blades +@itso_users
```

Now users that are members of the itso_users netgroup can remotely execute commands from a remote host that is a member of the itso_blades netgroup (Example 5-22).

Example 5-22 Remotely execute command when user and host are netgroup members

```
% hostname
bs01b07
%
% id
uid=20000(user1) gid=20000(itso) groups=20001(redbook)
%
% rsh bs01b11 hostname
bs01b11.itsc.austin.ibm.com
```

5.5 automount configuration

The AIX 5L automounter is LDAP enabled. The automount daemon (automountd) can retrieve automount information from LDAP or local files, according to information entered in /etc/irs.conf. The AIX 5L Version 5.3 automount daemon supports two automount objectclass pairs. The nisobject and nismap objectclass pair was in the original RFC2307 definition. The automount and automountmap objectclass pair was added in the RFC2307bis definition. This section describes an automount implementation using the automount/automountmap objectclass pair.

5.5.1 Create auto_master mapping in the LDAP server

The auto_master map is read by automountd when it starts. Example 6-21 shows the .ldif file that defines the auto_master map and an entry for the /home file system. The .ldif file is followed by output of the command used to add those definitions to the LDAP server. Blank lines are used to separate stanzas in .ldif files. The /home stanza tells automountd to search for automount information for the /home file system in the LDAP table auto_home.

If automount managed another file system, /apps for example, there would be another stanza in the .ldif file similar to the /home stanza (Example 5-23) to tell automountd where to look for information for mounting /apps.

Example 5-23 Adding an auto_master map to the LDAP server

```
#hostname
bs01b07
# cat /tmp/auto_master.ldif
dn: automountMapName=auto_master,dc=example,dc=com
```

```
objectClass: automountMap
objectClass: top
automountMapName: auto_master

dn: automountKey=/home,automountMapName=auto_master,dc=example,dc=com
objectClass: automount
objectClass: top
automountKey: /home
automountInformation: auto_home

# ldapadd -h sunldapserver -D "cn=directory manager" -w password -f
/tmp/auto_master.ldif
adding new entry automountMapName=auto_master,dc=example,dc=com

adding new entry
automountKey=/home,automountMapName=auto_master,dc=example,dc=com
```

Note: The AIX 5L client also supports the nisMap objectclass. In this example, the auto_master entry is created with the nisMap objectclass:

```
dn: nismapname=auto_master,dc=example,dc=com
nismapname: auto_master
objectClass: top
objectClass: nisMap

dn: cn=/home,nismapname=auto_master,dc=example,dc=com
objectClass: nisObject
objectClass: top
nisMapName: auto_master
nisMapEntry: auto_home
cn: /home
```

5.5.2 Create an auto_home entry in LDAP

In Example 5-23 on page 173 an entry for /home was added to the LDAP server stating that information for mounting /home is specified in a table called auto_home. The auto_home table contains entries mapping individual user names to the full path location of their home directory. Example 5-24 shows a .ldif file with an auto_home definition for two users with home directories on different hosts plus the output of the command adding those definitions to the LDAP server. In the user stanzas the automountMAP attribute contains the key (user ID) and the automountInformation shows where the user's home directory is located.

Example 5-24 ldif file with auto_home and multiple users with home directory on different hosts

```
# cat /tmp/auto_home.ldif
dn: automountMapName=auto_home,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto_home

dn: automountKey=user1,automountMapName=auto_home,dc=example,dc=com
objectClass: automount
objectClass: top
automountKey: user1
automountInformation: bs01b07:/var/tmp/home/user1

dn: automountKey=user2,automountMapName=auto_home,dc=example,dc=com
objectClass: automount
objectClass: top
automountKey: user2
automountInformation: bs01b06:/var/tmp/home/user2


# ldapadd -h sunldapserver -D "cn=directory manager" -w password -f
/tmp/auto_home.ldif
adding new entry automountMapName=auto_home,dc=example,dc=com

adding new entry
automountKey=user1,automountMapName=auto_home,dc=example,dc=com

adding new entry
automountKey=user2,automountMapName=auto_home,dc=example,dc=com
```

Note: The AIX 5L client supports the isobject objectclass as well as the automount objectclass. In this example, the user1 auto_home entry is created with the nisObject objectclass:

```
dn: nismapname=auto_home,dc=example,dc=com
nismapname: auto_home
objectClass: top
objectClass: nisMap

dn: cn=user1,nismapname=auto_home,dc=example,dc=com
objectClass: nisObject
objectClass: top
nisMapName: auto_home
nisMapEntry: bs01b07:/var/tmp/home/user1
cn=user1
```

Tip: The automounter expects either the automount or nisobject objectclass, but not both. Administrators should be consistent in defining automount objects.

5.5.3 Update /etc/irs.conf

Append entries to /etc/irs.conf to specify the name service that automount should use to retrieve automount information and the name service that exportfs will use to resolve netgroup information. In Example 5-25 both automount and netgroup lookup is configured to search from LDAP only.

Example 5-25 Modified /etc/irs.conf for automount and netgroup lookup via LDAP

```
# cat /etc/irs.conf
automount nis_ldap
netgroup nis_ldap
```

Note: It is possible to specify multiple name services for automount in the /etc/irs.conf file, separated by white space. The automount daemon will resolve automount entries according to the order specified in irs.conf. In the example of *automount files nis_ldap*, automount searches from local files first, prior to LDAP. This can be used to create local automount entries to override entries in the LDAP database.

5.5.4 Export NFS file systems by LDAP netgroup

The file systems referenced by the auto_home table entries need to be NFS exported from the systems where they reside. AIX 5L Version 5.3 RML02 allows the hosts permitted to mount an exported file system to be defined using LDAP netgroups in the exportfs command. The nisNetGroup objectclass, defined in RFC2307, names the hosts defined in the netgroups. Please refer to 3.2.4, “RFC2307bis schema” on page 65. The nisNetgroupTriple should contain fully qualified host names, especially if DNS is used for host resolution. A sample host type netgroup and file system export is shown in Example 5-26. In a production environment the /etc/exports file and **`exportfs`** commands would be defined and executed on every system where home directories need to be exported from.

Example 5-26 Sample host type netgroup and file system export

```
# lsldap -a netgroup itso_blades
dn: cn=itso_blades,ou=netgroup,dc=example,dc=com
cn: itso_blades
objectClass: top
objectClass: nisNetGroup
```

```
nisNetgroupTriple: (bs01b01.itsc.austin.ibm.com,-,)
nisNetgroupTriple: (bs01b02.itsc.austin.ibm.com,-,)
nisNetgroupTriple: (bs01b11.itsc.austin.ibm.com,-,)
nisNetgroupTriple: (bs01b07.itsc.austin.ibm.com,-,)

# cat /etc/exports
/test/home -rw,access=itso_blades
#
# exportfs -a
# exportfs
/test/home    -rw,access=itso_blades
```

5.5.5 Start automount daemon

Once configuration is complete, start automountd by running /usr/sbin/automount manually or rebooting the system. The automountd daemon should start.

To verify that automount is working properly, log in as a user and the home directory should mount automatically (Example 5-27).

Example 5-27 Log in with automount of the home directory

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2005.
login: user1
user1's Password:
% hostname
bs01b11.itsc.austin.ibm.com
%
% pwd
/home/user1
%
% df -k .
Filesystem    1024-blocks      Free %Used    Iused %Iused Mounted on
bs01b07:/var/tmp/home/user1      524288    514404    2%      355     1%
/home/user1
%
% ls -al
total 9
drwxrwxr-x   2 user1    itso          4096 Sep 22 09:32 .
dr-xr-xr-x   2 root     system           2 Sep 21 18:58 ..
-rw-r--r--   1 user1    itso            24 Sep 22 09:32 test.user1
%
% cat test.user1
this file is on bs01b07
%
```

5.6 SSL configuration

SSL is optionally used to implement secure communications between the LDAP client and server. To be operational SSL has to be configured on both the client and the server. Server configuration includes generating a certificate request, obtaining a server cert from a certificate authority (CA), and installing the server and CA certificates. Particular configuration of SSL for the Sun ONE directory is beyond the scope of this book. Please refer to the Sun ONE Directory Server installation guide for more details.

In this section, we assume a valid SSL configuration on the LDAP server and go through the steps to configure SSL on the AIX 5L client. The configuration steps are to create a key database on the AIX 5L client, install the CA certificate (the same certificate used on the server) in the client database, and make changes to the /etc/security/ldap/ldap.cfg file to specify use of SSL.

5.6.1 Create a key database and install the CA certificate

The following packages are required for SSL configuration on AIX 5L Version 5.3 client:

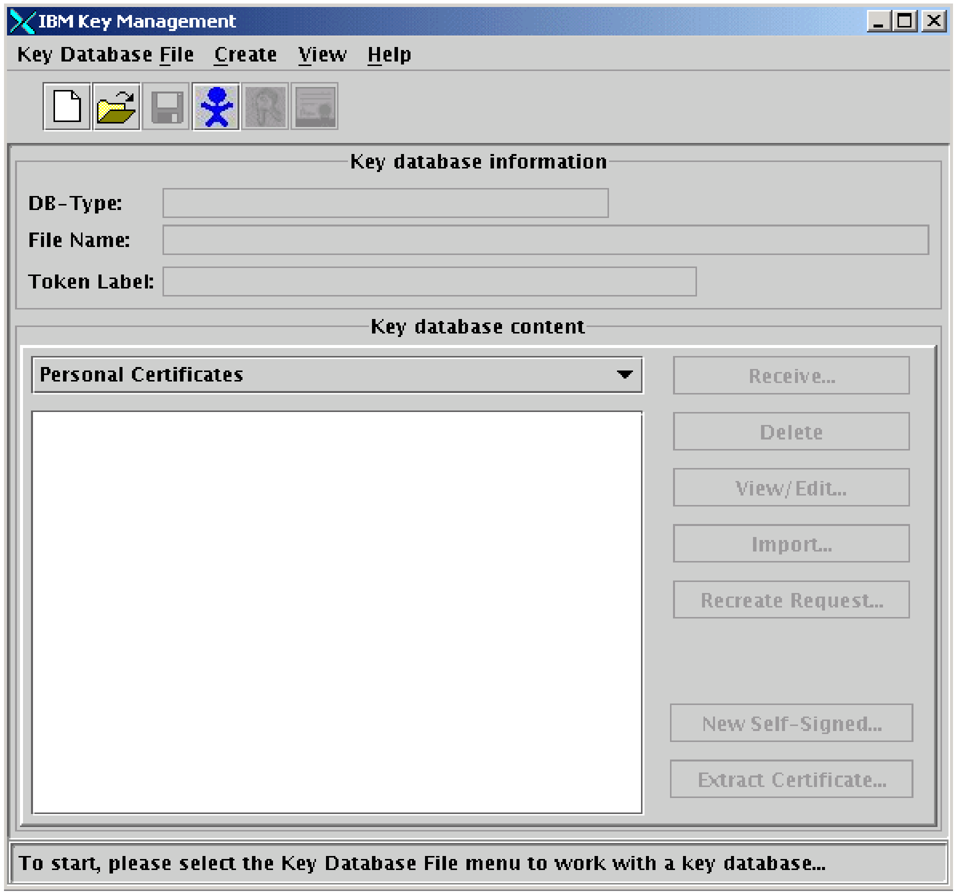
- gskta.rte
- ldap.max_crypto_client.adt
- ldap.max_crypto_client.rte

If the above packages are not installed on the system, install them from the AIX 5L Version 5.3 Expansion Pack CD (5705-603).

The gskta.rte component includes a Java-based key manager utility called GSK that creates a key database. GSK also imports the client and CA certificates into the key database.

Use **gsk7ikm** to create a key database on the LDAP client system:

```
# /usr/opt/ibm/gskta/bin/gsk7ikm
```

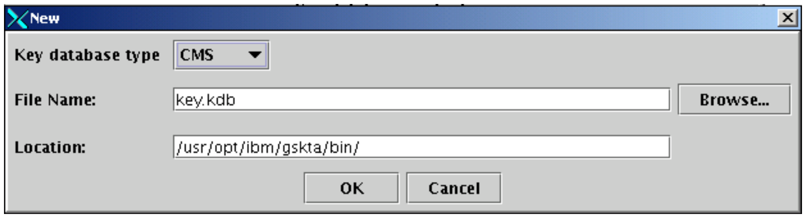


To start, please select the Key Database File menu to work with a key database...

Figure 5-3 Initial screen of gsk7ikm

On the Key Manager screen shown in Figure 5-3 select **Key Database File** → **New** and then choose **CMS** as the key database type. Then enter the file name and directory for the key database.

This would bring up a new window (Figure 5-4). Use the menu to set the key database type to CMS. Then specify the location and file name of the database file, and click **OK** to proceed.



Key database type: CMS
File Name: key.kdb
Location: /usr/opt/ibm/gskta/bin/

Figure 5-4 Input screen to create new key database

Enter a password for the database (Figure 5-5). Select the **Stash the password to a file** option if you do not want to enter the password next time.

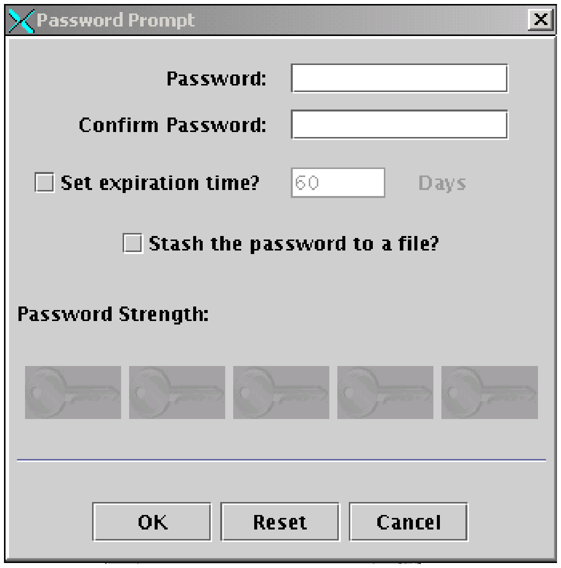
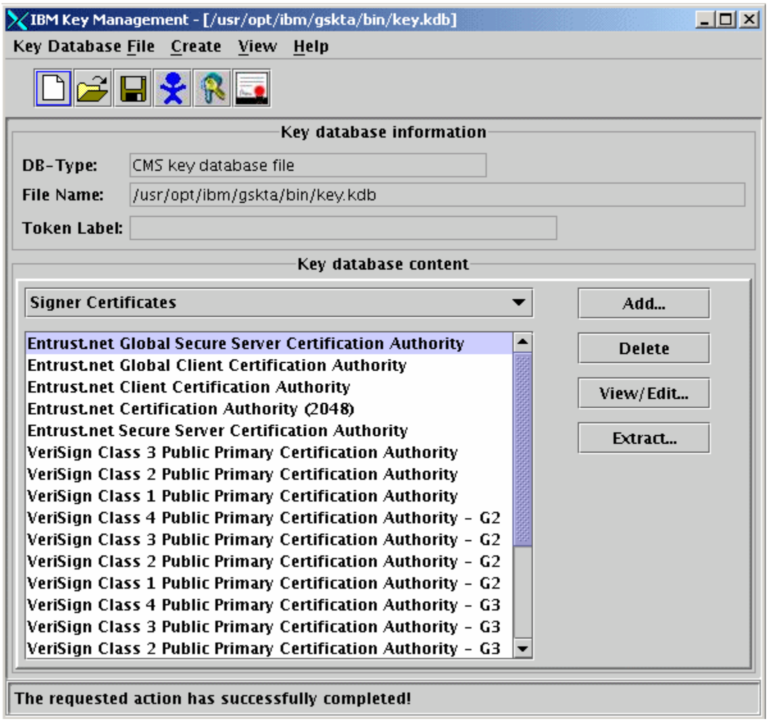


Figure 5-5 Set database password screen

Figure 5-6 shows the signer certificates that are currently installed in the database.


IBM Key Management - [/usr/opt/ibm/gskta/bin/key.kdb]
Key Database File Create View Help
Key database information
Key database content
Add...
Delete
View/Edit...
Extract...

DB-Type: CMS key database file
File Name: /usr/opt/ibm/gskta/bin/key.kdb
Token Label:

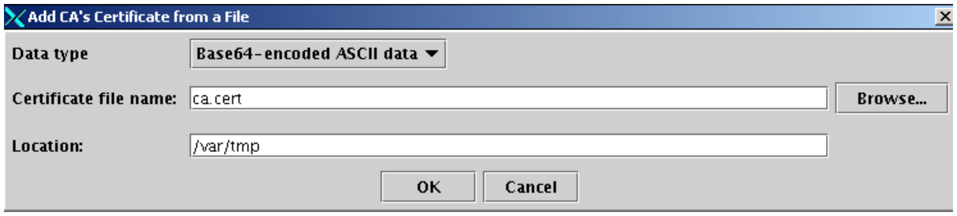
Signer Certificates

Entrust.net Global Secure Server Certification Authority
Entrust.net Global Client Certification Authority
Entrust.net Client Certification Authority
Entrust.net Certification Authority (2048)
Entrust.net Secure Server Certification Authority
VeriSign Class 3 Public Primary Certification Authority
VeriSign Class 2 Public Primary Certification Authority
VeriSign Class 1 Public Primary Certification Authority
VeriSign Class 4 Public Primary Certification Authority - G2
VeriSign Class 3 Public Primary Certification Authority - G2
VeriSign Class 2 Public Primary Certification Authority - G2
VeriSign Class 1 Public Primary Certification Authority - G2
VeriSign Class 4 Public Primary Certification Authority - G3
VeriSign Class 3 Public Primary Certification Authority - G3
VeriSign Class 2 Public Primary Certification Authority - G3

The requested action has successfully completed!

Figure 5-6 Default server certificate panel

As seen in Figure 5-6, the GSK automatically includes certificate signatures for many of the common CA authorities. If one of these CA authorities is used then no further actions are required and you can exit the GSK utility. For this book the server and CA certificates were both issued by an in-house CA, so the CA certificate has to be added to the key database on each client. To add a CA certificate to the key database, select the **ADD** button. Enter the file name of where you have previously saved the CA certificate (Figure 5-7).



Data type: Base64-encoded ASCII data
Certificate file name: ca.cert
Location: /var/tmp

Figure 5-7 Input screen to add CA certificate to the key database

In Figure 5-7 on page 181, /var/tmp/ca.cert is the CA certificate. The content of the certificate is shown in Example 5-28.

Example 5-28 A sample CA certificate (modified)

```
# cat /var/tmp/ca.cert
-----BEGIN CERTIFICATE-----
ZnxzGzCCAoSgAwIBAgIBADANBgkqhkiG9wOBAQQacaroMlliBQYDVQQGEwJ1czE7
Miprf1UECBMCVFgxDzANBgNVBAcTBkF1c3RpbjkheisHlihPChMLZXhhbXBsZSfj
b2lzcJAJBgNVBAMTAkNBMROwGwYJKoZIhvcNAQkreweoLnitraMtcGxlLmNvboAe
FwOlwpe5MDgxNTIwNTJaFwOwNjA5MDgxNTIwNTJythgareGdEgNVBAYTAnVztQsw
CQYDVQ4vEwJUWDEPMAOGA1UEBxMGQXVzdGluMRQdnomlAsirhCtleGFtcGxeLmNv
bTELMotcp1UEAxMCQOExHTAbBgkqhkiG9wOBCQEWDmNhQGV4YW1wbGUuY2ntMIGf
MAOGCS3uawb3DQEBAQUAA4GNADCBiQKBgQC8K8kCW83CDm36X+COqQyaywFhKaBO
kM1C5+78ptaSdxJjFg/XftqzYIf6i1Jt5fPiiO3x/C1BLbY+8U3OFyPtpOiKOKkR
fDPIEsR8+ibmAsFaR/d9YBytRDjwP1qz2YT3NXIekwfi91JlV3LGQM1qhI++cJeO
rRxvKvHemaXnywIDAQABo4HKMIHHMBOGA1UdDgQWBBRP+zR+YlZ312w9Dgf7QAIt
ft9CuDqBlwYD5ROjBIGPMIGMgBRP+zR+YlZ312w9DMf7QAItft9CucFxpGbwbTEL
MAkGAsUEBhMCdOMxCzAJBgNVBAgTAlRYMQ8wDQYDVQQHEwZBdXNOrW4xFDAyBgNV
BAoTH2V4YW1wbGUuY29tMQswCQYDVQQDEwJDQTEdMBsGCSqGSIb3DQEJARYOw2FA
ZXhnbXBsZS5jb22CAQAwDAYDVROTBAUwAwEB/zANBgkqhkiG9wOBAQQFAAOBg=Br
8mi+/PWQtueDI+g+eFMHOcy5iKQniBo1/oyv2DGC5SLZNnsqMCjLE7B4nWEepgdJ
Q8bcni1l7xiGIQqGMGprM8ClTU9jvZzC3a9dFerCWOaVTOVJIy3PjdXkyK5TYtXK
1h5LmDrORi/OJNCEWeQHXquT8NaDP8NL+j4QacbP1Q==
-----END CERTIFICATE-----
```

Enter a label for the certificate (Figure 5-8).

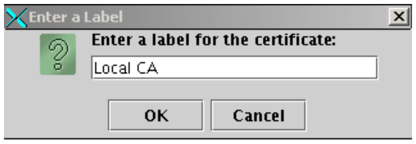
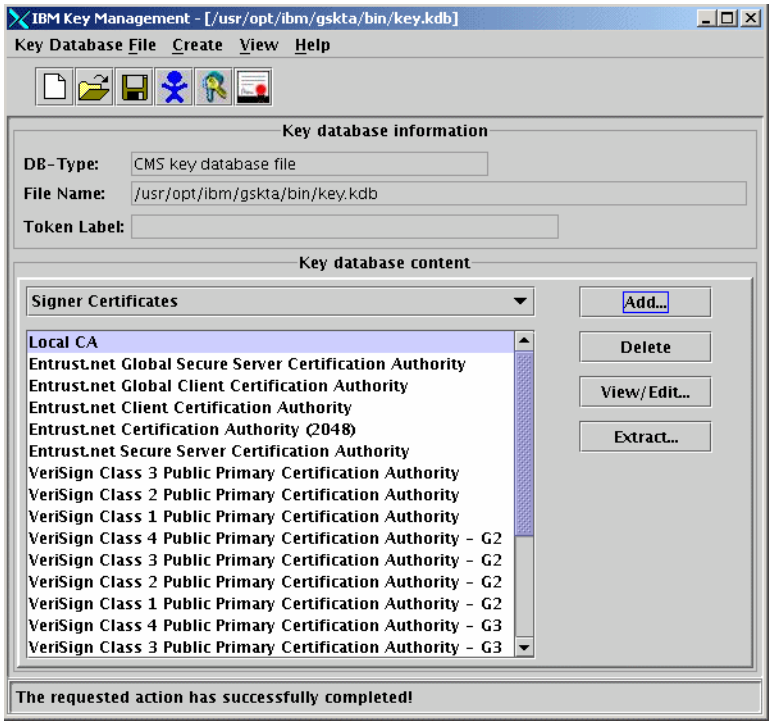


Figure 5-8 Certificate label


IBM Key Management - [/usr/opt/ibm/gskta/bin/key.kdb]

Key Database File Create View Help

Key database information

DB-Type: CMS key database file

File Name: /usr/opt/ibm/gskta/bin/key.kdb

Token Label:

Key database content

Signer Certificates

Local CA
Entrust.net Global Secure Server Certification Authority
Entrust.net Global Client Certification Authority
Entrust.net Client Certification Authority
Entrust.net Certification Authority (2048)
Entrust.net Secure Server Certification Authority
VeriSign Class 3 Public Primary Certification Authority
VeriSign Class 2 Public Primary Certification Authority
VeriSign Class 1 Public Primary Certification Authority
VeriSign Class 4 Public Primary Certification Authority - G2
VeriSign Class 3 Public Primary Certification Authority - G2
VeriSign Class 2 Public Primary Certification Authority - G2
VeriSign Class 1 Public Primary Certification Authority - G2
VeriSign Class 4 Public Primary Certification Authority - G3
VeriSign Class 3 Public Primary Certification Authority - G3

The requested action has successfully completed!

Figure 5-9 Local CA added to key database

Once the CA certificate has been added to the database, we can verify the SSL connection. An easy way to test SSL is to run the **ldapsearch** command with the -Z option to establish an SSL connection to the server.

If SSL is configured properly on the server and the client, then the **ldapsearch -Z** option should respond without any error (Example 5-29). In this example **ldapsearch** uses the -K option to specify the location of the key database file, and the -P option to specify the key database password.

Example 5-29 ldapsearch with SSL

```
# ldapsearch -h sunldapserver -b "dc=example,dc=com" -Z -K
/usr/opt/ibm/gskta/bin/key.kdb -P password uid=user1 dn
uid=user1,ou=people,dc=example,dc=com
```

5.6.2 Change /etc/security/ldap/ldap.cfg to enable SSL

To enable SSL on the AIX 5L client, modify the following settings in the /etc/security/ldap/ldap.cfg:

1. Update useSSL to yes:

   ```
   useSSL:yes
   ```

2. Specify the SSL key database file location:

   ```
   ldapsslkeyf:/usr/opt/ibm/gskta/bin/key.kdb
   ```

3. Specify the SSL key database file password:

   ```
   ldapsslkeypwd:password
   ```

4. Change the SSL port if necessary.

   By default, the system uses 636 as the SSL port. If it is otherwise, change the port number by modifying the ldapsslport setting:

   ```
   ldapsslport:636
   ```

5. Restart secldapclntd.

The change is only reflected when secldapclntd restarts. Run **`ls-secldapntd`** to confirm that the client daemon connects to the server on the SSL port 636 (Example 5-30).

Example 5-30 Restart of secldapclntd and output of ls-secldapclntd with SSL active

```
# restart-secldapclntd
The secldapclntd daemon terminated successfully.
Starting the secldapclntd daemon.
The secldapclntd daemon started successfully.
#
# ls-secldapclntd
ldapservers=sunserver
ldapport=636
ldapversion=3
userbasedn=ou=people,dc=example,dc=com
groupbasedn=ou=group,dc=example,dc=com
idbasedn=
usercachesize=1000
usercacheused=0
groupcachesize=100
groupcacheused=0
cachetimeout=300
heartbeatT=300
numberofthread=10
connectionsperserver=5
alwaysmaster=no
authtype=UNIX_AUTH
```

```
searchmode=ALL
defaultentrylocation=LDAP
ldaptimeout=60
userobjectclass=account,posixaccount,shadowaccount
groupobjectclass=posixgroup
```

Another way to verify that SSL is working properly is to monitor the LDAP traffic. Use tcpdump or check the LDAP server log to confirm that LDAP traffic is sending across the SSL port.

After one client has been successfully configured it may take less time to configure other clients by transferring the key database file and the /etc/security/ldap/ldap.cfg files with FTP (binary mode) or another file transfer utility followed by secldapclntd restart.

Important: Prior to the release of APAR IY77904, the AIX 5L Version 5.3 ML03 client was not able to mount any automount directories if SSL was enabled. This problem should not occur if you apply that APAR IY77904: “Automount with SSL not Working.” The full APAR description can be found at the following Web site:

http://www-1.ibm.com/support/docview.wss?rs=111&context=SWG10&q1=IY77904&uid=isg1IY77904&loc=en_US&cs=utf-8&lang=en

When the fix is publicly released, it can be downloaded from the following Web site:

http://www.ibm.com/servers/eserver/support/unixservers/aixfixes.html

In the meantime, if you want a subscription service to notify you by e-mail when the PTF for the APAR is available, you can do so at the following Web site:

http://www14.software.ibm.com/webapp/set2/aparsubscriptions/bjyqvcmjd

6

Scenario: IBM Tivoli Directory Server V6.0 integration

This chapter contains a hands-on example of how to set up AIX 5L Version 5.3 LDAP clients to authenticate against an LDAP server (ITDS6.0). We show here:

- How to migrate users from a existing AIX environment to an LDAP server
- How to set up AIX clients to authenticate against an LDAP server with:
  - Administrative user ID
  - Proxy user ID using ldap_auth
- How to work with the LDAP directory using AIX commands

6.1 Existing environment

We want to migrate from an RS6000 SP environment with user IDs stored in files to an environment with LDAP as the authentication method and naming service.

The system has centralized user management by using files on the Control Workstation (CWS). The files are distributed by supper mechanisms. Basically supper distributes the files by copying them to the clients.

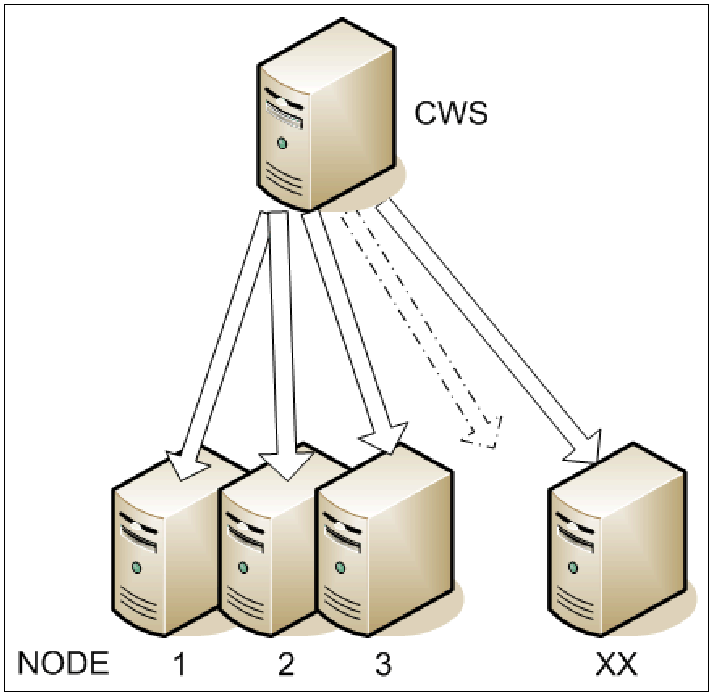


Figure 6-1 SP user ID management

6.2 Planned environment

The environment we plan will consist of a single centralized LDAP server. The existing nodes will be migrated to have access to the directory with a proxy user.

To manage the directory a single LDAP client will be created to maintain the user ID. Therefore this client needs administrative access to the directory.

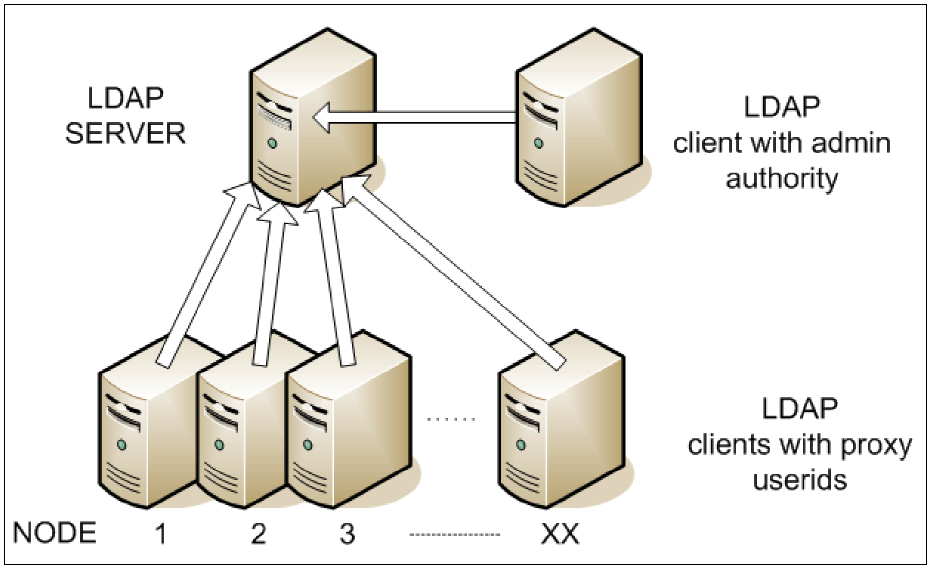


Figure 6-2 Planned LDAP environment

Note: In this example scenario only one LDAP server is used. For production environments this will most likely differ.

6.2.1 Schema layout and user ID considerations

The schema for this example will be the RFC2307AIX schema as explained in the 3.2.3, “RFC2307AIX schema” on page 63. All client nodes will have the ability to update the directory for the necessary AIX values.

The distinguished names used in this example are described in Table 6-1.

Table 6-1 Schema layout for RFC2307AIX

Distinguished name	Purpose
dc=example.dc=com	Base DN
ou=People,dc=example.dc=com	Storing the USERIDs
ou=Groups,dc=example,dc=com	Storing group IDs
ou=clients,dc=example,dc=com	Storing the IDs for the proxy clients
ou=System,dc=example,dc=com	AIX administrative information

Note: In this example we simplify things by assuming that there will be no clean up steps required to eliminate duplicate IDs, normalize formatting, and so on. In a production environment you may need to take special care to make sure that user information is prepared in advance.

We assume that since user IDs are already integrated in one file on the CWS there are no problems with duplicated user or group IDs when moving to a central directory.

To prevent conflicting entries between the local system IDs and the LDAP user IDs, the LDAP IDs will start in this example at UID 500. The GID will start at 500.

The local /etc/passwd and /etc/group files are shown in Example 6-1.

Example 6-1 /etc/passwd and /etc/group files for local user IDs

```
# cat /etc/passwd
root:!:0:0::/home/root:/usr/bin/ksh
daemon:!:1:1::/etc:
bin:!:2:2::/bin:
sys:!:3:3::/usr/sys:
adm:!:4:4::/var/adm:
uucp:!:5:5::/usr/lib/uucp:
guest:!:100:100::/home/guest:
nobody:!:4294967294:4294967294::/:
lpd:!:9:4294967294::/:
lp:*:11:11::/var/spool/lp:/bin/false
snapp:*:200:12:snapp login user:/usr/sbin/snapp:/usr/sbin/snappd
nuucp:*:6:5:uucp login user:/var/spool/uucppublic:/usr/sbin/uucp/uucico
invscout:*:7:13::/var/adm/invscout:/usr/bin/ksh
ipsec:*:201:1::/etc/ipsec:/usr/bin/ksh
sshd:*:202:201::/var/empty:/usr/bin/ksh
idsldap:!:203:202::/home/idsldap:/usr/bin/ksh
ldapdb2:!:205:14::/home/ldapdb2:/usr/bin/ksh
ldap:*:206:1::/home/ldap:/usr/bin/ksh

[ root@bs01b02:/mnt/installp/ppc ] # cat /etc/group
system:!:0:root
staff:!:1:ipsec,sshd,idsldap,ldapdb2,ldap,local2
bin:!:2:root,bin
sys:!:3:root,bin,sys
adm:!:4:bin,adm
uucp:!:5:uucp,nuucp
mail:!:6:
security:!:7:root
cron:!:8:root
printq:!:9:lp
```

```
audit:!:10:root
ecs:!:28:
nobody:!:4294967294:nobody,lpd
usr:!:100:guest
perf:!:20:
shutdown:!:21:
lp:!:11:root,lp,printq
snapp:!:12:snapp
invscout:!:13:invscout
ipsec:!:200:
sshd:!:201:sshd
idsldap:!:202:idsldap,root,ldapdb2
dbsysadm:!:14:ldapdb2,root
ldap:!:203:ldap
```

6.2.2 Client setup considerations

Normal clients for the LDAP directory do not need to have full rights. Therefore they will use a proxy ID, which is an ID that can still access the directory for reading but will have a limited write access. For example, the proxy ID has read access to the password but no write access.

