



# Configuring AIX 5L for Kerberos Based Authentication Using Network Authentication Service

*February 1, 2006*

**Ufuk Çelikkan**  
**IBM Corporation**

# CONTENTS

|  |           |
|--|-----------|
| <b>CONTENTS</b>  | <b>2</b>  |
| <b>1 OVERVIEW</b>  | <b>5</b>  |
| <b>2 AIX USER AUTHENTICATION AND IDENTIFICATION</b>  | <b>5</b>  |
| 2.1 Loadable Authentication/Identification Mechanism History                                   | 5         |
| 2.2 "registry" and "SYSTEM" Attributes   | 7         |
| <b>3 KERBEROS AND LDAP</b>   | <b>8</b>  |
| 3.1 Kerberos Tickets   | 8         |
| 3.2 KDC  | 8         |
| 3.3 Kerberos Database  | 9         |
| 3.4 LDAP   | 9         |
| <b>4 AUTHENTICATING TO AIX USING NETWORK AUTHENTICATION SERVICE</b>                            | <b>9</b>  |
| <b>5 INSTALLING AND CONFIGURING THE KERBEROS SERVER ON AIX</b>                                 | <b>10</b> |
| 5.1 Configuring the Network Authentication Service Server with Legacy Database Storage         | 13        |
| 5.2 Configuring Network Authentication Service Server with LDAP Storage                        | 16        |
| 5.2.1 Configuring LDAP   | 17        |
| 5.2.1.1 Configuration of LDAP using mksecldap  | 18        |
| 5.2.1.2 Configuration of LDAP using the conventional method                                    | 19        |
| 5.2.2 Loading Kerberos Schemas   | 22        |
| 5.2.3 Configuration of Kerberos Server   | 24        |
| <b>6 INSTALLING AND CONFIGURING THE KERBEROS CLIENT ON AIX FOR INTEGRATED LOGIN USING KRB5</b> | <b>26</b> |
| 6.1 Configuring Kerberos Integrated Login  | 26        |
| 6.2 Eliminating Kadmind Daemon Dependency during Authentication                                | 29        |

|          |   |           |
|----------|---|-----------|
| <b>7</b> | <b>KRB5 AUTHENTICATION LOAD MODULE QUESTIONS AND TROUBLESHOOTING INFORMATION</b>              | <b>30</b> |
| 7.1      | How do I Create AIX Kerberos Authenticated Users  | 30        |
| 7.2      | How do I Remove a Kerberos Authenticated User   | 31        |
| 7.3      | How do I Change the Password of a Kerberos Authenticated User                                 | 31        |
| 7.4      | What are AIX Kerberos Extended Attributes   | 31        |
| 7.5      | How do I list the AIX Kerberos Extended Attributes  | 34        |
| 7.6      | How do I modify the AIX Kerberos Extended Attributes  | 35        |
| 7.7      | Listing all the Users Defined in KRB5files  | 36        |
| 7.8      | How do I convert an AIX User to a Kerberos Authenticated User                                 | 37        |
| 7.9      | What do I do if the Password is Forgotten   | 39        |
| 7.10     | No TGT after a Successful Login When “needchange” Flag is Set                                 | 39        |
| 7.11     | AIX Does Not Accept My Password   | 39        |
| 7.12     | Can a Kerberos-Authenticated User Become Authenticated Using Only Standard AIX Authentication | 40        |
| 7.13     | How to Change Client Kadmind Port   | 41        |
| 7.14     | How to Destroy Kerberos Credentials   | 41        |
| 7.15     | Changing Ticket Life Time on KDC  | 42        |
| 7.16     | What will Happen if Kadmind Daemon is Not Available   | 42        |
| 7.17     | Configuring AIX for Kerberos Integrated Login with LDAP AIX User/Group Management.            | 43        |
| 7.18     | How to Use Kerberized r-cmds After a Successful Login   | 43        |
| <b>8</b> | <b>REFERENCES</b>   | <b>48</b> |
|          | <b>APPENDIX A</b>   | <b>49</b> |
|          | <b>APPENDIX B</b>   | <b>51</b> |

|  |    |
|--|----|
| Figure 1: Loadable identification and authentication.....  | 7  |
| Figure 2: Integrated Login Configuration Scenarios. ....   | 12 |
| Figure 3: Sample configuration for Kerberized r-cmds. .... | 44 |

# 1 Overview

Kerberos is a third party authentication system that originated at MIT as part of Project Athena [1]. This document describes the use of Kerberos as an alternative authentication mechanism to AIX® (We shall use the terms AIX and AIX 5L interchangeably.). The loadable identification and authentication framework of AIX naturally lends itself to the use of Kerberos. Kerberos technology combined with LDAP user/group management provides a robust, centralized and scalable authentication and identification mechanism for AIX.

## 2 AIX User Authentication and Identification

Authentication is the process of proving the identity of a user, device or other entity in a computer system. AIX standard authentication mechanism authenticates users using a one-way hash function called `crypt()` which is based on a modified DES algorithm. `crypt()` uses user password to calculate a hash. This hash is stored on the system. When a user tries to login, the hash is recalculated using the entered password and compared against the one maintained by the system. Authentication applications on AIX do not require any change to alternatively perform Kerberos authentication as it is woven into the fabric of the AIX security subsystem. By utilizing AIX loadable identification and authentication framework, the system directs authentication requests to use Kerberos instead of calling `crypt()`.

User identification is the process that enables recognition of an entity by a system, generally by the use of unique machine-readable user names. Historically, in UNIX environments the identification is performed using the data stored in ASCII text files and managed by standard interfaces defined by POSIX. Any UNIX system that claims compliance to POSIX Single UNIX specification has to provide these interfaces. The AIX loadable identification and authentication framework allows the identification information to be stored in additional domains like LDAP and accessed by calling the same standard interfaces. Applications that use these interfaces need not be aware of where this information is stored.

### 2.1 Loadable Authentication/Identification Mechanism History

AIX has supported the loadable identification and authentication framework since Version 4.1. AIX services that previously implemented their own authentication mechanism were modified to use this common set of authentication interfaces. These interfaces hide details related to authentication from other parts of the operating system. Any module that implements this interface can be used to perform authentication and/or identification. Authentication functions include password verification and modification. Identification functions include storage, retrieval and modification of user/group account information.

The framework defines a common interface and modules that implement this interface can be plugged into the framework to provide support to various technologies used for identification and/or authentication such as Kerberos or LDAP. When AIX needs to access one of these mechanisms it loads to necessary module and that module in turn interacts with the mechanism. Since each of these load modules have interfaces defined by this framework, it is easy to add a new module. The first load module shipped with AIX was the DCE load module. With AIX V4.3.3 the framework was extended to a more general one to exploit LDAP. The extended framework is designed to support data storage and retrieval mechanisms. The modified framework introduced new interfaces for getting and setting user and group account information. The **LDAP** load module has been implemented based on this extended interface.

AIX 5L™ V5.1 introduced compound load modules. A compound load module is a combination of an authentication load module and a database load module. The authentication half of the compound load module provides authentication services and the database side provides AIX identification services. An authentication only module is required to implement only authentication interfaces and a database only module is required to implement only account management interfaces. Some modules implement both interfaces. Such a module can be used as either the database or authentication side of a compound load module or as stand-alone module.

AIX provides an authentication only Kerberos module, **KRB5**, for use in the authentication half of a compound load module. The KRB5 load module enables Kerberos as an alternative authentication mechanism. AIX 5L V5.1 also introduced a pseudo load module called BUILTIN which provides access to the same AIX identification and authentication functionality via the loadable module paradigm. The idea behind the BUILTIN load module is to provide the database half of a compound load module which could be combined with KRB5 (or with any other authentication only load module) to provide a compound load module. The BUILTIN module provides legacy identification functionality (legacy functionality refers to all files being on the system). **LDAP** load module could also be used with KRB5 as the database side, to provide a compound load module.

When the KRB5 load module is combined with a database load module, an AIX system administrator can manage Kerberos authenticated users transparently using the existing user administration commands. This transparency allows the manipulation of Kerberos security registry (Kerberos database) through the AIX library calls and commands. For instance the existing mkuser command can now create a Kerberos principal in the Kerberos security registry associated with an AIX id.

Starting with AIX 5L V5.2, a second Kerberos authentication module, **KRB5A**, is provided for the purpose of authenticating against Microsoft® Active Directory 2000. For further information on the KRB5A load module see the whitepaper titled “Configuring AIX 5L for Kerberos Based Authentication using KRB5A”[13].

The following figure gives a simplified view of loadable identification and authentication framework of AIX. It demonstrates the authentication flow of the su command.

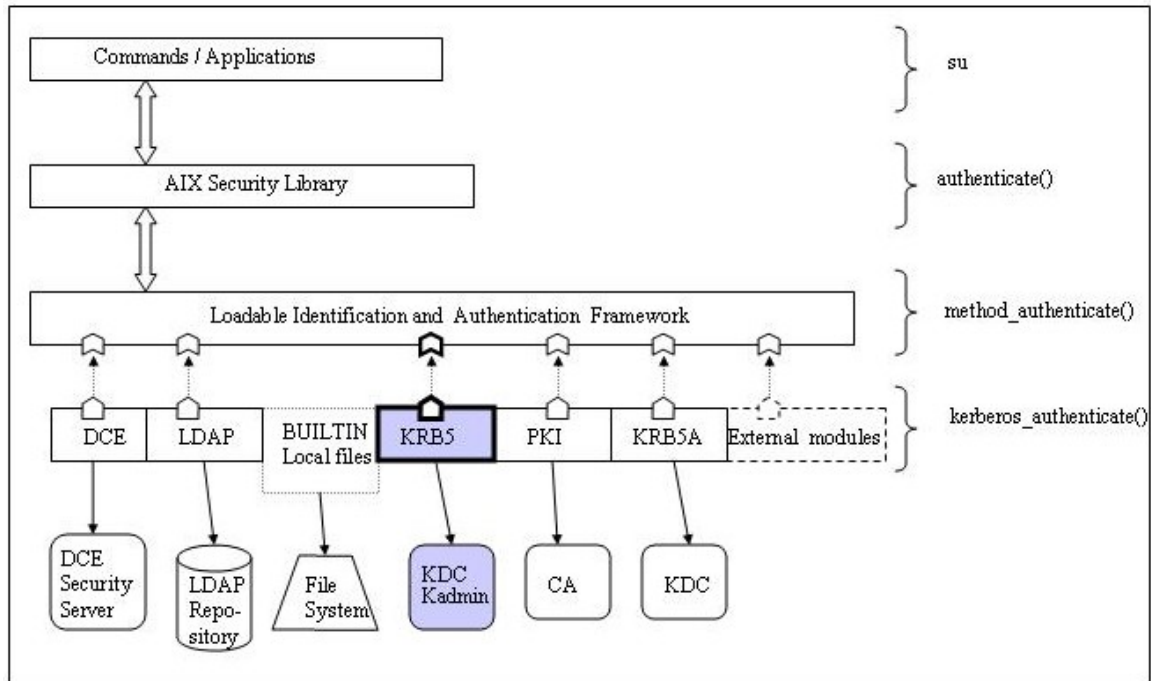


Figure 1: Loadable identification and authentication

## 2.2 "registry" and "SYSTEM" Attributes

AIX security subsystem directs authentication and identification requests to the proper method by using two attributes. These attributes are “registry” and “SYSTEM”. The registry attribute specifies where the user or group identification information is stored and administrated. The SYSTEM attribute controls which methods will be used and how the methods result will affect the overall authentication. **registry** and **SYSTEM** attributes play an important role in AIX user/group account management and authentication. Every user on AIX has a value for the registry and SYSTEM attribute. Groups only have registry values.