To have the possibility to manage the LDAP directory in the same way as the CWS before the migration, a client will be installed that has full administrative rights on the directory. This client can use the normal AIX high-level commands to manage the LDAP server.

6.2.3 Data migration considerations

To move the data from the local file to the LDAP server the **sectoldif** command will be used. The **sectoldif** command will export all data from the local system files to a single LDIF file.

The LDIF file can then be modified to contain only the entries above the user ID 500 and group ID 500. Changes to the LDIF need to be made carefully. AIX handles LDAP users only with LDAP groups already defined. Pay careful attention to specification of groups, as discussed in 6.4.1, “Creating new groups and users” on page 204.

6.2.4 Restricting user access

The access for LDAP users to a machine can be restricted in the following ways:

- ▸ Setting the SYSTEM attribute in the local /etc/security/user file for default and for the user

- Using the NIS combat mode in combination with netgroups on the LDAP server (see Chapter 5, “Scenario: Sun ONE LDAP server integration” on page 157)
- Using the hostsallowedlogin and the hostsdeniedlogin attributes of the user entry on the LDAP

Here we want to be able to control the access to the systems from a centralized place. Therefore we consider the netgroup or the LDAP mode hostsallowedlogin options. The hostsallowed login and hostsdeniedlogin are chosen here because they are configured on the LDAP server, but the netgroups have to be defined on the host itself.

Therefore the default stanza in the /etc/security/user file will be changed to SYSTEM=”LDAP or files” to allow any login to appear according to the hostallowed and hostdenied entries.

6.2.5 The LDAP server

In this chapter we use ITDS Version 6.0 as the LDAP server. The LDAP server that is used makes no difference in this chapter. Any other LDAP server that is LDAP V3 compliant is able to handle the setup. Otherwise it will be stated directly at the section.

The LDAP server used here has the following configuration:

admin DN	cn=admin
admin password	its0g00d
hostname	bs01b02

6.3 Migration steps

The installation can proceed as described in Figure 6-3.

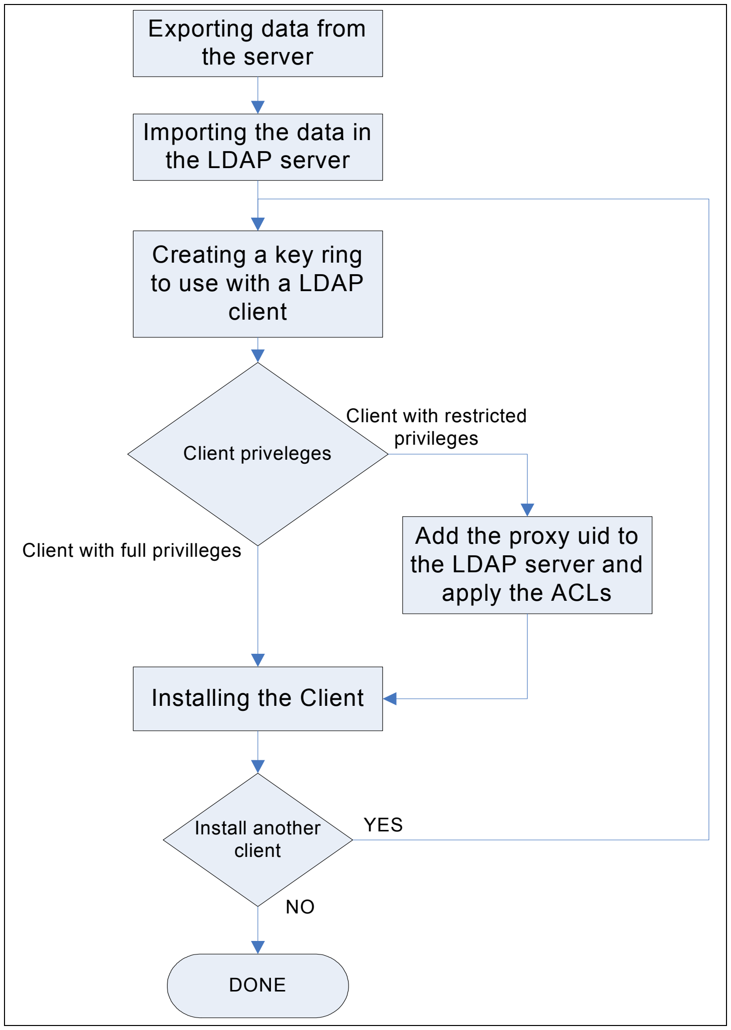

Exporting data from the server
Importing the data in the LDAP server
Creating a key ring to use with a LDAP client
Client priveleges
Client with restricted privileges
Add the proxy uid to the LDAP server and apply the ACLs
Client with full privilleges
Installing the Client
Install another client
YES
NO
DONE

Figure 6-3 Flowchart for the migration steps

First of all the data must be transferred from the CWS to the LDAP directory. The installation of the LDAP server must already be completed. The installation of ITDS 6.0 is described in Appendix A, “IBM Tivoli Directory Server V6.0 installation” on page 273. We assume here that the installation was done with the -u none flag. This means that no data is in the directory.

6.3.1 Step 1: Exporting data from the server

To export the data from the server we use the **sectoldif** command, as shown Example 6-2. The data will be redirected to a file so that it can be easily copied between servers.

The command options used here for sectoldif are:

-d "example,dc=com"	This defines the base DN that we use for the AIX user information.
-S RFC2307AIX	This defines the schema to use for the exported LDIF files, as described in 6.2.1, "Schema layout and user ID considerations" on page 189.

Example 6-2 Exporting user data from AIX: sample of sectoldif output

```
# sectoldif -d "dc=example,dc=com" -S RFC2307AIX > user_dat
# more user_data
dn: ou=People,dc=example,dc=com
ou: People
objectClass: organizationalUnit


dn: uid=default,ou=People,dc=example,dc=com
uid: default
objectClass: aixauxaccount
objectClass: shadowaccount

       .
       .
#
```

At this point in our user_dat output file we have a complete copy of all the information that is needed. The information about all user and group IDs below 500 are also in the file, so it is necessary now to edit the file manually. Just use an editor and remove all of those entries. Be careful not to remove the top-level containers from the directory, and the default entry should remain in the LDIF file also.

6.3.2 Step 2: Importing the data into the LDAP directory

The **ldapadd** command is used to transfer data in the user_dat file to the LDAP server. This can be done from any system that has network connectivity to the LDAP server and the **ldapadd** command installed.

The **ldapadd** command is used with the following parameters:

-h bs01b02	Host name of the LDAP server
-D cn=admin	Bind DN for the LDAP server
-w its0g00d	Bind password
-i user_data	The LDIF file with the user information

Example 6-3 Importing data to the LDAP directory

```
[ root@bs01b02:/home/root ] # ldapadd -h bs01b02  -D cn=admin -w its0g00d -i
user_data
adding new entry ou=People,dc=example,dc=com
adding new entry uid=default,ou=People,dc=example,dc=com
adding new entry uid=user500,ou=People,dc=example,dc=com
adding new entry uid=user501,ou=People,dc=example,dc=com
adding new entry uid=user502,ou=People,dc=example,dc=com
adding new entry uid=user503,ou=People,dc=example,dc=com
adding new entry uid=user504,ou=People,dc=example,dc=com
...
adding new entry uid=user510,ou=People,dc=example,dc=com
adding new entry ou=Groups,dc=example,dc=com
adding new entry cn=grp500,ou=Groups,dc=example,dc=com
adding new entry cn=grp501,ou=Groups,dc=example,dc=com
adding new entry cn=grp502,ou=Groups,dc=example,dc=com
adding new entry cn=grp503,ou=Groups,dc=example,dc=com
adding new entry cn=grp504,ou=Groups,dc=example,dc=com
...
adding new entry cn=grp510,ou=Groups,dc=example,dc=com
adding new entry ou=System,dc=example,dc=com
adding new entry cn=aixid,ou=System,dc=example,dc=com
adding new entry cn=aixbaseid,ou=System,dc=example,dc=com
```

The the server has now the data that is needed for the LDAP user authentication.

6.3.3 Step 3: Creating a key database to use with SSL clients

To establish a secure connection between the client and the LDAP server, the SSL client needs an SSL key database containing the certificate from the certificate authority that the LDAP server is using.

To create a key database we only need two steps (Example 6-4 on page 196):

1. Create a key database with the **gsk7cmd** command with the following parameters:

-keydb	This defines the type of operation.
-create	The action is to create a key database.
-db <filename>	The file name of the key database (key.kdb).

-pw <password>	This specifies the password to encrypt the key database (its0g00d).
-type <type>	This specifies the type of the key database file (cms).

2. Import the CA’s certificate into the key database using the **`gsk7cmd`** command with the following parameters:

-cert	This defines the type of operation.
-add	This adds the certificate to the key database.
-db <filename>	This is the file name of the key database file (key.kdb).
-pw <password>	This specifies the password to encrypt the key database (its0g00d).
-label <name>	This is the name under which the CA certificate should be listed (example.com).
-format <ascii>	This defines that the certificate is in ASCII format.
-trust <enable>	This defines that the certificate should be treated as trusted.
-file <filename>	This is the file name of the CA certificate (/home/root/cacert.pem).

Example 6-4 Creating a key database and importing the CA certificate

```
# cat /home/root/cacert.pem
-----BEGIN CERTIFICATE-----
MIIDGzCCAoSgAwIBAgIBADANBgkqhkiG9w0BAQQFADBtMQswCQYDVQQGEwJ1czEL
MAkGA1UECBMCVFgxDzANBgNVBAcTBkF1c3RpbjEUMBIGA1UEChMLZXhhbXBsZS5j
b20xCzAJBgNVBAMTAkNBMR0wGwYJKoZIhvcNAQkBFg5jYUBleGFtcGxlLmNvbTAe
Fw0wNTA5MDgxNTIwNTJaFw0wNjA5MDgxNTIwNTJaMG0xCzAJBgNVBAYTAnVzMQsw
CQYDVQQIEwJUWDEPMA0GA1UEBxMGQXVzdGluMRQwEgYDVQQKEwtleGFtcGxlLmNv
bTELMAkGA1UEAxMCQ0ExHTAbBgkqhkiG9w0BCQEWDmNhQGV4YW1wbGUuY29tMIGf
MA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC8K8kCW83CDm36X+COqQ3ayDFhKaBO
kM1C5+7UMoOSdxJjFg/XftqzYIf6i1Jt5fPii03x/C1BLbY+8U30FyPo+0iK0KkR
fDPIEsR8XZYtAsFaR/d9YBytRDjwP1qz2YT3NXIekwfi91JlV3LGQM1rcI++cJe0
rRxvKvHGmYXgLwIDAQABo4HKMIHHMB0GA1UdDgQWBBRP+zR+YlZ312w9DMf7QAIt
ft9CuDCBlwYDVR0jBIGPMIGMgBRP+zR+YlZ312w9DMf7QAItft9CuKFxpG8wbTEL
MAkGA1UEBhMCdXMxCzAJBgNVBAgTAlRYMQ8wDQYDVQQHEwZBdXN0aW4xFDASBgNV
BAoTC2V4YW1wbGUuY29tMQswCQYDVQQDEwJDQTEdMBsGCSqGSIb3DQEJARYOY2FA
ZXhhbXBsZS5jb22CAQAwDAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQQFAAOBgQBr
8m+/PWQtueDI+g+eFMHOcy5iKQniBo1/oyv2DGC5SLZNnsqMCjLE7B4nWEepgpJ4
Q+bcni1l7xiGIQqGMGprM8ClTU9jvZzC3a9dFerCWOaVTOVJIy3PjdXkyK5TYtXA
dh5LmDrORi/OJNCEWeQHXquT8NaDP8NL+j4QacbP1Q==
-----END CERTIFICATE-----
# gsk7cmd -keydb -create -db /etc/security/ldap/key.kdb -pw its0g00d -type cms
# gsk7cmd -cert -add -db /etc/security/ldap/key.kdb -pw its0g00d -label
example.com -format ascii -trust enable -file /home/root/cacert.pem
```

6.3.4 Step 4: Setting up AIX 5L LDAP client with administrative rights

The administrative client will have all rights to manage the directory. Therefore it uses SSL to secure the connections. Also, the client will be able to use all normal AIX high-level commands to manage the users in the LDAP directory.

The installation of the file sets is described in 4.2, “Installation and basic client configuration” on page 105. The **`mksecldap`** is used with the following parameters (Example 6-5):

-c	Client configuration
-h <hostname>	The host name (bs01b02)
-a <bind dn>	Bind DN (cn=admin)
-p<password>	Password for the bind DN (its0g00d)
-d <base dn>	Base DN ("dc=example,dc=com")
-k <filename>	Path to the key file (/etc/security/ldap/key.kdb)
-w <passord>	Password for the key database (its0g00d)

Example 6-5 Configuring the admin LDAP client

```
# mksecldap -c -h bs01b02  -a cn=admin -p its0g00d -d "dc=example,dc=com" -k
/etc/security/ldap/key.kdb -w its0g00d
gsksa.rte
gskta.rte
# lsldap
dn: ou=People,dc=example,dc=com

dn: ou=Groups,dc=example,dc=com

dn: ou=System,dc=example,dc=com
# lsldap -a passwd
dn: uid=default,ou=People,dc=example,dc=com
uid: default
objectClass: aixauxaccount
objectClass: shadowaccount
objectClass: posixaccount
```

The LDAP client is now active but no user can log in up to now because the SYSTEM and registry entries in /etc/security/user are not set. The entries can be changed for each user or for the default.

The default entry should be modified to contain the following lines:

SYSTEM = "LDAP or files"	This means that the authentification is done against the LDAP as the default and to the local files as a backup mechanisem.
registry = LDAP	This entry desribes where the user registry is stored. AIX 5L is able to know where a user registry is without that entry. The purpose is to resolve conflicting entries in local and remote registries.

The setup can be done with the **`chsec`** command:

```
chsec -f /etc/security/user -s default -a "SYSTEM=LDAP or files"
chsec -f /etc/security/user -s default -a "registry=LDAP"
```

6.3.5 Step 5: Installing a LDAP client with a proxy user ID

For non-administrative clients, the installation of the client using a proxy user ID consists of two parts:

1. Setting up the proxy user ID with the necessary ACLs on the LDAP server:
   – With the default ACLs that comes with AIX
   – With more restrictive ACLs
2. Setting up that actual client

Setting up the proxy user ID on the LDAP server

First we need to create a container in the directory for storing the user IDs for the clients as defined in Table 6-1 on page 189.

Therefore we create a LDIF file and apply it to the LDAP server with the **`ldapadd`** command. See Example 6-6.

Example 6-6 Creating a container for clients

```
# cat clients.ldif
dn: ou=clients,dc=example,dc=com
ou: clients
objectClass: organizationalUnit
# ldapadd -Z -K key.kdb -P its0g00d -D "cn=admin" -w its0g00d  -i clients.ldif
adding new entry ou=clients,dc=example,dc=com
```

```
# ldapsearch -b "dc=example,dc=com" -Z -K key.kdb -P its0g00d -D "cn=admin" -w
its0g00d  ou=clients

ou=clients,dc=example,dc=com
ou=clients
objectClass=organizationalUnit
objectClass=top
```

Defining the access with the default ACLs

Important: The example in this section is only valid for the ITDS Server. For other servers the access control methods are different.

After the container is created the actual ID can be inserted to the directory. We use the template at /etc/security/ldap/proxy.ldif.template. Create a copy of it and replace the following values with a reasonable value:

proxyDN	The distinguisehed name of the proxy (cn=bs01b03,ou=clients,dc=example,dc=com)
proxyUser	The name of the proxy user as defined in the DN (bs01b03)
proxyPWD	The password for the proxy user (its0g00d)
baseDN	The base DN to where the proxy user rights should be applied (dc=example,dc=com)

An explanation of the format of ACL entries in the LDIF file is outside the scope of this book. For more information about this see IBM Tivoli Directory Server information center, or the IBM Redbook *Understanding LDAP - Design and Implementation,* SG24-4986.

Example 6-7   LDIF file for creating a proxy user with the default

```
# Create the proxy user
dn: cn=bs01b03,ou=clients,dc=example,dc=com
changetype: add
cn: bs01b03
sn: bs01b03
userpassword: its0g00d
objectclass: person

# Modify proxy user password if previously existed (create above will fail).
dn: cn=bs01b03,ou=clients,dc=example,dc=com
changetype: modify
replace: userpassword
userpassword: its0g00d
```

```
# Set the baseDN ACL
dn: dc=example,dc=com
changetype: modify
replace: aclentry
aclentry: access-id:cn=this:at.userPassword:grant:w
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.userPassword:grant:r
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.passwordchar:grant:r
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.gecos:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.hostlastlogin:grant:rscw
aclentry:
access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.hostlastunsuccessfullogin:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.ixtimelastlogin:grant:rscw
aclentry:
access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.ixtimelastunsuccessfullogin:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.loginshell:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.passwordflags:grant:rw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.passwordhistlist:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.shadowlastchange:grant:rw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.ixlastupdate:grant:rw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.terminallastlogin:grant:rscw
aclentry:
access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.terminallastunsuccessfullogin:grant:rscw
aclentry:
access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.unsuccessfullogincount:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:normal:grant:rsc
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:critical:grant:rsc
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:sensitive:grant:rsc
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:object:deny:ad

# Propagate the ACL from the baseDN
dn: dc=example,dc=com
changetype: modify
replace: aclpropagate
aclpropagate: TRUE
```

In Example 6-8 we use **ldapadd** to import the LDIF data file shown in Example 6-7 on page 199.

Import the proxy user with **ldapadd**.

Example 6-8 Adding the administrative user ID to the server

```
# ldapadd -Z -K key.kdb -P its0g00d -D "cn=admin" -w its0g00d  -i bs01b03.ldif
adding new entry cn=bs01b03,ou=clients,dc=example,dc=com

modifying entry cn=bs01b03,ou=clients,dc=example,dc=com

modifying entry dc=example,dc=com
modifying entry dc=example,dc=com

#
```

Proxy user ID and restrictive ACLs

The ACLs shown above contain write entries. The AIX 5L client would be able to operate without any write access to the directory. Nevertheless, the client tries to update attributes at the LDAP server during every login of a user. These attributes are:

- ixtimelastlogin
- terminallastlogin
- hostlastlogin
- unsuccesfulllogincount

For an unsuccessful login the following attributes are updated:

- unsuccessfulllogincount
- terminallastunsuccessfullogin
- ixtimelastunsuccessfullogin
- hostlastunsuccessfullogin

After a password change the following attributes are updated:

- passwordflags
- shadowlastchange
- passwordhistlist (only if the list is used)

Also, the client tries to update the passwordchar where he has read-only rights, so he never succeeds.

The client has the ability to change the gecos field and the loginshell entry. Depending on the way the machine is used it is not necessary to change these values on every client, so this too can be removed from the ACL entries.

The modification that can be done without losing functionality is to restrict the read access to the password hash. In this case the client must use ldap_auth, as shown Example 6-10.

All other modifications will lead to a reduced functionality of the AIX 5L client with the RFC2307AIX schema. This consideration should be made with the target environment in mind. For this scenario we will only reduce the userpassword read access, access to the gecos, and loginshell from the client.

The ACLs to use are shown in Example 6-9.

Example 6-9 Restrictive ACLs for the proxy user ID

```
# Set the baseDN ACL
dn: dc=example,dc=com
changetype: modify
replace: aclentry
aclentry: access-id:cn=this:at.userPassword:grant:w
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.passwordchar:grant:r
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.hostlastlogin:grant:rscw
aclentry:
access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.hostlastunsuccessfullogin:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.ixtimelastlogin:grant:rscw
aclentry:
access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.ixtimelastunsuccessfullogin:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.passwordflags:grant:rw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.passwordhistlist:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.shadowlastchange:grant:rw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.ixlastupdate:grant:rw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.terminallastlogin:grant:rscw
aclentry:
access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.terminallastunsuccessfullogin:grant:rscw
aclentry:
access-id:cn=bs01b03,ou=clients,dc=example,dc=com:at.unsuccessfullogincount:grant:rscw
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:normal:grant:rsc
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:critical:grant:rsc
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:sensitive:grant:rsc
aclentry: access-id:cn=bs01b03,ou=clients,dc=example,dc=com:object:deny:ad

# Propagate the ACL from the baseDN
dn: dc=example,dc=com
changetype: modify
replace: aclpropagate
aclpropagate: TRUE
```

Setting up the LDAP client machine with the proxy user ID

To set up the LDAP client machine we go to that machine and use the **`mksecldap`** command. But first a key database must be created to use SSL connections against the server. The creation of the key database is described in 6.3.3, “Step 3: Creating a key database to use with SSL clients” on page 195. The result of the installation is shown in Example 6-10 on page 203.

`mksecldap` will be used with the following options:

-c	Create a client.
-h serverlist	The list of LDAP servers to connect to (bs01b02).
-a bindDN	The DN with which the bind is made. This is the DN of the proxy user cn=bs01b03,ou=clients,dc=example,dc=com.

-p bindPasswd	The password for the DN.
-d baseDN	The base DN where the user data is stored (dc=example,dc=com).
-k SSLkeypath	The path to the SSL key database (/etc/security/ldap/key.kdb).
-w SSLkeypasswd	The password for the SSL key database (its0g00d).
-A ldap_auth	The type of authentication that is used. Here the ldap_auth is used, which means that the password from the user will be sent to the server.

Example 6-10 Installing LDAP client with proxy user

```
# mksecldap -c -h bs01b02 -a "cn=bs01b03,ou=clients,dc=example,dc=com" -p
its0g00d -d "dc=example,dc=com" -k /etc/security/ldap/key.kdb -w its0g00d -A
ldap_auth
gsksa.rte
gskta.rte
# ls-secldapclntd
ldapservers=bs01b02
ldapport=636
ldapversion=3
userbasedn=ou=People,dc=example,dc=com
groupbasedn=ou=Groups,dc=example,dc=com
idbasedn=cn=aixid,ou=System,dc=example,dc=com
usercachesize=1000
usercacheused=0
groupcachesize=100
groupcacheused=0
cachetimeout=300
heartbeatT=300
numberofthread=10
connectionsperserver=10
alwaysmaster=no
authtype=LDAP_AUTH
searchmode=ALL
defaultentrylocation=LDAP
ldaptimeout=60
userobjectclass=account,posixaccount,shadowaccount,aixauxaccount,ibm-securityId
entities
groupobjectclass=posixgroup,aixauxgroup
#
```

Now you need to repeat the steps for setting the SYSTEM and registry values, as described in 6.3.4, “Step 4: Setting up AIX 5L LDAP client with administrative rights” on page 197.

6.4 Working with the AIX 5L clients

Table 6-2 on page 204 shows possible client-side operations. The administrative client refers to an AIX 5L client with administrative rights on the directory. The normal client refers to a client with a proxy user ID on one of the two ACLs shown in the previous chapter. The **`ldapadd`** and **`ldapmodify`** commands can be executed from everywhere. The general changes are intended to be done with an administrative user ID. The user changes are intended to be done with the user ID of the user.

Table 6-2 Client ID management privileges

Action	Admin client	Normal client	ldapadd/ ldapmodify
Creating USERS	Yes	No	Yes
Creating groups	Yes	No	Yes
Deleting USERS	Yes	No	Yes
Changing user attributes	Yes	No	Yes
USERs can change their own password	Yes	Yes	Yes
USERs can change their own attributes	Yes	Yes	No

6.4.1 Creating new groups and users

To create a new user or group login to the admin client, the user can be created from there with the **`mkuser`** command. But first we create a group for that user with the **`mkuser`** command:

```
mkgroup -R LDAP id=511 grp511
```

Important: The **mkuser** command relies on the defaults specified in the /usr/lib/security/mkuser.default file. If the specified group does not exist **mkuser** is not able to set the group, even if an existing group is specified on the command line. In our example the default is the staff group and it does not exist in the LDAP directory. Therefore we have to specify a group there that is in the LDAP server:

```
mkuser -R LDAP id=400 users
```

And change the values in the /usr/lib/security/mkuser.default file to:

```
user:
        pgrp = users
        groups = users
        shell = /usr/bin/ksh
        home = /home/$USER
```

Note that after changing this to a group that does not exist in the local files you are no longer able to create local user IDs.

Example 6-11   Creating a user with the mkuser command

```
mkuser -R LDAP id=511 grp user511
# lsldap -a passwd uid=user511
dn: uid=user511,ou=People,dc=example,dc=com
homedirectory: /home/user511
isadministrator: false
loginshell: /usr/bin/ksh
uidnumber: 511
gidnumber: 511
passwordchar: *
uid: user511
cn: user511
objectclass: account
objectclass: posixaccount
objectclass: shadowaccount
objectclass: aixauxaccount
objectclass: ibm-securityIdentities
objectclass: top
[ root@bs01b02:/home/root ] #
```

6.4.2 Deleting users

On the administrative client users can be deleted with the **rmuser** command:

```
rmuser -R LDAP user510
```

6.4.3 Changing passwords and other attributes

Changing passwords can be done in two ways:

- The user can change the password by himself. This means that a user logs in to a machine and types the **passwd** command. This works on a client with administrative rights and on a client with a proxy user ID when LDAP authentication is used. It will not work on a client with a proxy user ID and UNIX authentication.
- The administrator can change the password for a user. The administrator can change the password for the user with the **passwd** or the **chpasswd** command. This will only work on a client with administrative rights.

Changing attributes in an user entry can be done with the normal AIX command **chuser**, for example:

```
chuser -R LDAP rlogin=false user500
```

6.4.4 Restricting access to machines with the LDAP attributes

The attributes to restrict the access to machines are defined in the entry of the user. The available attributes are:

hostsallowedlogin	This is a comma-separated list of the hosts a user is allowed to log in to.
hostsdeniedlogin	This is a comma-separated list of the hosts a user is not allowed to log in to.

In the list of hosts networks can be specified (for example, 192.168.1.0/24).

If only the hostsallowedlogin field is available, the user can only log in to the hosts that are specified there. If only the hostsdeniedlogin is available, then the user can log in to every host except the hosts listed in the field. If both are available then the user can log in to all hosts that are listed in hostsallowedlogin and not in hostsdeniedlogin (Example 6-12).

Example 6-12 Using hosts allowed and host denied

```
# chuser -R LDAP hostsdeniedlogin=9.3.5.140,9.3.5.139,9.3.5.138    user500
# chuser -R LDAP hostsdeniedlogin=9.3.5.140,9.3.5.139    user500
# ldapsearch -h bs01b02 -D "cn=admin" -w its0g00d -b "dc=example,dc=com" 
cn=user500
uid=user500,ou=People,dc=example,dc=com
uid=user500
objectClass=aixauxaccount
....
unsuccessfullogincount=0
hostsallowedlogin=9.3.5.0/24
```

```
hostsdeniedlogin=9.3.5.140
hostsdeniedlogin=9.3.5.139
hostsdeniedlogin=9.3.5.138
```

That will allow user500 to log in to all hosts on the 9.3.5.0/24 network but not on 9.3.5.138, 9.3.5.139, and 9.3.5.140.

Also, the hostsallowedlogin and the hostsdeniedlogin can be defined in the default entry on the LDAP server. But be aware that the entry on the user stanza overides the value in the default entry.

7

Scenario: OpenLDAP integration

This chapter describes how to integrate the OpenLDAP Directory Server on SUSE Enterprise Linux 9 with an AIX 5L LDAP client.

In this chapter the following topics are described:

- LDAP schema used by SUSE
- Setting up an AIX client manually to authenticate against a openldap server
- Using anonymous bind on an AIX client

7.1 Overview

In this example scenario we show how to manually configure the AIX 5L LDAP client to authenticate against an OpenLDAP server. We have chosen to use anonymous bind in this example, as this is a very common method. We describe RFC2307 schema considerations specific to this scenario as well.

7.2 The environment

In this scenario we utilize a Suse Linux Enterprise Server 9 (SLES 9). The server installs and uses a LDAP server for user authentication during a standard installation.

In this scenario it is assumed that the server is up and running and that there are user IDs already deployed. To the LDAP server there are a couple of Linux servers connected already, and a new AIX 5L Version 5.3 Client should be deployed, as shown in Figure 7-1.

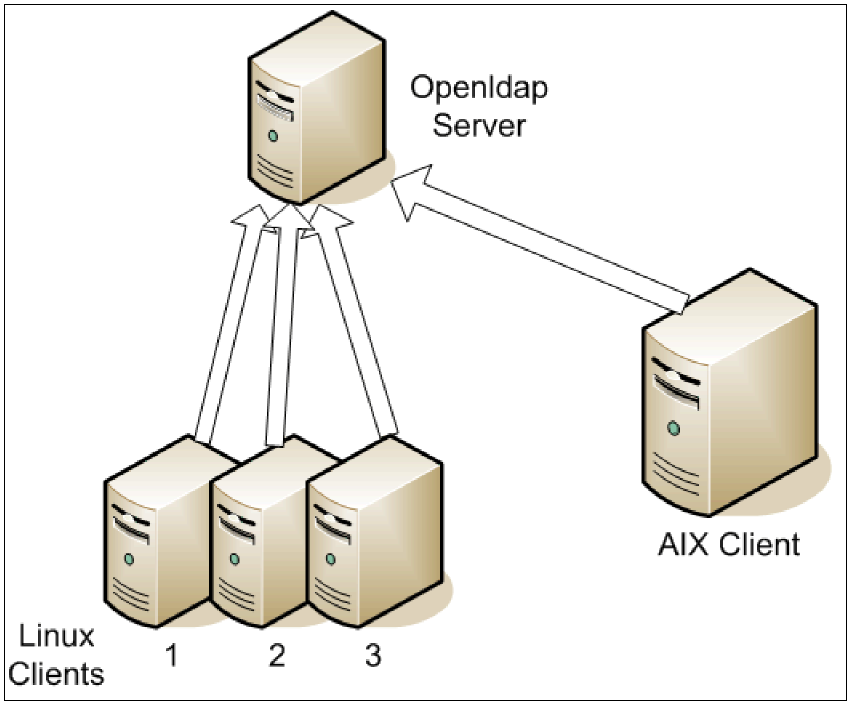


Figure 7-1 Environment for OpenLDAP

7.2.1 Schema considerations

SLES 9 is using a schema that is not a plain RFC2307 schema. It has the extensions that are defined in RFC2307BIS. That will not create a problem for the user IDs as long as they are stored under a certain DN. But the system stores group members with a DN.

The Linux client is able to handle both types of entries itself. Therefore we can use the normal RFC2307 group definition. See Example 7-1 and Example 7-2 for the difference.

Example 7-1 RFC2307 group entry

```
dn: cn=mygroup,ou=group,dc=example,dc=com
cn: mygroup
gidNumber: 2000
memberUid: test
memberUid: user500
objectClass: top
objectClass: namedObject
objectClass: posixGroup
```

Example 7-2 RFC2307BIS like group entry

```
dn: cn=ulinux,ou=group,dc=example,dc=com
cn: ulinux
gidNumber: 1000
member: uid=test,ou=people,dc=example,dc=com
objectClass: top
objectClass: posixGroup
objectClass: groupOfNames
```

This modification can be made, but then you should be aware that other system tools that depend on specific schema definitions may not work correctly anymore (such as YAST).

To prevent these issues the memberlists can be copied together in one entry that would work with both groups, as shown in Example 7-3. This synchronization could be accomplished using IBM Tivoli Directory Integrator. For this scenario we leave this up to the reader. It is assumed that we have the necessary entries for the posixGroup that AIX 5L can work with.

Example 7-3 Combined entry

```
dn: cn=mygroup,ou=group,dc=example,dc=com
cn: mygroup
gidNumber: 2000
member: uid=user500,ou=people,dc=example,dc=com
```

```
memberUid: user500
objectClass: top
objectClass: posixGroup
objectClass: groupOfNames
```

7.2.2 User ID and access rights

The access rights in the directory are very simple:

- Everybody can read everything except for the password hashes. Therefore the AIX 5L client must use the ldap_auth for the authentication requests.
- The AIX 5L client normally uses a user ID on the LDAP server. In this scenario we deploy the client without a user ID and force it to use anonymous bind.

7.3 Client setup

The client software must be installed before the client can be set up. See 4.2.1, “AIX 5L LDAP client software installation” on page 105.

The **mksecldap** command cannot be used in this scenario because it requires a binddn and pasword. It could be used to set up the client when using a binddn and password against openldap, so the configuration here will be done manually.

The following steps are needed:

1. Define the LDAP authentication method in methods.cfg.
2. Configuring the client daemon.
3. Start the secldapclntd and check that it will start at system boot.
4. Configure the /etc/security/users file.

7.3.1 Define the LDAP authentication method in methods.cfg

The /usr/lib/security/methods.cfg file must be modified to contain the LDAP stanza, as shown in Example 7-4.

Example 7-4 LDAP stanza in /usr/lib/security/methods.cfg

```
LDAP:
        program = /usr/lib/security/LDAP
        program_64 =/usr/lib/security/LDAP64
```

7.3.2 Configuring the client daemon

Note: This section demonstrates the manual configuration steps to set up the client with null credentials (a *null bind*). The recommended way under normal circumstances would be to set up the client so that it binds with a proxy DN (*proxy bind*, or sometimes referred to as *admin bind*). You should consider using the method in this section only if a standard proxy bind setup will not meet your requirements.

In a proxy bind setup you can use **mksecldap** to set up the client in a single step (the procedure for that would be similar to that shown in 6.3.5, "Step 5: Installing a LDAP client with a proxy user ID" on page 198, and specifically Example 6-10 on page 203).

By editing the /etc/security/ldap/ldap.cfg file the LDAP client daemon must be configured. Normally you could use the **mksecldap** command to do the basic LDAP client configuration, but the **mksecldap** command does not yet support the anonymous bind.

Example 7-5 shows the minimal LDAP client configuration used for this scenario. Modify the /etc/security/ldap/ldap.cfg file and change the following settings.

Example 7-5 AIX 5L LDAP client daemon configuration in /etc/security/ldap/ldap.cfg

```
# Comma separated list of ldap servers this client talks to
ldapservers:bs01b04

# Authentication type. Valid values are unix_auth and ldap_auth.
# Default is unix_auth.
# unix_auth - Retrieve user password and authenticate user locally.
# ldap_auth - Bind to LDAP server to authenticate user remotely through LDAP.
authtype:ldap_auth

# AIX-LDAP attribute map path.
userattrmappath:/etc/security/ldap/2307user.map
groupattrmappath:/etc/security/ldap/2307group.map

# Base DN where the user and group data are stored in the LDAP server.
# e.g., if user foo's DN is: uid=foo,ou=people,cn=aixdata
# then the user base DN is: ou=people,cn=aixdata
userbasedn:ou=people,dc=example,dc=com
groupbasedn:ou=group,dc=example,dc=com
```

Note: At this stage the LDAP client is sending clear text passwords over the network. For a production environment SSL should be enabled.

After the configuration of the ldap daemon we can start it and verify that it is working, as shown in the Example 7-6.

Example 7-6 Starting the secldapclntd

```
# start-secldapclntd
Starting the secldapclntd daemon.
The secldapclntd daemon started successfully.
#
# lsldap -a passwd
dn: uid=user500,ou=people,dc=itsc,dc=austin,dc=ibm,dc=com
cn: user500
gidNumber: 1000
givenName: l
homeDirectory: /home/user500
loginShell: /bin/bash
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
objectClass: inetOrgPerson
shadowInactive: -1
shadowLastChange: 13068
shadowMax: 99999
shadowMin: 0
shadowWarning: 7
sn: user500
uid: user500
uidNumber: 1001
#
```

The client is running now. To enable users to log in the /etc/security/users file must be modified. We modify the default entry to allow every user on the LDAP server to log in. This can be done with the following commands:

```
chsec -f /etc/security/user -s default -a "SYSTEM=LDAP or files"
chsec -f /etc/security/user -s default -a "registry=LDAP"
```

After this modification the users should be able to log in to the AIX 5L system.

Important: If you are using the anonymous or *null bind* method as described above, then make sure you have applied APAR IY79263. Otherwise it will not be possible to change the password of a user on the OpenLDAP server from the AIX 5L client. The AIX 5L Version 5.3 ML03 client would not be able to mount any automount directories if SSL is enabled. APAR IY79263: "LDAP: Fails to Reset User Password Against OpenLDAP," can be found at the following Web site:

```
http://www-1.ibm.com/support/docview.wss?rs=111&context=SWG10&q1=IY79263&uid
=isg1IY79263&loc=en_US&cs=utf-8&lang=en
```

When the fix is publicly released, it can be downloaded from the following Web site:

```
http://www.ibm.com/servers/eserver/support/unixservers/aixfixes.html
```

In the meantime, if you want a subscription service to notify you by e-mail when the PTF for the APAR is available, you can do so at the following Web site:

```
http://www14.software.ibm.com/webapp/set2/aparsubscriptions/bjyqvcmjd
```

7.3.3 Automatic start of the client daemon

At this stage the client will run but it will not start at system boot. To get it started properly a inittab entry is needed. The **mksecldap** command normally creates an entry like this:

```
ldapclntd:2:once: /usr/sbin/secldapclntd > /dev/console 2>&1
```

To create a entry like that use the **mkitab** command as follows:

```
mkitab 'ldapclntd:2:once: /usr/sbin/secldapclntd > /dev/console 2>&1'
```

This will add the entry to the end of the inittab file. If it is necessary to start the **secldapclntd** before another entry use the -i ident option.

7.3.4 Configuring the AIX 5L client daemon to use SSL

To use SSL on the AIX client the CA certificate needs to be imported to the key ring file. This is already described in the 4.3.1, "Configuring SSL" on page 113. Therefore we assume here we already have a working key ring file and it must only be used by the client. So we need to modify the ldap.cfg file to contain the entries shown in Example 7-7.

Example 7-7 SSL config for AIX 5L client

```
# Whether to use SSL to communicate with the LDAP server. Valid value
# is either "yes" or "no". Default is "no".
# Note: you need a SSL key and a password to the key to enable this.
useSSL: yes

# SSL key file path and key password
ldapsslkeyf:/etc/security/ldap/keyring.kdb
ldapsslkeypwd:its0g00d
```

After the configuration is made to the ldap.cfg file the client daemon needs to be restarted with the **`restart-secldapclntd`** command.

Scenario: Microsoft Windows 2003 Active Directory integration

This chapter discusses scenarios in which AIX 5L hosts can be integrated into an existing Microsoft Windows 2003 Active Directory (MSAD) environment. There are several variations possible when using Kerberos and LDAP technologies in a Microsoft environment. In this chapter we discuss the following three scenarios:

- In the *Kerberos only* scenario, the AIX 5L user authenticates against Microsoft's Kerberos Key Distribution Center (KDC) and receives a ticket-granting ticket (TGT) upon successful authentication. Normal user information is stored locally in the normal files /etc/passwd and /etc/group and other files in the /etc/security directory. This scenario is described in 8.1, "Scenario: Kerberos only" on page 219.
- In the *Kerberos and LDAP* scenario, the AIX 5L user again authenticates against the Microsoft KDC, receiving a TGT upon successful authentication. The user information is now stored in the Microsoft Active Directory. The scenario is described in 8.2, "Scenario: Kerberos and LDAP" on page 233.
- In the *LDAP only* scenario, the AIX 5L machine will use the MSAD for authentication and storage of user information. This scenario is described in 8.3, "Scenario: LDAP only" on page 259.

These scenarios were built with the following software:

- The LDAP and Network Authentication Service (NAS) clients are included with AIX 5L Version 5.3 Maintenance Release 3 (ML03).
- The Active Directory and Kerberos services are running on Microsoft Windows 2003 Server.
- Active Directory support for an RFC2307 compatible schema and UNIX/Windows password synchronization is accomplished using the Microsoft Windows Services for UNIX 3.5.

The following scenarios assume that a Microsoft Windows domain is already installed and configured. The configuration instructions for the Microsoft Windows 2003 Server used in these scenarios are in Appendix B, “Microsoft Windows 2003 Active Directory configuration” on page 283.

At the time of writing this book, IBM only supports Kerberos authentication against a Microsoft Windows 2000 Server KDC.

8.1 Scenario: Kerberos only

In the *Kerberos only* scenario, the AIX 5L user authenticates against Microsoft's Kerberos Key Distribution Center (KDC) and receives a ticket-granting ticket (TGT) upon successful authentication. The user information such as user ID, group ID, home directory, and shell is stored locally in the usual files /etc/passwd and /etc/group and files in the /etc/security directory.

Starting with AIX 5L Version 5.1, the Network Authorization Service (NAS) software on AIX 5L shipped with two Kerberos authentication load modules, KRB5 and KRB5A. The original KRB5 module allows you to authenticate against a Kerberos V5 KDC that supports the kadmin interface. The KRB5A authentication module only performs authentication and is used to authenticate against KDCs that do not support the kadmin interface, such as the Microsoft KDC. (The kadmin interface allows Kerberos clients to remotely manage Kerberos principals, policies, and service key tables.)

Important: Attempting to authenticate using the KRB5 module with a KDC that does not support the kadmin interface will cause user logon to fail.

Note: In AIX 5L Version 5.2 ML02 and Version 5.3 ML01 an option is introduced to eliminate authentication time dependency on the kadmin server. IBM white paper *Configuring AIX 5L for Kerberos Based Authentication Using Network Authentication Service* includes a section covering this option. The paper can be found at the following URL:

http://www-03.ibm.com/servers/aix/whitepapers/aix_kerberos.pdf

8.1.1 Installation and configuration of AIX 5L NAS Client

In this section we discuss the installation and configuration of the AIX 5L NAS Client.

Installation of NAS client software

In order to use the AIX 5L NAS client, you must install the required Licensed Product Packages (LPPs) from the AIX 5L Expansion Pack. The NAS client is packaged in the krb5.client and krb5.lic packages. Use the **`installp`** command, SMIT, or Web-based System Manager to install the required LPPs.

Example 8-1 shows the installation of the NAS client, license, and documentation. Replace the LPPSOURCE tag in the following commands with the correct location of your LPPs in your environment.

Example 8-1 Installation of Network Authentication Service client

```
# installp -acgXYpd LPPSOURCE krb5.client krb5.doc.en_US krb5.lic
<excess output removed>
# lslpp -l "krb5*"
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  krb5.client.rte            1.4.0.3  COMMITTED  Network Authentication Service
                                                 Client
  krb5.client.samples        1.4.0.3  COMMITTED  Network Authentication Service
                                                 Samples
  krb5.doc.en_US.html        1.4.0.3  COMMITTED  Network Auth Service HTML
                                                 Documentation - U.S. English
  krb5.doc.en_US.pdf         1.4.0.3  COMMITTED  Network Auth Service PDF
                                                 Documentation - U.S. English
  krb5.lic                   1.4.0.3  COMMITTED  Network Authentication Service
                                                 License

Path: /etc/objrepos
  krb5.client.rte            1.4.0.3  COMMITTED  Network Authentication Service
                                                 Client
```

Update system-wide PATH variable

After the installation of the NAS client, you should add the /usr/krb5/bin and /usr/krb5/sbin directories to the system-wide PATH variable. The easiest place to make this change is to modify the PATH line in the /etc/environment file. In order to avoid naming conflicts with other commands with the same name, make sure you add the new directories before the /usr/java14 directories. By default AIX 5L installs the java.sdk LPP, which also contains a program called **kinit** and may cause unexpected results if you run this program instead of the NAS client one. The following example shows the new PATH line in the /etc/environment file. Users must log off, then log in to have the new PATH take effect:

```
PATH=/usr/bin:/etc:/usr/sbin:/usr/ucb:/usr/bin/X11:/sbin:/usr/krb5/bin:/usr/krb
5/sbin:/usr/java14/jre/bin:/usr/java14/bin
```

Set up time synchronization

In order to prevent malicious users from replaying network traffic to authenticate to the KDC as legitimate users, the Kerberos clients' and servers' system clocks must be synchronized. The Kerberos server will automatically discard all authentication requests over the allowable clock skew. By default, the maximum clock skew allowed by Kerberos is 300 seconds. One method of maintaining minimal clock skew is by using the standard Network Time Protocol (NTP).

In this scenario we synchronize the AIX 5L clients to the same servers used to synchronize the domain controller. The following xntpd configuration file instructs the NTP client to synchronize with the NTP servers timeserver1, timeserver2, and timeserver3.example.com. Modify the /etc/ntp.conf file as shown in Example 8-2.

Example 8-2 ntp.conf

```
server timeserver1.example.com
server timeserver2.example.com
server timeserver3.example.com

driftfile /etc/ntp.drift
tracefile /etc/ntp.trace
```

Now that you have the NTP client configured, you must now synchronize the system clock and then start the NTP client to keep the clock synchronized. Before starting the xntpd daemon for the first time, you should run the **ntpdate** command to do a one-time synchronization. You must then start the **xntpd** daemon using the **startsrc** command. After the xntpd daemon is running for some time, you can use the **ntpq** command to check the synchronization status. Example 8-3 shows the initial time sync, starting the xntpd daemon, and using **ntpq** to check its status.

Example 8-3 Configuring and starting AIX 5L NTP client

```
# ntpdate timeserver1.example.com
15 Sep 12:54:50 ntpdate[110834]: adjust time server 9.60.53.163 offset
-0.001759 sec
# startsrc -s xntpd
0513-059 The xntpd Subsystem has been started. Subsystem PID is 192706.
# ntpq
ntpq> peers
     remote           refid      st t when poll reach   delay   offset    disp
==============================================================================
+timeserver1.exa ntp-gps1.example 2 u   20   64   37    52.14   -1.349  875.15
*timeserver2.exa ntp-gps2.example 2 u   20   64   37    47.81    1.508  875.38
+timeserver3.exa ntp-gps1.example 2 u   20   64   37    55.02   -2.027  875.23
ntpq> associations
ind assID status  conf reach auth condition  last_event cnt
===========================================================
  1 62927  9414   yes   yes  none   synchr.   reachable  1
  2 62928  9614   yes   yes  none  sys.peer   reachable  1
  3 62929  9414   yes   yes  none   synchr.   reachable  1
ntpq> quit
```

You should now ensure that the xntpd daemon starts automatically upon startup. You must uncomment the start /usr/sbin/xntpd line in /etc/rc.tcpip. The following example shows the start line for xntpd uncommented:

```
# Start up Network Time Protocol (NTP) daemon
start /usr/sbin/xntpd "$src_running"
```

Configure NAS client

You must now configure the NAS client either using the **`config.krb5`** or **`mkkrb5clnt`** command. Which command you use will depend on the whether your Kerberos servers supports the kadmin interface. For this MSAD scenario we use the KRB5A module, as the Windows 2003 does not support the kadmin interface.

Example 8-4 shows us configuring the NAS client using the **`config.krb5`** command and the following parameters:

- -C

  This parameter specifies that we are configuring an NAS client.

- -c bs01b06.example.com

  This is the host name of the machine running the Kerberos KDC server. In this scenario the KDC will be running on the Windows domain controller.

- -d example.com

  This is the domain name that maps host names to the Kerberos realm specified by the -r parameter.

- -r EXAMPLE.COM

  This is the Windows domain name also known as the Kerberos realm. The realm name must be capitalized.

- -s bs01b06.example.com

  This is the host name of the machine running the kadmin server. This scenario uses the KRB5A authentication module, which does not use the kadmin interface. This parameter is still required, so we will just give the host name of the domain controller.

Example 8-4 Configuration of NAS client using config.krb5 command

```
# config.krb5 -C -r EXAMPLE.COM -d example.com -c bs01b06.example.com -s
bs01b06.example.com
Initializing configuration...
```

```
Creating /etc/krb5/krb5_cfg_type...
Creating /etc/krb5/krb5.conf...
The command completed successfully.
#
```

Attention: The Windows domain name, specified with the -r option of the **`config.krb5`** command, must be capitalized.

Update encryption types

The **`config.krb5`** command creates the NAS client configuration file /etc/krb5/krb5.conf. The default values for the default_tkt_enctypes and default_tgs_enctypes attributes in the [libdefaults] stanza are as follows:

```
[libdefaults]
      default_realm = EXAMPLE.COM
      default_keytab_name = FILE:/etc/krb5/krb5.keytab
      default_tkt_enctypes = des3-cbc-sha1 arcfour-hmac aes256-cts des-cbc-md5
      des-cbc-crc.
      default_tgs_enctypes = des3-cbc-sha1 arcfour-hmac aes256-cts des-cbc-md5
      des-cbc-crc.
```

Currently Windows 2000 and 2003 domain controllers only support the encryption types des-cbc-md5 and des-cbc-crc. You must modify the [libdefaults] stanza in /etc/krb5/krb5.conf file with the correct encryption types. See Example 8-5 for the krb5.conf file for this scenario.

Example 8-5 NAS client configuration program /etc/krb5/krb5.conf

```
# cat /etc/krb5/krb5.conf
[libdefaults]
        default_realm = EXAMPLE.COM
        default_keytab_name = FILE:/etc/krb5/krb5.keytab
        default_tkt_enctypes = des-cbc-md5 des-cbc-crc
        default_tgs_enctypes = des-cbc-md5 des-cbc-crc

[realms]
        EXAMPLE.COM = {
                kdc = bs01b06.example.com:88
                admin_server = bs01b06.example.com:749
                default_domain = example.com
        }

[domain_realm]
        .example.com = EXAMPLE.COM
        bs01b06.example.com = EXAMPLE.COM

[logging]
```

```
        kdc = FILE:/var/krb5/log/krb5kdc.log
        admin_server = FILE:/var/krb5/log/kadmin.log
        default = FILE:/var/krb5/log/krb5lib.log
```

Define required authentication methods in methods.cfg

You must now modify the /usr/lib/security/methods.cfg file to add the KRB5A and KRB5Afiles stanzas. The KRB5A stanza defines a simple load module for the Kerberos V5 authentication-only load module. The KRB5Afiles stanza defines a compound load module that will use KRB5A for authentication and the BUILTIN load module for storage of user information. The BUILTIN load module will store the user information in the normal files /etc/passwd, /etc/groups, and so on.

Add the lines to the /usr/lib/security/methods.cfg file as shown in Example 8-6.

Example 8-6 KRB5A and KRB5A file stanzas in methods.cfg

```
KRB5A:
        program = /usr/lib/security/KRB5A
        options = authonly

KRB5Afiles:
        options = db=BUILTIN,auth=KRB5A
```

8.1.2 Configuring AIX 5L NAS client

The following sections cover the steps required to have your AIX 5L NAS client join your Windows Kerberos Domain. They discuss:

- Creating the host principal for the AIX 5L NAS client
- Generating the keytab file using the **`ktpass`** command and securely transporting the keytab to the AIX 5L client
- Importing the host principal keys into the default NAS client keytab

Create the host principal in the Windows Kerberos domain

You must now create a Kerberos host principal for each of your AIX 5L clients in the Windows domain. The host principal is used by the NAS client to verify that the ticket-granting ticket (TGT) received from the KDC is valid. A Kerberos principal name is multipart in a specific format. The following section describes the multiple parts of the kerberos principal name.

`SERVICE/HOSTNAME@REALM`

SERVICE	Specifies the service name
HOSTNAME)	Specifies the host name of the server
REALM	Specifies the Kerberos realm principal is a member

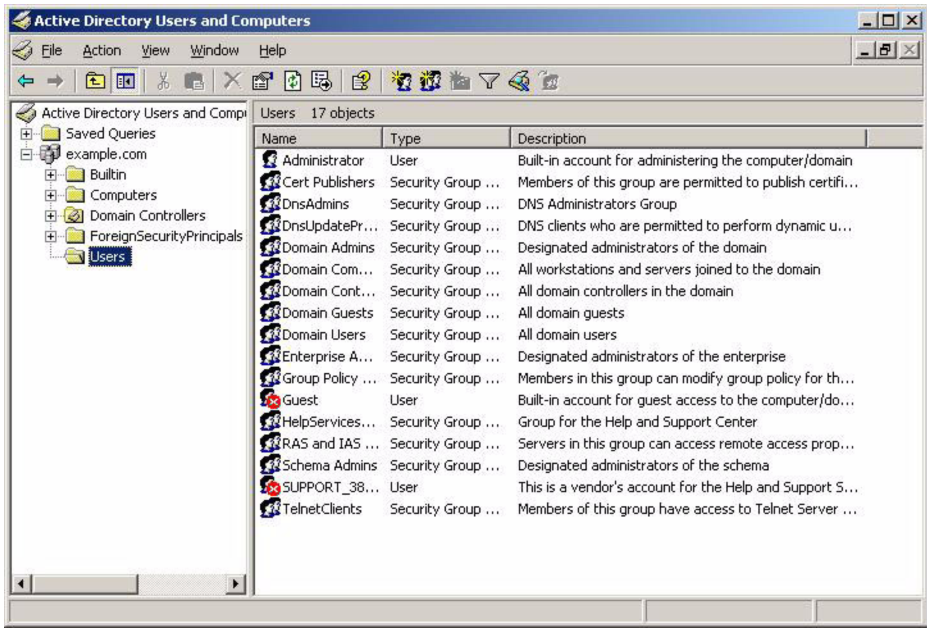
Important: If your NAS client has problems determining the fully qualified domain name of the local host, your TGT verification might fail. To avoid this problem make sure that the AIX naming server returns the fully qualified domain name for the local server. This problem is often seen when the local host name resolves to the short name because the short name is in the /etc/hosts file and /etc/netsvc.conf specifies hosts=local,bind.

Note: In AIX 5L Version 5.2 ML02 and Version 5.3 ML01 an option is introduced to make TGT verification optional. IBM white paper *Configuring AIX 5L for Kerberos Based Authentication Using Windows Kerberos Service* includes a section covering this option. The paper can be found at the following Web site:

`http://www-03.ibm.com/servers/aix/whitepapers/aix_kerberos2.pdf`

In this scenario we create a host principal for our AIX 5L client with a host name of bs01b08.example.com. The Kerberos principal is named host/bs01b08.example.com@EXAMPLE.COM. Windows servers do not support the creation of multipart account names, so you will need to create an account and use the ktpass program, packaged with the Windows support tools, to map the Windows user name to the Kerberos principal name. By choosing descriptive account names such as host_bs01b08 or service_bs01b08 you can easily distinguish between user accounts and host, service, and user principals.

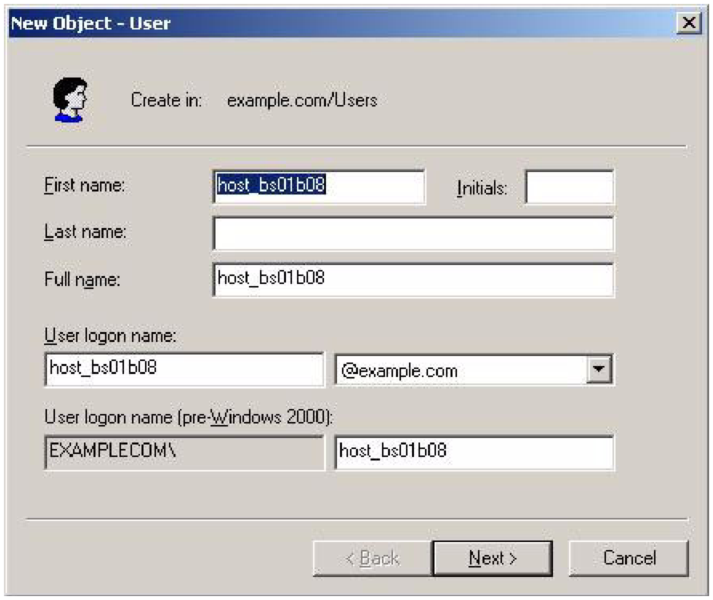
One method to create a host account is to use the Windows Active Directory Users and Computers management tool. Start the tool by selecting **Start** → **Administrative tools** → **Active Directory Users and Computers**. When the Active Directory Users and Computers management tool starts, it will look similar to Figure 8-1.


Active Directory Users and Computers
File Action View Window Help
Active Directory Users and Compu
Saved Queries
example.com
Builtin
Computers
Domain Controllers
ForeignSecurityPrincipals
Users
Users 17 objects

Name	Type	Description
Administrator	User	Built-in account for administering the computer/domain
Cert Publishers	Security Group ...	Members of this group are permitted to publish certifi...
DnsAdmins	Security Group ...	DNS Administrators Group
DnsUpdatePr...	Security Group ...	DNS clients who are permitted to perform dynamic u...
Domain Admins	Security Group ...	Designated administrators of the domain
Domain Com...	Security Group ...	All workstations and servers joined to the domain
Domain Cont...	Security Group ...	All domain controllers in the domain
Domain Guests	Security Group ...	All domain guests
Domain Users	Security Group ...	All domain users
Enterprise A...	Security Group ...	Designated administrators of the enterprise
Group Policy ...	Security Group ...	Members in this group can modify group policy for th...
Guest	User	Built-in account for guest access to the computer/do...
HelpServices...	Security Group ...	Group for the Help and Support Center
RAS and IAS ...	Security Group ...	Servers in this group can access remote access prop...
Schema Admins	Security Group ...	Designated administrators of the schema
SUPPORT_38...	User	This is a vendor's account for the Help and Support S...
TelnetClients	Security Group ...	Members of this group have access to Telnet Server ...

Figure 8-1 Active Directory Users and Computers management tool

In order to create a new user, you must right-click the **Users** folder inside the example.com container and select **New** → **User**. The first of three New Object - User dialogs will then be displayed. You must first enter the user name information, host_bs01b08, in the First name and User logon name fields. The remaining fields will be filled in automatically and do not need to be changed. Your dialog should look similar to Figure 8-2.



Create in: example.com/Users

First name: host_bs01b08

Initials:

Last name:

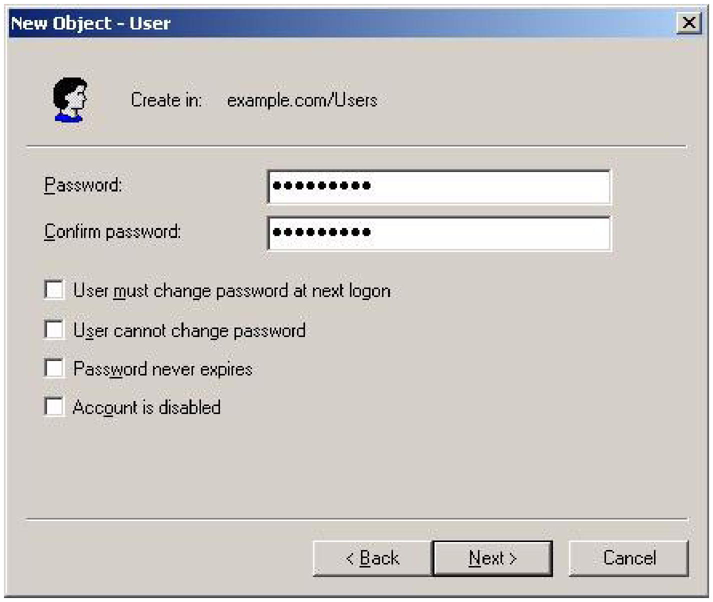
Full name: host_bs01b08

User logon name: host_bs01b08 @example.com

User logon name (pre-Windows 2000): EXAMPLECOM\ host_bs01b08

Figure 8-2 Creating host principal in MSAD (1 of 3)

You must then click the **Next** button to get to the next dialog. You must now choose a strong password for your host principal, that will pass all your password rules set in your domain security policy. The default security policy for the Windows 2003 Server requires that the minimum password length be seven characters to meet complexity requirements. You must uncheck the “User must change password at next logon” option. Your dialog should look similar to Figure 8-3.



Password: ••••••••••

Confirm password: ••••••••••

☐ User must change password at next logon

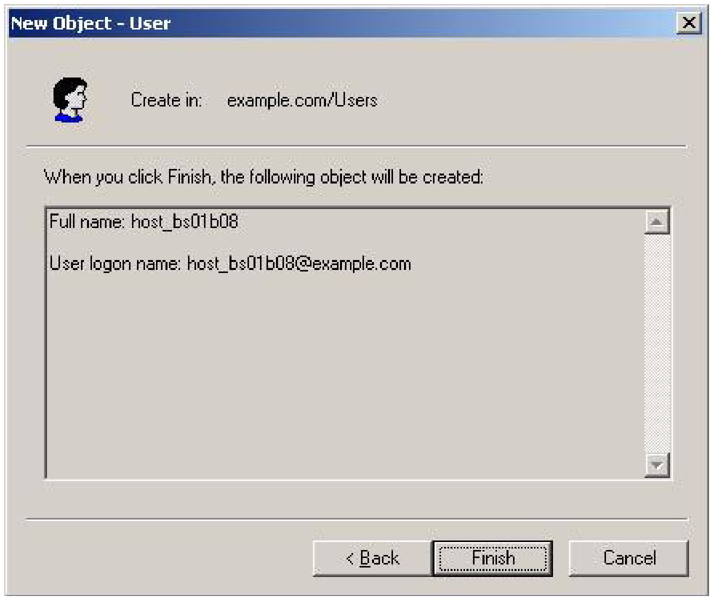
☐ User cannot change password

☐ Password never expires

☐ Account is disabled

Figure 8-3 Creating host principal in MSAD (2 of 3)

You must then select the **Next** button to get to the next dialog. You will now see the new user account confirmation dialog. To create the new user account select the **Next** button. The confirmation dialog should look similar to Figure 8-4.



Create in: example.com/Users

When you click Finish, the following object will be created:

Full name: host_bs01b08

User logon name: host_bs01b08@example.com

Figure 8-4 Creating host principal in MSAD (3 of 3)

Generating host principal keytab file

After the host principal is created, a keytab file needs to be generated and securely transferred to the AIX 5L client machines. The keytab file contains pairs of principals and encrypted keys. These keys are generated from the principal's password and host key. The keytab files are normally used to allow processes to authenticate with Kerberos without interactively entering a password or having a plaintext password in a file. The keytab file must be protected by UNIX permissions and should be readable only by root.

You must now run the **`ktpass`** command to create the keytab file and create a mapping between the user name with the service principal name (SPN). Example 8-7 on page 230 shows the use of the **`ktpass`** command to create a keytab for principal host/bs01b08.example.ibm@EXAMPLE.COM into the file host_bs01b08.keytab. The keytab file is then securely transferred to the AIX 5L client using a secure file transfer program, such as scp.[1]

- -mapuser host_bs01b08

  Specifies the Windows account name to map to the Kerberos principal

- -princ host/bs01b08.example.ibm@EXAMPLE.COM

  Specifies the Kerberos principal that you are generating the keytab file for
- -pass passw0rd!

  Specifies the password the host principal was created with
- -out host_bs01b08.keytab

  Specifies the file name of the keytab that will be generated

Example 8-7 Using ktpass command to map principal name and generate keytab file

```
C:\>ktpass -princ host/bs01b08.example.com@EXAMPLE.COM -mapuser host_bs01b08
-pass passw0rd! -out host_bs01b08.keytab
Targeting domain controller: bs01b06.example.com
Successfully mapped host/bs01b08.example.com to host_bs01b08.
Key created.
Output keytab to host_bs01b08.keytab:
Keytab version: 0x502
keysize 63 host/bs01b08.example.com@EXAMPLE.COM ptype 1 (KRB5_NT_PRINCIPAL) vno
3 etype 0x3 (DES-CBC-MD5) keylength 8 (0xa4466868fb15e986)
Account host_bs01b08 has been set for DES-only encryption.

C:\>scp host_bs01b08.keytab root@bs01b08:/etc/krb5
root@bs01b08's password:
host_bs01b08.keytab                           100%   69     0.1KB/s   00:00
C:\>
```

The **ktpass** and **setspn** commands are not installed on Windows by default. They must be installed from the Windows Support Tools package on the product media. If you need to install these commands, please use the instructions in "Installation of Microsoft Windows Support Tools" on page 286.

After using the **ktpass** command with the -mapuser option to map the principal name to local user name, you will be unable to delete or modify the mappings. In order to view, modify, or delete a mapping you must use the **setspn** command. Example 8-8 on page 231 shows the SPN associated with the user name host_bs01b08.

[1] The **scp** command is part of the openSSH package. The openSSH LPPs for AIX 5L can be downloaded from the following Web site: `https://sourceforge.net/projects/openssh-aix`. openSSH is also available for Windows machines from the Cygwin toolkit Web site: `http://www.cygwin.com`. More information about openSSH can be found at the following Web site: `http://www.openssh.org`.

Example 8-8 Using the setspn command to list the SPN associated with host_bs01b08

```
C:\>setspn -L host_bs01b08
Registered ServicePrincipalNames for
CN=host_bs01b08,CN=Users,DC=example,DC=com:

    host/bs01b08.example.com

C:\>
```

Importing host principal keytab

After the keytab file is transferred over to the AIX 5L client, you must then use the **ktutil** command to import it into the default keytab used by the NAS client /etc/krb5/krb5.keytab. After importing the keytab file and confirming that the keytab is valid with the **kinit** command, you should delete the host_bs01b08.keytab file. Example 8-9 shows the commands to use the **ktutil**, **kinit**, and **klist** commands.

Example 8-9 Import host principal keytab into krb5.keytab using ktutil command

```
# ktutil
ktutil:  rkt /etc/krb5/host_bs01b08.keytab
ktutil:  list
slot   KVNO   Principal
------ ------ ------------------------------------------------------
     1      3     host/bs01b08.example.com@EXAMPLE.COM
ktutil:  wkt /etc/krb5/krb5.keytab
ktutil:  quit
# kinit -kt /etc/krb5/krb5.keytab
# klist
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_0
Default principal:  host/bs01b08.example.com@EXAMPLE.COM

Valid starting     Expires            Service principal
09/13/05 16:23:31  09/14/05 02:23:32  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        Renew until 09/14/05 16:23:31
# rm host_bs01b08.keytab
```

8.1.3 Creating Kerberos principal and local AIX 5L users

In order for a user to log in to the AIX 5L machine, the user must have a Kerberos principal defined in the KDC and a local user defined on the AIX 5L machine. The following sections cover the steps required to create the Kerberos principal in your Microsoft KDC and create local AIX 5L users.

Creating Kerberos principals in Microsoft KDC

In order to create Kerberos principals use the Active Directory Users and Computers management tool. This scenario uses the user account test01. Using the documentation in "Create the host principal in the Windows Kerberos domain" on page 224.

Creating local users on AIX 5L

You must now create a the local user entries on your AIX 5L client by either using the **mkuser** command or SMIT. The local user information includes information such as the user ID, group ID, group membership, home directory, and shell. The following example shows the creation of the test01 user using the **mkuser** command. You must set both the registry and SYSTEM attributes to the compound load module KRB5Afiles:

```
# mkuser registry=KRB5Afiles SYSTEM=KRB5Afiles test01
```

Testing user logon for the Kerberos only scenario

After the Kerberos principal and local AIX 5L user are set up, you should then be able to log on to the AIX 5L machine by authenticating against a Microsoft KDC. Example 8-10 shows a sample log on session and how to verify the user's environment. The AUTHSTATE environment variable should be set to KRB5Afiles. Use the **id** command to check the user's primary group and additional group memberships, and use the **klist** command to check whether the user received a Kerberos TGT.

Example 8-10 Testing AIX 5L logon using Kerberos login using Microsoft KDC

```
# telnet bs01b08
Trying...
Connected to bs01b08.
Escape character is '^]'.

 telnet (bs01b08)

AIX Version 5
(C) Copyrights by IBM and by others 1982, 2005.
login: test01
test01's Password:

*******************************************************************************
*                                                                             *
*                                                                             *
*  Welcome to AIX Version 5.3!                                                *
*                                                                             *
*                                                                             *
*  Please see the README file in /usr/lpp/bos for information pertinent to    *
*  this release of the AIX Operating System.                                  *
*                                                                              *
*                                                                              *
********************************************************************************
$ echo $AUTHSTATE
KRB5Afiles
$ id
uid=209(test01) gid=1(staff)
$ klist
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_test01@EXAMPLE.COM_209
Default principal:  test01@EXAMPLE.COM

Valid starting     Expires            Service principal
09/13/05 16:32:45  09/14/05 02:32:45  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        Renew until 09/14/05 16:32:45
09/13/05 16:32:45  09/14/05 02:32:45  host/bs01b08.example.com@EXAMPLE.COM
$
```

8.2 Scenario: Kerberos and LDAP

The *Kerberos and LDAP* scenario is very similar to the *Kerberos only* scenario in the previous section. The user again authenticates against the Microsoft KDC, receiving a TGT upon successful authentication. The difference is that now the user and group information is stored in the Active Directory (AD).

By default, the Active Directory schema does not have the required POSIX attributes (such as user ID, group ID, and login shell) defined. Without these attributes, a user would be unable to log on to any UNIX client, using Active Directory. In order to integrate UNIX clients into Active Directory, Microsoft released a product called Microsoft Windows Services for UNIX (SFU). The server for the NIS component of SFU extends the Active Directory schema with object classes required to support UNIX clients. For more information about and how to get SFU, go to the following Web site:

http://www.microsoft.com/windowsserversystem/sfu/default.mspx

One complication using the SFU schema is that it does not adhere to the RFC2307 standard and different versions of SFU have different object class and attribute names. There are two versions of the SFU schemas, SFU 2.0 and SFU 3.0/3.5. You can recognize the SFU 3.0/3.5 schema, as it has object class and attribute names prefixed with msSFU30. In order to handle the nonstandard schema, the LDAP client must support object class and attribute mapping.

The AIX 5L Version 5.2 and later LDAP client supports both object class and attribute mapping for the user and group maps. This allows the AIX 5L client to support standard and nonstandard schemas. AIX 5L ships with predefined map files for the AIX, RFC2307, and RFC2307AIX schemas. In order to integrate your AIX 5L client with Active Directory, you must create map files that match your SFU schema version.

The Active Directory integration scenarios in this book use the Microsoft SFU 3.0/3.5 schema and will only use the user and group maps. This is because AIX 5L only supports object class and attribute mapping for users and groups.