By default users are defined in the "files" registry. This means that user account and group information is stored in the flat-ASCII files. The introduction of loadable identification and authentication modules allows definition of users in other registries as well. For example, one such registry, LDAP, enables user and group definitions to be stored in an LDAP repository. When LDAP is used, there is no need for user and group entries in the flat-ASCII files with the exception of the SYSTEM and registry attributes. These two attributes are always stored on the local host file system in `/etc/security/user` file. When a compound load module is used for user administration, the database half determines where AIX user/group account information is administrated and the authentication half describes the authentication and password related administration. The

authentication half may also describe authentication specific account administration by implementing certain interfaces of the framework.

The SYSTEM attribute allows the system administrator to specify to a fine granularity which method (or methods) a user must successfully authenticate to in order to gain access to the system. The well-known method tokens are **compat** and **DCE**. **compat** is the system default. SYSTEM=compat tells the system to use local authentication (crypt()), whereas SYSTEM=DCE results in a DCE authentication flow. The value of the SYSTEM attribute allows for interpretation of a logical grammar. One can specify two or more methods and combine them with logical constructors 'AND' and 'OR'. For instance "SYSTEM=DCE OR compat" indicates that the user is allowed to login if either DCE or crypt() authentication succeeds in this given order.

In a similar fashion a system administrator can use a compound load module name for the SYSTEM attribute. For instance, when SYSTEM attribute is set to "SYSTEM=KRB5files OR compat", the AIX host will first try a Kerberos flow for authentication and if it fails then it will try standard AIX authentication. A brief introduction to SYSTEM attribute values can be found in APPENDIX A.

## 3 Kerberos and LDAP

Kerberos is a network authentication service. It provides a means of verifying the identities of principals on an open, unprotected network under the assumption that packets traveling along the network can be read, modified, and inserted at will. A Kerberos principal is a unique identity that uses Kerberos authentication services. Kerberos verifies identities without relying on authentication by the host operating system, without basing trust on host addresses and without requiring physical security of all the hosts on the network. Refer to [1] and [2] for a detailed discussion of subject matter. In the following we shall briefly define the three major components of the Kerberos authentication mechanism: Kerberos tickets, KDC and Kerberos database.

### 3.1 Kerberos Tickets

Kerberos tickets are credentials that prove your identity. There are two types of tickets, a ticket granting ticket (TGT) and a service ticket. The TGT serves to validate the requested identity and is obtained after some form of authentication. Much like logging onto a host system, retrieval of your TGT will require something which will verify your identity -- a password or a token such as a smart card or the retina of your eye. Once you have a TGT you may then use your TGT to request service tickets for specific services. The service ticket grants you the permission to use the services. In summary, your TGT proves to the Kerberos server that you are who you say you are and your service ticket is your secure introduction to the service. This two-ticket method provides the "trusted third-party" security of Kerberos.

### 3.2 KDC

The trusted "third-party" or intermediary in Kerberos is called a KDC- Key Distribution Center. It issues all the Kerberos tickets to the clients. Each Kerberos administrative domain or realm has one or more KDCs. One of these KDCs is setup as master KDC and the rest are setup as slave KDCs. The master KDC contains the master copy of the database and propagates it to slave KDCs. Slave KDC's do not provide database administration but offer redundancy and allow clients to acquire tickets when the master KDC is unavailable.

### 3.3 Kerberos Database

The Kerberos database keeps a record of every principal, containing the name, private key and expiration date of the principal along with some administrative information about each principal. This information has historically been stored on the local file system. In recent years, software vendors provided features to alternatively store this information in LDAP. For the purpose of this paper, we shall use *legacy database* to refer to local file system storage.

### 3.4 LDAP

LDAP, Lightweight Directory Access Protocol, is an internet protocol that is used for accessing and updating information stored on directory servers. It was developed as a lightweight alternative to X.500 Directory Access Protocol. The directory stores and organizes data in a hierarchical manner in the form of *entries*. Each entry has a name that uniquely identifies it. This name is called *Distinguished Name*. LDAP is based on the client/server model of distributed computing and uses TCP/IP [4].

IBM Tivoli® Directory Server (ITDS) is a highly scalable reliable and robust implementation of LDAP protocol. It provides centralized information management, remote administration and replication. It uses IBM DB2® as the backing store. In the rest of the paper we shall use the term LDAP and IBM Tivoli Directory Server interchangeably. **LDAP** (in bold) shall refer to the AIX load module that manages user/group information stored on ITDS.

## 4 Authenticating to AIX Using Network Authentication Service

Network Authentication Service is IBM's implementation of MIT Kerberos Version 5. It is designed to interoperate with MIT Kerberos V5 distributions. The terms "Kerberos" and "Network Authentication Service" are used interchangeably in this paper.

Network Authentication Service based Kerberos authentication is provided in AIX through the authentication load modules **KRB5** and **KRB5A**. These modules seamlessly integrate into loadable identification and authentication module framework of AIX. Both

modules perform Kerberos authentication but differ in the way Kerberos principal management is performed.

The KRB5 load module enables a system administrator to manage Kerberos principals using existing AIX user administration commands. In order to use KRB5 module for principal management, the Kerberos server must support kadmin administration protocol. Network Authentication Service provides this support through the **kadmin** daemon.

The KRB5A load module is tailored to an environment where Kerberos principals are stored on a non-AIX system and cannot be managed from AIX by using the Kerberos database interface kadmin. Kerberos principal management is not possible through KRB5A load module and must be performed separately by using Kerberos principal-management tools. These tools may be part of a Kerberos product developed by software vendors or they may be integrated into the OS such as the case in Windows® 2000. The original goal of KRB5A was to provide authentication against Windows 2000 Active Directory server where Kerberos principal management is performed using the Active Directory account management tools and APIs. However, it can easily be used against other compliant KDC's where Kerberos admin interface is not supported.

## 5 Installing and Configuring the Kerberos Server on AIX

Network Authentication Service is shipped on AIX Expansion Pack. To install the server package, install the **krb5.server.rte** fileset. If the machine being configured as a Kerberos server will also be used as a Kerberos client, install the entire **krb5** package.

The following command can be used to install the Network Authentication Service server package:

```
# installp -aqXYgd . krb5.server
```

DCE has its own set of Kerberos client utilities with the same names. To avoid namespace collisions between DCE and Kerberos commands (that is, between the **klist**, **kinit**, and **kdestroy** commands), the Kerberos commands are installed in the **/usr/krb5/bin** and the **/usr/krb5/sbin** directories. It is recommended to add these directories to your **PATH** definition. Otherwise, to execute the Kerberos commands, you must specify fully qualified command path names.

```
# export PATH=$PATH:/usr/krb5/sbin:/usr/krb5/bin
```

Note: Java14 sdk also installs its own kinit and may precede other kinit programs in the PATH environment variable. Change PATH accordingly if Network Authentication Service commands needed to be executed instead of Java14 kinit program.

Network Authentication Service documentation is provided in the **krb5.doc.lang.pdf/html** package, where *lang* represents the supported language.

As described in Section 2, KRB5 load module must be combined with a database module to form a compound load module. The database module serves to store the user and group information. AIX has two database modules available for this purpose: LDAP and BUILTIN. LDAP module is used to access information stored on “LDAP” registry (directory) and BUILTIN module is used access information stored on “files” registry (local file system). The compound load module created is typically named KRB5files or KRB5LDAP to describe that KRB5 is used for authentication and local files or LDAP for the database.

Separately, IBM Network Authentication Service supports storing Kerberos information in either the local file system (Kerberos Legacy database) or LDAP. This leads to 4 possible configurations.

1. KRB5files with Kerberos information stored in Kerberos Legacy database
2. KRB5files with Kerberos information stored in Kerberos LDAP database
3. KRB5LDAP with Kerberos information stored in Kerberos Legacy database
4. KRB5LDAP with Kerberos information stored in Kerberos LDAP database.

When LDAP is chosen as the storage mechanism for storing Kerberos principals or AIX user and group information, LDAP needs to be configured before invoking Kerberos configuration commands. Configuring LDAP for storing Kerberos principals and LDAP for storing AIX user and group information are two distinct steps. Once LDAP configuration is done, mkkrb5srv command can be used is used to configure Kerberos servers. For information on how to configure Kerberos with an LDAP database backend, skip to section 5.2. The following figure shows the possible configuration scenarios.

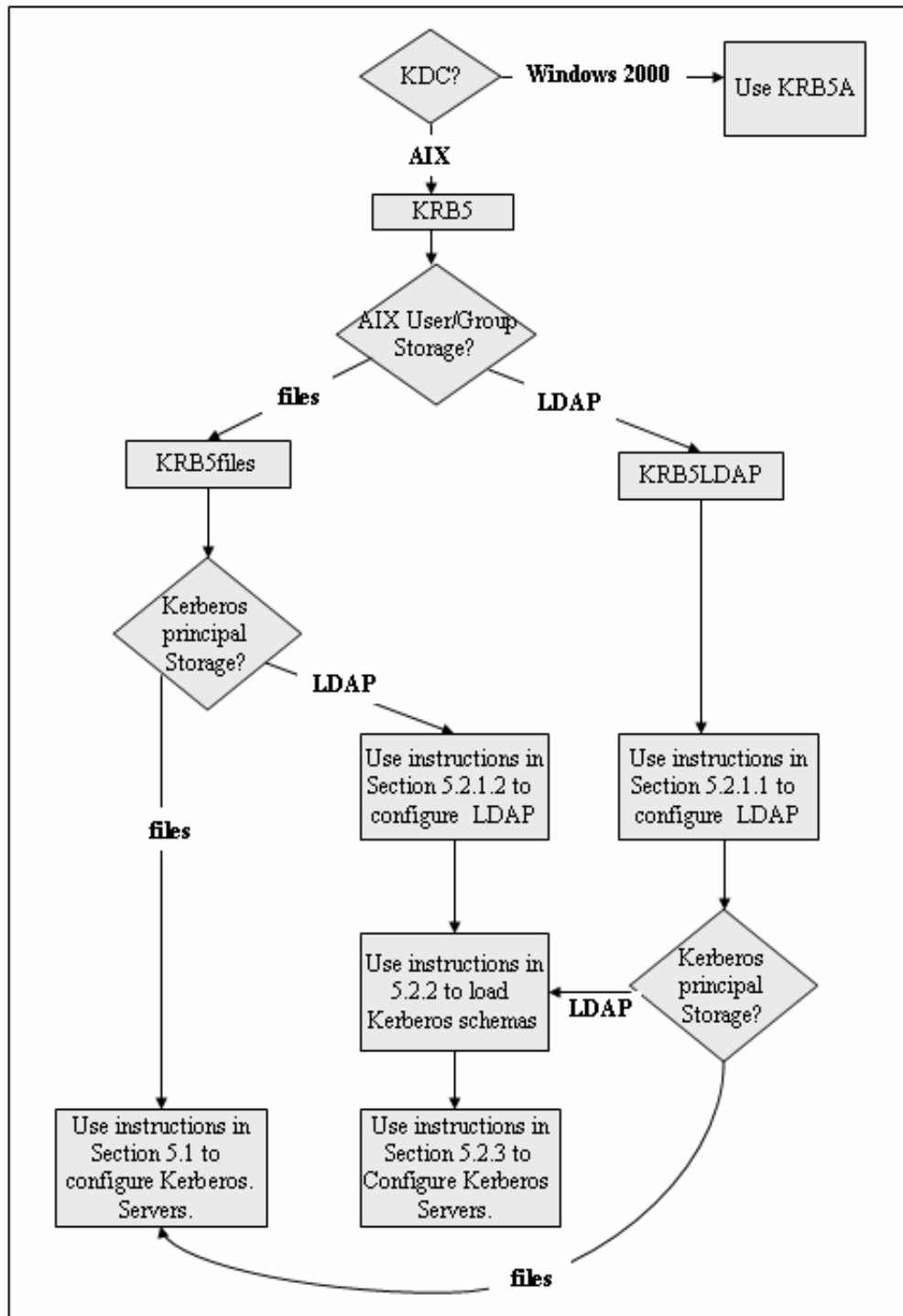


Figure 2: Integrated Login Configuration Scenarios.