Microsoft announced that the upcoming Windows 2003 Server R2 will now support a RFC2307 compliant schema. Since Windows 2003 Server R2 is not released, this book does not cover this topic.

For more discussion on the planning considerations for integrating with MSAD refer to 3.4, "Microsoft Active Directory considerations" on page 73.

8.2.1 Microsoft Windows 2003 Active Directory configuration

The following sections discuss the steps to configure the Microsoft Windows 2003 Active Directory to support UNIX integration. The basic steps are:

1. Install the Active Directory Schema MMC Snap-in.
2. Download and install Microsoft Windows Services for UNIX 3.5.
3. Create users and groups in the Active Directory domain with UNIX attributes.
4. Synchronize group membership attribute msSFU30MemberUid from msSFU30PosixMember.

Installing the Active Directory Schema MMC Snap-in

Install the Active Directory Schema snap-in for MMC so you can browse and modify the schema in your directory. The procedure to install the MMC snap-in is as follows:

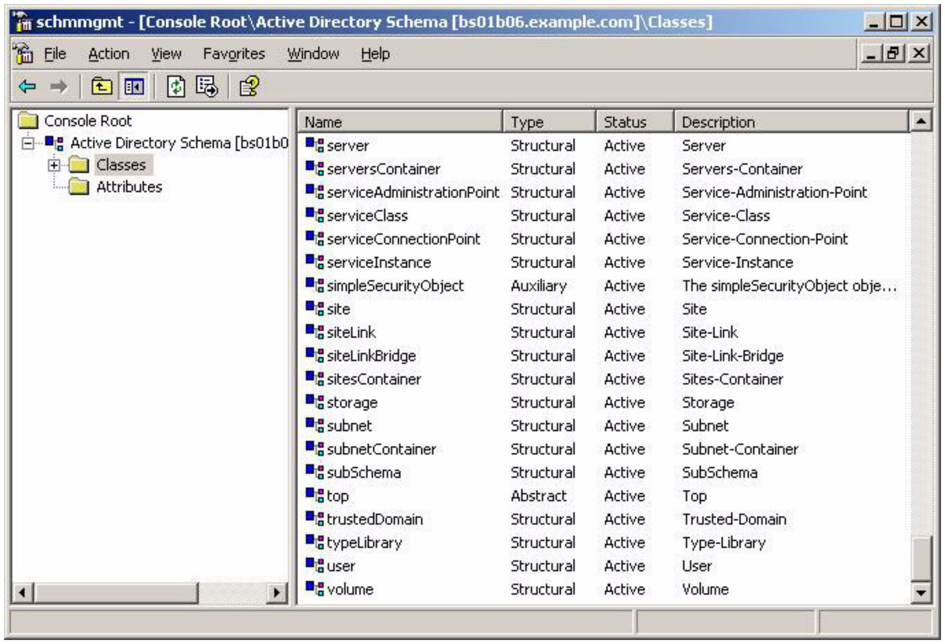
1. Register the Active Directory Schema Master snap-in. This is accomplished by running the following command:

   ```
   regsvr32 schmmgmt.dll
   ```

2. Start the Management Console (MMC) by selecting **Start**, then **Run**. Type `mmc /a` and select **OK**.
3. Select the **File** → **Add/Remove Snap-in** menu to open the Add/Remove Snap-in dialog.
4. Select the **Add** button to open the Add Standalone Snap-In dialog. Select the **Active Directory Schema snap-in** and then the **Add** button.
5. Select **Close** and then the **OK** button to complete the snap-in installation.

6. Select **File** → **Save as** and save the console configuration in the %SYSTEMROOT%\system32 directory with a file name of schmmgmt.msc.
7. Create a shortcut in the Administrative Tools folder in your Start menu by right-clicking **Start** and then **Open All Users**. Select the **Program** folder and then the **Administrative Tools** folder.
8. Select **File** → **New** → **Shortcut**. Then enter the location of the saved console, `%SYSTEMROOT%\system32\schmmgmt.msc`, in the “Type the location of the item” field. Select **Next** to continue.
9. Enter the name of the new shortcut, `Active Directory Schema`, in the “Type a name for this shortcut” field.
10. Start the Windows Active Directory Schema management tool by selecting **Start** → **Administrative tools** → **Active Directory Schema**.

When the Active Directory Schema management tool starts, it will look similar to Figure 8-5.


schmmgmt - [Console Root\Active Directory Schema [bs01b06.example.com]\Classes]
File
Action
View
Favorites
Window
Help
Console Root
Active Directory Schema [bs01b0
Classes
Attributes

Name	Type	Status	Description
server	Structural	Active	Server
serversContainer	Structural	Active	Servers-Container
serviceAdministrationPoint	Structural	Active	Service-Administration-Point
serviceClass	Structural	Active	Service-Class
serviceConnectionPoint	Structural	Active	Service-Connection-Point
serviceInstance	Structural	Active	Service-Instance
simpleSecurityObject	Auxiliary	Active	The simpleSecurityObject obje...
site	Structural	Active	Site
siteLink	Structural	Active	Site-Link
siteLinkBridge	Structural	Active	Site-Link-Bridge
sitesContainer	Structural	Active	Sites-Container
storage	Structural	Active	Storage
subnet	Structural	Active	Subnet
subnetContainer	Structural	Active	Subnet-Container
subSchema	Structural	Active	SubSchema
top	Abstract	Active	Top
trustedDomain	Structural	Active	Trusted-Domain
typeLibrary	Structural	Active	Type-Library
user	Structural	Active	User
volume	Structural	Active	Volume

Figure 8-5 Active Directory Schema management tool

Install Microsoft Windows Services for UNIX (SFU)

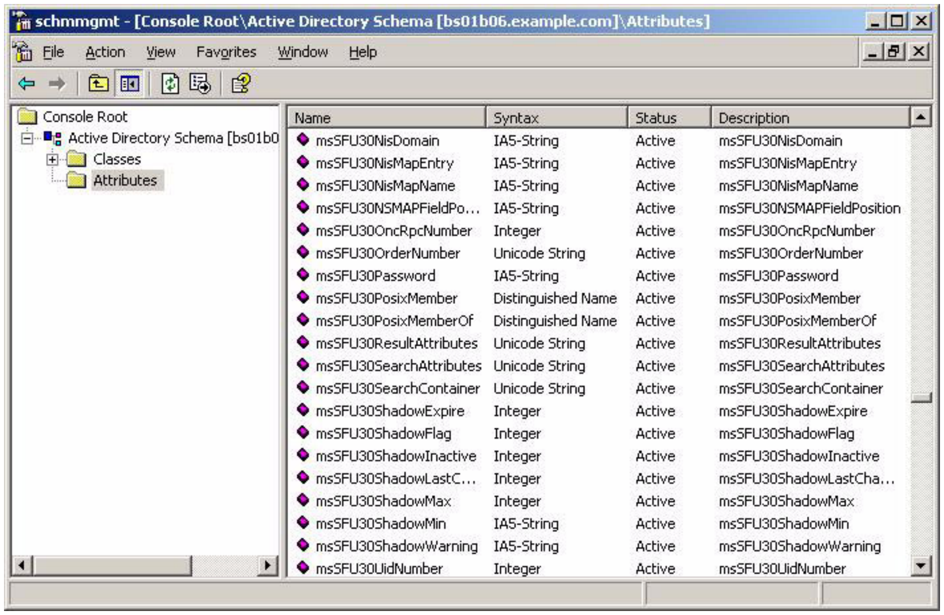
You must now download and install the Microsoft SFU 3.5 package. The following section describes the steps required. For more information please refer to the Microsoft SFU site at the Web site given in step 1.

1. Download Microsoft Windows Services for UNIX 3.5 from the Microsoft Web site:

   `http://www.microsoft.com/windowsserversystem/sfu/default.mspx`

2. Unzip the downloaded file by executing it and select the destination folder to unzip the SFU installation into.
3. Start the installation of SFU by selecting **Start**, then **Run**. Select **Browse** and select the program **setup.exe** in the folder you unpacked SFU into. Select **Open** to select the program to execute, then select **OK** to continue.
4. At the welcome to SFU dialog, select **Next** to continue.
5. At the Customer Information dialog, enter the appropriate information into the User name and Organization fields. Select **Next** to continue.
6. At the License and Support Information dialog, read the license agreement and select the **I accept the agreement** option. Select **Next** to continue.
7. At the Installation options dialog, select the **Custom installation** option. Select **Next** to continue.
8. At the Selecting Components dialog, select the **NIS Server** for installation. Select **Next** to continue.
9. Select **Next** to continue.
10. Accept the default installation location of C:\SFU by selecting the **Next** button.
11. Installation of SFU is complete. Select **Finish** to continue.
12. After the installation, the installer will upgrade the Active Directory schema with the SFU schema. This upgrade is not reversible. Select **Next** to continue.
13. Reboot the server to complete the installation.

Now that SFU is installed, you will see object classes and attribute names from the SFU schema in the Active Directory Schema management tool. Figure 8-6 shows an example of what the schema management tool would look like.


schmmgmt - [Console Root\Active Directory Schema [bs01b06.example.com]\Attributes]
File
Action
View
Favorites
Window
Help
Console Root
Active Directory Schema [bs01b0
Classes
Attributes

Name	Syntax	Status	Description
msSFU30NisDomain	IA5-String	Active	msSFU30NisDomain
msSFU30NisMapEntry	IA5-String	Active	msSFU30NisMapEntry
msSFU30NisMapName	IA5-String	Active	msSFU30NisMapName
msSFU30NSMAPFieldPo...	IA5-String	Active	msSFU30NSMAPFieldPosition
msSFU30OncRpcNumber	Integer	Active	msSFU30OncRpcNumber
msSFU30OrderNumber	Unicode String	Active	msSFU30OrderNumber
msSFU30Password	IA5-String	Active	msSFU30Password
msSFU30PosixMember	Distinguished Name	Active	msSFU30PosixMember
msSFU30PosixMemberOf	Distinguished Name	Active	msSFU30PosixMemberOf
msSFU30ResultAttributes	Unicode String	Active	msSFU30ResultAttributes
msSFU30SearchAttributes	Unicode String	Active	msSFU30SearchAttributes
msSFU30SearchContainer	Unicode String	Active	msSFU30SearchContainer
msSFU30ShadowExpire	Integer	Active	msSFU30ShadowExpire
msSFU30ShadowFlag	Integer	Active	msSFU30ShadowFlag
msSFU30ShadowInactive	Integer	Active	msSFU30ShadowInactive
msSFU30ShadowLastC...	Integer	Active	msSFU30ShadowLastCha...
msSFU30ShadowMax	Integer	Active	msSFU30ShadowMax
msSFU30ShadowMin	IA5-String	Active	msSFU30ShadowMin
msSFU30ShadowWarning	IA5-String	Active	msSFU30ShadowWarning
msSFU30UidNumber	Integer	Active	msSFU30UidNumber

Figure 8-6 Active Directory Schema Management tool after SFU installation

Create user and group with UNIX attributes

Now that you have installed the Microsoft Windows Services for UNIX, you will be able to use the Active Directory Users and Computers management tool to create and modify users and groups with UNIX POSIX attributes. Start the tool by selecting **Start** → **Administrative tools** → **Active Directory Users and Computer**.

This section discusses the creation of the user and groups needed for this scenario. This scenario uses one user, named test02, and three different groups named testgrp1, testgrp2, and testgrp3. The user would have the following entry in the /etc/passwd file:

```
test02:!:10000:10000:test02:/home/test02:/usr/bin/ksh
```

The three groups would have the following entries in the /etc/group file:

```
testgrp1:!:10000:test02
testgrp2:!:10001:test02
testgrp3:!:10002:test02
```

To create the users and groups:

1. Create the groups testgrp1, testgrp2, and testgrp3. Assign group UNIX attributes, group ID to testgrp1, testgrp2, testgrp3.
2. Create the user test02. Assign UNIX attributes user ID, primary group, home directory, and login shell for user test02. Reset the user password to synchronize the Windows and UNIX passwords.
3. Add user test02 as a member of testgrp2 and testgrp3.

Steps to create groups with UNIX attributes

This list goes through the steps required to create the group and assign the UNIX attributes for the testgrp1 group. Repeat this list to create groups testgrp2 and testgrp3:

1. Open the New Object - Group dialog by selecting the **example.com**/**Users** container, then right-click and select **New** → **Group**.
2. Enter `testgrp1` in the Group name field. The default values **Global** and **Security** for Group scope and Group type should be selected. Select **OK** to create the group.
3. In order to add the UNIX attributes to your new group, open the testgrp1 Properties dialog by selecting the **testgrp1** entry, and then RMB select **Properties**. Then select the **UNIX Attributes** tab.
4. Select **example** for the NIS Domain menu field. The next available UNIX Group ID will be chosen. Select **OK** to accept the default GID and the save changes.

Steps to create users with UNIX attributes

This list goes through the steps required to create the user, assign the UNIX attributes, and synchronize the windows and UNIX password for the user test02:

1. Open the New Object - User dialog by selecting the **example.com**/**Users** container, then right-click and select **New** → **User**.
2. Enter `test02` in the First name and User logon name fields. The Full name and User logon name (pre-Windows 2000) fields will be completed automatically. Select **Next** to continue.
3. You must now create an strong initial password for the user and enter it into the Password and Confirm Password fields. You should also uncheck the **User must change password at next logon** option. Select **Next** to continue. Select **Finish** to create the account.
4. In order to add the UNIX attributes to your new user, open the test02 Properties dialog by selecting the **test02** entry, then right-click and select **Properties**. Then select the **UNIX Attributes** tab.

5. Select **example** for the NIS Domain menu field. The next available UNIX User ID will be chosen. Enter `/usr/bin/ksh` for the login shell. Select **testgrp1** in the Primary group name/GID menu. Select **OK** to accept the default UID and save the changes.
6. You must now synchronize the user's password with the SFU password attribute, msSFUPassword or msSFU30Password, by selecting the **test02** entry, and then right-clicking and selecting **Reset Password**.
7. You must now enter a strong password for the user and enter it into the New Password and Confirm Password fields. You should confirm that the "User must change password at next logon" option is unchecked. Select **OK** to continue.

Steps to add user as a member of a UNIX group

Here we give the steps required to add the user test02 as a member of the testgrp2 group. Repeat these steps to add test02 to the testgrp3 group:

1. In order to add UNIX users to additional groups, open the testgrp2 Properties dialog by selecting the **testgrp2** entry, and then right-click and select **Properties**. Then select the **UNIX Attributes** tab and then the **Add** button.
2. Select the user **test02**, and then select the **Add** button. You can then select **Add** again to add additional group members. Select **OK**, then **OK** again, to accept the additional group members.

Synchronizing group membership attributes

Microsoft SFU determines group membership with two different attributes. One attribute uses user login names and the other uses distinguished names (DNs).

The msSFU30MemberUid attribute is defined in the msSFU30PosixGroup object class as a multi-valued list of user names, where the msSFU30PosixMember attribute is defined in the msSFU30PosixGroup object class as a multi-valued list of distinguished names (DN). Using the msSFU30PosixMember attribute allows you to specify a group membership that extends beyond a single container. It would also be possible to have group members with the same user login names from different containers.

While the ability to have group membership lists that are uniquely identifiable and cross multiple containers is very useful, the UNIX LDAP client support is not very widespread. Currently AIX 5L does not support defining group memberships using DNs. The easiest method to resolve this problem is by synchronizing the DN and user login name membership lists with an external program. This allows you to support clients who use user login names and DNs for group membership. One problem with this solution is that you would be unable to distinguish between users with the same user name in multiple containers.

Important: If you do not synchronize your group membership attributes, your users will only have a primary group set when logging into an AIX 5L machine.

For a more detailed discussion of the group membership attributes please refer to “Multiple attributes to specify group memberships” on page 79.

Example 8-11 shows the msSFU30PosixMember and msSFU30MemberUid attributes after they were synchronized externally. The test02 and test03 users are both in the same container.

Example 8-11 lsldap output showing memberUid and posixMember group membership

```
# lsldap -a group testgrp2
dn: CN=testgrp2,CN=Users,DC=example,DC=com
objectClass: top
objectClass: group
cn: testgrp2
distinguishedName: CN=testgrp2,CN=Users,DC=example,DC=com
instanceType: 4
whenCreated: 20050915223328.0Z
whenChanged: 20050919053249.0Z
uSNCreated: 20494
uSNChanged: 20720
name: testgrp2
objectGUID: ;&H[0l
objectSid:
sAMAccountName: testgrp2
sAMAccountType: 268435456
groupType: -2147483646
objectCategory: CN=Group,CN=Schema,CN=Configuration,DC=example,DC=com
msSFU30Name: testgrp2
msSFU30GidNumber: 10001
msSFU30MemberUid: test03
msSFU30MemberUid: test02
msSFU30NisDomain: example
msSFU30PosixMember: CN=test03,CN=Users,DC=example,DC=com
msSFU30PosixMember: CN=test02,CN=Users,DC=example,DC=com
#
```

8.2.2 AIX 5L LDAP client installation

The following sections discuss the steps to install the AIX 5L LDAP client. The steps are as follows:

1. Install and configure the AIX 5L NAS client as in the previous scenario (8.1, “Scenario: Kerberos only” on page 219, except for the following sections:

"Define required authentication methods in methods.cfg" on page 224 and 8.1.3, "Creating Kerberos principal and local AIX 5L users" on page 231).

2. Install the AIX 5L LDAP client using the instructions in 4.2.1, "AIX 5L LDAP client software installation" on page 105.
3. Define the KRB5A, LDAP, and KRB5ALDAP authentication methods in the methods.cfg file.

Define required authentication methods in methods.cfg

You must now modify the /usr/lib/security/methods.cfg file to add the KRB5A, LDAP, and KRB5ALDAP stanzas. The KRB5A and LDAP stanzas define the simple load modules used for Kerberos V5 *authentication only* and LDAP support, respectively. The KRB5ALDAP stanza defines a compound load module that will use KRB5A for authentication and the LDAP load module for storage of user information (user ID, primary groups, shell, home directory, and so on). Add the stanzas shown in Example 8-12 to the /usr/lib/security/methods.cfg file.

Example 8-12 KRB5A, LDAP, and KRB5ALDAP stanzas in methods.cfg

```
KRB5A:
        program = /usr/lib/security/KRB5A
        options = authonly

LDAP:
        program = /usr/lib/security/LDAP
        program_64 = /usr/lib/security/LDAP64

KRB5ALDAP:
        options = db=LDAP,auth=KRB5A
```

8.2.3 AIX 5L LDAP client configuration

This section discusses the steps to configure the AIX 5L LDAP client. The steps are:

1. Configure the AIX 5L LDAP client daemon.
2. Create object class and attribute mapping files for SFU schema.
3. Start and test the AIX 5L LDAP client daemon.
4. Test user logon in the Kerberos and LDAP scenario.

Configuring the AIX 5L LDAP client daemon

After the LDAP client is installed, you must configure the LDAP client daemon by editing the /etc/security/ldap/ldap.cfg file. Normally you could use the **mksecldap** command to do the basic LDAP client configuration, but the **mksecldap** command does not yet support the Microsoft Active Directory schema.

Example 8-13 shows the minimal LDAP client configuration used for this scenario. Modify the /etc/security/ldap/ldap.cfg file and change the settings, as shown in Example 8-13.

Example 8-13 AIX 5L LDAP client daemon configuration in /etc/security/ldap/ldap.cfg

```
# Comma separated list of ldap servers this client talks to
ldapservers:bs01b06.example.com

# LDAP server bind distinguished name (DN) and password
binddn:cn=host_bs01b08,cn=Users,dc=example,dc=com
bindpwd:passw0rd!

# AIX 5L-LDAP attribute map path.
userattrmappath:/etc/security/ldap/MSSFU30user.map
groupattrmappath:/etc/security/ldap/MSSFU30group.map

# Base DN where the user and group data are stored in the LDAP server.
userbasedn:cn=Users,dc=example,dc=com
groupbasedn:cn=Users,dc=example,dc=com

# LDAP class definitions
userclasses:User
groupclasses:Group
```

The following is a description of the settings used in the previous configuration file to connect to the Active Directory. For a detailed discussion about the AIX 5L LDAP client settings please refer to 4.2, “Installation and basic client configuration” on page 105.

- binddn, bindpwd

  These options specify the DN and password to bind to the Active Directory. Unlike the Windows 2000 Server, the Windows 2003 Server Active Directory does not allow anonymous bind by default.

- groupattrmappath

  This option specifies the file name of the objectclass and attribute map files for the group maps.

- groupclasses

  This option specifies the objectclasses that the group entries use.

- userattrmappath

  This option specifies the file name of the objectclass and attribute map files for the user maps.

- userclasses

  The option specifies the objectclasses that the user entries use.

- userbasedn,groupbasedn

  This option specifies the location of the user and group containers in the directory. Most UNIX LDAP implementations split the user and group entries into different containers. Microsoft uses the same container for both user and group entries. Microsoft also supports the use of multiple containers to store user and group information, but the AIX 5L LDAP client only supports one container at a time.

Create object class and attribute name map files

AIX 5L does not ship with map files for any version of the Microsoft SFU schema, so you are required to create user and group map files for your version of SFU. Map files for SFU 2.0 and SFU 3.0/3.5 can be found at the end of this chapter and can be customized to match your environment. For more information about possible customization to the map files please refer to “Microsoft Windows Services for UNIX mappings” on page 314.

This scenario is based on SFU 3.5 and used the map files found in 8.3.6, “Object class and attribute mapping files for SFU” on page 266. The map files for SFU 2.0 were not tested, but are included for your reference.

Starting AIX 5L LDAP client daemon

Now that you have created the user and group mapping tables, you can start the AIX 5L LDAP client daemon. The following example shows how to start the LDAP client daemon with the **start-secldapclntd** command:

```
# start-secldapclntd
Starting the secldapclntd daemon.
The secldapclntd daemon started successfully.
#
```

You can confirm the LDAP client daemon started successfully by looking for errors in the syslog messages. You will see the following message in the syslog if the client daemon started successfully:

```
Oct  2 10:31:42 bs01b08 daemon:info secldapclntd: Successfully started
secldapclntd.
```

You must now add an entry to /etc/inittab so the LDAP client daemon will start when the machine is rebooted. The following example uses the **mkitab** command to create the appropriate entry in /etc/inittab:

```
# mkitab "ldapclntd:2:once:/usr/sbin/secldapclntd > /dev/console 2>&1”
```

Testing LDAP client daemon using the lsldap command

You should now test the AIX 5L LDAP client daemon using the **lsldap** command. The **lsldap** command allows you see whether the options such as the userbasedn, groupbased, userclasses, groupclasses, and other options specified in ldap.cfg are correct. Example 8-14 shows two uses of the **lsldap** command. The first **lsldap** shows the user account information or the user test02 in the directory. The second **lsldap** shows the group information for the testgrp2 group.

Example 8-14 Testing the AIX 5L LDAP client using the lsldap client

```
# lsldap -a passwd test02
dn: CN=test02,CN=Users,DC=example,DC=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: user
cn: test02
givenName: test02
distinguishedName: CN=test02,CN=Users,DC=example,DC=com
instanceType: 4
whenCreated: 20050915223856.0Z
whenChanged: 20050915224109.0Z
displayName: test02
uSNCreated: 20541
uSNChanged: 20548
name: test02
objectGUID: zA Ñ½äEŽā-ŐçŠ,%
userAccountControl: 512
badPwdCount: 0
codePage: 0
countryCode: 0
badPasswordTime: 127718156514330590
lastLogoff: 0
lastLogon: 127719843552028366
pwdLastSet: 127712975366868032
primaryGroupID: 513
objectSid:
accountExpires: 9223372036854775807
logonCount: 7
sAMAccountName: test02
sAMAccountType: 805306368
```

```
userPrincipalName: test02@example.com
objectCategory: CN=Person,CN=Schema,CN=Configuration,DC=example,DC=com
msSFU30Name: test02
msSFU30UidNumber: 10000
msSFU30GidNumber: 10000
msSFU30LoginShell: /usr/bin/ksh
msSFU30Password: ABCD!efgh12345$67890
msSFU30NisDomain: example
msSFU30HomeDirectory: /home/test02
msSFU30PosixMemberOf: CN=testgrp6,CN=Users,DC=example,DC=com
msSFU30PosixMemberOf: CN=testgrp3,CN=Users,DC=example,DC=com
msSFU30PosixMemberOf: CN=testgrp2,CN=Users,DC=example,DC=com
#
# lsldap -a group testgrp2
dn: CN=testgrp2,CN=Users,DC=example,DC=com
objectClass: top
objectClass: group
cn: testgrp2
distinguishedName: CN=testgrp2,CN=Users,DC=example,DC=com
instanceType: 4
whenCreated: 20050915223328.0Z
whenChanged: 20050919053249.0Z
uSNCreated: 20494
uSNChanged: 20720
name: testgrp2
objectGUID: ;&-H^Ê[O—1Û
objectSid:
sAMAccountName: testgrp2
sAMAccountType: 268435456
groupType: -2147483646
objectCategory: CN=Group,CN=Schema,CN=Configuration,DC=example,DC=com
msSFU30Name: testgrp2
msSFU30GidNumber: 10001
msSFU30MemberUid: test03
msSFU30MemberUid: test02
msSFU30NisDomain: example
msSFU30PosixMember: CN=test03,CN=Users,DC=example,DC=com
msSFU30PosixMember: CN=test02,CN=Users,DC=example,DC=com
#
```

Tip: If **`lsldap`** returns an error about being unable to contact the secldapclntd daemon, you will need to start the secldapclntd daemon using the **`start-secldapclntd`** command. If the secldapclntd fails to start, there is probably a error in the LDAP client daemon configuration file. In order to fix the problem, look in the syslog messages for errors from the LDAP client daemon.

Testing user logon for the Kerberos and LDAP scenario

Now that the client daemon is configured, you should be able to log on to the AIX 5L machine. Example 8-15 on page 246 shows a sample logon session and how to verify the user's environment. The AUTHSTATE environment variable should be set to KRB5ALDAP. Use the **id** command to check the user's primary group and additional group memberships, and use the **klist** command to check whether the user received a Kerberos TGT.

Example 8-15 Testing user login for the Kerberos and LDAP scenario

```
# telnet bs01b08.example.com
Trying...
Connected to bs01b08.example.com.
Escape character is '^]'.

 telnet (bs01b08)

AIX Version 5
(C) Copyrights by IBM and by others 1982, 2005.
login: test02
test02's Password:

*******************************************************************************
*                                                                             *
*                                                                             *
*  Welcome to AIX Version 5.3!                                                *
*                                                                             *
*                                                                             *
*  Please see the README file in /usr/lpp/bos for information pertinent to    *
*  this release of the AIX Operating System.                                  *
*                                                                             *
*                                                                             *
*******************************************************************************

$ echo $AUTHSTATE
KRB5ALDAP
$
$ id
uid=10000(test02) gid=10000(testgrp1) groups=10001(testgrp2),10002(testgrp3)
$ klist
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_test02@EXAMPLE.COM_10000
Default principal:  test02@EXAMPLE.COM

Valid starting     Expires            Service principal
09/23/05 16:25:55  09/24/05 02:25:55  krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

```
        Renew until 09/24/05 16:25:55
09/23/05 16:25:55  09/24/05 02:25:55  host/bs01b08.example.com@EXAMPLE.COM

$
```

8.2.4 Configuring LDAP to use SSL

This section discusses how to configure the AIX 5L LDAP client and Active Directory server to use SSL to encrypt LDAP communications. This section covers the following:

- Configuring the Windows Active Directory to use SSL
- Configuring the AIX 5L LDAP client to use SSL
- Validating that your SSL setup is working as expected

For more information about why using SSL to secure your LDAP client and server communications please refer to 3.1.8, “Using SSL to secure LDAP connection” on page 59.

Windows Server SSL configuration

In order to configure SSL for LDAP on the Windows Server, you must install the MMC snap-in to manage local certificates, create a certificate request, have CER signed by a CA, import the signed certificate into the local keystore, import a CA certificate as a trusted root CA, and then reboot the server for the new configuration to take effect.

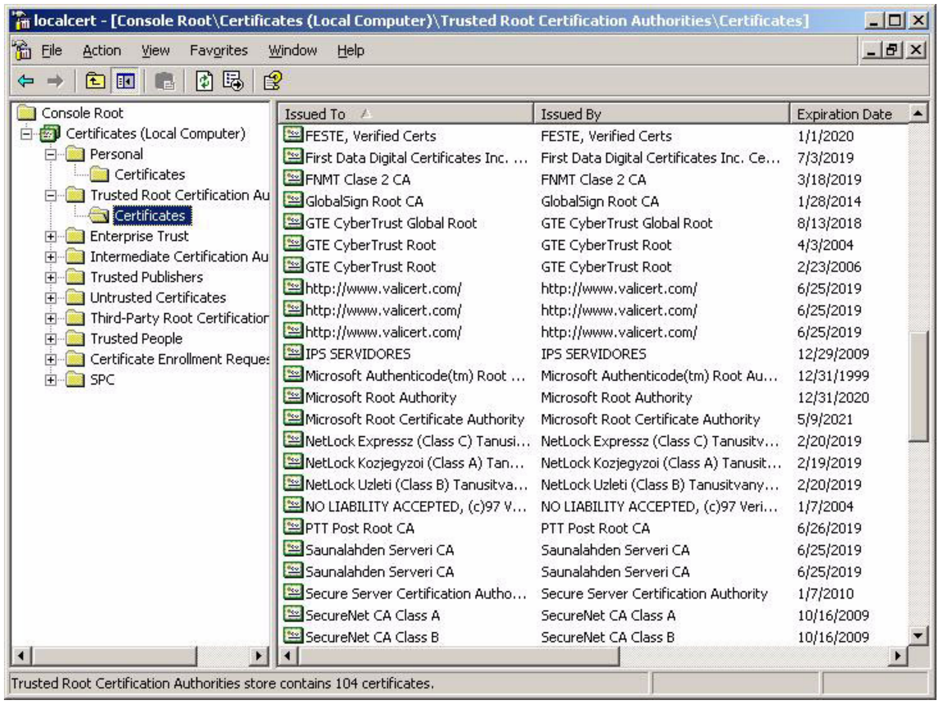
Installation of the local certificates MMC snap-in

Install the certificate snap-in for MMC so you can manage the certificates in your local machine keystore. The procedure to install the MMC snap-in is as follows.

1. Start the Management Console (MMC) by selecting **Start**, then **Run**. Type `mmc /a` and select **OK**.
2. Select the **File** → **Add/Remove Snap-in** menu to open the Add/Remove Snap-in dialog.
3. Select the **Add** button to open the Add Standalone Snap-In dialog. Select the **Certificates** snap-in and then the **Add** button.
4. Select the **Computer Account** option to manage system-wide certificates. Select the **Next** button to continue.
5. Select the **Local Computer** option to manage certificates on the local computer only. Select the **Finish**, **Close**, and then the **OK** button to complete the snap-in installation.
6. Select **File** → **Save as** and save the console configuration in the %SYSTEMROOT%\system32 directory with a file name of localcert.msc.

7. Create a shortcut in the Administrative Tools folder in your Start menu by selecting right-clicking the **Start** menu and then **Open All Users**. Select the **Program** folder and then the **Administrative Tools** folder.
8. Select the **File** → **New** → **Shortcut** menu. Then enter the location of the saved console, `%SYSTEMROOT%\system32\localcert.msc`, in the “Type the location of the item” field. Select **Next** to continue.
9. Enter the name of the new shortcut, `Certificates (Local Computer)`, in the “Type a name for this shortcut” field.
10. To start the local certificate management tool, select **Start** → **Administrative tools** → **Certificates** (**Local Computer)**.

When the local certificate management tool starts, it will look similar to Figure 8-7. The certificates used by Active Directory are located in the **Console Root** → **Certificates (Local Computer)** → **Personal** → **Certificates** folder. The list of trusted root certificates authorities is located in the **Console Root** → **Certificates (Local Computer)** → **Trusted Certification Authorities** → **Certificates** folder.


localcert - [Console Root\Certificates (Local Computer)\Trusted Root Certification Authorities\Certificates]
File
Action
View
Favorites
Window
Help
Console Root
Certificates (Local Computer)
Personal
Certificates
Trusted Root Certification Au
Certificates
Enterprise Trust
Intermediate Certification Au
Trusted Publishers
Untrusted Certificates
Third-Party Root Certificatior
Trusted People
Certificate Enrollment Reques
SPC

Issued To	Issued By	Expiration Date
FESTE, Verified Certs	FESTE, Verified Certs	1/1/2020
First Data Digital Certificates Inc. ...	First Data Digital Certificates Inc. Ce...	7/3/2019
FNMT Clase 2 CA	FNMT Clase 2 CA	3/18/2019
GlobalSign Root CA	GlobalSign Root CA	1/28/2014
GTE CyberTrust Global Root	GTE CyberTrust Global Root	8/13/2018
GTE CyberTrust Root	GTE CyberTrust Root	4/3/2004
GTE CyberTrust Root	GTE CyberTrust Root	2/23/2006
http://www.valicert.com/	http://www.valicert.com/	6/25/2019
http://www.valicert.com/	http://www.valicert.com/	6/25/2019
http://www.valicert.com/	http://www.valicert.com/	6/25/2019
IPS SERVIDORES	IPS SERVIDORES	12/29/2009
Microsoft Authenticode(tm) Root ...	Microsoft Authenticode(tm) Root Au...	12/31/1999
Microsoft Root Authority	Microsoft Root Authority	12/31/2020
Microsoft Root Certificate Authority	Microsoft Root Certificate Authority	5/9/2021
NetLock Expressz (Class C) Tanusi...	NetLock Expressz (Class C) Tanusitv...	2/20/2019
NetLock Kozjegyzoi (Class A) Tan...	NetLock Kozjegyzoi (Class A) Tanusit...	2/19/2019
NetLock Uzleti (Class B) Tanusitva...	NetLock Uzleti (Class B) Tanusitvany...	2/20/2019
NO LIABILITY ACCEPTED, (c)97 V...	NO LIABILITY ACCEPTED, (c)97 Veri...	1/7/2004
PTT Post Root CA	PTT Post Root CA	6/26/2019
Saunalahden Serveri CA	Saunalahden Serveri CA	6/25/2019
Saunalahden Serveri CA	Saunalahden Serveri CA	6/25/2019
Secure Server Certification Autho...	Secure Server Certification Authority	1/7/2010
SecureNet CA Class A	SecureNet CA Class A	10/16/2009
SecureNet CA Class B	SecureNet CA Class B	10/16/2009

Trusted Root Certification Authorities store contains 104 certificates.

Figure 8-7 Windows certificate MMC snap-in (local computer)

Generating Windows Server certificate

You must use the **certreq** command to generate a certificate request. The **certreq** command uses a policy file, which specifies the attributes needed to generate a certificate. It contains attributes such as the subject's common name, certificate key length, and additional key usage extensions. The Active Directory requires that the certificate meet the following requirements:

- The private key and certificate for the local machine must be imported into the local computer's personal keystore.
- The fully qualified domain name (FQDN) for the Active Directory must appear in the common name (CN) in the subject field or DNS entry in the subject alternative name extension.
- The certificate must be issued by a CA that the domain controller and the LDAP clients trust.
- The certificate must contain the enhanced key usage extension that specifies the server authentication object identifier (OID) 1.3.6.1.5.5.7.3.1. This OID indicates that the certificate will be used as a SSL server certificate.

Example 8-16 shows the policy file used to generate the certificate for the example.com domain controller. The subject field is set to CN=bs01b06.example.com, which is the FQDN of the domain controller. You then use the **certreq** command to generate the certificate request file.

Example 8-16 Creating the certificate request on Windows server™

```
C:\>type bs01b06.example.com_req.inf
[Version]
Signature="$Windows NT$

[NewRequest]
Subject="CN=bs01b06.example.com"
KeySpec = 1
KeyLength = 2048
Exportable = TRUE
MachineKeySet = TRUE
SMIME = False
PrivateKeyArchive = FALSE
UserProtected = FALSE
UseExistingKeySet = FALSE
ProviderName = "Microsoft RSA SChannel Cryptographic Provider"
ProviderType = 12
RequestType = PKCS10
KeyUsage = 0xa0

[EnhancedKeyUsageExtension]
OID=1.3.6.1.5.5.7.3.1
; this is for Server Authentication
```

```
C:\>
C:\>certreq -new bs01b06.example.com_req.inf bs01b06.example.com_req.pem
C:\>
```

Signing and importing Windows server certificate

After the CER is generated, you must send the request to the certificate authority to be signed. For more information about signing this certificate see “Signing a certificate for bs01b06” on page 295. After the signed certificate is returned, you must import the certificate into the local machines’s personal keystore.

Example 8-17 shows how to import the signed certificate using the **certreq** command. Confirm that the certificate is imported correctly by using the **certutil** command.

Example 8-17 Accepting the signed certificate into local certificate keystore

```
C:\>certreq -accept bs01b06.example.com_cert.pem
C:\>certutil -store my
================ Certificate 0 ================
Serial Number: 07
Issuer: E=ca@example.com, CN=CA, O=example.com, L=Austin, S=TX, C=us
Subject: CN=bs01b06.example.com
Non-root Certificate
Cert Hash(sha1): e2 25 17 4d 44 a6 8a 16 a7 da 79 71 ea 12 31 44 2d ab c1 98
  Key Container =
26dc2f1282dbc19cce4b01e2716c3df0_d8b07f77-4c5f-4dca-a0ee-4b6bc
f55e605
  Provider = Microsoft RSA SChannel Cryptographic Provider
Encryption test passed
CertUtil: -store command completed successfully.
C:\>
```

Importing Certificate Authority certificate

Until the example.com CA is designated as a trusted root, any certificates signed by that CA will be untrusted. You must import the CA’s certificate, using the local certificate management tool, into the Trusted Certification Authorities folder in the local keystore.

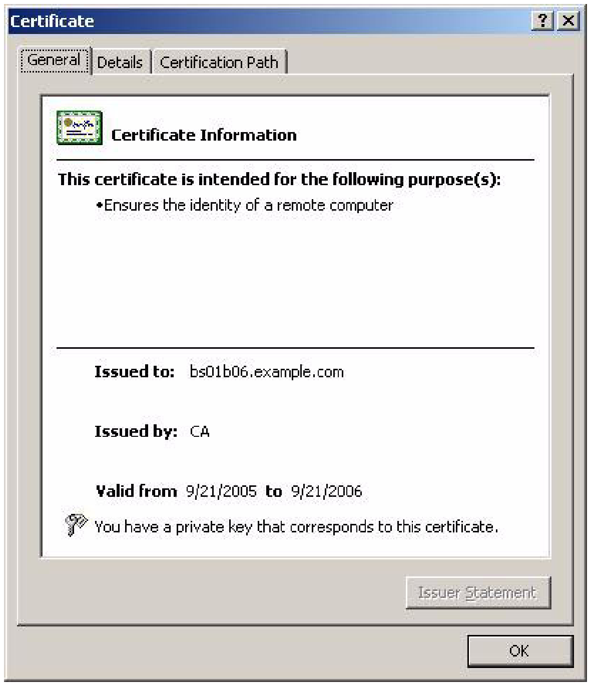
To start the local certificate management tool, you need to select **Start** → **Administrative tools** → **Certificates** (**Local Computer)**.

1. After the certificate tool opens, select the **/Console Root/Certificates (Local Computer)/Trusted Certification Authorities** folder.
2. Start the certificate import wizard by selecting the **Action** → **All Tasks** → **Import** menu. Select the **Next** button to continue.

3. Select the file you want to import. The example.com CA certificate is located in the cacert.pem file. Select the **Next** button to continue.
4. Select the **Place all certificates in the following store** option and make sure the certificate store field is set to **Trusted Root Certification Authorities**. Select the **Next** button to continue.
5. The CA certificate is now imported. Select the **Finish** button to close the wizard.

After the CA and server certificates are imported into the local keystore, you can then use the local certificate management tool to check whether the certificates are correctly imported.

Open the **Console Root** → **Certificates (Local Computer)** → **Personal** → **Certificates** folder and select the certificate issued to b01bs06.example.com. Figure 8-8 shows that the certificate issued to b01b06.example.com is valid and was issued by the example.com CA. The certificate has a corresponding private key in the keystore. The `Ensures the identity of the remote computer` text indicates that the certificate has the required server authentication key usage defined.



Certificate Information

This certificate is intended for the following purpose(s):
- Ensures the identity of a remote computer

Issued to: bs01b06.example.com

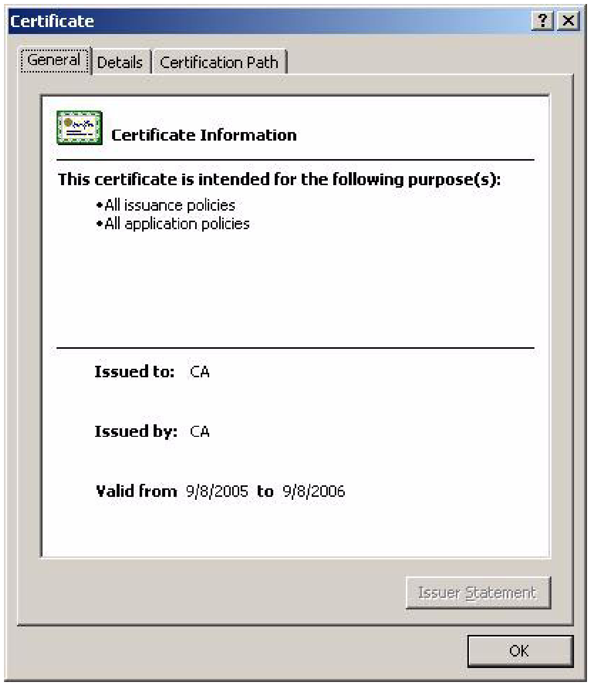
Issued by: CA

Valid from 9/21/2005 **to** 9/21/2006

You have a private key that corresponds to this certificate.

Figure 8-8 Certificate information dialog for Windows Server

To check the example.com certificate, open the **Console Root** → **Certificates (Local Computer)** → **Trusted Certification Authorities** → **Certificates** folder and select the certificate issued to b01bs06.example.com. Figure 8-9 on page 252 shows that the certificate issued to and by the example.com CA is valid.



This certificate is intended for the following purpose(s):
- All issuance policies
- All application policies

Issued to: CA

Issued by: CA

Valid from 9/8/2005 to 9/8/2006

Figure 8-9 Certificate information dialog for example.com certificate authority

Note: In order to complete the configuration of SSL for the Active Directory, you must reboot the Windows server.

AIX 5L LDAP Client SSL configuration

You must use the **gsk7cmd** command to generate a key database and then import the example.com CA certificate. This allows the LDAP client to validate that the server's certificate was issued by a trusted CA. This mitigates the problem with an LDAP client connecting against a rogue server.

Creating key database and importing CA certificate

Example 8-18 on page 253 shows the creation of the certificate management system (CMS) key database and the import of the CA certificate. The first **gsk7cmd** command creates a CMS key database named

/usr/ldap/etc/bs01b08_keydb.kdb with a password passw0rd!. The second command adds the example.com CA certificate from the /usr/ldap/etc/cacert.pem file into the key database. The trust enable parameter marks the certificate as a trusted root certificate. The third command allows you to check that the certificate was imported correctly into the key database.

Example 8-18 Creating key database and importing CA certificate on AIX 5L

```
# gsk7cmd -keydb -create -db /usr/ldap/etc/bs01b08_keydb.kdb -pw passw0rd!
-type cms -stash
#
# gsk7cmd -cert -add -db /usr/ldap/etc/bs01b08_keydb.kdb -pw passw0rd! -label
"example.com CA Certificate" -format binary -trust enable -file
/usr/ldap/etc/cacert.pem
#
# gsk7cmd -cert -details -db /usr/ldap/etc/bs01b08_keydb.kdb -pw passw0rd!
-label "example.com CA Certificate"

Label: example.com CA Certificate
Key Size: 1024
Version: X509 V3
Serial Number: 00
Issued By: CA
example.com
Austin, TX, us
Subject: CA
example.com
Austin, TX, us
Valid From: Thursday, September 8, 2005 10:20:52 AM CDT To: Friday, September
8, 2006 10:20:52 AM CDT
Fingerprint: 20:69:65:06:1B:F4:5A:AD:0E:F8:3E:11:BB:58:96:72:DF:7C:87:72
Signature Algorithm: 1.2.840.113549.1.1.4
Trust Status: enabled
#
```

Configuration of AIX 5L LDAP client

Now that you have the Active Directory server configured with SSL certificates and have created the key database on the AIX 5L client, you must now configure the AIX 5L LDAP client to use SSL. You need to change the options in the /etc/security/ldap/ldap.cfg file as shown in Example 8-19, and then restart the secldapclntd daemon using the **`restart-secldapclntd`** command. The ldapsslkeyf option is the location of the local key database you created in the previous section. The ldapsslkeypwd option is the password needed to open the key database.

Example 8-19 AIX 5L LDAP client SSL configuration section from /etc/security/ldap/ldap.cfg

```
# SSL Key database and password
useSSL: yes
ldapsslkeyf:/usr/ldap/etc/bs01b08_keydb.kdb
ldapsslkeypwd:passw0rd!
```

Example 8-20 Restarting the secldapclntd daemon to reread the configuration file

```
# restart-secldapclntd
The secldapclntd daemon terminated successfully.
Starting the secldapclntd daemon.
The secldapclntd daemon started successfully.
#
```

Validation of your SSL configuration

The following section discusses methods that you can use to verify that your LDAP client and server SSL configuration is correct. After introducing any security-related technologies in your environment, you should always go back and verify that it is working as expected.

When using SSL to secure another network protocol such as LDAP, you will often receive misleading and incorrect error messages from the application. The problem is that applications might not receive detailed diagnostics of low-level SSL failures or might just ignore failure information. For example, you might receive an application error saying unable to contact LDAP server, when your actual problem is that your LDAP server certificate failed to verify. While the error might be technically correct, the diagnostic message is not very helpful. The best way to determine where the problem is, is to start at the bottom and work your way up the application stack. In the LDAP over SSL scenario, you need to test the following layers:

- Test the low-level SSL protocol by using the s_client function of the **`openssl`** command.
- Test whether your local AIX 5L keystore and LDAP server is working properly with SSL by using the **`ldapsearch`** command.
- Test that your AIX 5L LDAP client is working correctly.

Low-level SSL validation using the openssl command

The easiest way to test the low-level SSL connection to the LDAP server is by using the **`openssl s_client`** command with the -showcerts option. This command will connect to the specified host and list the server certificate, the certificate authority chain, supported ciphers, SSL session information, and verify return code. If the SSL connection worked, the **openssl s_client** command result in the verify return code will be 0 (Ok).

Example 8-21 shows the output of the **openssl s_client** command connecting the AIX 5L machine (bs01b08.example.com) to the Active Directory server (bs01b06.example.com). The parameters used are as follows:

- -CAfile /usr/ldap/etc/cacert.pem

  This option specifies the PEM file containing the example.com's CA certificate. This allows the SSL client to verify that the server is using a certificate signed by the trusted CA.
- -connect bs01b06.example.com:636

  This connects to the host name of the Active Directory server using the secure LDAP port (636).
- -showcerts

  This option specifies that you want to display the SSL certificates of the remote server and the CA that signed it.
- -ssl2

  This option specifies that you want to connect using SSL Version 2 protocol.

Example 8-21 Low-level SSL validation using the openssl s_client

```
# openssl s_client -ssl2 -connect bs01b06.example.com:636 -CAFile
/usr/ldap/etc/cacert.pem -showcerts
CONNECTED(00000003)
depth=1 /C=us/ST=TX/L=Austin/O=example.com/CN=CA/emailAddress=ca@example.com
verify return:1
depth=0 /CN=bs01b06.example.com
verify return:1
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIDnzCCAwigAwIBAgIBBzANBgkqhkiG9w0BAQQFADBtMQswCQYDVQQGEwJ1czEL
MAkGA1UECBMCVFgxDzANBgNVBAcTBkF1c3RpbjEUMBIGA1UEChMLZXhhbXBsZS5j
b20xCzAJBgNVBAMTAkNBMR0wGwYJKoZIhvcNAQkBFg5jYUBleGFtcGxlLmNvbTAe
Fw0wNTA5MjExNjU2NDJaFw0wNjA5MjExNjU2NDJaMB4xHDAaBgNVBAMTE2JzMDFi
MDYuZXhhbXBsZS5jb20wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDq
nwZsBwamzyaE/qIeVChBvMtysmuEK4QnD8eWZGvOFEsctFmXfaj5/Lw146TkWDGJ
UkQ8dJbFnbcZDN4TntHCWS2S57ZBL+ygD9ckML6f75o+eLG1DF8mDfq+AfBB1d5u
cnz8/IVtVp52MnRo3B2DvR3u8RyJ1bWZeYeAvezdeZjiqPRc9sdLgvgpMn8La+JU
5mCPjjyhX+049m5o7wJxIRPGcC++IgGmktf0d2jZ4TY/o4FghcU0g4SRPky/tu2p
myTLs3X30hwuJCvaMGAYRznNk8wByOoANNAPlJ4eKM6oW51L36o8NyAZrU6yjH5A
yKuuuYD8Gf4L9PhmvfyNAgMBAAGjggEYMIIBFDAJBgNVHRMEAjAAMCwGCWCGSAGG
+EIBDQQfFh1PcGVuU1NMIEdlbmVyYXRlZCBDZXJ0aWZpY2F0ZTAdBgNVHQ4EFgQU
f16kZry6799UsEMZcIBC5cSnsS4wgZcGA1UdIwSBjzCBjIAUT/s0fmJWd9dsPQzH
```

```
+OACLX7fQrihcaRvMGOxCzAJBgNVBAYTAnVzMQswCQYDVQQIEwJUWDEPMAOGA1UE
BxMGQXVzdGluMRQwEgYDVQQKEwtleGFtcGxlLmNvbTELMAkGA1UEAxMCQ0ExHTAb
BgkqhkiG9wOBCQEWDmNhQGV4YW1wbGUuY29tggEAMBMGA1UdJQQMMAoGCCsGAQUF
BwMBMAsGA1UdDwQEAwIFoDANBgkqhkiG9wOBAQQFAAOBgQCzRmVqOqOA3WeQ//al
vsUiRYul/ZzZFMk4ZnIq/NNgB9w1BEcEXDPLeRkCY7Qm4CIg4GFxojjZ468VDy6D
+kAEE5TemLCCQcdQju6lj8BmlCLhgT/CD9V5v79yQcdpFhwj14nafu3H3HMnUNxX
r9OujGOspd2eqShWlfI9oEYnIA==
-----END CERTIFICATE-----
subject=/CN=bs01b06.example.com
issuer=/C=us/ST=TX/L=Austin/O=example.com/CN=CA/emailAddress=ca@example.com
---
No client certificate CA names sent
---
Ciphers common between both SSL endpoints:
RC4-MD5         EXP-RC4-MD5     RC2-CBC-MD5
EXP-RC2-CBC-MD5 DES-CBC-MD5     DES-CBC3-MD5
---
SSL handshake has read 1064 bytes and written 367 bytes
---
New, SSLv2, Cipher is DES-CBC3-MD5
Server public key is 2048 bit
SSL-Session:
    Protocol  : SSLv2
    Cipher    : DES-CBC3-MD5
    Session-ID: 7A230000C1C03D58BA5DD89C1D0C6DA5
    Session-ID-ctx:
    Master-Key: 2FE71F0B199339A2B5F239FC72966C5B94917371131A88F4
    Key-Arg   : 6842E42B41C3D5BF
    Start Time: 1127489298
    Timeout   : 300 (sec)
    Verify return code: 0 (ok)
---
#
```

Basic secure LDAP validation using the ldapsearch command

After you have confirmed that the SSL connection is working properly. You should then confirm that you are able to search your LDAP directory using LDAP on a secure port. This will confirm that the LDAP server can communicate with SSL and that your local AIX 5L certificate keystore was correctly created. Example 8-22 on page 257 shows the use of the **`ldapsearch`** command to test secure LDAP search. The following is a partial list of **`ldapsearch`** options used to perform the test:

- -D 'cn=host_bs01b08,cn=Users,dc=example,dc=com'

  This option specifies the DN used to bind to the Active Directory. By default, Windows Server 2003 has anonymous bind to the directory disabled. You can use the host principal for the AIX 5L client as the bind DN.

- -h bs01b06.example.com

  This option specifies the host name of the LDAP server to connect to.
- -K /usr/ldap/etc/bs01b08_keydb.kdb

  This option specifies the location of the local keystore where the example.com CA certificate was imported.
- -P 'passw0rd!'

  This option specifies the password needed to open the keystore.
- -p 636

  This option specifies the port number to connect to. The default port for secure LDAP is 636.
- -w 'passw0rd!'

  This option specifies the password to bind to the LDAP server with.
- -Z

  This option specifies that the **ldapsearch** command should use SSL to connect to the LDAP server.

Example 8-22 Testing LDAP over SSL using ldapsearch command

```
# ldapsearch  -b 'cn=Users,dc=Example,dc=Com' -h bs01b06.example.com -s base -Z
-p 636 -K /usr/ldap/etc/bs01b08_keydb.kdb  -P 'passw0rd!' -D
'cn=host_bs01b08,cn=Users,dc=example,dc=com' -w 'passw0rd!'  '(objectclass=*)'
objectClass=top
objectClass=container
cn=Users
description=Default container for upgraded user accounts
distinguishedName=CN=Users,DC=example,DC=com
instanceType=4
whenCreated=20050913201915.0Z
whenChanged=20050913201915.0Z
uSNCreated=4304
uSNChanged=4304
showInAdvancedViewOnly=FALSE
name=Users
objectGUID=NOT ASCII
systemFlags=-1946157056
objectCategory=CN=Container,CN=Schema,CN=Configuration,DC=example,DC=com
isCriticalSystemObject=TRUE
#
```

Full LDAP client validation using the lsldap command

After receiving satisfactory results using the **ldapsearch** command you should now test the AIX 5L LDAP client using the **lsldap** command. Refer back to "Testing LDAP client daemon using the lsldap command" on page 244.

8.2.5 Kerberos bind

Starting with AIX 5L Version 5.3, you can now configure the LDAP client daemon to bind to the Active Directory server using Kerberos. The Kerberos bind allows the LDAP client daemon to authenticate to the KDC instead of binding with a DN and password. Since all user and LDAP client daemon authentication is handled by Kerberos, no passwords are sent in clear text across the network. Depending on your data confidentiality policy, this might remove the need to deploy LDAP over SSL. If you are already using Kerberos and LDAP, there is no reason you should not use this feature, as you already have a host principal and keytab generated for each Kerberos client.

Attention: There is a known problem with the AIX 5L Version 5.3 LDAP client binding with Kerberos to an Active Directory server. IBM is aware of the problem and has released APAR IY79120 to resolve this problem. The full APAR description can be found at the following Web site:

http://www-1.ibm.com/support/docview.wss?rs=111&context=SWG10&q1=IY79120&uid=isg1IY79120&loc=en_US&cs=utf-8&lang=en

When the fix is publicly released, it can be downloaded at the following Web site:

http://www.ibm.com/servers/eserver/support/unixservers/aixfixes.html

In the meantime, if you want a subscription service to notify you by e-mail when the PTF for the APAR is available, you can do so at the following Web site:

http://www14.software.ibm.com/webapp/set2/aparsubscriptions/bjyqvcmjd

Example 8-23 shows the configuration to be added to the LDAP client daemon configuration file to enable Kerberos bind. The settings are as follows.

useKRB5	This setting makes the LDAP client daemon bind to the Active Directory using Kerberos. You will need to set this to yes to enable Kerberos bind.
krbprincipal	This setting specifies the Kerberos principal to authenticate as when accessing the Active Directory. This scenario uses the host principal for this value.

krbkeypath	This setting specifies the location of the keytab file for the krbprincipal.
krbcmddir	This setting specifies the location of the `kinit` command. The default value is correct and should not be modified.

Example 8-23 LDAP client daemon configuration for Kerberos bind

```
# Whether to use Kerberos for initial bind to the server.
# useKRB5       - valid value is either "yes" or "no". Default is "no".
# krbprincipal  - Kerberos principal used to bind to the server.
# krbkeypath    - Path to the kerberos keytab, default to
#                 /etc/security/ldap/krb5.keytab
# krbcmddir     - Directory that contains the Kerberos kinit command.
#                 Default to /usr/krb5/bin/.
useKRB5:yes
krbprincipal:host/bs01b08.example.com
krbkeypath:/etc/krb5/krb5.keytab
krbcmddir:/usr/krb5/bin/
```

For more information about solving problems see 4.4.2, “Kerberos bind to LDAP server” on page 137.

8.3 Scenario: LDAP only

The *LDAP only* scenario is very similar to the *LDAP and Kerberos* scenario in the previous section, except now user authentication is done using LDAP and not Kerberos. The update and group information is still stored in Active Directory (AD).

Since this scenario is nearly identical to the LDAP and Kerberos scenario, this section will refer you to the previous section for the common tasks. For a complete description of LDAP in the MSAD scenario, please refer to the scenario description in 8.2, “Scenario: Kerberos and LDAP” on page 233.

When using the AIX 5L LDAP client to authenticate against an LDAP server, you have two different authentication types. The first is called client-side or UNIX authentication, and the second server-side or LDAP authentication.

Client-side authentication is when the LDAP client encrypts the user’s password locally and compares it against the encrypted password from the LDAP server. The UNIX authentication method is available on AIX 4.3.3 and later. Server-side authentication is when the LDAP client binds to the LDAP server using the user’s distinguished name (DN), which is derived from the user’s name, and the entered password. The user’s password is sent across the network to the LDAP server in clear text. The LDAP authentication method is only available on AIX 5L Version 5.3.

For more information about the different authentication types please refer to 3.5.3, “Client-side and server-side authentication” on page 86.

The first part of the scenario will use the UNIX authentication method, as this is the default authentication type, and the second part will use the LDAP authentication method. For more information describing the advantages and disadvantages of each authentication type refer to 3.5.3, “Client-side and server-side authentication” on page 86.

8.3.1 Windows 2003 Active Directory configuration

This section discusses the steps to configure the Microsoft Windows 2003 Active Directory to support UNIX integration. These steps are similar to the Kerberos and LDAP scenario. Refer to 8.2.1, “Microsoft Windows 2003 Active Directory configuration” on page 234.

- Install the Active Directory Schema MMC Snap-in by using the instructions in “Installing the Active Directory Schema MMC Snap-in” on page 234.
- Install the Microsoft Windows Services for UNIX 3.5 by using the instructions in “Install Microsoft Windows Services for UNIX (SFU)” on page 236.
- Create a user and groups in the Active Directory domain with UNIX attributes using the instructions in “Create a user account and groups with UNIX attributes” on page 260.
- Synchronize group membership attributes msSFU30MemberUid and msSFU30PosixMember using the instructions in “Synchronizing group membership attributes” on page 239.

Create a user account and groups with UNIX attributes

This section discusses the creation of the user and groups needed for this scenario. This scenario uses one user, named test03, and two different groups named testgrp2 and testgrp3. The user would have the following entry in the /etc/passwd file:

```
test03:!:10001:10002:test03:/home/test03:/usr/bin/ksh
```

The two groups would have the following entries in the /etc/group file:

```
testgrp2:!:10001:test03
testgrp3:!:10002:test03
```

Steps to create groups with UNIX attributes

Here we go through the steps required to create the group and assign the UNIX attributes for the testgrp2 group. Repeat this list to create group testgrp3:

1. Open the New Object - Group dialog by selecting the **example.com**/**Users** container, and then right-click and select **New** → **Group**.
2. Enter `testgrp2` into the Group name field. The default values **Global** and **Security** for Group scope and Group type should be selected. Select **OK** to create the group.
3. In order to add the UNIX attributes to your new group, open the testgrp2 Properties dialog by selecting the **testgrp2** entry, and then right-click and select **Properties**. Then select the **UNIX Attributes** tab.
4. Select **example** for the NIS Domain menu field. The next available UNIX Group ID will be chosen. Select **OK** to accept the default GID and save changes.

Steps to create users with UNIX attributes

Here we go through the steps required to create the user, assign the UNIX attributes and synchronize the windows and UNIX password for the user test03:

1. Open the New Object - User dialog by selecting the **example.com**/**Users** container, and then right-click and select **New** → **User**.
2. Enter `test03` in the First name and User logon name fields. The Full name and User logon name (pre-Windows 2000) fields are completed automatically. Select **Next** to continue.
3. You must now create a strong initial password for the user and enter it into the Password and Confirm Password fields. You should also uncheck the **User must change password at next logon** option. Select **Next** to continue. Select **Finish** to create the account.
4. In order to add the UNIX attributes to your new user, open the test03 Properties dialog by selecting the **test02** entry, and then right-click and select **Properties**. Then select the **UNIX Attributes** tab.
5. Select **example** for the NIS Domain menu field. The next available UNIX user ID will be chosen. Enter `/usr/bin/ksh` for the login shell. Select **testgrp3** in the Primary group name/GID menu. Select **OK** to accept the default UID and save the changes.
6. You must now synchronize the user's password with the SFU password attribute, msSFUPassword or msSFU30Password, by selecting the **test03** entry, and then right-click and select **Reset Password**.
7. You must now enter a strong password for the user and enter it into the New Password and Confirm Password fields. You should confirm that the **User**

must change password at next logon option is unchecked. Select **OK** to continue.

Steps to add user as a member of a UNIX group

This list gives the steps required to add the user test03 as a member of the testgrp2 group.

1. In order to add UNIX users to additional groups, open the testgrp2 Properties dialog by selecting the **testgrp2** entry, and then right-click and select **Properties**. Then select the **UNIX Attributes** tab and then the **Add** button.
2. Select the user **test03**, and then select the **Add** button. You can then select **Add** again to add additional group members. Select **OK**, then **OK** again to accept the additional group members.

8.3.2 AIX 5L LDAP client installation

The installation steps are:

1. Install the AIX 5L LDAP client using the instructions in 4.2.1, "AIX 5L LDAP client software installation" on page 105.
2. Define LDAP authentication methods in the methods.cfg file using instructions in "Define required authentication methods in methods.cfg" on page 262.
3. Unlike the Microsoft Windows 2000 Active Directory, Windows 2003 disables anonymous bind to the Active Directory. Create a host account to be used to bind to the Active Directory server using the instructions in "Create host account used to bind to Active Directory" on page 262.

Define required authentication methods in methods.cfg

You must now modify the /usr/lib/security/methods.cfg file to add the LDAP stanza. The LDAP stanza defines a simple load module for authentication and storage of user information (uid, primary groups, shell, home directory, and so on). Add the stanza shown in Example 8-24 to the /usr/lib/security/methods.cfg file.

Example 8-24 LDAP stanza in methods.cfg

```
LDAP:
        program = /usr/lib/security/LDAP
        program_64 = /usr/lib/security/LDAP64
```

Create host account used to bind to Active Directory

This list goes through the steps required to create the host account used by the AIX 5L LDAP client daemon to bind to the Active Directory. Unlike normal UNIX users, this user does not need to have UNIX attributes. The host account will be named host_bs01b08:

1. Open the New Object - User dialog by selecting the **example.com/Users** container, and then right-click and select **New** → **User**.
2. Enter `host_bs01b08` in the First name and User logon name fields. The Full name and User logon name (pre-Windows 2000) fields are completed automatically. Select **Next** to continue
3. You must now create a strong initial password for the user and enter it into the Password and Confirm Password fields. You should also uncheck the **User must change password at next logon** option. Select **Next** to continue. Select **Finish** to create the account.

8.3.3 AIX 5L LDAP client configuration

The following section discusses the steps to configure the AIX 5L LDAP client to use the UNIX authentication method. Since UNIX authentication is the default, configure the AIX 5L LDAP client using all the instructions in 8.2.3, "AIX 5L LDAP client configuration" on page 241, except the "Testing user logon for the Kerberos and LDAP scenario" on page 246. The basic configuration steps are:

1. Configure the AIX 5L LDAP client daemon.
2. Create object class and attribute mapping files for SFU schema.
3. Start and test the AIX 5L LDAP client daemon.
4. Confirm that the UNIX crypt password is set.
5. Test the user logon in the LDAP UNIX authentication scenario.

Confirming UNIX crypt password is set

In "Steps to create users with UNIX attributes" on page 261 you had to synchronize the password with the SFU password attribute msSFU30Password. This section discussed how to confirm that the password was synchronized correctly.

Note: The SFU attribute msSFU30Password, which contains the user's password in UNIX crypt format, is only necessary if you are using client-side authentication with LDAP. When using Kerberos or server-side authentication, the unicodePwd is used to authenticate users.

When a user account is added to a Windows domain, the users's password is stored in the unicodePwd attribute, in a Unicode format. When UNIX attributes are added to the user, the msSFU30Password containing the user's password is given the default value of ABCD!efgh12345$67890. The user or the administrator must change the password again and Windows will synchronize the unicodePwd and msSFU30Password passwords in their respective formats.

This was the reason for resetting the password for the second time when creating the user accounts above.

Example 8-25 shows the use of the **lsldap** command to display the msSFU30Password attributes of two different users, test02 and test03. The msSFU30Password attribute for user test02 is still set to the default ABCD!efgh12345$67890 and will not allow that user to log on. The msSFU30Password attribute for user test03 has been synchronized properly and that user should be able to log on.

Example 8-25 Confirming UNIX crypt password exists in MSAD with lsldap

```
# lsldap -a passwd test02 | grep msSFU30Password
msSFU30Password: ABCD!efgh12345$67890
#
# lsldap -a passwd test03 | grep msSFU30Password
msSFU30Password: jSngJNOwqZyOA
```

Testing user logon for the LDAP-only scenario

Now that the LDAP client daemon is configured, you should be able to log on to the AIX 5L machine. Example 8-26 shows a sample logon session and how to verify the user's environment. The AUTHSTATE environment variable should be set to LDAP. Use the **id** command to check the user's primary group and additional group memberships.

Example 8-26 Testing user login for the LDAP-only scenario

```
# telnet bs01b08.example.com
Trying...
Connected to bs01b08.example.com.
Escape character is '^]'.

 telnet (bs01b08)

AIX Version 5
(C) Copyrights by IBM and by others 1982, 2005.
login: test03
test03's Password:

*******************************************************************************
*                                                                             *
*                                                                             *
*  Welcome to AIX Version 5.3!                                                *
*                                                                             *
*                                                                             *
*  Please see the README file in /usr/lpp/bos for information pertinent to    *
*  this release of the AIX Operating System.                                  *
*                                                                             *
*                                                                              *
********************************************************************************

$ echo $AUTHSTATE
LDAP
$ id
uid=10001(test03) gid=10002(testgrp3) groups=10001(testgrp2)
$
```

8.3.4 Configuring LDAP to use SSL

The steps required to configure SSL on the client side and server side can be found in 8.2.4, "Configuring LDAP to use SSL" on page 247.

8.3.5 Configuring server-side authentication

The first part of the scenario used client-side authentication to authenticate against the Active Directory. This section discusses how to configure the AIX 5L Version 5.3 LDAP client to use server-side authentication when authenticating users.

Attention: When using server-side authentication, the user's password is sent in clear text to the LDAP server. We highly recommend using SSL to encrypt the LDAP connection from client to server.

In order to switch from client-side to server-side authentication, you must set the authtype setting to ldap_auth in the /etc/security/ldap/ldap.cfg file.

Example 8-27 shows the authentication section of the LDAP client daemon configuration file, with the authtype setting set to ldap_auth.

Example 8-27 AIX 5L LDAP client authentication type section from ldap.cfg

```
# Authentication type
# ldap_auth - Bind to LDAP server to authenticate user remotely through LDAP.
authtype:ldap_auth
```

After the LDAP client daemon's configuration file is changed, you must restart the daemon using the **restart-secldapclntd** command, as shown in Example 8-28.

Example 8-28 Restarting the secldapclntd daemon to reread the configuration file

```
# restart-secldapclntd
The secldapclntd daemon terminated successfully.
Starting the secldapclntd daemon.
The secldapclntd daemon started successfully.
#
```

8.3.6 Object class and attribute mapping files for SFU

This section contains the mapping files to allow the AIX 5L LDAP client to map the Microsoft SFU schemas. The mapping files for SFU 2.0 are here for your reference and have not been tested in this scenario.

Example 8-29 AIX 5L user entity mapping file for Microsoft SFU 2.0

```
# ========================================================================
#
# MSSFU20user.map
# AIX 5L -> Microsoft Windows Services for UNIX 2.0
#
#      LDAP user attribute name mapping table
#
# Format:
# AIX_ATTR   AIX_ATTR_TYPE   LDAP_ATTR   LDAP_VALUE
#
# AIX_ATTR:     AIX attribute name
# AIX_ATTR_TYPE AIX attribute type - SEC_CHAR, SEC_INT, SEC_LIST, SEC_BOOL
# LDAP_ATTR     LDAP attribute name
# LDAP_VALUE    LDAP attribute type - "s" for single-valued attribues
#
# ========================================================================

# The following entry contains the object class used in the filter
# for queries on user data.  All users should have this class.
keyobjectclass  SEC_CHAR        User                    s

# The following attributes are required by AIX to be functional
username        SEC_CHAR        msSFUName               s
id              SEC_INT         UidNumber               s
pgrp            SEC_CHAR        GidNumber               s
home            SEC_CHAR        msSFUHomeDirectory      s
shell           SEC_CHAR        msSFULoginShell         s
gecos           SEC_CHAR        displayName             s
spassword       SEC_CHAR        msSFUPassword           s
```

```
lastupdate      SEC_INT         ShadowLastChange        s

# The following attributes are optional
maxage          SEC_INT         ShadowMax               s
minage          SEC_INT         ShadowMin               s
maxexpired      SEC_INT         ShadowExpire            s
pwdwarntime     SEC_INT         ShadowWarning           s
```

Example 8-30 AIX 5L group entity mapping file for Microsoft SFU 2.0