## 5.1 Configuring the Network Authentication Service Server with Legacy Database Storage

This section describes how to setup Network Authentication Service KDC and admin servers with a legacy Kerberos database. To configure Network Authentication Service servers, **mkkrb5srv** command is used. Its syntax is given below. See Appendix B for further details.

```
mkkrb5srv -h | {-r <realm> -s <server> -d <domain> [-a <admin name>]
                [[-l { ldapserver | ldapserver:port }]
                [-u ldap_DN -p ldap_DN_pw]
                [-f {keyring | keyring:entry_dn} -k keyring_pw]
                [-m masterkey_location] [-b bind_type]] | -U }
```

A typical invocation of the **mkkrb5srv** command is:

```
# mkkrb5srv -r Realm_Name -s KDC_Server -d Domain_Name -a Admin_Name
```

Notes:

1. **DCE consideration:** It is not recommended that both DCE and Kerberos server software be installed on the same physical system. If you must do so, the default operational Internet port numbers must be changed for the DCE clients and server or for the Kerberos clients and server. In either case, such a change can affect interoperability with existing DCE and Kerberos deployments in your environment. For information about coexistence of DCE and Kerberos, refer to Network Authentication Service documentation [3].
2. **Kerberos clock skew:** Kerberos Version 5 is set up to reject ticket requests from any host whose clock is not within the specified maximum clock skew of the KDC. The default value for maximum clock skew is 300 seconds (five minutes). Kerberos requires that some form of time synchronization be configured between the servers and the clients. It is recommended that you use the Network Time Protocol **-xntpd** or Time Server **-timed** daemons for time synchronization. To use the **timed** daemon, do the following:
  - a. Set up the KDC server as a time server by starting the **timed** daemon, as follows:
    - i. `timed -M`
  - b. Start the **timed** daemon on each Kerberos client.
    - i. `timed -t`
3. **File System Size Consideration:** Network Authentication Service uses space in **/var** filesystem to store information. This information includes database, logs and credential cache files of the authenticated users. This information has potential to grow large. Ensure that **/var** filesystem has sufficient free space to hold this information. Regularly monitoring the amount of free space in **/var** filesystem is strongly suggested.

The following definitions are used in this example to configure Network Authentication Service servers with legacy database:

**KDC Server** : *kdcsrv.austin.ibm.com*  
**Realm Name** : *MYREALM*  
**Domain Name** : *austin.ibm.com*  
**Administrator Name** : *admin/admin*

Invoke `mkkrb5srv` command to configure Network Authentication Service servers using the above definitions. If an existing Kerberos server configuration exists it could be removed by using `mkkrb5srv -U` or `unconfig.krb5` command. If the existing configuration needed to be preserved then the following steps should not be performed.

*Example: Configuring the Network Authentication Service server components with legacy database using mkkrb5srv.*

---

```
# mkkrb5srv -r MYREALM -s kdcsrv.austin.ibm.com -d austin.ibm.com -a
admin/admin
```

| Fileset                 | Level   | State     | Description                           |
|-------------------------|---------|-----------|---------------------------------------|
| -----                   |         |           |                                       |
| Path: /usr/lib/objrepos |         |           |                                       |
| krb5.server.rte         | 1.4.0.1 | COMMITTED | Network Authentication Service Server |
|                         |         |           |                                       |
| Path: /etc/objrepos     |         |           |                                       |
| krb5.server.rte         | 1.4.0.1 | COMMITTED | Network Authentication Service Server |

The -s option is not supported.  
The administration server will be the local host.  
Initializing configuration...  
Creating /etc/krb5/krb5\_cfg\_type...  
Creating /etc/krb5/krb5.conf...  
Creating /var/krb5/krb5kdc/kdc.conf...  
Creating database files...  
Initializing database '/var/krb5/krb5kdc/principal' for realm 'MYREALM'  
master key name 'K/M@MYREALM'

---

The command will prompt for a master database password. It is important that this password is not forgotten.

**Note:** Network Authentication Service does not support a configuration where KDC and the administrative server are on different hosts. The local host is used both for the KDC and administrative server. Therefore ignore the error message saying “The -s option is not supported.”

---

```
You are prompted for the database Master Password.
It is important that you DO NOT FORGET this password.
Enter database Master Password:
Re-enter database Master Password to verify:
WARNING: no policy specified for admin/admin@MYREALM;
defaulting to no policy. Note that policy may be overridden by
ACL restrictions.
```

---

Next, the command will prompt for the password for the administrative principal which was specified as *admin/admin* and then start the *kadmind* and *krb5kdc* daemon from **/etc/inittab**.

---

```
Enter password for principal "admin/admin@MYREALM":
Re-enter password for principal "admin/admin@MYREALM":
Principal "admin/admin@MYREALM" created.
Creating keytable...
Creating /var/krb5/krb5kdc/kadm5.acl...
Starting krb5kdc...
krb5kdc was started successfully.
Starting kadmind...
kadmind was started successfully.
The command completed successfully.
Restarting kadmind and krb5kdc
#
```

---

It may be necessary to wait a few minutes for the **kadmind** and **krb5kdc** daemons to start from **/etc/inittab**. Once the command completes the entries in **/etc/inittab** can be verified with the following commands:

*Example: /etc/inittab entries added by mkkrb5srv for Network Authentication Service*

---

```
# lsitab krb5kdc
krb5kdc:2:once:/usr/krb5/sbin/krb5kdc

# lsitab kadm
kadm:2:once:/usr/krb5/sbin/kadmind
```

---

Verify that KDC and kadmind servers have started.

---

```
#ps -ef | grep -v grep | grep krb5
root 405694      1    0 14:09:05    -   0:00 /usr/krb5/sbin/kadmind
root 590058      1    0 14:09:05    -   0:00 /usr/krb5/sbin/krb5kdc
```

---

**mkkrb5srv** creates the master KDC and the *kadmind* administrative servers for the Kerberos realm *MYREALM*. It also creates the configuration files, initializes the principal database and starts the KDC and *kadmind* servers.

Running **mkkrb5srv** command with legacy database results in the following actions:

1. Creates the **/etc/krb5/krb5.conf** file. Values for realm name, Kerberos admin server, and domain name are set as specified on the command line. The **/etc/krb5/krb5.conf** file also sets the paths for the **default\_keytab\_name**, **kdc**, and **admin\_server** log files.
2. Creates the **/var/krb5/krb5kdc/kdc.conf** file. The **/var/krb5/krb5kdc/kdc.conf** file sets the values for the **kdc\_ports**, **kadmind\_port**, **max\_life**,

- max\_renewable\_life**, **master\_key\_type**, and **supported\_encetypes** variables. This file also sets the paths for the **database\_name**, **admin\_keytab**, **acl\_file**, **dict\_file**, and **key\_stash\_file** variables.
3. Creates the **/var/krb5/krb5kdc/kadm5.acl** file. Sets up the access control for **admin**, **root**, and **host** principals. This enables AIX **root** user to perform Network Authentication Service administrative operations.
  4. Creates the database and one database administration principal. You are asked to set a Kerberos master key and set the password for a Kerberos administrative principal identity. For disaster-recovery purposes, it is critical that the master key and administrative principal identity and password are securely stored away.

See Appendix B and the Network Authentication Service Administrator and User's guide [3] for more information on configuration problems and recovery actions.

## 5.2 Configuring Network Authentication Service Server with LDAP Storage

Before configuring Network Authentication Service with LDAP as backend store, IBM Tivoli Directory Server and DB2 packages need to be installed. The packages can be found on AIX Base Installation Media. The examples in the rest of the paper use IBM Tivoli Directory Version 5.2 and DB2 Version 8.1.1.16. Installing the **ldap.server.rte** package for IBM Directory Server Version 5.2 will install the following filesets:

Note: The kernel must be running in 64-bit mode when IBM Tivoli Directory Server version 5.2 or later is used. This requirement may change in the future.

*Example: Successful install of IBM Tivoli Directory Server Version 5.2*

```
# installp -aXYgd . ldap.server.rte
+-----+
+                               Summaries:                               +
+-----+

Installation Summary
-----
Name                               Level           Part            Event            Result
-----
ldap.client.rte                    5.2.0.0        USR             APPLY            SUCCESS
ldap.client.adt                    5.2.0.0        USR             APPLY            SUCCESS
ldap.client.rte                    5.2.0.0        ROOT           APPLY            SUCCESS
db2_08_01.pext                     8.1.1.16      USR             APPLY            SUCCESS
db2_08_01.msg.en_US.iso8859       8.1.1.16      USR             APPLY            SUCCESS
db2_08_01.jhlp.en_US.iso885      8.1.1.16      USR             APPLY            SUCCESS
db2_08_01.icut                    8.1.1.16      USR             APPLY            SUCCESS
db2_08_01.icuc                    8.1.1.16      USR             APPLY            SUCCESS
db2_08_01.db2.samples             8.1.1.16      USR             APPLY            SUCCESS
db2_08_01.client                  8.1.1.16      USR             APPLY            SUCCESS
db2_08_01.cj                     8.1.1.16      USR             APPLY            SUCCESS
ldap.server.java                   5.2.0.0        USR             APPLY            SUCCESS
ldap.server.rte                    5.2.0.0        USR             APPLY            SUCCESS
ldap.server.com                    5.2.0.0        USR             APPLY            SUCCESS
```

|                    |          |      |       |         |
|--------------------|----------|------|-------|---------|
| ldap.server.cfg    | 5.2.0.0  | USR  | APPLY | SUCCESS |
| ldap.server.com    | 5.2.0.0  | ROOT | APPLY | SUCCESS |
| ldap.server.cfg    | 5.2.0.0  | ROOT | APPLY | SUCCESS |
| db2_08_01.sqlproc  | 8.1.1.16 | USR  | APPLY | SUCCESS |
| db2_08_01.repl     | 8.1.1.16 | USR  | APPLY | SUCCESS |
| db2_08_01.ldap     | 8.1.1.16 | USR  | APPLY | SUCCESS |
| db2_08_01.jdbc     | 8.1.1.16 | USR  | APPLY | SUCCESS |
| db2_08_01.db2.rte  | 8.1.1.16 | USR  | APPLY | SUCCESS |
| db2_08_01.db2.engn | 8.1.1.16 | USR  | APPLY | SUCCESS |
| db2_08_01.das      | 8.1.1.16 | USR  | APPLY | SUCCESS |
| db2_08_01.cs.rte   | 8.1.1.16 | USR  | APPLY | SUCCESS |
| db2_08_01.conv     | 8.1.1.16 | USR  | APPLY | SUCCESS |
| db2_08_01.conn     | 8.1.1.16 | USR  | APPLY | SUCCESS |

---

Verify that asynchronous Input/Output is enabled by executing the **lsattr** command.

*Example: Asynchronous Input/Output not enabled*

---

```
# lsattr -El aio0
autoconfig defined STATE to be configured at system restart True
fastpath enable State of fast path True
kproprio 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
```

---

If the output shows the value of STATE as defined instead of available then use the **chdev** command to change the STATE value to available. This enables asynchronous I/O.

```
# chdev -l aio0 -a autoconfig='available'
```

*Example: Correct aio0 configuration*

---

```
# lsattr -El aio0
autoconfig available STATE to be configured at system restart True
fastpath enable State of fast path True
kproprio 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
```

---

Once the installation of the filesets is complete and the state of asynchronous IO has been verified, configuration of Network Authentication Service server components with LDAP backing store can begin. This involves three major steps:

1. Configuring LDAP,
2. Loading Network Authentication Service schema definitions,
3. Configuring Network Authentication Service KDC and kadmind servers.