```
# ===========================================================================
#
# MSSFU20group.map
# AIX -> Microsoft Windows Services for UNIX 2.0
#
#       LDAP group attribute name mapping table
#
# Format:
# AIX_ATTR   AIX_ATTR_TYPE   LDAP_ATTR   LDAP_VALUE
#
# AIX_ATTR:     AIX attribute name
# AIX_ATTR_TYPE AIX attribute type - SEC_CHAR, SEC_INT, SEC_LIST, SEC_BOOL
# LDAP_ATTR     LDAP attribute name
# LDAP_VALUE    LDAP attribute type - "s" for single-valued attribues
#               or "m" for multi-valued attributes.
#
# NOTE:
# In case the client needs to talk to a LDAP server with different
# schema (attributes) mapping than the following, modify the corresponding
# LDAP attributes to match these defined in the new LDAP server, and
# comment out the lines where the attribute(s) is not defined.
#
# Save a copy of this file before you modify entries in this file.
#
# ===========================================================================

# The following entry contains the object class used in the filter
# for queries on group data.  All groups should have this class.
keyobjectclass  SEC_CHAR        Group                   s

# The following attributes are required by AIX to be functional
groupname       SEC_CHAR        msSFUName               s
id              SEC_INT         GidNumber               s
users           SEC_LIST        MemberUid               m
```

Example 8-31   AIX 5L user entity mapping file for Microsoft SFU 3.0/3.5

```
# ==========================================================================
#
# MSSFU30user.map
# AIX 5L -> Microsoft Windows Services for UNIX 3.0/3.5
#
#       LDAP user attribute name mapping table
#
# Format:
# AIX_ATTR   AIX_ATTR_TYPE   LDAP_ATTR   LDAP_VALUE
#
# AIX_ATTR:     AIX attribute name
# AIX_ATTR_TYPE AIX attribute type - SEC_CHAR, SEC_INT, SEC_LIST, SEC_BOOL
# LDAP_ATTR     LDAP attribute name
# LDAP_VALUE    LDAP attribute type - "s" for single-valued attribues
#
# ==========================================================================

# The following entry contains the object class used in the filter
# for queries on user data.  All users should have this class.
keyobjectclass  SEC_CHAR        User                    s

# The following attributes are required by AIX to be functional
username        SEC_CHAR        msSFU30Name             s
id              SEC_INT         msSFU30UidNumber        s
pgrp            SEC_CHAR        msSFU30GidNumber        s
home            SEC_CHAR        msSFU30HomeDirectory    s
shell           SEC_CHAR        msSFU30LoginShell       s
gecos           SEC_CHAR        displayName             s
spassword       SEC_CHAR        msSFU30Password         s
lastupdate      SEC_INT         msSFU30ShadowLastChange s

# The following attributes are optional
maxage          SEC_INT         msSFU30ShadowMax        s
minage          SEC_INT         msSFU30ShadowMin        s
maxexpired      SEC_INT         msSFU30ShadowExpire     s
pwdwarntime     SEC_INT         msSFU30ShadowWarning    s
```

Example 8-32   AIX 5L group entity mapping file for Microsoft SFU 2.0

```
# ==========================================================================
#
# MSSFU30group.map
# AIX 5L -> Microsoft Windows Services for UNIX 3.0/3.5
#
#       LDAP group attribute name mapping table
#
# Format:
```

```
# AIX_ATTR   AIX_ATTR_TYPE   LDAP_ATTR   LDAP_VALUE
#
# AIX_ATTR:     AIX attribute name
# AIX_ATTR_TYPE AIX attribute type - SEC_CHAR, SEC_INT, SEC_LIST, SEC_BOOL
# LDAP_ATTR     LDAP attribute name
# LDAP_VALUE    LDAP attribute type - "s" for single-valued attribues
#               or "m" for multi-valued attributes.
#
# NOTE:
# In case the client needs to talk to a LDAP server with different
# schema (attributes) mapping than the following, modify the corresponding
# LDAP attributes to match these defined in the new LDAP server, and
# comment out the lines where the attribute(s) is not defined.
#
# Save a copy of this file before you modify entries in this file.
#
# ==========================================================================

# The following entry contains the object class used in the filter
# for queries on group data.  All groups should have this class.
keyobjectclass  SEC_CHAR        Group                   s

# The following attributes are required by AIX to be functional
groupname       SEC_CHAR        msSFU30Name             s
id              SEC_INT         msSFU30GidNumber        s
users           SEC_LIST        msSFU30MemberUid        m
```

8.4 Troubleshooting

For help troubleshooting the Active Directory, see:

- Microsoft Support: How to configure Active Directory diagnostic event logging in Windows Server

  http://support.microsoft.com/default.aspx?scid=kb;en-us;314980&sd=tech

- Microsoft Support: Description of the requirements and of the troubleshooting methods that you can use to enable an LDAP client to communicate with an LDAP server over SSL

  http://support.microsoft.com/default.aspx?scid=kb;en-us;290483

For help troubleshooting Kerberos, see:

- Microsoft TechNet: Kerberos Authentication Tools and Settings

  http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/TechRef/b36b8071-3cc5-46fa-be13-280aa43f2fd2.mspx

- Microsoft TechNet: Troubleshooting Kerberos Error

  http://www.microsoft.com/technet/prodtechnol/windowsserver2003/technologies/security/tkerberr.mspx

Part 3

Appendixes

A

IBM Tivoli Directory Server V6.0 installation

This chapter describes the installation of IBM Tivoli Directory Server V6.0 (ITDS) for the use as an authentication back-end. The installation will be done with ITDS Version 6.0. The installation would have the same structure for ITDS5.2.

The ITDS V6.0 product is shipped with either a native installer or with the Install Shield Multi-platform (ISMP) installer. This scenario use installs ITDS V6.0 using the native installer.

Preparing the system for installation

Before the system can be installed, it must be ensured that the system setup is correctly. Check that the system is able to run in 64-bit mode and that it is running in 64-bit mode (Example A-1).

Example: A-1 Using the lsconf command to check for 64-bit kernel

```
# lsconf -ck
CPU Type: 64-bit
Kernel Type: 64-bit
#
```

The system is running in 64-bit mode if CPU type and kernel type are 64-bit.

Also the system must have asynchronous I/O (AIO) switched on. Example A-2 shows the use of the **lsattr** command to determine the status of AIO.

Example: A-2 Check whether AIO is available

```
# lsattr -El aio0
autoconfig defined STATE to be configured at system restart True
fastpath   enable  State of fast path                        True
kprocprio  39      Server PRIORITY                           True
maxreqs    4096    Maximum number of REQUESTS                True
maxservers 10      MAXIMUM number of servers per cpu         True
minservers 1       MINIMUM number of servers                 True
#
```

Example A-4 on page 275 shows how to use the **chdev** command to enable AIO and to make it available after reboot.

Example: A-3 Use the chdev command to enable AIO

```
# chdev -l aio0 -P -a autoconfig=available
aio0 changed
#
# lsattr -El aio0
autoconfig available STATE to be configured at system restart True
fastpath   enable    State of fast path                        True
kprocprio  39        Server PRIORITY                           True
maxreqs    4096      Maximum number of REQUESTS                True
maxservers 10        MAXIMUM number of servers per cpu         True
minservers 1         MINIMUM number of servers                 True
#
```

The ITDS will be installed via the **mksecldap** command. For most system administrators it is convenient to install the data on a separate partition, volume group, or both. The **mksecldap** command will create a DB2 instance ldapdb2. With a home directory at /home/ldapdb2. To separate the data just create a file system at this directory. See Example A-4.

Also, the DB2 UDB packages consume a lot of space. They should be separated at another partition. The directory is /usr/opt/db2_08_01 and it needs approximately 500 MB.

Example: A-4 File system for db2 and the ldapdb2 instance

```
# lsfs /dev/db2_ldap_lv  /dev/db2_install
Name            Nodename   Mount Pt               VFS   Size    Options    Auto
/dev/db2_ldap_lv --         /home/ldapdb2          jfs2  1310720 rw         yes
/dev/db2_install --         /usr/opt/db2_08_01     jfs2  1048576 rw         yes
#
```

Attention: Ensure that you have mounted the newly created directories before continuing.

Installing DB2

The DB2 UDB file sets are available with the ITDS installation package. To install DB2 for use with the ITDS Server just go to the directory where you have your installation packages (untared if delivered as tar.gz or the mounted cd). This should be named itdsV60.

For the DB2 installation just switch to the DB2 directory and use the **db2_install** command. It will ask you what type of server should be installed. Use the DB2 UDB Enterprise Server Edition.

Example: A-5 db2install

```
root@bs01b02:/mnt/ITDSv60/itdsV60/db2 ] # ./db2_install


Specify one or more of the following keywords,
separated by spaces, to install DB2 products.
 Keyword        Product Description
 DB2.ESE        DB2 Enterprise Server Edition for AIX

Enter "help" to redisplay product names.
Enter "quit" to exit.
***********************************************************
```

```
DB2.ESE
....
....
db2_install program completed successfully.
```

If needed, a DB2 UDB FixPak can be applied. The list of supported FixPaks in use with ITDS can be found here:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.IBMDS.doc/install06.htm

Installing the ITDS file sets

To install the ITDS6.0 server completely the file sets listed in Table A-1 have to be installed. This can be done via **`smit install`** or via the **`installp`** command. The result is shown with the **`lslpp`** command (Example A-6 on page 277).

Table A-1 ITDS V6.0 file sets to install

File set	Description
idsldap.cltbase60.rte	Base client runtime
idsldap.cltbase60.adt	Base client SDK
idsldap.clt64bit60.rte	64-bit client (no SSL)
idsldap.clt_max_crypto64bit60.rte	64-bit client (SSL)
idsldap.cltjava60.rte	Java client (no SSL)
idsldap.cltjava_max_crypto60.rte	Java client (SSL)
idsldap.srvproxy64bit60	Proxy server 64-bit (no SSL)
idsldap.srv_max_cryptoproxy64bit60	Proxy server (64-bit) (SSL)
idsldap.srv64bit60.rte	Full server (no SSL) (64-bit)
idsldap.msg60.en_US	English messages
gsksa.rte	AIX Certificate and SSL Base Runtime ACME Toolkit
gskta.rte	

Note: All file sets to use on AIX 5L should be the 64-bit ones. The 32-bit file sets are not supported on every platform.

Example: A-6 Installed file sets for ITDS V6.0

```
# lslpp -L idsldap* gsk*
  Fileset                      Level  State  Type  Description (Uninstaller)
  ----------------------------------------------------------------------------
  gsksa.rte                  7.0.3.3    C     F    AIX Certificate and SSL Base
                                                   Runtime ACME Toolkit
  gskta.rte                  7.0.3.3    C     F    AIX Certificate and SSL Base
                                                   Runtime ACME Toolkit
  idsldap.clt64bit60.rte     6.0.0.0    C     F    Directory Server - 64 bit
                                                   Client
  idsldap.clt_max_crypto64bit60.rte
                             6.0.0.0    C     F    Directory Server - 64 bit
                                                   Client (SSL)
  idsldap.cltbase60.adt      6.0.0.0    C     F    Directory Server - Base Client
  idsldap.cltbase60.rte      6.0.0.0    C     F    Directory Server - Base Client
  idsldap.cltjava60.rte      6.0.0.0    C     F    Directory Server - Java Client
  idsldap.cltjava_max_crypto60.rte
                             6.0.0.0    C     F    Directory Server - Java Client
                                                   (SSL)
  idsldap.msg60.en_US        6.0.0.0    C     F    Directory Server - Messages -
                                                   U.S. English (en)
  idsldap.srv64bit60.rte     6.0.0.0    C     F    Directory Server - 64 bit
                                                   Server
  idsldap.srv_max_cryptoproxy64bit60.rte
                             6.0.0.0    C     F    Directory Server - Proxy
                                                   Server (SSL)
  idsldap.srvproxy64bit60.rte
                             6.0.0.0    C     F    Directory Server - Proxy
                                                   Server
```

Configuring ITDS using the mksecldap command

The **mksecldap** command will do all of the configuration on the LDAP server. As base DN, “dc=example,dc=com” is used. The schema to be applied to the directory is RFC2307AIX.

During the **mksecldap** command the local users and groups can be imported to the directory server, which is the default.

```
mksecldap -s -a "cn=admin" -p its0g00d -d "dc=example,dc=com" -x
"cn=proxy,ou=People,dc=example,dc=com" -X its0g00d -S rfc2307aix
```

To not import the local users the -u NONE option must be specified:

```
mksecldap -s -a "cn=admin" -p its0g00d -d "dc=example,dc=com" -x
"cn=proxy,ou=People,dc=example,dc=com" -X its0g00d -S rfc2307aix -u NONE
```

Attention: If the local security database is not imported to the LDAP server, the containers for ou=People and ou=Group will not be created. An LDAP client can only be created with **mksecldap** if there are users and groups in the containers.

During the installation the password for the ldapdb2 user and an encryption seed must be specified.

Example: A-7 mksecldap command with users

```
# mksecldap -s -a "cn=admin" -p its0g00d -d "dc=example,dc=com" -x
"cn=proxy,ou=People,dc=example,dc=com" -X its0g00d -S rfc2307aix>
ldapdb2's New password:
Enter the new password again:
Enter an encryption seed to generate key stash files:
...
...
...
LPSRV009I IBM Tivoli Directory (SSL), 6.0      Server started.
migrating users/groups to LDAP server.
#
```

Note: The proxy user ID must be specified with an existing suffix. During the installation only the “ou=People”+custom suffix exists. Which means for our example that during the installation only “ou=People,dc=example,dc=com” is valid. If it is necessary to create the proxy user ID at another suffix, or create it more than once, it must be done manually. There is an example template at /etc/security/ldap/proxy.ldif.template.

You should then use the **ldapsearch** command to verify the directory installed successfully. Example A-8 shows the use of the **ldapsearch** command to test the newly installed directory server.

Example: A-8 Verifying the installation

```
# ldapsearch -b "" -s base "objectclass=*" namingcontexts
namingcontexts=CN=SCHEMA
namingcontexts=CN=LOCALHOST
namingcontexts=CN=PWDPOLICY
namingcontexts=CN=IBMPOLICIES
namingcontexts=DC=EXAMPLE,DC=COM
#
# ldapsearch -b "dc=example,dc=com"  -D "cn=admin" -w its0g00d  uid=test1
uid=test1,ou=People,dc=example,dc=com
uid=test1
objectClass=aixauxaccount
```

```
objectClass=shadowaccount
objectClass=posixaccount
objectClass=account
objectClass=ibm-securityidentities
objectClass=top
cn=test1
passwordchar=*
uidnumber=204
gidnumber=1
homedirectory=/home/test1
loginshell=/usr/bin/ksh
isadministrator=false
#
# ldapsearch -b "dc=example,dc=com" -D "cn=admin" -w its0g00d cn=proxy
cn=proxy,ou=People,dc=example,dc=com
cn=proxy
sn=proxy
objectclass=person
objectclass=top
userpassword={crypt}Tnq4gXhZqliMo
#
```

After the **mksecldap** command has successfully created the ITDS Server, the DB2 must be configured to start the database automatically.

Starting and stopping ITDS on reboot

The ITDS will start with the current configuration, but the DB2 will not. The ITDS will be able to serve his own config information, but no real data. Therefore the DB2 must be started before the ITDS.

We want to shut down the database in a consistent way, so it is necessary to stop the ITDS Server before and stop the database afterwards.

Starting DB2

There are different ways to start the DB2. One way is to utilize the fault monitor. The way that we show here is to write a little script to start the DB2 instance from the inittab file. This way is used here, because we create the inittab entry in a way that it will be executed completely before the rest of the inittab will be executed. That way we ensure the way that DB2 is starting before the ITDS.

1. The script is just one line. We store it at /etc/rc.db2:

   ```
   #!/bin/sh
   /usr/bin/su - ldapdb2 -c /home/ldapdb2/sqllib/adm/db2start
   ```

2. Make the script executable:

   ```
   chmod 755 /etc/rc.db2
   ```

3. Create the inittab entry:

   ```
   mkitab -i ids0 "db2:2:wait:/etc/rc.db2 >/dev/null 2>&1"
   ```

Now it is time to test the start script.

Stopping DB2 and ITDS

To stop the DB2 and ITDS we need to execute a script during a shutdown. For this purpose the /etc/rc.shutdown can be used. Be aware that this script will only be executed when a shutdown command is used.

The script in Example A-9 will be placed in /etc/rc.shutdown. The script should be made executable:

```
chmod 755 /etc/rc.shutdown
```

Example: A-9 Shutdown script for ITDS

```
#!/bin/sh
# shutdown the ITDS instance
/usr/bin/idsslapd -I ldapdb2 -k
# shutdown the directory admin deamon
/usr/bin/idsdiradm -I ldapdb2 -k

#stopping the db2 instance
/usr/bin/su -  ldapdb2 -c /home/ldapdb2/sqllib/adm/db2stop

exit 0
```

It is time to test both scripts now.

Generating a key database to use for SSL with ITDS 6.0

The GSKit should already be installed. A key database is a file that stores the certificates and the private key to use with ITDS. The key database can be created with the IKeyman GUI utility or with the command-line version called **ikeycmd**.

1. Create a key database.
2. Import the certificate authority (CA) certificate into the key database.
3. Create a key and a certificate signing request (CSR).
4. Have CSR signed by the example.com certificate authority.
5. Import the signed certificate into the key database.

For a more detailed explanation of how to work with GSKit please see 4.3.1, “Configuring SSL” on page 113. The steps described there for client certificates are the same as for the server certificate here.

The following example uses the **gsk7cmd** command to create the key database, add a trusted CA certificate to the key database, and then create a certificate request for the directory server.

Example: A-10 Step 1–3: Creating a key ring for ITDS

```
# gsk7cmd -keydb -create -db ./key.kdb  -pw its0g00d -type cms -stash
# gsk7cmd -cert -add -db ./key.kdb -label example.com -format binary -pw
its0g00d -trust enable -file  /home/root/cacert.pem
# gsk7cmd -certreq -create -db ./key.kdb -pw its0g00d -label bs01b02 -dn
"CN=bs01b02,O=example.com,L=Austin,ST=TX, C=us" -file bs01b02.req
```

Attention: The certificate that will be received from the CA should be in a file without any other text comment or information.

After the server certificate is signed by the CA, you must use the **gsk7cmd** command to import the signed certificated into the key database. The following example shows using the **gsk7cmd** command to receive the signed server certificate. The second **gsk7cmd** command will list the details of the signed certificate.

Example: A-11 Step 5 and showing the certificate

```
# /home/ldapdb2/idsslapd-ldapdb2/etc
# gsk7cmd -cert -receive -file bs01b02.cert.pem  -db ./key.kdb -pw its0g00d
-format ascii -default_cert yes
# gsk7cmd -cert -details   -db ./key.kdb -pw its0g00d -label bs01b02

Label: bs01b02
Key Size: 1024
Version: X509 V3
Serial Number: 03
Issued By: CA
example.com
Austin, TX, us
Subject: bs01b02
example.com
Austin, TX, us
Valid From: Wednesday, September 14, 2005 7:13:46 PM CDT To: Thursday,
September 14, 2006 7:13:46 PM CDT
Fingerprint: F6:B9:8A:AA:6E:DA:0F:7A:4E:27:E1:0C:78:4D:FC:FC:46:15:76:5F
Signature Algorithm: 1.2.840.113549.1.1.4
Trust Status: enabled
```

Enabling SSL

To enable SSL the following values must be specified in the ITDS configuration file:

- ibm-slapdSslKeyDatabase: /home/ldapdb2/key.kdb

  This defines the location of the key ring file with the CA certificate, the certificate for the LDAP server, and the private key of the LDAP server.

- ibm-slapdSecurePort: 636

  This specifies the port on which the server listens for SSL connections.

- ibm-slapdSecurity: SSL

  This specifies the modes of SSL security. The usable values are none, SSL, and SSLonly. None means that no SSL is used at all. SSLonly means that only SSL is used on that server. SSL means that SSL and none SSL connections can be established.

- ibm-slapdSslAuth: serverauth

  This value specifies whether server-side authentication or server-side and client-side authentication should be used. Possible values are serverauth and serverClientAuth. Serverauth means that only the server has to have a valid certificate. ServerClientAuth means that the clients needs a valid certificate as well.

After the modifications the server and the administration server have to be restarted.

B

Microsoft Windows 2003 Active Directory configuration

This appendix discusses the installation and configuration of the Microsoft Windows 2003 Active Directory (MSAD) environment used to test the scenarios covered in Chapter 8, “Scenario: Microsoft Windows 2003 Active Directory integration” on page 217, of this book. Properly designing and deploying Microsoft Active Directory is a involved process that considers corporate structure, security policies, organizational boundaries, existing IT infrastructure, and many other factors. Designing an Active Directory domain structure for your enterprise is beyond the scope of this book.

MSAD scenario test environment

In order to simplify the MSAD scenario test environment, the simplest of domain structures, the single domain model was chosen. With the single domain model, all objects are within a single security boundary and there are no trust relationships with other domains. This simplicity reduces the complexity and administrative overhead of the test environment.

The Active Directory (AD) integration scenarios were written about a fictitious company called example.com. example.com migrated all their existing Windows clients to Windows 2003 Active Directory servers and now want to integrate their AIX 5L client machines as well. Their goal is to allow centralized user and group management for all of their platforms.

Example.com's AD domain configuration is as follows:

- The example.com company chose a single domain.
- The common domain name service (DNS) name used by their AD is example.com.
- Their domain controllers also double as DNS servers for the example.com zone.

Active Directory configuration

Following are the steps used to install the Active Directory server used in the test environment:

1. Install Microsoft Windows 2003 Server and all of the latest service packs and fixes.
2. In order to prevent malicious users from replaying network traffic to authenticate to the KDC as legitimate users, the Kerberos clients' and servers' system clock's must be synchronized. The Kerberos server automatically discards all authentication requests over the allowable clock skew. By default, the maximum clock skew allowed by Kerberos is 300 seconds. One method of maintaining minimal clock skew is by using the standard Network Time Protocol (NTP). Example B-1 shows how to use the **w32tm** command to synchronize the domain controller with the NTP server timeserver1.example.com and allow the domain controller to act as a timeserver for other clients.

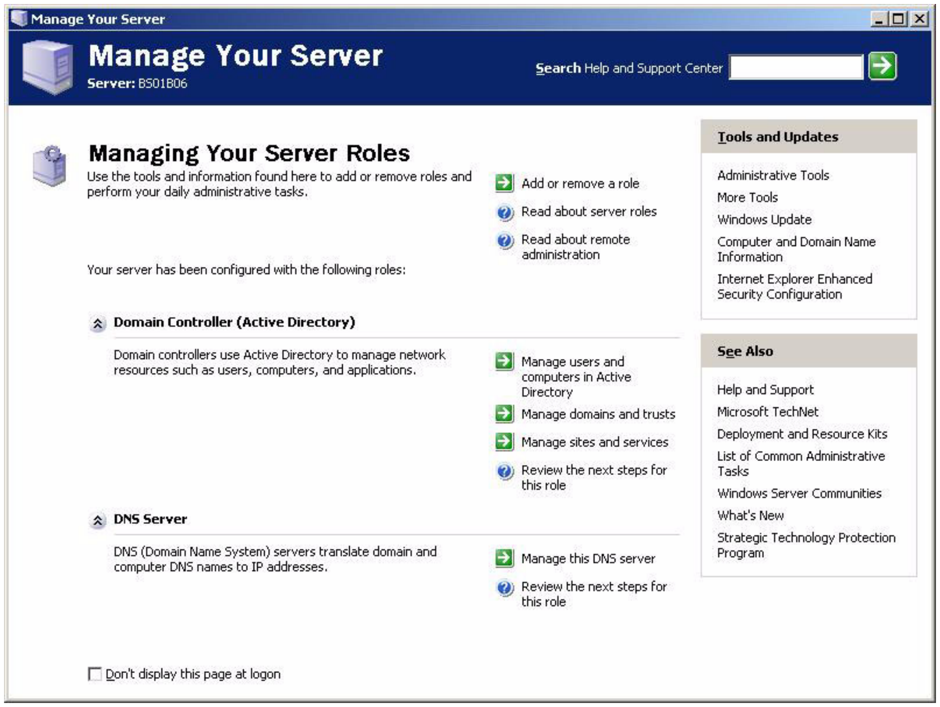
Example: B-1 Synchronizing Windows server using NTP protocol

```
C:\>w32tm /config /update /manualpeerlist:timeserver1.example.com
/syncfromflags:MANUAL /reliable:YES
The command completed successfully.
```

```
C:\>>w32tm /monitor
bs01b06.example.com *** PDC *** [9.3.5.143]:
    ICMP: 0ms delay.
    NTP: +0.0000000s offset from bs01b06.example.com
        RefID: timeserver1.example.com [9.10.225.159]
C:\>
```

3. Start the Active Directory Installation Wizard by selecting **Start** → **Run**, and type `dcpromo`, and select **OK** to continue.
4. Select the **Next** button to continue.
5. Review the operating system compatibility dialog and select **Next** to continue.
6. Select the **Domain Controller for a new domain** option to install the first domain controller in the domain. Select **Next** to continue.
7. This test environment only has one domain controller in new forest or domain. Select the **Domain in a new forest** option, then select **Next** to continue.
8. Enter `example.com` as the fully qualified DNS name for the new domain, then select **Next** to continue.
9. Enter `EXAMPLECOM` as the Windows domain NETBIOS name, then select **Next** to continue.
10. Accept the default Active Directory database and log folder locations by selecting the **Next** button.
11. Accept the default shared system volume folder location by selecting the **Next** button.
12. The wizard will check if the DNS servers delegated for the example.com domain support dynamic updates. Since the example.com domain does not exist, the domain controller will also be the DNS server. Select the **Install and configure the DNS server on this computer, and set this computer to use this DNS server as its preferred DNS server** option, and select **Next** to continue.
13. Select the **Permissions compatible only with Windows 2000 and Windows Server 2003 operating systems** option for the default permissions for user and group objects. Select **Next** to continue.
14. Enter a strong password used for the directory services restore administration password. Select **Next** to continue.
15. Review the options you have selected and select **Next** to start the installation.
16. Active Directory is now installed on this server. Close the wizard by selecting the **Finish** button. You will then be required to restart Windows to complete the installation.

One way of confirming that your Windows server has both a domain controller and a DNS role configured is by running the Manage Your Server tool. To start the Manage Your Server tool select **Start** → **Administrative Tools** → **Manage Your Server**. Your Manage Your Server screen should look similar to Figure B-1.


Manage Your Server

Manage Your Server

Server: B501B06

Search Help and Support Center

Managing Your Server Roles

Use the tools and information found here to add or remove roles and perform your daily administrative tasks.

Add or remove a role

Read about server roles

Read about remote administration

Your server has been configured with the following roles:

Domain Controller (Active Directory)

Domain controllers use Active Directory to manage network resources such as users, computers, and applications.

Manage users and computers in Active Directory

Manage domains and trusts

Manage sites and services

Review the next steps for this role

DNS Server

DNS (Domain Name System) servers translate domain and computer DNS names to IP addresses.

Manage this DNS server

Review the next steps for this role

Don't display this page at logon

Tools and Updates

Administrative Tools

More Tools

Windows Update

Computer and Domain Name Information

Internet Explorer Enhanced Security Configuration

See Also

Help and Support

Microsoft TechNet

Deployment and Resource Kits

List of Common Administrative Tasks

Windows Server Communities

What's New

Strategic Technology Protection Program

Figure B-1 Windows server configured in both domain controller and DNS roles

Additional software installation

In order to complete the test environment, you must install some additional software tools supplied by Microsoft.

Installation of Microsoft Windows Support Tools

The first software package is the Microsoft Windows Support Tools. This package allows administrators and support personnel to manage their networks and troubleshoot problems. The tools are located on the \Support\Tools folder on the Windows operating system CD. The AIX 5L integration scenarios use the **`ktpass`**, **`ldp`**, and **`setspn`** commands from the MS Windows Support Tools.

Installation of Windows Server 2003 Resource Kit Tools

The second software package to be installed is the Windows Server 2003 Resource Kit. The resource kit can be downloaded from Microsoft using the following Web site. The **`klist`** and **`kerbtray`** commands included with the resource kit are very useful when working with Kerberos on Windows:

`http://www.microsoft.com/downloads/details.aspx?familyid=9d467a69-57ff-4ae7-96ee-b18c4790cffd&displaylang=en`

C

Example.com certificate authority setup

This appendix describes the setup and use of the certificate authority that was used with all example scenarios in this book to issue certificates.

Installing OpenSSL

OpenSSL comes with most Linux distributions by default. Information about OpenSSL can be found at the OpenSSL Web site:

http://www.openssl.org/

For AIX OpenSSL is required to use OpenSSH. The OpenSSL package can be found at the AIX Toolbox for Linux Web site:

http://www-03.ibm.com/servers/aix/products/aixos/linux/download.html

OpenSSL is located under the cryptographic content section.

The installation has to be done through the **rpm** command, as shown in Example C-1.

Example: C-1 RPM installation of OpenSSL

```
# rpm -iv openssl-0.9.7d-2.aix5.1.ppc.rpm
openssl-0.9.7d-2
#
```

Creating the CA certificate

To set up the CA for the example.com domain we need to make some assumptions. We modify the openssl.cnf to reflect these assumptions to the CA. The file can be found at /var/ssl/openssl.cnf and the interesting sections are shown in Example C-2.

Example: C-2 openssl.cnf

```
[ CA_default ]
dir             = /var/ssl/example.com  # Where everything is kept
certs           = $dir/certs            # Where the issued certs are kept
crl_dir         = $dir/crl              # Where the issued crl are kept
database        = $dir/index.txt        # database index file.
unique_subject  = yes
new_certs_dir   = $dir/newcerts         # default place for new certs.
certificate     = $dir/cacert.pem       # The CA certificate
serial          = $dir/serial           # The current serial number
                                        # must be commented out to leave a V1
CRL
crl             = $dir/crl.pem          # The current CRL
private_key     = $dir/private/cakey.pem# The private key
RANDFILE        = $dir/private/.rand    # private random number file
x509_extensions = usr_cert              # The extentions to add to the cert
name_opt        = ca_default            # Subject Name options
```

```
cert_opt        = ca_default            # Certificate field options
copy_extensions = copy
default_days    = 365                   # how long to certify for
default_crl_days= 30                    # how long before next CRL
default_md      = md5                   # which md to use.
preserve        = no                    # keep passed DN ordering
policy          = policy_match
   .
   .
   .
[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = us
countryName_min                 = 2
countryName_max                 = 2
stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = TX
localityName                    = Locality Name (eg, city)
0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = example.com
organizationalUnitName          = Organizational Unit Name (eg, section)
commonName                      = Common Name (eg, YOUR name)
commonName_max                  = 64
emailAddress                    = Email Address
emailAddress_max                = 64
```

Also, the directories to store the certificates and keys must be created:

```
mkdir /var/ssl/example.com /var/ssl/example.com/certs /var/ssl/example.com/crl
/var/ssl/example.com/newcert /var/ssl/example.com/private
```

OpenSSL is using a couple of files, which it uses to maintain the CA. These files must be created:

```
touch /var/ssl/example.com/index.txt
echo "01" >> /var/ssl/example.com/serial
```

The access rights on the directories and files should be reviewed to restrict access to the CA and, most importantly, to the private key as far as possible.

To create the CA certificate the OpenSSL command is issued directly, as shown in Example C-3.

Example: C-3 Generating the CA certificate

```
# /opt/freeware/bin/openssl req  -new -x509 -keyout \
/var/ssl/example.com/private/cakey.pem -out /var/ssl/example.com/cacert.pem
Generating a 1024 bit RSA private key
............++++++
........................++++++
writing new private key to '/var/ssl/example.com/private/example.com.CA.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [us]:
State or Province Name (full name) [TX]:
Locality Name (eg, city) []:
Organization Name (eg, company) [example.com]:
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:CA
Email Address []:CA@example.com
#
```

During the creation of the certificate missing information must be provided. Also, the information that has been defined as default in the openssl.cnf file must be confirmed. The password for the CA private key must be given during the creation process. This password is needed whenever the CA's private key is used.

The certificate can be displayed as in Example C-4.

Example: C-4 Certificate fields

```
# /opt/freeware/bin/openssl x509 -in cacert.pem -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 0 (0x0)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=us, ST=TX, L=Austin, O=example.com,
CN=CA/emailAddress=ca@example.com
        Validity
            Not Before: Sep  8 15:20:52 2005 GMT
            Not After : Sep  8 15:20:52 2006 GMT
        Subject: C=us, ST=TX, L=Austin, O=example.com,
CN=CA/emailAddress=ca@example.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:bc:2b:c9:02:5b:cd:c2:0e:6d:fa:5f:e0:8e:a9:
```

```
                    0d:da:c8:31:61:29:a0:74:90:cd:42:e7:ee:d4:32:
                    83:92:77:12:63:16:0f:d7:7e:da:b3:60:87:fa:8b:
                    52:6d:e5:f3:e2:8b:4d:f1:fc:2d:41:2d:b6:3e:f1:
                    4d:f4:17:23:e8:f8:e8:8a:d0:a9:11:7c:33:c8:12:
                    c4:7c:5d:96:2d:02:c1:5a:47:f7:7d:60:1c:ad:44:
                    38:f0:3f:5a:b3:d9:84:f7:35:72:1e:93:07:e2:f7:
                    52:65:57:72:c6:40:cd:6b:70:8f:be:70:97:b4:ad:
                    1c:6f:2a:f1:c6:99:85:e0:2f
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                4F:FB:34:7E:62:56:77:D7:6C:3D:0C:C7:FB:40:02:2D:7E:DF:42:B8
            X509v3 Authority Key Identifier:

keyid:4F:FB:34:7E:62:56:77:D7:6C:3D:0C:C7:FB:40:02:2D:7E:DF:42:B8

DirName:/C=us/ST=TX/L=Austin/O=example.com/CN=CA/emailAddress=ca@example.com
                serial:00

            X509v3 Basic Constraints:
                CA:TRUE
    Signature Algorithm: md5WithRSAEncryption
        6b:f2:6f:bf:3d:64:2d:b9:e0:c8:fa:0f:9e:14:c1:f4:73:2e:
        62:29:09:e2:06:8d:7f:a3:2b:f6:0c:60:b9:48:b6:4d:9e:ca:
        8c:0a:32:c4:ec:1e:27:58:47:a9:82:92:78:43:e6:dc:9e:2d:
        65:ef:18:86:21:0a:86:30:6a:6b:33:c0:a5:4d:4f:63:bd:9c:
        c2:dd:af:5d:15:ea:c2:5b:46:95:4c:e5:49:23:2d:cf:8d:d5:
        e4:c8:ae:53:62:d5:c0:76:1e:4b:98:3a:f4:46:2f:ce:24:d0:
        84:59:e4:07:5e:ab:93:f0:d6:83:3f:c3:4b:fa:3e:10:69:c6:
        cf:d5
-----BEGIN CERTIFICATE-----
MIIDGzCCAoSgAwIBAgIBADANBgkqhkiG9w0BAQQFADBtMQswCQYDVQQGEwJ1czEL
MAkGA1UECBMCVFgxDzANBgNVBAcTBkF1c3RpbjEUMBIGA1UEChMLZXhhbXBsZS5j
b20xCzAJBgNVBAMTAkNBMR0wGwYJKoZIhvcNAQkBFg5jYUBleGFtcGxlLmNvbTAe
Fw0wNTA5MDgxNTIwNTJaFw0wNjA5MDgxNTIwNTJaMG0xCzAJBgNVBAYTAnVzMQsw
CQYDVQQIEwJUWDEPMA0GA1UEBxMGQXVzdGluMRQwEgYDVQQKEwtleGFtcGxlLmNv
bTELMAkGA1UEAxMCQ0ExHTAbBgkqhkiG9w0BCQEWDmNhQGV4YW1wbGUuY29tMIGf
MA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC8K8kCW83CDm36X+COqQ3ayDFhKaB0
kM1C5+7UMo0SdxJjFg/XftqzYIf6i1Jt5fPii03x/C1BLbY+8U30FyPo+0iK0KkR
fDPIEsR8XZYtAsFaR/d9YBytRDjwP1qz2YT3NXIekwfi91JlV3LGQM1rcI++cJe0
rRxvKvHGmYXgLwIDAQABo4HKMIHHMB0GA1UdDgQWBBRP+zR+YlZ312w9DMf7QAIt
ft9CuDCBlwYDVR0jBIGPMIGMgBRP+zR+YlZ312w9DMf7QAItft9CuKFxpG8wbTEL
MAkGA1UEBhMCdXMxCzAJBgNVBAgTAlRYMQ8wDQYDVQQHEwZBdXN0aW4xFDASBgNV
BAoTC2V4YW1wbGUuY29tMQswCQYDVQQDEwJDQTEdMBsGCSqGSIb3DQEJARY0Y2FA
ZXhhbXBsZS5jb22CAQAwDAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQQFAAOBgQBr
8m+/PWQtueDI+g+eFMHOcy5iKQniBo1/oyv2DGC5SLZNnsqMCjLE7B4nWEepgpJ4
```

```
Q+bcni1l7xiGIQqGMGprM8ClTU9jvZzC3a9dFerCWOaVTOVJIy3PjdXkyK5TYtXA
dh5LmDrORi/OJNCEWeQHXquT8NaDP8NL+j4QacbP1Q==
-----END CERTIFICATE-----
```

Signing a certificate

The client or server that needs to get a certificate must create a certificate signing request and send this request to the CA.

The certificate request we want to sign here is shown in Example C-5.

Example: C-5 Certificate signing request

```
# /opt/freeware/bin/openssl req -in bs01b02.req  -text
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=us, ST=TX, L=Austin, O=example.com, CN=bs01b02
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1023 bit)
                Modulus (1023 bit):
                    73:27:03:52:83:5a:af:a5:0e:f0:08:a0:a5:c9:63:
                    6b:b6:40:c3:6a:70:b6:ab:3c:e1:6c:3b:e5:e0:68:
                    c0:67:21:88:38:32:5a:b3:77:b9:e9:3a:e3:f1:72:
                    19:31:f1:bc:c4:b0:2f:a5:de:1c:33:81:e5:10:db:
                    93:80:99:e2:82:97:34:a2:2f:03:fa:37:74:5f:10:
                    4d:73:1d:ee:7f:6b:e6:1e:96:6a:69:c8:d2:90:83:
                    55:ba:58:49:3a:ad:a8:23:ea:f9:c1:3d:73:69:8d:
                    67:48:c9:82:8e:8c:7b:98:77:1b:81:e7:1d:84:6f:
                    69:bf:6e:a1:13:6c:e6:5f
                Exponent: 65537 (0x10001)
        Attributes:
            a0:00
    Signature Algorithm: md5WithRSAEncryption
        4d:b1:a6:e3:06:8f:25:7c:f1:22:3a:c3:a0:0e:31:c0:c8:e6:
        14:c5:1d:bb:fc:2c:14:66:e0:16:cb:6a:eb:6c:1a:96:03:c4:
        4b:76:f3:9a:2f:91:74:72:bc:84:95:dd:f1:54:98:b7:b0:bc:
        2d:7c:52:fd:1a:23:c1:6c:e0:c5:ce:fa:2e:eb:04:0f:9a:16:
        0e:f7:d3:b0:0a:53:73:22:91:e2:ad:5d:30:8c:26:07:8f:69:
        26:1f:c0:c3:dc:01:46:78:59:5b:37:74:e0:ca:ae:4f:8e:e3:
        8e:a6:5a:3b:ce:ad:31:e8:58:87:3b:75:0b:ff:37:3b:a7:1d:
        1c:16
-----BEGIN CERTIFICATE REQUEST-----
MIIBkTCB+wIBADBTMQswCQYDVQQGEwJ1czELMAkGA1UECBMCVFgxDzANBgNVBAcT
BkF1c3RpbjEUMBIGA1UEChMLZXhhbXBsZS5jb20xEDAOBgNVBAMTB2JzMDFiMDIw
gZ4wDQYJKoZIhvcNAQEBBQADgYwAMIGIAoGAcycDUoNar6UO8Aigpclja7ZAw2pw
```

```
tqs84Ww75eBowGchiDgyWrN3uek64/FyGTHxvMSwL6XeHDOB5RDbk4CZ4oKXNKIv
A/o3dF8QTXMd7n9r5h6WamnIOpCDVbpYSTqtqCPq+cE9c2mNZOjJgo6Me5h3G4Hn
HYRvab9uoRNs5l8CAwEAAaAAMA0GCSqGSIb3DQEBBAUAA4GBAE2xpuMGjyV88SI6
w6AOMcDI5hTFHbv8LBRm4BbLautsGpYDxEt285ovkXRyvISV3fFUmLewvC18UvOa
I8Fs4MXO+i7rBA+aFg7307AKU3MikeKtXTCMJgePaSYfwMPcAUZ4WVs3dODKrk+O
446mWjvOrTHoWIc7dQv/NzunHRwW
-----END CERTIFICATE REQUEST-----
```

To sign the certificate the **openssl** command is used with a specified policy, as shown in Example C-6.

Example: C-6 Signing a certificate

```
# /opt/freeware/bin/openssl ca  -policy policy_anything -in  bs01b02.req  -out
example.com/certs/bs01b02.cert
Using configuration from /var/ssl/openssl.cnf
Enter pass phrase for /var/ssl/example.com/private/cakey.pem:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName           :PRINTABLE:'us'
stateOrProvinceName   :PRINTABLE:'TX'
localityName          :PRINTABLE:'Austin'
organizationName      :PRINTABLE:'example.com'
commonName            :PRINTABLE:'bs01b02'
Certificate is to be certified until Oct 17 23:49:08 2006 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Signing a certificate for bs01b06

It is also possible that X509 extensions must be present to use a certificates. See the OpenSSL Web site for details:

http://www.openssl.org/

Example C-7 shows a certificate with extensions for key usage.

Example: C-7 Certificate for bs01b06

```
# cd /var/ssl/example.com
# openssl ca -policy policy_anything -cert cacert.pem -keyfile
private/cakey.pem   -out bs01b06.example.com_cert.pem  -in
bs01b06.example.com_req.pem
```

```
Using configuration from /var/ssl/openssl.cnf
Enter pass phrase for private/cakey.pem:
DEBUG[load_index]: unique_subject = "no"
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 7 (0x7)
        Validity
            Not Before: Sep 21 16:56:42 2005 GMT
            Not After : Sep 21 16:56:42 2006 GMT
        Subject:
            commonName                = bs01b06.example.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                7F:5E:A4:66:BC:BA:EF:DF:54:B0:43:19:70:80:42:E5:C4:A7:B1:2E
            X509v3 Authority Key Identifier:

keyid:4F:FB:34:7E:62:56:77:D7:6C:3D:0C:C7:FB:40:02:2D:7E:DF:42:B8

DirName:/C=us/ST=TX/L=Austin/O=example.com/CN=CA/emailAddress=ca@example.com
                serial:00

            X509v3 Extended Key Usage:
                TLS Web Server Authentication
            X509v3 Key Usage:
                Digital Signature, Key Encipherment
Certificate is to be certified until Sep 21 16:56:42 2006 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated

#
```

D

Schemas and migration tools

This appendix is a reference for the different LDAP schemas, the object class and attribute mapping tables, and the AIX data migration tools used in this book. This appendix covers the following IBM supported schemas: AIX, RFC2307 and RFC2307AIX, and several unsupported schemas such as RFC2307bis and Microsoft Windows Services for UNIX. This appendix does not discuss the differences or how to choose the appropriate schema. For more planning information related to schemas please read 3.2, “Which schema you should use” on page 60.

This appendix also includes a reference for the data migration tools supplied with AIX.

The following sections are found in this appendix:

AIX schema

The following section discusses the original AIX schema. The AIX schema is the only supported schema for AIX 4.3.3 and AIX 5L Version 5.1 clients. The schema uses custom object class and attribute names and is not RFC2307 compliant.

Table D-1 lists each object class and its required and optional attributes.

Table D-1 AIX schema object classes and attribute names

Object class name	Description	Required and optional attributes
AIXAccount	AIX user information object class	**Required:** gid, passwordChar, uid, userName **Optional**: adminGroupNames, AIXDefaultMACLevel, AIXFuncMode, AIXisDCEExport, AIXLowMACLevel, AIXPromptMAC, AIXScreens, AIXUpperMACLevel, auditClasses, authMethod1, authMethod2, caption, coreSizeLimit, coreSizeLimitHard, cpuSize, cpuSizeHard, dataSegSize, dataSegSizeHard, description, filePermMask, fileSizeLimit, fileSizeLimitHard, gecos, groupList, groupsIds, groupSwitchUserAllowed, homeDirectory, host, hostLastLogin, hostLastUnsuccessfulLogin, isAccountEnabled, isAdministrator, isDaemon, isLoginAllowed, isRemoteAccessAllowed, isSwitchUserAllowed, ixLastUpdate, ixTimeLastLogin, ixTimeLastUnsuccessfulLogin, l, loginShell, loginTimes, maxFailedLogins, maxLogin, numberWarnDays, o, ou, openFileLimit, openFileLimitHard passwordCheckMethods, passwordDictFiles, passwordExpiredWeeks, passwordExpireTime, passwordFlags, passwordHistSize, passwordMaxAge, passwordMaxRepeatedChars, passwordMinAge, passwordMinAlphaChars, passwordMinDiffChars, passwordMinLength, passwordMinOtherChars, physicalMemLimit, physicalMemLimitHard, principalPtr, roleList, seeAlso, stackSizeLimit, stackSizeLimitHard, systemEnvironment, terminalAccess, terminalLastLogin, terminalLastUnsuccessfulLogin, timeExpiredLogout, timeExpireLockout, trustedPathStatus, unsuccessfulLoginCount, userCertificate, userEnvironment, userPassword, unsuccessful_login_times, passwordHistList, hostsAllowedLogin, hostsDeniedLogin, passwordHistExpire, capability, rcmds

Object class name	Description	Required and optional attributes
AIXaccessGroup	AIX group information	**Required**: gid, GroupName **Optional**: AIXGroupAdminList, AIXisDCEExport, AIXScreens, groupPassword, ibm-aixProjectNameList, isAdministrator, member
ibm-SecurityIdentities	Defines the security identities of a user (user could be a person or a service)	**Required**: N/A **Optional**: altSecurityIdentities, userPrincipalName
AIXAdmin	AIX class to store user/group administration attributes	**Required**: N/A **Optional**: AIXAdminGroupId, AIXAdminUserId, AIXGroupID, AIXUserID, cn

```
# The following entry contains the object class used in the filter
# for queries on user data.  All users should have this class.
keyobjectclass  SEC_CHAR        aixaccount              s

# The following attributes are required by AIX to be functional
username        SEC_CHAR        username                s
spassword       SEC_CHAR        userpassword            s
id              SEC_INT         uid                     s
pgrp            SEC_CHAR        gid                     s
gecos           SEC_CHAR        gecos                   s
home            SEC_CHAR        homedirectory           s
shell           SEC_CHAR        loginshell              s
lastupdate      SEC_INT         ixlastupdate            s
```

Figure D-1 Required attributes for AIX user schema

AIX attribute list

The default user LDIF stanza for AIX schema is shown in Figure D-2.

```
dn: username=default,ou=People,ou=xyzcomp,ou=puzzle
username: default
objectClass: aixaccount
objectClass: ibm-securityidentities
objectClass: account
uid: -1
gid: -1
passwordchar: *
isadministrator: false
isloginallowed: true
isswitchuserallowed: true
isdaemon: true
isremoteaccessallowed: true
groupswitchuserallowed: ALL
terminalaccess: ALL
authmethod1: SYSTEM
authmethod2: NONE
trustedpathstatus: nosak
filepermmask: 022
timeexpirelockout: 0
numberwarndays: 0
isaccountenabled: false
maxfailedlogins: 0
passwordhistexpire: 0
passwordhistsize: 0
passwordminage: 0
passwordmaxage: 0
passwordexpiredweeks: -1
passwordminalphachars: 0
passwordminotherchars: 0
passwordminlength: 0
passwordmindiffchars: 0
passwordmaxrepeatedchars: 8
filesizelimit: 2097151
coresizelimit: 2097151
cpusize: -1
datasegsize: 262144
physicalmemlimit: 65536
stacksizelimit: 65536
openfilelimit: 2000
aixscreens: *
aixfuncmode: roles+acl
```

Figure D-2 Default user LDIF stanza

Figure D-3 shows a typical user AIX schema LDIF stanza.

```
dn: username=bthere,ou=People,ou=xyzcomp,ou=puzzle
username: bthere
objectClass: aixaccount
objectClass: ibm-securityidentities
objectClass: account
passwordchar: !
uid: 500
gid: 1
gecos: Ben There
homedirectory: /home/bthere
loginshell: /usr/bin/ksh
rolelist: ManageAllUsers
isadministrator: true
admingroupnames: security,printq,system
userpassword: AeAYkSFisG/QM
ixlastupdate: 1114093114
ixtimelastlogin: 1127313326
terminallastlogin: /dev/dtlogin/_0
hostlastlogin: matilda
unsuccessfullogincount: 0
ixtimelastunsuccessfullogin: 1126642932
terminallastunsuccessfullogin: /dev/vty0
hostlastunsuccessfullogin: matilda
hostlastunsuccessfullogin: matilda
```

Figure D-3 Typical user AIX schema LDIF stanza

The required group attributes in the AIX schema are shown in Figure D-4 on page 302. Figure D-4 on page 302 shoes the required stanza from /etc/security/ldap/aixgroup.map.

```
# The following entry contains the object class used in the filter
# for queries on group data.  All groups should have this class.
keyobjectclass  SEC_CHAR        aixaccessgroup         s

# The following attributes are required by AIX to be functional
groupname       SEC_CHAR        groupname              s
id              SEC_INT         gid                    s
users           SEC_LIST        member                 m
primary         SEC_LIST        primary                s
adms            SEC_LIST        aixgroupadminlist      s
admin           SEC_BOOL        isadministrator        s
dce_export      SEC_BOOL        aixisdceexport         s
screens         SEC_LIST        aixscreens             s
```

Figure D-4 Required AIX schema group attributes

RFC2307 schema

This section discusses the RFC2307 schema that is available on AIX 5L Version 5.2 and later. Table D-2 lists the object classes used by AIX the LDAP client. All the object classes except account and AIXadmin are defined by the RFC2307 standard.

Table D-2 RFC2307 schema object classes and attribute names

Object class name	Description	Required and optional attributes
account	Common account information.	**Required**: uid **Optional**: description, host, l, o, ou, seeAlso
posixAccount	Abstraction of an account with POSIX attributes.	**Required**: cn, gidNumber, homeDirectory, uid, uidNumber **Optional**: description, gecos, loginShell, userPassword

Object class name	Description	Required and optional attributes
shadowAccount	Additional attributes of shadow passwords.	**Required**: uid **Optional**: description, shadowExpire, shadowFlag, shadowInactive, shadowLastChange, shadowMax, shadowMin, shadowWarning, userPassword
posixGroup	Abstraction of a group of accounts.	**Required**: cn, gidNumber **Optional**: description, memberUid, userPassword
bootableDevice	A device with boot parameters; device *should* be used as a structural class.	**Required**: N/A **Optional**: bootFile, bootParameter
ieee802Device	A device with a MAC address; device *should* be used as a structural class.	**Required**: N/A **Optional**: macAddress
ipHost	Abstraction of a host, an IP device. The distinguished value of the cn attribute denotes the host's canonical name. Device *should* be used as a structural class.	**Required**: cn, ipHostNumber **Optional**: l, description, manager
ipNetwork	Abstraction of a network. The distinguished value of the cn attribute denotes the network's canonical name.	**Required**: cn, ipNetworkNumber **Optional**: ipNetmaskNumber, l, description, manager
ipProtocol	Abstraction of an IP protocol. Maps a protocol number to one or more names. The distinguished value of the cn attribute denotes the protocol's canonical name.	**Required**: cn, ipProtocolNumber, description **Optional**: description
ipService	Abstraction an Internet Protocol service. Maps an IP port and protocol (such as TCP or UDP) to one or more names. The distinguished value of the cn attribute denotes the service's canonical name.	**Required**: cn, ipServicePort, ipServiceProtocol **Optional**: description

Object class name	Description	Required and optional attributes
nisMap	A generic abstraction of a NIS map.	**Required**: nisMapName **Optional**: description
nisNetgroup	Abstraction of a netgroup. May refer to other netgroups.	**Required**: cn **Optional**: nisNetgroupTriple, memberNisNetgroup, description
nisObject	An entry in an NIS map.	**Required**: cn, nisMapEntry, nisMapName **Optional**: description
oncRpc	Abstraction of an Open Network Computing (ONC™) [RFC1057] Remote Procedure Call (RPC) binding. This class maps an ONC RPC number to a name. The distinguished value of the cn attribute denotes the RPC service's canonical name.	**Required**: cn, oncRpcNumber, description **Optional**: description
AIXAdmin	AIX class to store user/group administration attributes.	**Required**: n/a **Optional**: AIXAdminGroupId, AIXAdminUserId, AIXGroupID, AIXUserID, cn

Figure D-5 shows the default RFC2307 schema user in LDIF format.

```
dn: uid=default,ou=People,
uid: default
objectClass: posixaccount
objectClass: shadowaccount
objectClass: account
cn: default
uidnumber: -1
gidnumber: -1
userpassword: {crypt}*
homedirectory: *
shadowwarning: 0
shadowmin: 0
shadowmax: 0
shadowexpire: -1
```

Figure D-5 Default RFC2307 schema user in LDIF format

As you can see, there are far fewer attributes defined than for the AIX schema shown earlier. This is also true of the typical user, as shown in Table D-3 on page 306.

```
dn: uid=bthere,ou=People,
uid: bthere
objectClass: posixaccount
objectClass: shadowaccount
objectClass: account
cn: bthere
uidnumber: 500
gidnumber: 1
gecos: Ben There
homedirectory: /home/jwtesch
loginshell: /usr/bin/ksh
userpassword: {crypt}AeQRvTRisG/QM
shadowlastchange: 12894
```

Figure D-6 Typical RFC2307 schema-based user in LDIF format

RFC2307AIX schema

This section discusses the RFC2307AIX schema that is available on AIX 5L Version 5.2 and later. The RFC2307AIX schema is composed of the RFC2307 object classes in the previous section and the AIX 5L-specific object classes aixAuxAccount and aixAuxGroup, which define the additional attributes used by the AIX 5L security subsystem. These additional attributes allow you to record unsuccessful login attempts, create additional password rules, and set the default file mask.

Table D-3 on page 306 lists the object classes that extend the RFC2307 schema and the required and optional attributes.

Table D-3 RFC2307 AIX schema object classes and attribute names

Object class name	**Description**	**Required and optional attributes**
aixAuxAccount	Auxiliary AIX user information object class, for use with posixAccount and shadowAccount object classes	**Required**: N/A **Optional**: passwordChar, adminGroupNames, aIXDefaultMACLevel, aIXFuncMode, aIXisDCEExport, aIXLowMACLevel, aIXPromptMAC, aIXScreens, aIXUpperMACLevel, ,auditClasses, authMethod1, authMethod2, coreSizeLimit, coreSizeLimitHard, cPuSize, cPuSizeHard, dataSegSize, dataSegSizeHard, filePermMask, fileSizeLimit, fileSizeLimitHard, groupList, groupSwitchUserAllowed, hostLastLogin, hostLastUnsuccessfulLogin, hostsAllowedLogin, hostsDeniedLogin, isAdministrator, isAccountEnabled, isDaemon, isLoginAllowed, isRemoteAccessAllowed, isSwitch, UserAllowed, ixTimeLastLogin, ixTimeLastUnsuccessfulLogin, loginTimes, maxFailedLogins, maxLogin, openFileLimit, openFileLimitHard, passwordCheckMethods, passwordDictFiles, passwordExpireTime, passwordHistSize, passwordMaxRepeatedChars, passwordMinAlphaChars, passwordMinDiffChars, passwordMinLength, passwordMinOtherChars, physicalMemLimit, physicalMemLimitHard, roleList, StackSizeLimit, StackSizeLimitHard, SystemEnvironment, terminalAccess, terminalLastLogin, terminalLastUnsuccessfulLogin, timeExpiredLogout, timeExpireLockout, trustedPathStatus, unsuccessfulLoginCount, userEnvironment, passwordFlags
aixAuxGroup	Auxiliary AIX group information object class, for use with the posixGroup object class	**Required**: N/A **Optional**: aIXGroupAdminList, aIXisDCEExport, aIXScreens, groupPassword, isAdministrator

The default RFC2307AIX schema is shown in the following LDIF file (see the following figure).

```
dn: uid=default,ou=People,ou=xyzcomp,ou=myorg
uid: default
objectClass: aixauxaccount
objectClass: shadowaccount
objectClass: posixaccount
objectClass: account
cn: default
uidnumber: -1
gidnumber: -1
passwordchar: *
userpassword: {crypt}*
homedirectory: *
isadministrator: false
isloginallowed: true
isswitchuserallowed: true
isdaemon: true
isremoteaccessallowed: true
groupswitchuserallowed: ALL
terminalaccess: ALL
authmethod1: SYSTEM
authmethod2: NONE
trustedpathstatus: nosak
filepermmask: 022
timeexpirelockout: 0
shadowwarning: 0
isaccountenabled: false
maxfailedlogins: 0
passwordhistexpire: 0
passwordhistsize: 0
shadowmin: 0
shadowmax: 0
shadowexpire: -1
passwordminalphachars: 0
passwordminotherchars: 0
passwordminlength: 0
passwordmindiffchars: 0
passwordmaxrepeatedchars: 8
filesizelimit: 2097151
coresizelimit: 2097151
cpusize: -1
datasegsize: 262144
physicalmemlimit: 65536
stacksizelimit: 65536
openfilelimit: 2000
aixscreens: *
aixfuncmode: roles+acl
```

RFC2307bis schema

The following section discusses the RFC2307bis schema, which was a follow-on to the original RFC2307 schema. RFC2307bis was an Internet Engineering Task Force (IETF) working draft that expired without becoming an official standard. Even though RFC2307bis is expired, vendors have already implemented some of the newer features. For example, the **`automount`** daemon on AIX 5L Version 5.3 and later supports the new automount and automountMap object classes.

Table D-4 lists the new object classes added by RFC2307bis and the required and optional attributes.

Table D-4 RFC2307bis schema object classes and attribute names

Object class name	Description	Required and optional attributes
automountMap	Automount map information	**Required**: automountMapName **Optional**: description
autoimmunity	Automount information	**Required**: automountKey, automountInformation **Optional**: description
nisDomainObject	Associates a NIS domain with a naming context	**Required**: nisDomain **Optional**
nisKeyObject	An object with a public and secret key	**Required**: cn, nisPublickey, nisSecretKey **Optional**: uidNumber, description

Microsoft Windows Services for UNIX schemas

This section is a reference for the different Microsoft Windows Service for UNIX (SFU) schemas. There are two different versions of the SFU schemas: one for SFU 2.0 and the other SFU 3.0/3.5. They can be distinguished by the object class and attribute names. SFU 3.0/3.5 names are normally prefixed by msSFU30, while the SFU 2.0 names use msSFU or no prefix at all.

The SFU 2.0 schema adds msSFUPosixAccount and msSFUShadowAccount as auxiliary classes to the user class, and the msSFUPosixGroup object class is added as an auxiliary class to the group class.

The SFU 3.0/3.5 schema adds msSFU30PosixAccount and msSFU30ShadowAccount as auxiliary classes to the user class, and the msSFU30PosixGroup is added as an auxiliary class to this class.

Table D-5 lists the auxiliary object classes added by SFU 2.0 and its required and optional attributes.

Table D-5 Microsoft SFU V2.0 schema object classes and attribute names

Object class name	**Required and optional attributes**
msSFUPosixAccount	**Required**: cn **Optional**: description, gecos, gidNumber, loginShell, msSFUHomeDirectory, msSFUName, msSFUPassword, posixMemberOf, syncNisDomain, uid, uidNumber
msSFUShadowAccount	**Required**: cn **Optional**: description, msSFUName, shadowWarning, shadowMax, shadowMin, shadowLastChange, shadowFlag, shadowExpire, shadowInactive, uid
msSFUPosixGroup	**Required**: cn **Optional**: description, GidNumber, MemberUid, msSFUName, msSFUPassword, PosixMember, syncNisDomain

Table D-6 lists the auxiliary object classes added by SFU 3.0 and 3.5 and their required and optional attributes.

Table D-6 Microsoft SFU V3.0 and V3.5 schema object classes and attribute names

Object class name	**Required and optional attributes**
msSFU30PosixAccount	**Required**: N/A **Optional**: description, msSFU30Gecos, msSFU30GidNumber, msSFU30HomeDirectory, msSFU30LoginShell, msSFU30Name, msSFU30NisDomain, msSFU30Password,, msSFU30PosixMemberOf, msSFU30UidNumber

Object class name	Required and optional attributes
msSFU30ShadowAccount	**Required**: N/A **Optional**: description, msSFU30Name, msSFU30ShadowWarning, msSFU30ShadowMax, msSFU30ShadowMin, msSFU30ShadowLastChange, msSFU30ShadowFlag, msSFU30ShadowExpire, msSFU30ShadowInactive
msSFU30PosixGroup	**Required**: N/A **Optional**: description, msSFU30GidNumber, msSFU30MemberUid, msSFU30Name, msSFU30NisDomain, msSFU30PosixMember, msSFU30Password

Object class and attribute name mappings

The following section is a reference for the different mapping files used by the AIX 5L LDAP client starting with AIX 5L Version 5.2. The mapping files allow AIX 5L to work with schemas with different object class and attribute names. AIX 5L currently supports mapping files for the user and group entities.

Note: AIX 5L also supports mapping for the AIXAdmin object class. This map is independent of the schemas used for user information and authentication. The AIXAdmin data is only used by commands like `mkuser` when adding new LDAP users. The map for the AIXAdmin class is located in /etc/security/ldap/aixid.map.

Table D-7 lists the supported AIX schemas and the file name of the mapping files used for the user and group entities. The mapping files are included in the bos.rte.security LPP and are installed in the /etc/security/ldap directory.

Table D-7 User and group entities mapping files shipped with AIX 5L

AIX 5L schema name	User entity mapping file	Group entity mapping file
AIX	aixuser.map	aixgroup.map
RFC2307	2307user.map	2307group.map
RFC2307AIX	2307aixuser.map	2307aixgroup.map

AIX, RFC2307 and RFC2307AIX Mappings

Table D-8 User and group entities object class name mapping

AIX	RFC2307 schema object class names	AIX schema object class names	RFC2307AIX schema object class names
keyobjectclass	posixAccount	aixAccount	posixAccount
keyobjectclass	posixGroup	aixAccessGroup	posixGroup

Table D-9 User attribute mapping from AIX to AIX, RFC2307, and RFC2307AIX schemas

AIX attribute names	RFC2307 schema attribute names	AIX schema attribute names	RFC2307AIX schema attribute names
username	uid	username	uid
id	uidNumber	uid	uidNumber
pgrp	gidNumber	gid	gidNumber
home	homeDirectory	homeDirectory	homeDirectory
shell	loginShell	loginShell	loginShell
gecos	gecos	gecos	gecos
spassword	userPassword	userPassword	userPassword
lastupdate	shadowLastChange	ixLastUpdate	shadowLastChange
Note - The attributes below are optional.			
maxage	shadowMax	passwordMaxAge	shadowMax
minage	shadowMin	passwordMinAge	shadowMin
maxexpired	shadowExpire	passwordExpiredWeeks	shadowExpire
pwdwarntime	shadowWarning	numberWarnDays	shadowWarning
N/A	shadowFlag	N/A	shadowFlag
N/A	shadowInactive	N/A	shadowInactive
account_locked	N/A	isAccountEnabled	isAccountEnabled
admgroups	N/A	adminGroupNames	adminGroupNames
admin	N/A	isAdministrator	isAdministrator
auditclasses	N/A	auditClasses	auditClasses

AIX attribute names	RFC2307 schema attribute names	AIX schema attribute names	RFC2307AIX schema attribute names
auth1	N/A	authMethod1	authMethod1
auth2	N/A	authMethod2	authMethod2
auth_name	N/A	userPrincipalName	userPrincipalName
auth_domain	N/A	altSecurityIdentities	altSecurityIdentities
core	N/A	coreSizeLimit	coreSizeLimit
core_hard	N/A	coreSizeLimitHard	coreSizeLimitHard
cpu	N/A	cPuSize	cPuSize
cpu_hard	N/A	cPuSizeHard	cPuSizeHard
d_level	N/A	aIXDefaultMACLevel	aIXDefaultMACLevel
daemon	N/A	isDaemon	isDaemon
data	N/A	dataSegSize	dataSegSize
data_hard	N/A	dataSegSizeHard	dataSegSizeHard
dce_export	N/A	aIXisDCEExport	aIXisDCEExport
dictionlist	N/A	passwordDictFiles	passwordDictFiles
expires	N/A	timeExpireLockout	timeExpireLockout
flags	N/A	passwordFlags	passwordFlags
fsize	N/A	fileSizeLimit	fileSizeLimit
fsize_hard	N/A	fileSizeLimitHard	fileSizeLimitHard
funcmode	N/A	aIXFuncMode	aIXFuncMode
groups	N/A	groupList	groupList
groupsids	N/A	groupsids	groupsids
histexpire	N/A	passwordHistExpire	passwordHistExpire
histsize	N/A	passwordHistSize	passwordHistSize
histlist	N/A	PasswordHistList	PasswordHistList
host_last_login	N/A	hostLastLogin	hostLastLogin
host_last_unsuccessful_ login	N/A	hostLastUnsuccessfulLo gin	hostLastUnsuccessfulLo gin

AIX attribute names	RFC2307 schema attribute names	AIX schema attribute names	RFC2307AIX schema attribute names
hostsallowedlogin	N/A	hostsAllowedLogin	hostsAllowedLogin
hostsdeniedlogin	N/A	hostsDeniedLogin	hostsDeniedLogin
l_level	N/A	aIXLowMACLevel	aIXLowMACLevel
login	N/A	isLoginAllowed	isLoginAllowed
loginretries	N/A	maxFailedLogins	maxFailedLogins
logintimes	N/A	loginTimes	loginTimes
maxrepeats	N/A	passwordMaxRepeatedChars	passwordMaxRepeatedChars
maxulogs	N/A	maxLogin	maxLogin
minalpha	N/A	passwordMinAlphaChars	passwordMinAlphaChars
mindiff	N/A	passwordMinDiffChars	passwordMinDiffChars
minlen	N/A	passwordMinLength	passwordMinLength
minother	N/A	passwordMinOtherChars	passwordMinOtherChars
nofiles	N/A	openFileLimit	openFileLimit
nofiles_hard	N/A	openFileLimitHard	openFileLimitHard
password	N/A	passwordChar	passwordChar
prompt_mac	N/A	aIXPromptMAC	aIXPromptMAC
pwdchecks	N/A	passwordCheckMethods	passwordCheckMethods
rcmds	N/A	rcmds	rcmds
rlogin	N/A	isRemoteAccessAllowed	isRemoteAccessAllowed
roles	N/A	roleList	roleList
rss	N/A	physicalMemLimit	physicalMemLimit
rss_hard	N/A	physicalMemLimitHard	physicalMemLimitHard
screens	N/A	aIXScreens	aIXScreens
stack	N/A	StackSizeLimit	StackSizeLimit
stack_hard	N/A	StackSizeLimitHard	StackSizeLimitHard
su	N/A	isSwitchUserAllowed	isSwitchUserAllowed

AIX attribute names	RFC2307 schema attribute names	AIX schema attribute names	RFC2307AIX schema attribute names
sugroups	N/A	groupSwitchUserAllowed	groupSwitchUserAllowed
sysenv	N/A	SystemEnvironment	SystemEnvironment
telnet	N/A	isRemoteAccessAllowed	isRemoteAccessAllowed
time_last_login	N/A	ixTimeLastLogin	ixTimeLastLogin
time_last_unsuccessful_login	N/A	ixTimeLastUnsuccessfulLogin	ixTimeLastUnsuccessfulLogin
tpath	N/A	trustedPathStatus	trustedPathStatus
tty_last_login	N/A	terminalLastLogin	terminalLastLogin
tty_last_unsuccessful_login	N/A	terminalLastUnsuccessfulLogin	terminalLastUnsuccessfulLogin
ttys	N/A	terminalAccess	terminalAccess
u_level	N/A	aIXUpperMACLevel	aIXUpperMACLevel
uactivity	N/A	timeExpiredLogout	timeExpiredLogout
umask	N/A	filePermMask	filePermMask
unsuccessful_login_count	N/A	unsuccessfulLoginCount	unsuccessfulLoginCount
usrenv	N/A	userEnvironment	userEnvironment
utocount	N/A	timeExpireLockout	timeExpireLockout
capabilities	N/A	capability	capability
projects	N/A	ibm-aixProjectNameList	ibm-aixProjectNameList

Microsoft Windows Services for UNIX mappings

The following section discusses the mapping files used to support Microsoft SFU 2.0 and 3.0/3.5 schemas. As AIX 5L does not officially support Microsoft Active Directory, it does not ship with mapping files for the SFU schema. Unsupported mapping files for SFU 2.0 and 3.0/3.5 can be can be found in 8.3.6, “Object class and attribute mapping files for SFU” on page 266.

Table D-10 and Table D-11 contain the mapping information for both SFU schemas. The tables have the following information:

- Mapping AIX 5L user and group entities to SFU object class names can be found in Table D-10.
- Mapping AIX 5L user attributes to SFU names can be found in Table D-11.
- Mapping AIX 5L group attributes to SFU names can be found in Table D-12 on page 316.