## 5.2.1 Configuring LDAP

The easiest method to configure LDAP is to use the **mksecldap** command. This will configure the LDAP server to store AIX user/group information as well and is useful if KRB5LDAP is chosen as the authentication mechanism. If this is not planned however, then LDAP can be configured using the traditional LDAP configuration methods as is documented in Section 5.2.1.2.

Note: The kernel must be running in 64-bit mode when IBM Tivoli Directory Server version 5.2 or later is used. This requirement may change in the future. Use **bootinfo -K** command to determine the kernel mode.

### 5.2.1.1 Configuration of LDAP using mksecldap

Invoke **mksecldap** command to configure. Its syntax is given below. See [7] for further details.

```
mksecldap -s -a adminDN -p adminPasswd -S schematype
          [-d baseDN] [-n port] [-k SSLkeypath] [-w SSLkeypasswd]
          [-x proxyuserDN -X proxyuserPasswd] [-u NONE] [-v LDAPVersion]
          [-U]
```

A typical invocation of the **mksecldap** command for server configuration is:

```
# mksecldap -s -a <admin Distinguished name> -p <Administrator
Password> -S <Schema Type>
```

#### *Example: Configuration of the IBM Directory Server using mksecldap*

---

```
[kdcsrv] # mksecldap -s -a cn=root -p secret -S aix
ldapdb2's New password:
Enter the new password again:

You have chosen the following actions:

Administrator DN 'cn=root' and password will be set.

Setting administrator DN 'cn=root' and password.
Set administrator DN 'cn=root' and password.

IBM Tivoli Directory Server Configuration complete.

You have chosen the following actions:

Database 'ldapdb2' will be configured in instance 'ldapdb2'.

Configuring IBM Tivoli Directory Server Database.
Creating instance: 'ldapdb2'.
Created instance: 'ldapdb2'.
Cataloging instance node: 'ldapdb2'.
Cataloged instance node: 'ldapdb2'.
Starting database manager for instance: 'ldapdb2'.
Started database manager for instance: 'ldapdb2'.
Creating database: 'ldapdb2'.
Created database: 'ldapdb2'.
Updating the database: 'ldapdb2'
```

Updated the database: 'ldapdb2'  
Updating the database manager: 'ldapdb2'  
Updated the database manager: 'ldapdb2'  
Enabling multi-page file allocation: 'ldapdb2'  
Enabled multi-page file allocation: 'ldapdb2'  
Configuring database: 'ldapdb2'  
Configured database: 'ldapdb2'  
Adding local loop back to database: 'ldapdb2'.  
Added local loop back to database: 'ldapdb2'.  
Stopping database manager for instance: 'ldapdb2'.  
Stopped database manager for instance: 'ldapdb2'.  
Starting database manager for instance: 'ldapdb2'.  
Started database manager for instance: 'ldapdb2'.  
Configured IBM Tivoli Directory Server Database.  
IBM Tivoli Directory Server Configuration complete.

You have chosen the following actions:

Suffix 'cn=aixdata' will be added to the configuration file.

Adding suffix: 'cn=aixdata'.  
Added suffix: 'cn=aixdata'.

IBM Tivoli Directory Server Configuration complete.

Server starting in configuration only mode.

Server starting.

Plugin of type EXTENDEDOP is successfully loaded from libevent.a.

Plugin of type PREOPERATION is successfully loaded from libDSP.a.

Plugin of type PREOPERATION is successfully loaded from libDigest.a.

Plugin of type EXTENDEDOP is successfully loaded from libevent.a.

Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.

Plugin of type AUDIT is successfully loaded from /lib/libldapaudit.a.

Plugin of type EXTENDEDOP is successfully loaded from libevent.a.

Plugin of type DATABASE is successfully loaded from /lib/libback-config.a.

Plugin of type EXTENDEDOP is successfully loaded from libloga.a.

Non-SSL port initialized to 389.

Stopping the LDAP server.

Server starting.

Plugin of type EXTENDEDOP is successfully loaded from libevent.a.

Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.

Plugin of type EXTENDEDOP is successfully loaded from libldaprepl.a.

Plugin of type PREOPERATION is successfully loaded from libDSP.a.

Plugin of type PREOPERATION is successfully loaded from libDigest.a.

Plugin of type EXTENDEDOP is successfully loaded from libevent.a.

Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.

Plugin of type AUDIT is successfully loaded from /lib/libldapaudit.a.

Plugin of type AUDIT is successfully loaded from

/usr/ccs/lib/libsecldapaudit64.a(shr.o).

Plugin of type EXTENDEDOP is successfully loaded from libevent.a.

Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.

Plugin of type DATABASE is successfully loaded from /lib/libback-rdbm.a.

Plugin of type REPLICATION is successfully loaded from /lib/libldaprepl.a.

Plugin of type EXTENDEDOP is successfully loaded from /lib/libback-rdbm.a.

Plugin of type EXTENDEDOP is successfully loaded from libevent.a.

Plugin of type DATABASE is successfully loaded from /lib/libback-config.a.

Plugin of type EXTENDEDOP is successfully loaded from libloga.a.

Non-SSL port initialized to 389.

Migrating users and groups to LDAP server.

---

## 5.2.1.2 Configuration of LDAP using the conventional method

Consult IBM Tivoli Directory Server documentation [4] for configuring the server using this method. In the following we give a sample configuration scenario using IBM Tivoli Directory Server V5.2. Authenticate as root user and execute the following steps. These steps assume that the user and group do not exist.

1. Create a group called `dbsysadm` and add `root` to this group:

```
# mkgroup dbsysadm
# chgrpmem -m + root dbsysadm
```

2. Add a user with the same name as the database that you will create

```
# mkuser pgrp=dbsysadm ldapdb2
```

3. Set the password for the `ldapdb2` user and clear the password change flag.

```
# passwd ldapdb2
# pwdadm -f NOCHECK ldapdb2
# pwdadm -c ldapdb2
```

4. Add an administrator ID and password.

*Example: Adding LDAP administrator ID and password.*

---

```
# ldapcfg -u cn=root -p secret
```

You have chosen the following actions:

Administrator DN 'cn=root' and password will be set.

Do you want to....

(1) Continue with the above actions, or

(2) Exit without making any changes: 1

Setting administrator DN 'cn=root' and password.

Set administrator DN 'cn=root' and password.

IBM Tivoli Directory Server Configuration complete.

```
ldapcfg -u cn=root -p secret
```

---

5. Invoke **ldapcfg** to configure LDAP. Make sure you have enough space in **/home**. **ldapcfg** requires at least 80MB of free space to complete.

*Example: Configuration of IBM Directory Server using ldapcfg*

---

```
# ldapcfg -l /home/ldapdb2 -a ldapdb2 -w ldapdb2 -d ldapdb2
```

You have chosen the following actions:

Database 'ldapdb2' will be configured in instance 'ldapdb2'.

Do you want to....

- (1) Continue with the above actions, or
- (2) Exit without making any changes: 1

```
Configuring IBM Tivoli Directory Server Database.
Creating instance: 'ldapdb2'.
Created instance: 'ldapdb2'.
Cataloging instance node: 'ldapdb2'.
Cataloged instance node: 'ldapdb2'.
Starting database manager for instance: 'ldapdb2'.
Started database manager for instance: 'ldapdb2'.
Creating database: 'ldapdb2'.
Created database: 'ldapdb2'.
Updating the database: 'ldapdb2'
Updated the database: 'ldapdb2'
Updating the database manager: 'ldapdb2'
Updated the database manager: 'ldapdb2'
Enabling multi-page file allocation: 'ldapdb2'
Enabled multi-page file allocation: 'ldapdb2'
Configuring database: 'ldapdb2'
Configured database: 'ldapdb2'
Adding local loop back to database: 'ldapdb2'.
Added local loop back to database: 'ldapdb2'.
Stopping database manager for instance: 'ldapdb2'.
Stopped database manager for instance: 'ldapdb2'.
Starting database manager for instance: 'ldapdb2'.
Started database manager for instance: 'ldapdb2'.
Configured IBM Tivoli Directory Server Database.
```

IBM Tivoli Directory Server Configuration complete.

---

6. Invoke **ibmslapd** to start the ldap server. View **/var/ldap/ibmslapd.log** to diagnose any problems.

*Example: Starting ibmslapd*

---

```
# ibmslapd
Server starting.
Plugin of type EXTENDEDOP is successfully loaded from libevent.a.
Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.
Plugin of type EXTENDEDOP is successfully loaded from libldaprepl.a.
Plugin of type PREOPERATION is successfully loaded from libDSP.a.
Plugin of type PREOPERATION is successfully loaded from libDigest.a.
Plugin of type EXTENDEDOP is successfully loaded from libevent.a.
Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.
Plugin of type AUDIT is successfully loaded from /lib/libldapaudit.a.
Plugin of type EXTENDEDOP is successfully loaded from libevent.a.
Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.
Plugin of type DATABASE is successfully loaded from /lib/libback-rdbm.a.
Plugin of type REPLICATION is successfully loaded from /lib/libldaprepl.a.
Plugin of type EXTENDEDOP is successfully loaded from /lib/libback-rdbm.a.
Plugin of type EXTENDEDOP is successfully loaded from libevent.a.
Plugin of type DATABASE is successfully loaded from /lib/libback-config.a.
Plugin of type EXTENDEDOP is successfully loaded from liblog.a.
Non-SSL port initialized to 389.
```

---

We have now successfully installed and configured all of the software that is required to support KDC with IBM directory server

## 5.2.2 Loading Kerberos Schemas

Now, we shall add two schema definitions into LDAP. The first one is the Kerberos schema and the second one is the realm schema. Following steps demonstrate how to do this.

1. Stop **ibmslapd** and add a new suffix using **ldapcfg**:

```
# ibmdirctl -D cn=root -w secret stop
Stop operation succeeded
```

Note: If IBM Directory Server administration daemon were not running then this command would fail. Start administration daemon by typing `ibmdiradm`. If this does not solve the problem you can manually stop `ibmslapd` by using the kill command.

```
# ldapcfg -q -s "ou=austin,o=ibm,c=us"
```

Now start `ibmslapd`:

```
# ibmdirctl -D cn=root -w secret start
Start operation succeeded
```

2. Add the Kerberos schema first. Network Authentication Service ships an `ldif` file in **`/usr/krb5/ldif`**.

*Example: Adding Kerberos schema into LDAP.*

---

```
# ldapmodify -h kdcsrv.austin.ibm.com -D cn=root -w secret -f
/usr/krb5/ldif/IBM.KRB.schema.ldif -v -c

ldap_init(localhost, 389)
add attributetypes:
    BINARY (85 bytes) ( 1.2.840.113556.1.4.49 NAME 'badPasswordTime' SYNTAX
1.3.6.1.4.1.1466.115.121.1.27 )
add ibmattributetypes:
    BINARY (67 bytes) ( 1.2.840.113556.1.4.49 DBNAME( 'badPwdTime'
'badPasswordTime' ) )
modifying entry cn=schema
...
...
```

---

3. Add the realm entry:

Network Authentication Service is shipped with a **`realm_add.ldif`** file in **`/usr/krb5/ldif`** directory. Modify that file appropriately to represent the Network Authentication Service database. The modification involves the replacement of default suffix and realm values with the actual ones. Following is an example **`realm_add.ldif`** file in which the shaded text represents the modifications made to the file.

*Example: Sample `realm_add.ldif` file*

---

```
# The suffix "ou=, o=, c= " should be defined before attempting to
```

```
# load this data. Or change the suffix to be an already defined object.
# Change all references to krbrealmName-V2 such that it represents your
# realm name.
```

```
dn: ou=austin, o=ibm, c=us
ou: austin
objectclass: organizationalUnit
```

```
dn: krbrealmName-V2=MYREALM, ou=austin, o=ibm, c=us
objectclass: KrbRealm-V2
objectclass: KrbRealmExt
krbrealmName-V2: MYREALM
krbprincSubtree: krbrealmName-V2=MYREALM, ou=austin, o=ibm, c=us
krbDeleteType: 3
krbMaxFailAuth: 0
krbDisableTimeInterval: 0
```

```
dn: cn=principal, krbrealmName-V2=MYREALM, ou=austin, o=ibm, c=us
objectclass: container
cn: principal
```

```
dn: cn=policy, krbrealmName-V2=MYREALM, ou=austin, o=ibm, c=us
objectclass: container
cn: policy
```

---

Use **realm\_add.ldif** file to create the realm entry. Use the hostname of the LDAP server instead if localhost if necessary.