Table D-10 AIX user and group entities to SFU object class mapping

AIX	**RFC2307 object class names**	**MS SFU V2.0 object class names**	**MS SFU V3.0 and V3.5 object class names**
keyobjectclass	posixAccount	User	User
keyobjectclass	posixGroup	Group	Group

Table D-11 AIX 5L user attribute mappings to Microsoft SFU

AIX attribute names	**RFC2307 attribute names**	**MS SFU 2.0 attribute names**	**MS SFU 3.0/3.5 attribute names**
username[1]	uid	msSFUName sAMAccountName	msSFU30Name sAMAccountName
id	uidNumber	UidNumber	msSFU30UidNumber
pgrp	gidNumber	GidNumber	msSFU30GidNumber
home	homeDirectory	msSFUHomeDirectory	msSFU30HomeDirectory
shell	loginShell	msSFULoginShell	msSFU30LoginShell
gecos[2]	gecos	displayName name Gecos	displayName name msSFU30Gecos
spassword[3]	`userPassword`	msSFUPassword unicodePwd	msSFU30Password unicodePwd
lastupdate	`shadowLastChange`	ShadowLastChange	msSFU30ShadowLastChange
Note - The attributes below this bar are optional.			
`maxage`	`shadowmax`	ShadowMax	msSFU30ShadowMax
`minage`	`shadowmin`	ShadowMin	msSFU30ShadowMin
`maxexpired`	`shadowexpire`	ShadowExpire	msSFU30ShadowExpire
`pwdwarntime`	`shadowwarning`	ShadowWarning	msSFU30ShadowWarning

[1] It is possible to map the sAMAccountName, msSFUName, or msSFU30Name attributes to the AIX username attribute as both attributes will have the same information if the user has the POSIX attributes defined. If you map the sAMAccountName attribute to the AIX username attribute, non-UNIX users in your Active Directory will now be visible to AIX 5L.This mapping would not be optimal, as non-UNIX users would not have the required POSIX attributes like user ID, group ID, home directory, or login shell defined. The preferred mapping for username is the msSFUName or msSFU30Name attribute as only UNIX users with POSIX attributes will be visible.
[2] There are several options for mapping the gecos attribute. If you are importing your data into the SFU's NIS server using the **nismap** command, the gecos or msSFU30Gecos attributes will be populated. If you are managing the UNIX users through the Active Directory Users and Computers tool, it is best to map the AIX gecos attribute to the displayName attribute. Microsoft suggests mapping the gecos attribute to name, but this attribute matches the sAMAccountName and for UNIX users the user names are normally complete names. For example, a user named John Doe might have the attributes sAMAccountName = jdoe, name = jdoe, gecos = <unset>, and displayName = John Doe.
[3] This attribute is unused if you are using the Kerberos authentication module for user authentication. The spassword attribute will only be used for LDAP or UNIX authentication. Microsoft suggests mapping the AIX spassword to unicodePwd, but AIX 5L does not support reading or writing a unicode formatted password. You should only map the AIX spassword attribute to either msSFUPassword or msSFUPassword.

Table D-12 AIX 5L group attribute mappings to Microsoft SFU

AIX attribute names	RFC2307 attribute names	MS SFU 2.0 attribute names	MS SFU 3.0/3.5 attribute names
groupname[1]	cn	msSFUName sAMAccountName	msSFU30Name sAMAccountName
id	gidNumber	GidNumber	msSFU30GidNumber
users[2]	memberUid	MemberUid PosixMember	msSFU30MemberUid msSFU30PosixMember

[1] It is possible to map the sAMAccountName, msSFUName, or msSFU30Name atttributes to the AIX groupname attribute as both attributes will have the same information if the group has the POSIX attributes defined. If you map the sAMAccountName attribute to the AIX groupname attribute, non-UNIX groups in your Active Directory will now be visible to AIX 5L. This mapping would not be optimal, as non-UNIX groups would not have the required POSIX attributes like group or user membership information defined. The preferred mapping for groupname is the msSFUName or msSFU30Name attribute, as only UNIX groups with POSIX attributes will be visible.
[2] Microsoft suggests mapping the AIX users attribute to PosixMember or msSFU30PosixMember, but AIX 5L does not support user group membership specified with distinguished names (DN) versus user names. You should only map the AIX users attribute to either the MemberUid or msSFU30MemberUid attribute. For more information about the different attributes used to define group memebership see “Multiple attributes to specify group memberships” on page 79.

Information about AIX Data Migration Tools

In this section we discuss AIX Data Migration Tools.

nistoldif - NIS to LDAP data migration

nistoldif is a migration tool provided with AIX 5.3 for migrating NIS data to LDAP. It can convert existing name service data directly from the NIS database, and load the data into the LDAP directory. If the NIS database cannot be found, the command would use local name service files.

The following options are available:

-a	Specifies the bind DN used to connect to the LDAP server. If this flag is used, -h and -p must also be used, and data will be loaded directly into the LDAP database.
-d	Specifies the suffix of the LDAP directory that the data should be added under.
-f	Specifies the directory to look for flat files in, or the name of the automount map file. If this flag is not used, nistoldif will look for files in /etc. This flag is required for autoimmunity maps.
-h	Specifies the host name that is running the LDAP server. If this flag is used, -a and -p must also be used, and data will be written directly to the LDAP server. This flag will be ignored for automount data.
-k	Specifies the SSL key path. If this flag is used, -w must also be used.
-m	Specifies the automount map on the LDAP server.
-n	Specifies the port to connect to the LDAP server on. If this flag is used, -a, -h, and -p must also be used; if it is not used, the default LDAP port is used.
-p	Specifies the password used to connect to the LDAP server. If this flag is used, -a and -h must also be used, and data will be written directly to the LDAP server.
-s	Specifies a set of maps to be written to the server. This flag should be followed by a list of letters representing the maps that should be migrated. If this flag is not used, all maps will be migrated. The letters are a for automount, e for netgroup, g for group, h for hosts, n for networks, p for protocols, r for rpc, s for services, and u for passwd.
-S	Specifies the LDAP schema to use for users and groups. This can be RFC2307 or RFC2307AIX. RFC2307AIX gives extended AIX schema support. If this flag is not used, RFC2307 is the default.

-w	Specifies the SSL password. If this flag is used, -k must also be used.
-y	Specifies the NIS domain to read maps from. If this flag is not used, the default domain will be used.

Example D-1 shows converting local name service files into the LDAP directory. The -S option is used to specify the LDAP schema use for the conversion. The -a, -h, and -p options are also used to import the data into the LDAP database directly. If the -a, -h, and -p are not specified, the command would export the data into LDIF format.

Example: D-1 nistoldif migrate all NIS local files

```
#This example shows nistoldif to migrate NIS data to LDAP according to
RFC2307AIX schema
# nistoldif -d "dc=example,dc=com" -a “cn=admin” -h ldapserver -p password -S
RFC2307AIX

#This example shows to convert nis data to ldap rfc2307 schema
# nistoldif -d "dc=example,dc=com" -a “cn=admin” -h ldapserver -p password -S
RFC2307
```

The -s option can be used to convert a particular NIS data file into LDIF. The -f allows the user to specify an alternate location of the NIS data file. This is useful to create a file with only a subset of data.

If the -a option is not used, then the command will convert the NIS data into LDIF format. The user can then save the output to a file, and use the **ldapmodify** or **ldapadd** command to load the LDIF data into LDAP directory.

In Example D-2 the *-s a* option was used to convert only the automount entries. The -f option was also used to specify the location of the automount files.

Example: D-2 Convert local auto_home files into LDIF

```
# cat /etc/auto_home
user1 bs01b07:/u/user1
user2 bs01b02:/u/user2
#
# nistoldif -s a -f /etc/auto_home -d "dc=example,dc=com"
dn: automountMapName=auto_home,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto_home

dn: automountKey=user1,automountMapName=auto_home,dc=example,dc=com
objectClass: automount
objectClass: top
```

```
automountKey: user1
automountInformation: bs01b07:/u/user1

dn: automountKey=user2,automountMapName=auto_home,dc=example,dc=com
objectClass: automount
objectClass: top
automountKey: user2
automountInformation: bs01b02:/u/user2
```

sectoldif - AIX local security to LDAP data migration

The **sectoldif** command is a migration tool provided with AIX for migrating local security data to LDAP. It can convert data from the local system directly to one of the three schemas: RFC2307, RFC2307AIX, or the AIX schema. The data will be written to standard output. The LDIF file can then be imported to the LDAP server using the **ldapadd** command.

The syntax of the **sectoldif** command is:

```
sectoldif -d baseDN [ -S schematype ] [ -u username ]
```

-d baseDN	Specifies the base DN to use for the DNs to be used in the LDIF file for all entries.
-S schematype	Defines which schema should be used. The default is the AIX schema.
-u username	If that option is given the command will only print the information for the user specified.

The **sectoldif** command will export the following from the local files:

- Users
- Groups
- aixid and aixbaseid

This is shown in Example D-3 with the RFC2307 schema.

The **sectoldif** command constructs the DNs based on the -d “baseDN” and putting the values specified in /etc/security/ldap/sectoldif.cfg in front of that base.

Example: D-3 Exporting data from the local system with sectoldif

```
# sectoldif -d "dc=example,dc=com" -S RFC2307
dn: ou=People,dc=example,dc=com
ou: People
objectClass: organizationalUnit
```

```
dn: uid=default,ou=People,dc=example,dc=com
uid: default
objectClass: posixaccount
objectClass: shadowaccount
objectClass: account
cn: default
uidnumber: -1
gidnumber: -1
userpassword: {crypt}*
homedirectory: *
shadowwarning: 0
shadowmin: 0
shadowmax: 0
shadowexpire: -1
   .
   .
   .
dn: ou=Groups,dc=example,dc=com
ou: Groups
objectClass: organizationalUnit

dn: cn=system,ou=Groups,dc=example,dc=com
cn: system
objectClass: posixgroup
gidnumber: 0

dn: cn=staff,ou=Groups,dc=example,dc=com
cn: staff
objectClass: posixgroup
gidnumber: 1
memberuid: ipsec
memberuid: sshd
memberuid: idsldap
memberuid: ldapdb2
memberuid: ldap
   .
   .
   .
dn: cn=aixid,ou=System,dc=example,dc=com
cn: aixid
objectClass: aixadmin
aixadminuserid: 8
aixuserid: 232
aixadmingroupid: 15
aixgroupid: 226

dn: cn=aixbaseid,ou=System,dc=example,dc=com
cn: aixbaseid
objectClass: aixadmin
```

```
aixadmingroupid: 1
aixadminuserid: 1
aixuserid: 200
aixgroupid: 200
```

secldifconv - Converting between schemas

AIX 5L Version 5.3 now supports three schemas: RFC2307, RFC2307AIX, and AIX. The **secldifconv** command provides the ability to migrate data in the LDIF format to another format. The conversion can be done from any of the three supported schemas to any other. Care should be taken, considering that when migrating from a schema to another data might get lost (AIX schema to RFC2307) or that data may be missing because it does not exist in the source schema (RFC2307 to RFC2307AIX).

The syntax of the command is:

```
secldifconv [-R load_module ] -S schematype -i inputFile [ -r ]
```

-R load_module	This defines the load module to get the passwords from if they are missing in the LDIF file.
-S schematype	This is the schema used to create the output.
-i inputFile	This is the file to read the source LDIF information from.
-r	This specifies to remove any attributes that are not defined in the specified schema type.

The -r flag changes the behavior of the command if a conversion is done from a schema with more attributes (AIX and RFC2307AIX schema) to a schema with less attributes (RFC2307 schema).

- If not specified, attributes that are defined in the source schema but not in the target schema and that are not converted are passed to the output file with no change.
- If specified, these attributes are not written to the output file.

The behavior of the -r flag is shown in Example D-4.

Example: D-4 Converting from RFC2307AIX to RFC2307

```
# ldapsearch -h bs01b03 -D cn=admin -w its0g00d -L -b "dc=example,dc=com"
objectclass=*  > rfc2307aix.ldif
# grep -p user500 rfc2307aix.ldif
dn: uid=user500,ou=People,dc=example,dc=com
uid: user500
objectClass: aixauxaccount
```

```
objectClass: shadowaccount
objectClass: posixaccount
objectClass: account
objectClass: ibm-securityidentities
objectClass: top
cn: user500
uidnumber: 219
gidnumber: 500
homedirectory: /home/user500
loginshell: /usr/bin/ksh
isadministrator: false
shadowlastchange: 13042
userpassword: {crypt}gr8dG1t3ZRnQI
passwordchar: !
hostlastlogin: localhost
ixtimelastlogin: 1126904896
terminallastlogin: /dev/pts/1
unsuccessfullogincount: 0
hostsallowedlogin: 9.3.5.140

dn: cn=grp500,ou=Groups,dc=example,dc=com
cn: grp500
objectClass: aixauxgroup
objectClass: posixgroup
objectClass: top
gidnumber: 500
memberuid: user500
isadministrator: false

# secldifconv -i rfc2307aix.ldif -S RFC2307 -r | grep -p user500
uid: user500
objectClass: top
objectclass: posixaccount
objectclass: shadowaccount
objectclass: account
uidnumber: 219
gidnumber: 500
homedirectory: /home/user500
loginshell: /usr/bin/ksh
shadowlastchange: 13042
userpassword: {crypt}gr8dG1t3ZRnQI
cn: user500

cn: grp500
objectClass: top
objectclass: posixgroup
gidnumber: 500
memberuid: user500
```

E

AIX LDAP quick reference

The purpose of this appendix is to provide a quick reference sheet to the AIX LDAP services as they are used for user authentication and identification. The following sections are included:

LDAP configuration files

This section contains a list of files that are associated with user management, either in base AIX file-based user management or with LDAP-based user management.

LDAP client configuration files

The following files are used for configuring the secldapclntd daemon or are used by the daemon to configure the way the client daemon interacts with the LDAP server.

- /etc/security/ldap/ldap.cfg

  LDAP client configuration file. Read by secldapclntd when it starts up or is refreshed. Contains information for binding to the LDAP servers. This file is customized by mksecldap -c.

- /etc/security/ldap/sectoldif.cfg

  Configuration file for determining the distinguished names used to create LDIF files with sectoldif.

- /etc/security/ldap/2307user.map

  The user map files 2307user.map, 2307aixuser.map, and aixuser.map are used by the secldapclntd daemon to map the LDAP user attributes for the RFC2307, RFC2307AIX, and AIX schemas to the AIX security user attributes.

- /etc/security/ldap/2307group.map

  The group map files 2307group.map, 2307aixgroup.map, and aixgroup.map are used by the secldapclntd daemon to map the LDAP group attributes for the RFC2307, RFC2307AIX, and AIX schemas to the AIX security user attributes.

- /etc/security/ldap/id.map

  This defines the mapping for the LDAP aixadmin class used with the AIX schema.

- /etc/security/ldap/proxy.ldif.template

  This is the template to be used when creating a proxy user on the LDAP server for secldapclntd to use for bind DN. This defines the Access Control Lists (ACLs) for access to the user information data trees.

- /etc/security/ldap/sectoldif.cfg

  # LDAP user/group directory information tree configuration table used by the **`mksecldap`**, **`sectoldif`** and **`nistoldif`** commands to determine the name of various data trees that will be created in LDAP.

- /etc/security/ldap/ldapid.ldif.template

  This is used to add the data directory tree to the LDAP server to control the minimum group and user IDs to be used when creating new users.
- /etc/security/ldap/nisSchema.ldif

  This is the LDIF file containing the RFC2307AIX object classes.

IBM Tivoli Directory Server configuration files

- /usr/ldap/etc/ibmslapd.conf

  This is the IBM Tivoli Directory Server configuration file.

AIX security configuration files

- /etc/security/user

  This is the user configuration file that determines the type of authentication based on the SYSTEM and registry attributes for the user.
- /etc/passwd

  This defines the user's UID and primary GID as well as the user's shell and home directory for local users. Entries are not required in this file for LDAP-based users.
- /etc/security/passwd

  This contains the user encrypted password and password flags for local (files-based) users. Entries are not required in this file for LDAP-based users.
- /etc/group

  This defines the group names, GID, and userlist for file-based user groups.
- /etc/security/group

  This defines whether a group is an administrative group and whether the users are administrative users for that group.
- /usr/lib/security/methods.cfg

  This is the loadable authentication module (LAM) definition file. This file maps the LDAP LAM from the /etc/security user SYSTEM attribute to the LDAP LAM module in /usr/lib/security.
- /etc/inittab

  This starts the secldapclntd client daemon and the ibmslapd LDAP server.

- /usr/lib/security/mkuser.default

  This defines the default pgrp, groups, shell, and home directory for users when using the mkuser command for both normal and administrative users.
- /etc/security/.ids

  This specifies the next UID and GID to use when creating a new user or group with the **mkuser** or **mkgroup** command.
- /etc/netsvc.conf

  This determines the search order for host naming services. An example is to look in /etc/hosts, first use host=local, bind8.
- /etc/irs.conf

  This is used to enable an AIX client to use LDAP to look up netgroup information (for example, netgroup nis_ldap).
- /etc/exports

  This is used to specify NFS exported file systems on the server used for HOME directories with automount.
- /etc/hosts.equiv

  This is used in configuring netgroup support.
- /etc/security/lastlog

  This is the last location that a user logged in from and the date and time of the last unsuccessful login and the unsuccessful login count.
- /etc/security/failedlogin

  This contains a record of unsuccessful login attempts stored in utmp format and used by the accounting and audit subsystems.
- /etc/security/environ

  This contains the environment attributes that can be defined for individual users. Global environment variables are defined in /etc/environment.
- /etc/security/limits

  This contains the process resource limits for users.

NIS map files

The NIS map files are:

- /etc/passwd

  This file is used to designate (using a plus sign at the end of the file) that users not listed in the file will use NIS type of authentication if the SYSTEM parameter is set to compat.

- /etc/auto_home

  This is NIS home directory automount information.

LDAP daemons

The daemons associated with LDAP user management include the following:

- secldapclntd

  AIX Security LDAP client daemon
- ibmslapd

  IBM Tivoli Directory Server daemon
- slapd

  Directory Server daemon on Solaris and OpenLDAP
- idsdiradm

  The Administrative Daemon for controlling LDAP

LDAP integration commands

This section contains a list of AIX commands and shell scripts that are used in conjunction with integrating AIX and an LDAP server:

- **`mksecldap`**

  The mksecldap shell script is used to configure both IBM Tivoli Directory LDAP server and the AIX LDAP client for user management.
- **`sectoldif`**

  This is a tool to convert files-based users to an LDIF file format for entry into an LDAP server. The tool can create files that conform to all three standard schema supported by AIX 5.2 and 5.3 by using the -S <schema> flag.
- **`secldifconf`**

  This tool is used to convert users from one schema format to another. The primary purpose of this is to provide an easy migration path for users defined with the AIX schema to move to the RFC2307 or RFC2307AIX schema once the operating system has been upgraded from AIX 4.3.3 or 5.1 to AIX 5.3.
- **`nistoldif`**

  This tool is used to convert NIS-based maps to an LDIF file format for entry into an LDAP server. The tool can create files that conform to all three standard schema supported by AIX 5.2 and 5.3 by using the -S <schema> flag.

- **`ldapchangepwd`**

  The ldapchangepwd program is an LDAP command-line client that can be used to change the LDAP-based password for an LDAP entry. This should not be confused with the AIX **`passwd`** command, which can also be used to change an AIX user's password stored on the LDAP server.

- **`ldapsearch`**

  The **`ldapsearch`** command is an LDAP command-line client that can be used to extract information from an LDAP directory by specifying a server, a base DN, and a search filter. The flags for the AIX **`ldapsearch`** client are different from those from OpenLDAP. The information will be returned in LDIF format.

- **`ldapadd`**

  The **`ldapadd`** command is an LDAP command-line client that can be used to import or add a directory from an LDIF file into an LDAP directory.

- **`ldapdelete`**

  The **`ldapdelete`** command is an LDAP command-line client that can be used to remove information from an LDAP directory.

- **`ldapdiff`**

  This is the LDAP replica synchronization tool used when running replica servers.

- **`ldapexop`**

  The ldapexop utility is an LDAP command-line client that provides the capability to bind to a directory server and issue a single extended operation along with any data that makes up the extended operation value.

- **`ibmdirctl`**

  This is the administration daemon control program. The administration daemon idsdiradm must be running to use this.

- **`ldapmodify`**

  This is a command-line interface to the ldap_modify, ldap_add, ldap_delete, and ldap_modrdn application programming interfaces that can be used to modify or add entries to the LDAP server from an LDIF file.

- **`ldapmodrdn`**

  This is a command-line interface to ldap_modrdn that can be used to modify Relative Distinguished Name (RDN) entries in the LDAP directory using the LDIF file format.

- **`ldaptrace`**

  This is the administration tracing utility that can be used in conjunction with IBM support to solve specific problems.

- **`lsldap`**

  This displays naming service objects from the configured LDAP directory server.

- **`gskit. gsk7ikm`**

  The Global Security toolkit is a required component for Secure Socket Layer (SSL) enablement. It is used to generate keys and certificates.

- **`ibmmultiadd`**

  This is an undocumented multi-threaded LDAP add utility.

- **`lsuser -R LDAP ALL`**

  This lists all user information stored in the LDAP server when the running system is an AIX LDAP client.

- **`mkuser -R LDAP`**

  This is run with the user's name, SYSTEM, and registry objects to add a user to the LDAP directory that can be used by the AIX client. This runs on an AIX LDAP client.

- **`passwd -R LDAP`**

  This is used by root to change the password of an LDAP-based user if the user's SYSTEM attribute is not set to LDAP. When passwd is run by a user with the SYSTEM attribute set to LDAP, there is no need to specify the -R LDAP flag for that user to change her own password.

- **`secldapclntd`**

  This is the LDAP user management and caching client daemon. This daemon must be running on the client for LDAP authentication and user management to work. The secldapclntd daemon reads the /etc/security/ldap/ldap.cfg file when it starts or is refreshed.

- **`flush-secldapclntd`**

  This flushes the cache for the secldapclntd daemon process causing the daemon to read all new information from the LDAP server rather than the local cached copy.

- **`ls-secldapclntd`**

  This lists the status of the secldapclntd daemon including the server that it tis talking to, the DNs being used, the cache sizes, the number of threads, and the object classes used in the LDAP server.

- **`restart-secldapclntd`**

  This is used to stop the current secldapclntd daemon process and restart it. The cache will be cleared and new configuration parameters will be in effect.

- **`start-secldapclntd`**

  This is the script to start the secldapclntd LDAP client daemon.

- **`stop-secldapclntd`**

  This is the script to terminate the secldapclntd LDAP client daemon.

- **`mkprtldap`**

  This is the script to set up directory-based printing on AIX.

- **`mkprojldap`**

  The script configures LDAP server and client machines for handling the advanced accounting subsystem data. To support advanced accounting on the LDAP server, the LDAP schema for advanced accounting must be uploaded to the server. This has nothing that is unique to LDAP user management, and additional information is beyond the scope of this book.

- **`mkldap`**

  This is the high-level directory setup command called by individual subsystem setup commands like **`mkprtldap`**. This command is not normally run the by the system administrator directly.

LDAP secldapclntd configuration options

The AIX LDAP security client secldapclntd reads the configuration options from /etc/security/ldap/ldap.cfg. This file is self documenting to a large extent, but the available options are reviewed in this section as a pat of the quick reference.

- ldapservers

  This is the separated list of LDAP servers this client talks to. The servers are listed in a priority order. If the first server is not available, authentication will fail over to the next server. Once the first server is available, the client will return to using the original server (for example, ldapservers:bs01,bs02,bso3).

- binddn

  This is the LDAP server bind distinguished name (DN) used by secldapclntd when talking to the server (for example, binddn:cn=admin).

- bindpwd

  This is the LDAP server bind DN password used by secldapclntd to bind to the LDAP server. Prior to AIX 5.3 this is a clear text password, but with AIX

5.3 this password can be encrypted with DES (for example, bindpwd:{DES}C718666...).

- authtype

  This is the authentication type. Valid values are unix_auth and ldap_auth. With unix_auth, the encrypted password is retrieved and authentication is done on the client. With ldap_auth, the password is sent to the LDAP server to bind and thus the authentication is done remotely at the LDAP server (for example, authtype:unix_auth).

- useSSL

  A boolean value of either yes or no specifies whether to use SSL to communicate with the LDAP server. (Note: ou needs a SSL key and a password to the key to enable this (for example, useSSL: yes).

- ldapsslkeyf

  This is the SSL key file path name (for example, ldapsslkeyf:/etc/security/ldap/key.kdb).

- ldapsslkeypwd

  This is the SSL key file password (for example, ldapsslkeypwd:{DES}61F48B79A17...).

- useKRB5

  This determines whether the daemon will do the initial bind to the LDAP server using Kerbreos. The options are yes and no. If yes is specified, then the Kerberos principle must also be specified (for example, useKRB5:yes).

- krbprincipal

  This is the Kerberos principal used to bind to the server (for example, krbprincipal:principal).

- krbkeypath

  This is the path to the Kerberos keytab. It defaults to /etc/security/ldap/krb5.keytab (for example, krbkeypath:/etc/security/ldap/krb5.keytab).

- krbcmddir

  This is the directory that contains the Kerberos **kinit** command. The default is /usr/krb5/bin/ (for example, krbcmddir:/usr/krb5/bin/).

- userattrmappath

  This is the AIX-LDAP attribute map path for user attributes (for example, userattrmappath:/etc/security/ldap/2307aixuser.map).

- groupattrmappath

  This is the AIX-LDAP attribute map path for group attributes (for example, groupattrmappath:/etc/security/ldap/2307aixgroup.map).

- idattrmappath

  This is the AIX-LDAP attribute map path for admin ID attributes (for example, idattrmappath:/etc/security/ldap/aixid.map).

- userbasedn

  This is the base DN where the user data is stored in the LDAP server. For example, if user foo's DN is uid=foo,ou=people,cn=aixdata, then the user base DN is ou=people,cn=aixdata (for example, userbasedn:ou=People,dc=example,dc=com).

- groupbasedn

  This is the base DN where the group data is stored in the LDAP server (for example, groupbasedn:ou=Groups,dc=example,dc=com).

- idbasedn

  This is the base DN where the admin user ID data is stored in the LDAP server (for example, idbasedn:cn=aixid,ou=System,dc=example,dc=com).

- hostbasedn

  This is the base DN where the host name data is stored in the LDAP server when LDAP is used to replace DNS.

- servicebasedn

  This is the base DN used for storing network services and ports in the LDAP server when using LDAP host name services.

- protocolbasedn

  This is the base DN used for storing network protocols in the LDAP server when using LDAP host naming services.

- networkbasedn

  This is the base DN used for storing network information in the LDAP server.

- netgroupbasedn

  This is the base DN used for storing netgroup information in the LDAP server when using LDAP for NIS type netgroups (for example, netgroupbasedn:ou=netgroup,dc=example,dc=com).

- rpcbasedn

  This is the base DN used for storing remote procedure call information in the LDAP server.

- automountbasedn

  This is the base DN used for storing automount maps in the LDAP server (for example, automountbasedn:ou=automount,cn=aixdata).

- aliasbasedn

  This is the base DN used for storing aliases in the LDAP server.

- bootparambasedn

  This is the base DN used for storing boot parameters in the LDAP server.

- etherbasedn

  This is the base DN used for storing ethers information in the LDAP server.

- userclasses

  LDAP user class definitions to be used by the client (for example, userclasses:account,posixaccount,shadowaccount,...).

- groupclasses

  This is LDAP group class definitions to be used by the client (for example, groupclasses:posixgroup,aixauxgroup).

- ldapversion

  This is the LDAP server version. Valid values are 2 and 3. The default is 3.

- ldapport

  This is the LDAP server port. It defaults to 389 for a non-SSL connection.

- ldapsslport

  This is the LDAP server port. This defaults to 636 for the SSL connection (for example, ldapsslport:636).

- followaliase

  Follow aliases. Valid values are NEVER, SEARCHING, FINDING, and ALWAYS. The default is NEVER.

- usercachesize

  This is the number of user cache entries. Valid values are 100–10000 entries. The default is 1000 (for example, usercachesize: 10000).

- groupcachesize

  This is the number of group cache entries. Valid values are 10–1000 entries. The default is 100 (for example, groupcachesize: 1000).

- cachetimeout

  This is the cache time-out value in seconds. Valid values are 60–60*60 seconds. The default is 300. This is set to 0 to disable caching (for example, cachetimeout: 300).

- heartbeatinterval

  This is the time interval in seconds that the secldapclntd daemon contacts the LDAP server for server status. Valid values are 60–60*60 seconds. The default is 300 (for example, heartbeatinterval: 300).

- numberofthread

  This is the number of threads the secldapclntd daemon uses to process jobs. Valid values are 1–1000. The default is 10 (for example, numberofthread: 20).

- connectionsperserver

  This is the number of LDAP server connections to maintain for each server in the list. When performing LDAP server operations, threads in the daemon will lock one of the connections and release it when finished. If the specified value is greater than numberofthread, then only numberofthread connections will be made. Valid values are 1–100. The default is 10.

- searchmode

  This is the mode used to search for information on the LDAP server. Valid values are ALL and OS and the default is ALL. With OS, only the information specific to AIX required attributes on an entry is returned. Non-OS attributes like telephone number, binary images, and so on are not returned in this mode. ALL returns all attributes of an entry. Note that for performance reasons, use OS only when the user entry has many non-OS required attributes or attributes with a large amount of data such as binary entries to reduce the sorting effort required on the LDAP server.

- defaultentrylocation

  This is the default user attribute entry location. Valid values are LDAP and local, and the default is LDAP. This means that the user named default will be used on the LDAP server to retrieve the default user attributes (for example, defaultentrylocation:local).

- ldaptimeout

  This is the timeout period (seconds) for LDAP client requests to the server. This value determines how long the client will wait or a response from the LDAP server. The valid range is 0–3600 (1 hour). The default is 60 seconds. Set this value to 0 to disable the time out (for example, ldaptimeout:120).

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information about ordering these publications, see “How to get IBM Redbooks” on page 337. Note that some of the documents referenced here may be available in softcopy only.

- *Understanding LDAP - Design and Implementation*, SG24-4986

  `http://www.redbooks.ibm.com/abstracts/sg244986.html?Open`
- *AIX 5L Differences Guide Version 5.3 Edition*, SG24-7463

  `http://www.redbooks.ibm.com/abstracts/sg247463.html?Open`
- Technote: *AIX - Migrating NIS Maps into LDAP*

  `http://www.redbooks.ibm.com/abstracts/tips0123.html?Open`

Online resources

These Web sites and URLs are also relevant as further information sources:

- Complete online AIX documentation

  `http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=/com.ibm.aix.doc/infocenter/base/aixpdfs.htm`
- AIX 5L Version 5.3 Security Guide

  `http://publib.boulder.ibm.com/infocenter/pseries/v5r3/topic/com.ibm.aix.doc/aixbman/security/security.pdf`
- IBM white paper: *Configuring AIX 5L for Kerberos Based Authentication Using Windows Kerberos Service*

  `http://www-03.ibm.com/servers/aix/whitepapers/aix_kerberos2.pdf`
- IBM white paper: *Configuring AIX 5L for Kerberos Based Authentication Using Network Authentication Service*

  `http://www-03.ibm.com/servers/aix/whitepapers/aix_kerberos.pdf`

The AIX development group published three white papers in March 2003 to help document the new features in AIX 5L Version 5.2:

- *Using LDAP for Naming Services in AIX 5L Version*

  http://www-03.ibm.com/servers/aix/whitepapers/ldap_naming.html

- *Configuring an IBM Directory Server for User Authentication and Management in AIX*

  http://www-03.ibm.com/servers/aix/whitepapers/ldap_server.html

- *Configuring an AIX Client System for User Authentication and Management Through LDAP*

  http://www-03.ibm.com/servers/aix/whitepapers/ldap_client.html

Online reference materials provided by Microsoft:

- Microsoft Solution Guide for Windows Security and Directory Services for UNIX

  http://www.microsoft.com/technet/itsolutions/cits/interopmigration/unix/usecdirw/01wsdsu.mspx

- Windows 2003 Server product documentation

  http://www.microsoft.com/windowsserver2003/proddoc/default.mspx

- How To Create an Active Directory Server in Windows Server 2003

  http://support.microsoft.com/default.aspx?scid=kb;en-us;324753

- Windows server schema reference

  http://msdn.microsoft.com/library/default.asp?url=/library/en-us/adschema/adschema/classes_all.asp

- Installing the Active Directory Schema MMC Snap-in

  http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/ServerHelp/8c76ff67-9e9d-4fc7-bfac-ffedee8a04d4.mspx

- Windows Services for UNIX 3.0: Schema Changes for Server for NIS

  http://www.microsoft.com/downloads/details.aspx?FamilyID=ccbb506e-418a-4b93-97c0-9cf1a9a11152&displaylang=en

- How to set a user's password with Ldifde

  http://support.microsoft.com/?kbid=263991

- TLS/SSL Tools and Settings

  http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/TechRef/3f98fdd9-ed64-49f7-9c20-a2d4581dfbea.mspx

- Advanced Certificate Enrollment and Management

  `http://www.microsoft.com/technet/prodtechnol/windowsserver2003/technologies/security/advcert.mspx`

- How to enable LDAP over SSL with a third-party certification authority

  `http://support.microsoft.com/?kbid=321051`

Other publications

Internet RFCs

Please go to the following URL for access to RFCs:

`http://www.ietf.org/rfc`

- 4120 The Kerberos Network Authentication Service (V5). C. Neuman, T. Yu, S. Hartman, K. Raeburn. July 2005. (Format: TXT=340314 bytes) (Obsoletes RFC1510) (Status: PROPOSED STANDARD)
- 4121 The Kerberos Version 5 Generic Security Service Application Program Interface (GSS-API) Mechanism: Version 2. L. Zhu, K. Jaganathan, S. Hartman. July 2005. (Format: TXT=340314 bytes) (Updates RFC1964) (Status: PROPOSED STANDARD)
- 3244 Microsoft Windows 2000 Kerberos Change Password and Set Password Protocols. M. Swift, J. Trostle, J. Brezak. February 2002. (Format: TXT=13334 bytes) (Status: INFORMATIONAL)
- 3244 The String Representation of LDAP Search Filters
- 2849 The LDAP Data Interchange Format (LDIF) - Technical Specification. G. Good. June 2000. (Format: TXT=26017 bytes) (Status: PROPOSED STANDARD)
- 2256 - A Summary of the X.500(96) User Schema for use with LDAPv3

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

`ibm.com/support`

IBM Global Services

`ibm.com/services`

Index

IBM

Redbooks

Integrating AIX into Heterogenous LDAP Environments

Integrating AIX into Heterogeneous LDAP Environments

Describes the latest integration methods based on AIX 5L Version 5.3 with the 5300-03 Recommended Maintenance Package

Provides detailed guidance for planning and implementation

Includes integration scenarios for IBM Tivoli Directory Server, Sun ONE Directory Server, Microsoft Active Directory Server, and OpenLDAP

This IBM Redbook is a technical planning reference for IT organizations that are adding AIX 5L clients to an existing LDAP authentication and user management environment. It presents integration scenarios for the AIX 5L LDAP client with IBM Tivoli Directory Server, the Sun ONE Directory Server, and Microsoft Active Directory.

The sample integration scenarios can be used as a road map for administrators migrating AIX 5L users from traditional local file authentication to an LDAP server, or for adding new AIX 5L boxes to an environment where there are users already defined in the aforementioned directory server products.

- Part 1 introduces the book and provide a history of AIX and LDAP integration and a detailed discussion of LDAP migration planning topics.
- Part 2 starts details AIX 5L LDAP client installation, integration, and configuration topics that apply to all of the following specific integration scenarios. It then describes integration scenarios for four LDAP server environments.
- Part 3 provides background and technical reference information supporting the integration scenarios we present.

SG24-7165-00 ISBN 0738497401