### *Example: Adding the realm entry into LDAP*

---

```
# ldapadd -h localhost -D cn=root -w secret -f /usr/krb5/ldif/realm_add.ldif -v
-c
```

```
ldap_init(localhost, 389)
add ou:
    BINARY (6 bytes) austin
add objectclass:
    BINARY (18 bytes) organizationalUnit
adding new entry ou=austin, o=ibm, c=us

add objectclass:
    BINARY (11 bytes) KrbRealm-V2
    BINARY (11 bytes) KrbRealmExt
add krbrealmName-V2:
    BINARY (7 bytes) MYREALM
add krbprincSubtree:
    BINARY (47 bytes) krbrealmName-V2=MYREALM, ou=austin, o=ibm, c=us
add krbDeleteType:
    BINARY (1 bytes) 3
add krbMaxFailAuth:
    BINARY (1 bytes) 0
add krbDisableTimeInterval:
    BINARY (1 bytes) 0
adding new entry krbrealmName-V2=MYREALM, ou=austin, o=ibm, c=us

add objectclass:
    BINARY (9 bytes) container
add cn:
```

```
        BINARY (9 bytes) principal
adding new entry cn=principal, krbrealmName-V2=MYREALM, ou=austin, o=ibm, c=us

add objectclass:
        BINARY (9 bytes) container
add cn:
        BINARY (6 bytes) policy
adding new entry cn=policy, krbrealmName-V2=MYREALM, ou=austin, o=ibm, c=us
```

---

At this point, we can verify that LDAP has all the container names that it must have.

*Example: LDAP naming contexts when mksecldap is used*

---

```
# ldapsearch -b "" -s base "objectclass=*" namingcontexts

namingcontexts=CN=SCHEMA
namingcontexts=CN=LOCALHOST
namingcontexts=CN=PWDPOLICY
namingcontexts=CN=IBMPOLICIES
namingcontexts=CN=AIXDATA
namingcontexts=OU=AUSTIN,O=IBM,C=US
```

---

*Example: LDAP naming contexts when mksecldap is not used*

---

```
# ldapsearch -b "" -s base "objectclass=*" namingcontexts

namingcontexts=CN=SCHEMA
namingcontexts=CN=LOCALHOST
namingcontexts=CN=PWDPOLICY
namingcontexts=CN=IBMPOLICIES
namingcontexts=OU=AUSTIN,O=IBM,C=US
```

---

## 5.2.3 Configuration of Kerberos Server

Run **mkkrb5srv** to setup the Network Authentication Service kadmind and KDC servers for AIX Kerberos integrated login. Please refer to Section 5.1 and Appendix B for **mkkrb5srv** command usage. The following definitions are used in the example:

|                                    |                        |
|------------------------------------|------------------------|
| <b>KDC Server</b>                  | :kdcsrv.austin.ibm.com |
| <b>Realm Name</b>                  | :MYREALM               |
| <b>Domain Name</b>                 | :austin.ibm.com        |
| <b>Administrator Name</b>          | :admin/admin           |
| <b>LDAP server</b>                 | :kdcsrv.austin.ibm.com |
| <b>LDAP administrator name</b>     | :cn=root               |
| <b>LDAP administrator password</b> | :secret                |

*Example: Configuring the Network Authentication Service server components with LDAP storage using mkkrb5srv*

---

```

# mkkrb5srv -r MYREALM -s kdcsrv.austin.ibm.com -d austin.ibm.com -a
admin/admin -l kdcsrv.austin.ibm.com -u cn=root -p secret
  Fileset                Level  State      Description
-----
Path: /usr/lib/objrepos
  krb5.server.rte        1.4.0.1  COMMITTED  Network Authentication Service
                        Server
Path: /etc/objrepos
  krb5.server.rte        1.4.0.1  COMMITTED  Network Authentication Service
                        Server

The -s option is not supported.
The administration server will be the local host.
Initializing configuration...
Creating /etc/krb5/krb5_cfg_type...
Creating /etc/krb5/krb5.conf...
Creating /var/krb5/krb5kdc/kdc.conf...
Creating database files...
Initializing database 'LDAP' for realm 'MYREALM'
master key name 'K/M@MYREALM'
Attempting to bind to one or more LDAP servers. This may take a while...
You are prompted for the database Master Password.
It is important that you DO NOT FORGET this password.
Enter database Master Password:
Re-enter database Master Password to verify:
Attempting to bind to one or more LDAP servers. This may take a while...
WARNING: no policy specified for admin/admin@MYREALM;
        defaulting to no policy. Note that policy may be overridden by
        ACL restrictions.
Enter password for principal "admin/admin@MYREALM":
Re-enter password for principal "admin/admin@MYREALM":
Principal "admin/admin@MYREALM" created.
Creating keytable...
Attempting to bind to one or more LDAP servers. This may take a while...
Creating /var/krb5/krb5kdc/kadm5.acl...
Starting krb5kdc...
Attempting to bind to one or more LDAP servers. This may take a while...
krb5kdc was started successfully.
Starting kadmind...
Attempting to bind to one or more LDAP servers. This may take a while...
kadmind was started successfully.
The command completed successfully.
Restarting kadmind and krb5kdc
Attempting to bind to one or more LDAP servers. This may take a while...
Attempting to bind to one or more LDAP servers. This may take a while...

```

Check that the KDC and kadmin servers have started.

```

# ps -ef | grep krb5 | grep -v grep
  root 770208      1   0 15:09:19      -   0:00 /usr/krb5/sbin/kadmind
  root 835684      1   0 15:09:19      -   0:00 /usr/krb5/sbin/krb5kdc

```

Running **mkkrb5srv** command with LDAP database results in actions similar to legacy database configuration except for database creation. Since LDAP is used as backend storage, database files are not created on the local file system. Instead a **.kdc\_ldap\_data file** is created under **/var/krb5/krb5kdc** to hold information about LDAP.

Setting up the Master/Slave configuration for KDC's, replica servers for LDAP and SSL are outside the scope of this paper. By using SSL, the Kerberos administration server or KDC authenticates the identity of the LDAP server through public key technology. The privacy and integrity of data is also protected during transmit with the use of SSL protocol. Please refer to Network Authentication Service [3] and IBM Tivoli Directory Server [4] documentation for this advanced configuration options.

## 6 Installing and Configuring the Kerberos client on AIX for Integrated Login using KRB5

Configuration of Kerberos integrated login requires Network Authentication Service client packages to be installed on the system. The client packages can be obtained from the AIX Expansion Pack. To install Network Authentication Service client packages the following command can be used:

```
# installp -aqXYgd . krb5.client
```

### 6.1 Configuring Kerberos Integrated Login

To configure Network Authentication Service client for Kerberos Integrated login *mkkrb5clnt* command is used. Its typical syntax for client configuration is given below. See Appendix B and AIX documentation for a detailed discussion of this command [10].

```
mkkrb5clnt -r <realm> -d <domain> {-c <KDC> -s <server> |  
-l {ldapservice | ldapservice:port} [-c <KDC> -s <server>]}  
[-a <admin>] [-A] [-i <data base>] [-K] [-T]
```

Note the following in regards to the optional flags for *mkkrb5clnt*:

- |           |   |
|-----------|---|
| -A and -T | Required for client to perform Kerberos principal management.                                   |
| -K        | Updates the SYSTEM line in the default stanza of /etc/security/user with "KRB5files OR compat". |
| -l        | Performs KDC and kadmin discovery if LDAP is in use.  |
| -i        | Specifies the database to use in the methods.cfg entry  |
| -a        | Specifies the Kerberos administrator principal.   |

Using the "**-i files**" option configures the system to use the local file system to query AIX user/group information. If the client has already been configured as an LDAP client then alternatively "**-i LDAP**" can be specified to query this information centrally in LDAP. Please refer to [6] for LDAP client configuration steps.

Note: Storing AIX user/group information on LDAP and storing Kerberos principals on LDAP are two distinct operations. From client point of view, the mechanism used to store Kerberos data is transparent.

The following definitions are used in the example configuration:

**Realm Name** : *MYREALM*  
**KDC Server** : *kdcsrv.austin.ibm.com*  
**Administration Server** : *kdcsrv.austin.ibm.com*  
**Domain Name** : *austin.ibm.com*  
**Administrator Principal** : *admin/admin*  
**AIX User/Group Database** : *files*

The following example configures the system for Kerberos integrated login with the local file system as the AIX user/group repository:

*Example: Output of mkkrb5clnt command*

---

```
# mkkrb5clnt -r MYREALM -c kdcsrv.austin.ibm.com -s kdcsrv.austin.ibm.com -a
admin/admin -d austin.ibm.com -A -i files -K -T
/etc/krb5/krb5.conf file already exists
Password for admin/admin@MYREALM:
Configuring fully integrated login
Making root a Kerberos administrator
Authenticating as principal admin/admin with existing credentials.
WARNING: no policy specified for root/kdcsrv.austin.ibm.com@MYREALM;
defaulting to no policy. Note that policy may be overridden by
ACL restrictions.
Enter password for principal "root/kdcsrv.austin.ibm.com@MYREALM":
Re-enter password for principal "root/kdcsrv.austin.ibm.com@MYREALM":
Principal "root/kdcsrv.austin.ibm.com@MYREALM" created.

Administration credentials NOT DESTROYED.
Configuring Kerberos as the default authentication scheme
Cleaning administrator credentials and exiting.
```

---

After the **mkkrb5clnt** command finishes the **/etc/krb5/krb5.conf** file will be updated with the realm name, KDC server, Kerberos admin server and domain name specified with the command invocation. The configuration can be verified by examining the **/etc/krb5/krb5.conf** file:

*Example: Sample /etc/krb5/krb5.conf file on the client*

---

```
[libdefaults]
    default_realm = MYREALM
    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

[realms]
    MYREALM = {
        kdc = kdcsrv.austin.ibm.com:88
        admin_server = kdcsrv.austin.ibm.com:749
        default_domain = austin.ibm.com
    }
```

```
[domain_realm]
.austin.ibm.com = MYREALM
kdcsrv.austin.ibm.com = MYREALM

[logging]
kdc = FILE:/var/krb5/log/krb5kdc.log
admin_server = FILE:/var/krb5/log/kadmin.log
default = FILE:/var/krb5/log/krb5lib.log
```

Note: If LDAP is used for Kerberos principal storage then krb5.conf file will contain the line `vdb_plugin_lib = /usr/lib/libkrb5ldapplug.a` under `[realms]` stanza.

If a client is located in a different DNS domain than the KDC, the DNS domains for both the client and the KDC must be mapped to the realm. This requires the following additional actions to be performed:

1. Edit the `/etc/krb5/krb5.conf` file and add another entry after `[domain realm]`.
2. Map the different domain to your realm.

For example, if you want to include a client that is in the `raleigh.ibm.com` domain into your `MYREALM` realm, modify the `/etc/krb5/krb5.conf` file as follows:

```
[domain realm]
.austin.ibm.com = MYREALM
.raleigh.ibm.com = MYREALM
```

When Network Authentication Service configuration is complete, it is not apparent to normal users that the Kerberos technology is in use. The login process to the operating system remains unchanged. However, after a successful login users will have Kerberos ticket-granting tickets (TGTs) associated with their running processes. The user's environment variable `$KRB5CCNAME` points to that TGT. To verify that the login is successful and the user has a TGT, use `klist` command.

*Example: Sample Kerberos integrated login session*

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2004.
login: foo
foo'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.
*
*
*****
[kdcsrv] $ echo $AUTHSTATE
KRB5files
[kdcsrv] $ /usr/krb5/bin/klist
Ticket cache: FILE:/var/krb5/security/creds/krb5cc_foo@MYREALM_203
Default principal: foo@MYREALM
```

```
Valid starting      Expires          Service principal
04/29/05 13:12:36  04/30/05 13:12:36  krbtgt/MYREALM@MYREALM
04/29/05 13:12:38  04/29/05 16:12:38  kadmin/admin@MYREALM
[kdcsvr] $ echo $KRB5CCNAME
FILE:/var/krb5/security/creds/krb5cc_foo@MYREALM_203
[kdcsvr] $
```

---

The next example demonstrates `su` command using Kerberos authentication. Note that the `SYSTEM` attribute controls which authentication method would be used. In this example `foo`'s `SYSTEM` attribute has a value of **KRB5files**.

*Example: Sample Kerberos based su authentication*

---

```
$ id
uid=100(guest) gid=100(usr)
$ su - foo
foo's Password:
[kdcsvr] $ id
uid=203(foo) gid=1(staff)
[kdcsvr] $ echo $AUTHSTATE
KRB5files
[kdcsvr] $ /usr/krb5/bin/klist
Ticket cache: FILE:/var/krb5/security/creds/krb5cc_foo@MYREALM_203
Default principal: foo@MYREALM

Valid starting      Expires          Service principal
04/29/05 13:17:19  04/30/05 13:17:19  krbtgt/MYREALM@MYREALM
04/29/05 13:17:22  04/29/05 16:17:22  kadmin/admin@MYREALM
[kdcsvr] $ echo $KRB5CCNAME
FILE:/var/krb5/security/creds/krb5cc_foo@MYREALM_203
[kdcsvr] $
```

---

## 6.2 Eliminating Kadmind Daemon Dependency during Authentication

If the `kadmind` daemon is not available, authentication through the KRB5 load module will take longer and may even fail. Setting the `kadmind` options parameter in the `methods.cfg` file eliminates dependency on the `kadmind` daemon during authentication. The possible option values are `no` or `false` to disable the **kadmind** lookups and `yes` or `true` to enable **kadmind** lookups. The default value is `yes`.

When the `kadmind` option is set to `no`, the **kadmind** daemon is not contacted during authentication. Therefore, users can log into the system regardless of the status of the **kadmind** daemon (provided that the user enters the correct password when the system prompts one). However, AIX user administration commands such as **mkuser**, **chuser**, or **rmuser** will fail to administrate Kerberos integrated users if the daemon is not available (this happens for example, when the daemon is down or the machine is not accessible).

To disable the checking of the **kadmind** daemon during authentication, modify the stanzas in the **methods.cfg** file as follows:

*Example: Sample /usr/lib/security/methods.cfg file.*

---

```
KRB5:
    program = /usr/lib/security/KRB5
    options = kadmind=no
KRB5files:
    options = db=BUILTIN,auth=KRB5
```

---

If the network where the **kadmind** daemon resides is not reachable or the system hosting the kadmind server is down the authentication may even fail. Setting the kadmind option to **no** in the **methods.cfg** file eliminates the delays or failures during authentication when the machine is not accessible

When you set **kadmind=no** but the **kadmind** daemon is running, **lsuser** will fail to retrieve the Kerberos related attributes. However, **login**, **su**, **passwd**, **mkuser**, **chuser**, and **rmuser** commands will succeed.

## 7 KRB5 Authentication Load Module Questions and Troubleshooting Information

Before performing any problem determination and troubleshooting make sure all the servers and daemons are running.

**LDAP** module uses syslog subsystem to write its error and debug information

IBM Network Authentication Service uses its own log files to log requests made to KDC and kadmind daemons. The log files are specified in the `[logging]` stanza of **krb5.conf** file. The default locations of these files are **/var/krb5/log/krb5kdc.log** and **/var/krb5/log/kadmin.log**. Please refer to NAS documentation for details.

When a problem appears to be related to IBM Tivoli Directory Server, it is suggested to check the log files generated by IBM Tivoli Directory Server. By default they are located in **/var/ldap/ibmslapd.log** and **/var/ldap/db2cli.log**.

### 7.1 How do I Create AIX Kerberos Authenticated Users

The root user needs to obtain Kerberos credentials which grant the required privilege to perform administrative tasks. This is done by invoking `#kinit root/<fully qualified host name>`. The credential lifetime is set during the configuration of the Network Authentication Service. This limit cannot be circumvented using the **-l** option. The following example demonstrates how to get needed privilege to perform

administrative tasks. `kdcsrv.austin.ibm.com` is the host where administrative tasks are being performed

```
# kinit root/kdcsrv.austin.ibm.com
```

To create a Kerberos authenticated user, type:

```
# mkuser -R KRB5files SYSTEM=KRB5files registry=KRB5files foo
```

This creates AIX user account `foo` and Kerberos principal `foo@MYREALM` on the Kerberos database. It also sets the authentication grammar for this user to `KRB5files`. This means this user goes through a Kerberos authentication and if the correct Kerberos password is not entered the authentication is denied.

One can also use `KRB5LDAP` as the registry value once LDAP is properly configured using `mksecldap`.

```
# mkuser -R KRB5LDAP SYSTEM=KRB5LDAP registry=KRB5LDAP foo
```

## 7.2 How do I Remove a Kerberos Authenticated User

To remove a Kerberos authenticated user, use the `rmuser` command. When the system is configured to use `KRB5` authentication module, `rmuser` will also remove the Kerberos principal from Kerberos database.

```
#rmuser -R KRB5files foo
```

One can also use `KRB5LDAP` as registry value, once LDAP is properly configured using `mksecldap`.

```
#rmuser -R KRB5LDAP foo
```

## 7.3 How do I Change the Password of a Kerberos Authenticated User

A user can change the password by invoking `passwd` command.

```
$ passwd -R KRB5files foo
```

## 7.4 What are AIX Kerberos Extended Attributes

When `KRB5` authentication module is used, it is possible to manipulate some of the Kerberos principal information through AIX `lsuser` and `chuser` commands. Kerberos principal information is manipulated using AIX extended attributes. In the following table you will find a list of AIX extended Kerberos attributes and their access status. `GET` means you can only display the attribute and `SET` means a privileged user (root on AIX)

can assign values to it. Note that an AIX Kerberos authenticated user is able to display his own Kerberos extended attributes in addition to the other allowed AIX attributes (i.e. id, pgrp, groups, gecost, home, shell).

| Extended attribute name | Description   | Access Mode |
|-------------------------|---|-------------|
| krb5_principal_name     | The name of the principal associated with the AIX user name.  | GET         |
| krb5_principal          | Same as krb5_principal_name.  | GET         |
| krb5_realm              | The Kerberos realm name that this principal belongs to.   | GET         |
| krb5_last_pwd_change    | The time this principal's password was last changed.  | GET         |
| krb5_attributes         | The set of attributes for use by the KDC. krb5_attribute is discussed in detail in Section krb5_attributes.             | GET/SET     |
| krb5_mod_name           | The name of the principal that most recently modified this principal.   | GET         |
| krb5_mod_date           | The time this principal was last modified.  | GET         |
| krb5_kvno               | The version of the principal's current key (password)   | GET/SET     |
| krb5_mkvno              | The database master key version number. This is provided for compatibility with other implementations. This field is 0. | GET         |
| krb5_max_renewable_life | The maximum renewable lifetime of any ticket issued for this principal  | GET/SET     |
| krb5_names              | A list of name:hostname pairs. This field is for future use. It is suggested that this attribute is not modified.       | GET/SET     |

**Table 1: AIX Kerberos extended attributes**

### **krb5\_attributes**

The AIX extended attribute *krb5\_attributes* represents the set of Kerberos principal attributes for use by the Key Distribution Center. The symbols used to set the *krb5\_attributes* are given below. A privileged user can modify these Kerberos attributes using *chuser* command (i.e. *chuser -R KRB5files krb5\_attributes=+requires\_preauth krb5user*). To set a flag, add '+' in front of the flag. When unsetting the flag, '-' is prefixed to the flag.

**allow\_postdated**

If set, postdated tickets will be allowed to be issued for the principal

Set : **+allow\_postdated**

Unset : **-allow\_postdated**

**allow\_forwardable**

If set, forwardable tickets will be allowed to be issued for the principal

Set : **+allow\_forwardable**

Unset : **-allow\_forwardable**

**allow\_tgs\_req**

If set, service tickets for this principal will be issued using a TGT

Set : **+allow\_tgs\_req**

Unset : **-allow\_tgs\_req**

**allow\_renewable**

If set, renewable tickets will be allowed to be issued for the principal.

Set : **+allow\_renewable**

Unset : **-allow\_renewable**

**allow\_proxiable**

If set, proxiable tickets will be allowed to be issued for the principal.

Set : **+allow\_proxiable**

Unset : **-allow\_proxiable**

**allow\_dup\_skey**

If set enables user-to-user authentication for this principal

Set : **+allow\_dup\_skey**

Unset : **-allow\_dup\_skey**

**allow\_tix**

If unset no tickets will be issued for this principal

Set : **+allow\_tix**

Unset : **-allow\_tix**

**requires\_preauth**

If set, software preauthentication is required before a ticket is issued

Set : **+requires\_preauth**

Unset : **-requires\_preauth**

**requires\_hwauth**

If set, software hardware preauthentication is required before a ticket is issued for the principal

Set : **+requires\_hwauth**

Unset : **-requires\_hwauth**

**needchange**

If set, the principals keys (password must be changed before tickets are issued)

Set : **+needchange**

Unset : **-needchange**

**allow\_svr**

If set service tickets is allowed to be issued for this principal

Set : **+allow\_svr**

Unset : **-allow\_svr**

### password\_changing\_service

If set the principal is the special principal for the password changing service

Set : **+password\_changing\_service**

Unset : **-password\_changing\_service**

### support\_desmd5

If set, the KDC may issue tickets which use the RSA MD5 checksum algorithm.

Turning on this option may cause interoperability problems. Refer to Network Authentication Service documentation for more detail.

Set : **+support\_desmd5**

Unset : **-support\_desmd5**

When lsuser displays Kerberos principal attributes, for readability purposes, it adds the prefix `dis` to some of the attributes if the attribute is unset. Below you will find what lsuser would display when an attribute is set or unset.

| Attribute name            | lsuser output when attribute is set | lsuser output when attribute is unset. |
|---------------------------|-------------------------------------|--|
| allow_postdated           | <NULL>                              | disallow_postdated                     |
| allow_forwardable         | <NULL>                              | disallow_forwardable                   |
| allow_tgs_req             | <NULL>                              | disallow_tgt_based                     |
| allow_renewable           | <NULL>                              | disallow_renewable                     |
| allow_proxiable           | <NULL>                              | disallow_proxiable                     |
| allow_dup_skey            | <NULL>                              | disallow_dup_skey                      |
| allow_tix                 | <NULL>                              | disallow_all_tix                       |
| requires_preauth          | requires_preauth                    | <NULL>                                 |
| requires_hwauth           | requires_hwauth                     | <NULL>                                 |
| needchange                | requires_pwchange                   | <NULL>                                 |
| allow_svr                 | <NULL>                              | disallow_svr                           |
| password_changing_service | Pwchange_service                    | <NULL>                                 |
| Support_desmd5            | support_desmd5                      | <NULL>                                 |

**Table 2: lsuser output for Kerberos extended attributes.**

When a user is created all the attributes except the following are set: **requires\_hwauth**, **needchange**, **password\_changing\_service**, **support\_desmd5**

If the user's **needchange** flag is set the user will go through a change password dialog the next time the user attempts to login. When such a situation happens, due to an AIX architectural limitation the user's TGT will not be stored on the file system. In other words the user would be authenticated using Kerberos but will not have a TGT. In such a scenario the user needs to invoke kinit to get a TGT. Note that **needchange** flag only applies to KRB5 module.

## 7.5 How do I list the AIX Kerberos Extended Attributes

Extended attributes are supported only by **KRB5** module. To list all the AIX user attributes including the extended attributes use the lsuser command.

```
# lsuser -R KRB5files foo
```

One can also list a select list of attributes using the **-a** option. The following example displays only the Kerberos extended attributes of a user:

*Example: Listing of extended Kerberos attributes for user foo*

---

```
# lsuser -R KRB5files -f -a \  
    krb5_principal \  
    krb5_principal_name \  
    krb5_realm \  
    krb5_last_pwd_change \  
    krb5_attributes \  
    krb5_mod_name \  
    krb5_mod_date \  
    krb5_kvno \  
    krb5_mkvno \  
    krb5_max_renewable_life \  
    krb5_names foo  
  
foo:  
    krb5_principal=foo@MYREALM  
    krb5_principal_name=foo@MYREALM  
    krb5_realm=MYREALM  
    krb5_last_pwd_change=1113949802  
    krb5_attributes=requires_preauth  
    krb5_mod_name=root/kdcsrv.austin.ibm.com@MYREALM  
    krb5_mod_date=1113949803  
    krb5_kvno=2  
    krb5_mkvno=0  
    krb5_max_renewable_life=604800  
    krb5_names=foo:kdcsrv.austin.ibm.com
```

---

## 7.6 How do I modify the AIX Kerberos Extended Attributes

Extended attributes are only supported by KRB5 module. Only a privileged user can modify extended attributes with a SET access mode. These are **krb5\_kvno**, **krb5\_max\_renewable\_life**, **krb5\_attributes** and **krb5\_names**. Following are examples of the use of `chuser` command to modify Kerberos extended attributes:

To change the maximum renewable lifetime of any ticket issued to `foo` to 5 days type:

---

```
# chuser -R KRB5files krb5_max_renewable_life=432000 foo
```

---

Note that `krb5_max_renewable_life` value is specified in terms of seconds.

To change the key (password) version number of the principal associated with `foo` type:

---

```
# chuser -R KRB5files krb5_kvno=4 foo
```

---

To set all the Kerberos principal attributes listed in Section 5.4.4 type:

---

```
# chuser -R KRB5files \  
krb5_attributes=+allow_postdated,+allow_forwardable,+allow_tgs_req,+allow_renewable,+allow_proxiabile,+allow_dup_skey,+allow_tix,+requires_preauth,+requires_hwa  
th,+needchange,+allow_svr,+password_changing_service,+support_desmd5 foo  
  
# lsuser -R KRB5files -a krb5_attributes foo  
foo krb5_attributes=requires_preauth,requires_hwauth,requires_pwchange,  
pwchange_service,support_desmd5
```

---

To unset all the Kerberos principal attributes listed in Section 5.4.4 type:

---

```
# chuser -R KRB5files krb5_attributes=-allow_postdated,-allow_forwardable,\  
-allow_tgs_req,-allow_renewable,-allow_proxiabile,-allow_dup_skey,\  
-allow_tix,-requires_preauth,-requires_hwauth,-needchange,-allow_svr,\  
-password_changing_service,-support_desmd5 foo  
  
# lsuser -R KRB5files -a krb5_attributes foo  
foo  
krb5_attributes=disallow_postdated,disallow_forwardable,disallow_tgt_based,disa  
llow_renewable,disallow_proxiabile,disallow_dup_skey,disallow_all_tix,disallow_s  
vr
```

---

To change the krb5\_names and add an AIX user name/host name pair type:

---

```
# lsuser -R KRB5files -a krb5_names foo  
foo krb5_names=foo:kdcsrv.austin.ibm.com  
  
# chuser -R KRB5files krb5_names=bar:greenjeans.austin.ibm.com foo  
  
# lsuser -R KRB5files -a krb5_names foo  
foo krb5_names=foo:kdcsrv.austin.ibm.com,bar:greenjeans.austin.ibm.com
```

---

## 7.7 Listing all the Users Defined in KRB5files

In order to list all the Kerberos authenticated users in the system, use **lsuser** command with ALL as the username.

---

```
# mkuser -R KRB5files user1  
# mkuser -R KRB5files user2  
# mkuser -R KRB5files user3  
# mkuser -R KRB5files user4  
  
# lsuser -R KRB5files -a registry ALL  
user1 registry=KRB5files  
user2 registry=KRB5files  
user3 registry=KRB5files  
user4 registry=KRB5files
```

---

## 7.8 How do I convert an AIX User to a Kerberos Authenticated User

The existing AIX users can be converted to Kerberos authenticated users by using the command **mkseckrb5**. Its syntax is as follows:

```
mkseckrb5 [-h | -r ] [user_name ....]
```

When **mkseckrb5** is called with a user name, then it converts that user if it is a non-administrative user (i.e. uid greater than 201). **mkseckrb5** asks Network Authentication Service administrative principal name and password. A password is also asked for each converted user if randomize option is not used.

```
# mkseckrb5 foo
Please enter the admin principal name: admin/admin
Password for admin/admin@MYREALM:
Importing foo
Enter password for principal "foo@MYREALM":
Re-enter password for principal "foo@MYREALM":
```

Before this user can login using Kerberos, the user's **SYSTEM** and registry attributes need to be set:

```
# chuser -R KRB5files SYSTEM=KRB5files registry=KRB5files foo
```

*Example: Integrated login after converting user foo*

---

```
# tn kdcsrv
Trying...
Connected to kdcsrv.austin.ibm.com.
Escape character is '^T'.
```

```
telnet (kdcsrv.austin.ibm.com)
```

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2004.
login: foo
foo'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.
*
*
*****
```

```
$ /usr/krb5/bin/klist
Ticket cache: FILE:/var/krb5/security/creds/krb5cc_foo@MYREALM_218
```

```
Default principal:  foo@MYREALM
```

```
Valid starting      Expires            Service principal
04/21/05 15:39:22  04/22/05 15:39:21  krbtgt/MYREALM@MYREALM
04/21/05 15:39:24  04/21/05 18:39:24  kadmin/admin@MYREALM
```

---

When `mkseckrb5` is called without any arguments, then it converts all the local non-administrative users whose ids are greater than 201. It asks the administrative principal name and the password. For each converted user a password is asked.

```
[kdcsrv] # mkseckrb5
Please enter the admin principal name: admin/admin
Password for admin/admin@MYREALM:
Importing user1
Enter password for principal "user1@MYREALM":
Re-enter password for principal "user1@MYREALM":
Importing user2
Enter password for principal "user2@MYREALM":
Re-enter password for principal "user2@MYREALM":
.
.
```

One can choose to randomize the passwords during the conversion with the use of `-r` option. Later on, AIX system administrator can set passwords by using the `passwd` command. This eliminates the password entry step during the migration. Asking user passwords during conversion can be a burden when converting a large number of users if no automation is available.

*Example: Converting a user by specifying `-r` option.*

---

```
# mkseckrb5 -r user1
Please enter the admin principal name: admin/admin
Password for admin/admin@MYREALM:
Importing user1
```

---

Once the migration is done, set the `SYSTEM`, registry attributes and the password:

```
# chuser -R KRB5files SYSTEM=KRB5files registry=KRB5files user1

# passwd -R KRB5files user1
Changing password for "user1"
user1's Old password:
user1's New password:
Enter the new password again:
#
```

As you have noticed the user's old password is asked. This is AIX 5L Version 5.3 behavior. You can enter any value, as it will not be used. If you want to eliminate the "Old Password" prompt please refer to Section 7.9 "What do I do if the Password is Forgotten" which describes the use of `rootpwdrequired` option. This option applies only to AIX 5L V5.3.

**mkseckrb5** converts only local users. The users in remote domains such as LDAP can not be converted using `mkseckrb5`.

## 7.9 What do I do if the Password is Forgotten

On AIX, the root user can set the password of a Kerberos principal using the `passwd` command.

```
# passwd -R KRB5files foo
Changing password for "foo"
foo's Old password:
foo's New password:
Enter the new password again:
```

The `passwd` command asks the old password. You can enter any value, as it will be ignored (On the other hand if a user is changing his/her own password the old password is always required.) You can disable the prompting of the old password, when root is changing the password using the **rootpwdrequired** option in `methods.cfg` file. Edit the `/usr/lib/security/methods.cfg` file as follows:

```
KRB5files:
    options = db=BUILTIN,auth=KRB5,rootrequiresopw=false
```

This option is effective only when **root** changing a Kerberos authenticated user's password. After modifying the `/etc/security/methods.cfg` file invoke `passwd` command again.

```
# passwd -R KRB5files foo
Changing password for "foo"
foo's New password:
Enter the new password again:
#
```

## 7.10 No TGT after a Successful Login When “needchange” Flag is Set

When the user's `needchange` flag is set, the user will go through a change password dialog at the next login attempt. However, due to an AIX architectural limitation the user's TGT will not be stored on the file system. In other words the user would be authenticated using Kerberos but will not have a TGT. In such a scenario the user needs to invoke **kinit** to get a TGT.

## 7.11 AIX Does Not Accept My Password

Check that the KDC and `kadmind` servers are running.

Check that the password meets the requirements of both AIX and Network Authentication Service. Both AIX and Network Authentication Service password policy rules are applied. The password must comply with both policies. You may change the password rules on AIX by properly modifying the password policy related attributes.

Refer to AIX documentation for how to do this. Network Authentication Server kadmin tool is used to modify the password policy on the Kerberos database.

## 7.12 Can a Kerberos-Authenticated User Become Authenticated Using Only Standard AIX Authentication

The answer is yes. Perform the following actions to authenticate the Kerberos-authenticated user `foo` using AIX `crypt()` authentication instead of Kerberos

1. Set the AIX password of user `foo` (`/etc/security/passwd`) using the `passwd` command. Choose a different password for testing purposes:

```
# passwd -R files foo
```

2. Change the **SYSTEM** attribute of the user, as follows:

```
# chuser -R KRB5files SYSTEM=compat foo
```

This changes the authentication from Kerberos to `crypt()`.

*Example: Login after conversion from Kerberos to `crypt()` authentication*

---

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2004.
login: foo
foo's Password:
[compat]: 3004-610 You are required to change your password.
      Please choose a new one.

foo's New password:
Enter the new password again:
Enter the new password again:
*****
*
*
* 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.
*
*
*****
[kdcsrv] $ id
uid=203(foo) gid=1(staff)
[kdcsrv] $ echo $AUTHSTATE
compat
[kdcsrv] $ /usr/krb5/bin/klist
Unable to get cache name (ticket cache: /var/krb5/security/creds/krb5cc_203).
      Status 0x96c73ac3 - No credentials cache found.
[kdcsrv] $
```

---

Note that AUTHSTATE value is **compat** and no TGT is issued as the user logged in using local authentication. If you want to use crypt() authentication as a backup mechanism, the **SYSTEM** attribute needs to be changed as follows:

```
# chuser -R KRB5files SYSTEM="KRB5files or compat" foo
```

## 7.13 How to Change Client Kadmin Port

The following applies to KRB5 load module, as user/principal management support is available only in KRB5. **kadmin** daemon is used to perform Kerberos principal management.

If the **kadmin** daemon is accepting requests other than the well-known port 749, the following steps need to be done on the client to change it. In this example **kadmin** daemon runs on *kdcsrv.austin.ibm.com* using port 812.

1. Configure the client using `config.krb5` command first.

```
# config.krb5 -C -r MYREALM -c kdcsrv.austin.ibm.com -s \
  kdcsrv.austin.ibm.com -d austin.ibm.com
```

2. Edit the **krb5.conf** file and change the port number. In the following example the port number 812 is used.

```
admin_server = kdcsrv.austin.ibm.com:812
```

3. Run **mkkrb5clnt** to complete the configuration.

```
# mkkrb5clnt -c kdcsrv.austin.ibm.com -r MYREALM -a admin/admin \
  -s kdcsrv.austin.ibm.com -d austin.ibm.com -A -i files -K -T
```

## 7.14 How to Destroy Kerberos Credentials

Each time a user logs in the previous Kerberos credentials are overwritten. However, when a user logs out the credentials do not get destroyed. Network Authentication Service **kdestroy** command removes the credential obtained through Kerberos integrated login.

```
$ /usr/krb5/bin/klist
Ticket cache: FILE:/var/krb5/security/creds/krb5cc_foo@MYREALM_205
Default principal: foo@MYREALM

Valid starting    Expires          Service principal
04/22/05 09:31:11 04/23/05 09:31:10  krbtgt/MYREALM@MYREALM
04/22/05 09:31:13 04/22/05 12:31:13  kadmin/admin@MYREALM /kdestroy
```

Invoke **kdestroy** to clean the Kerberos credentials

```
$ /usr/krb5/bin/kdestroy
```

```
$ /usr/krb5/bin/klist
Unable to get cache name (ticket cache:
/var/krb5/security/creds/krb5cc_foo@MYREALM_205).
Status 0x96c73ac3 - No credentials cache found.
```

## 7.15 Changing Ticket Life Time on KDC

You can change the maximum time period for which a ticket is valid in **kdc.conf** file. Modify the `max_life` to a new value and stop and start **krb5kdc** and **kadmind** daemons.

Example:

```
max_life = 8h 0m 0s
```

Also modify the `max_life` of **krbtgt/MYREALM** and **kadmin/admin** principals to the same value.

```
# kadmin.local
kadmin.local: modify_principal -maxlife "8 hours" krbtgt/MYREALM
Principal "krbtgt/MYREALM@MYREALM" modified.

kadmin.local: modify_principal -maxlife "8 hours" kadmin/admin
Principal "kadmin/admin@MYREALM" modified.
```

## 7.16 What will Happen if Kadmind Daemon is Not Available

If the **kadmind** server is not available, the authentication may take longer. The authentication may even fail in the case where the part of the network in which **kadmind** daemon resides is not reachable or the system hosting the **kadmind** server is down. Setting the **kadmind** option to `no` in the **methods.cfg** file eliminates the delays during authentication when the machine is not accessible. See section 6.2 for details.

When the **kadmind** daemon is down, users cannot log in if their passwords are expired. Expired passwords need to be changed. Password changes require the availability of the **kadmind** daemon. Consequently, users who require a password change or have expired passwords, will not be able to log in when the **kadmind** daemon is down.

If the **kadmind** daemon is not available (the daemon is down or not reachable), the **mkuser** command gives the following error:

```
3004-694 Error adding "krb5user": You do not have permission.
```

Additionally, the **chuser** and **lsuser** commands will manage only AIX related attributes, not Kerberos related attributes. The **rmuser** command will not delete the Kerberos principal and the **passwd** command will fail for Kerberos authenticated users.

When the **kadmind** daemon is not available, the root user will not be able to change user passwords. In a situation such as a forgotten password, you must make the **kadmind** daemon available. Also, if a user chooses to enter a Kerberos principal name at the login prompt, the primary name of the principal name will be truncated according to the AIX user name length limitation. This truncated name will be used for AIX user identification information retrieval (for example, to retrieve your home directory value).

## 7.17 Configuring AIX for Kerberos Integrated Login with LDAP AIX User/Group Management.

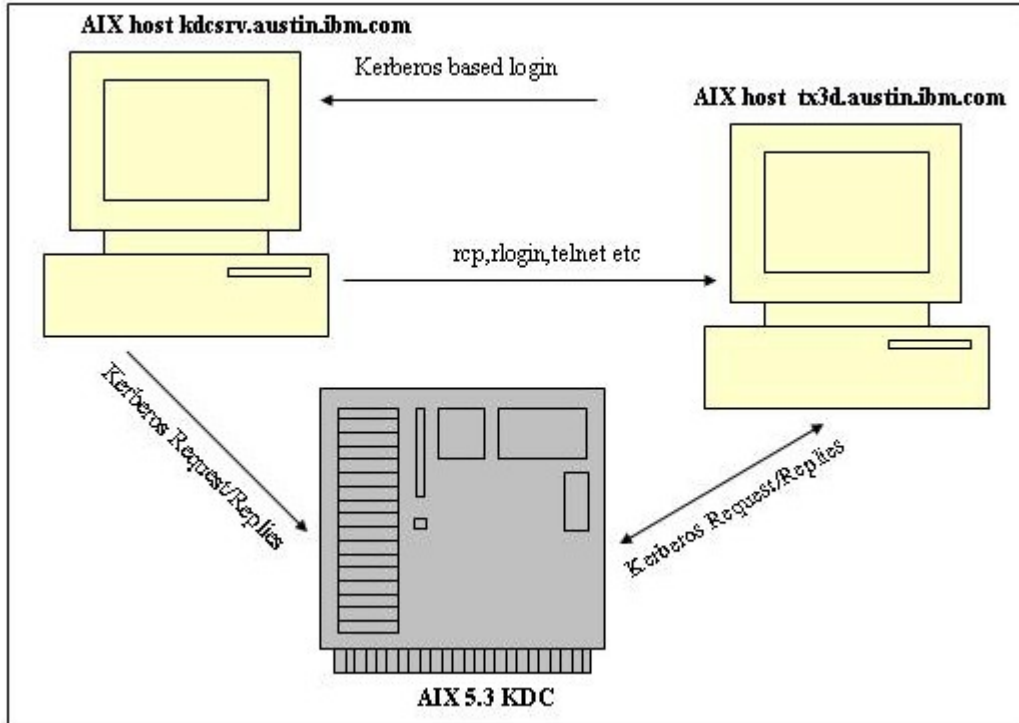
If LDAP is planned to store AIX user/group information, **mksecdap** command must be executed to configure the LDAP server and client before executing **mkkrb5srv** and **mkkrb5clnt**. Refer to Section 5.2.1.1 for configuring the LDAP server using **mksecdap**. LDAP client configuration is also done using **mksecdap**. To configure the Kerberos servers use **mkkrb5srv**. See Section 5.1 for how to do this. Finally configure the Kerberos client using the **mkkrb5clnt** command with **-i** LDAP option. The following example demonstrates the Kerberos client configuration for Kerberos integrated login with LDAP as the AIX user/group repository:

```
# mkkrb5clnt -r MYREALM -c kdcsrv.ustin.ibm.com -s kdcsrv.austin.ibm.com -a
admin/admin -d austin.ibm.com -A -i LDAP -K -T
```

## 7.18 How to Use Kerberized r-cmds After a Successful Login

When an AIX user authenticates to the system using Kerberos, the resulting TGT can be used for Kerberized r-cmds. Kerberos is a single-sign-on system. Once the user has Kerberos credentials, the credentials can be passed on to certain commands such as login and telnet and authenticate without entering a password. This requires the system to be configured for Kerberized r-cmds. In the following example, the Network Authentication Service server is configured on *kdcsrv.austin.ibm.com* using **mkkrb5srv** command. The same system is also configured for Kerberos based logins using **mkkrb5clnt**. A second system *tx3d.austin.ibm.com* is configured as a client using **mkkrb5clnt**.

```
[tx3d] # mkkrb5clnt -r MYREALM -c kdcsrv.ustin.ibm.com \
-s kdcsrv.austin.ibm.com -a admin/admin -d austin.ibm.com -A -i files -K -T
```



**Figure 3: Sample configuration for Kerberized r-cmds.**

The keys for host principal **host/tx3d.austin.ibm.com** need to be saved to file **/etc/krb5/krb5.keytab** on tx3d. Since we have used **mkkrb5clnt** to configure the client machine these keys have already been extracted to **/var/krb5/security/keytab/tx3d.austin.ibm.com.keytab**. Link this file to **/etc/krb5/krb5.keytab** file.

```
# ln -s /var/krb5/security/keytab/tx3d.austin.ibm.com.keytab
/etc/krb5/krb5.keytab
```

Note: If *tx3d.austin.ibm.com* is configured using **config.krb5** instead of **mkkrb5clnt**, then we have to explicitly create a host principal and extract the keys. The following example demonstrates how to do this.

```
[tx3d] # kadmin -p admin/admin
Authenticating as principal admin/admin with password.
Password for admin/admin@MYREALM:

kadmin: addprinc -randkey host/tx3d.austin.ibm.com
WARNING: no policy specified for host/tx3d.austin.ibm.com@MYREALM;
defaulting to no policy. Note that policy may be overridden by
ACL restrictions.
Principal "host/tx3d.austin.ibm.com@MYREALM" created.

kadmin: ktadd -k /etc/krb5/krb5.keytab host/tx3d.austin.ibm.com
Entry for principal host/tx3d.austin.ibm.com with kvno 3, encryption type
Triple DES cbc mode with HMAC/sha1 added to keytab
WRFIELD:/etc/krb5/krb5.keytab.
Entry for principal host/tx3d.austin.ibm.com with kvno 3, encryption type
ArcFour with HMAC/md5 added to keytab WRFIELD:/etc/krb5/krb5.keytab.
```

```
Entry for principal host/tx3d.austin.ibm.com with kvno 3, encryption type AES-
256 CTS mode with 96-bit SHA-1 HMAC added to keytab
WRFILe:/etc/krb5/krb5.keytab.
Entry for principal host/tx3d.austin.ibm.com with kvno 3, encryption type DES
cbc mode with RSA-MD5 added to keytab WRFILe:/etc/krb5/krb5.keytab.
kadmin:
```

Note that since kadmin tool is invoked from tx3d, the keys are extracted to **/etc/krb5/krb5.keytab** file on tx3d. This step can also be done on the machine that hosts Kerberos admin server (i.e. kdcsrv). After extracting the keys into a keytab file, the file is transferred and merged with the **/etc/krb5/krb5.keytab** file on tx3d.

Enable r-cmds on both systems to use Kerberos Version 5 authentication.

On tx3d.austin.ibm.com:

```
[tx3d] # lsauthent
Standard Aix
[tx3d] # chauthent -k5 -std
[tx3d] # lsauthent
Kerberos 5
Standard Aix
```

On kdcsrv.austin.ibm.com:

```
[kdcsrv] # chauthent -k5 -std
[kdcsrv] # lsauthent
Kerberos 5
Standard Aix
```

Create a Kerberos authenticated user **foo** on *kdcsrv* and set the password.

```
[kdcsrv] # mkuser -R KRB5files SYSTEM=KRB5files registry=KRB5files foo
[kdcsrv] # passwd -R KRB5files foo
```

Create the user **foo** on tx3d:

```
[tx3d] # mkuser -R files foo
```

Now telnet to *kdcsrv.austin.ibm.com* using Kerberos authentication. Make sure that we got a TGT by invoking **klist** command. We shall use the TGT we obtain after a successful login to invoke remote commands on *tx3d.austin.ibm.com*

```
telnet kdcsrv
Trying...
Connected to kdcsrv.austin.ibm.com.
Escape character is '^'.
```

```
telnet (kdcsrv.austin.ibm.com)
```

```
AIX Version 5
(C) Copyrights by IBM and by others 1982, 2004.
login: foo
foo'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.
*
*
*****

```

```

$ /usr/krb5/bin/klist
Ticket cache: FILE:/var/krb5/security/creds/krb5cc_foo@MYREALM_203
Default principal: foo@MYREALM

```

```

Valid starting    Expires          Service principal
04/28/05 14:05:50 04/29/05 14:05:50  krbtgt/MYREALM@MYREALM
04/28/05 14:05:52 04/28/05 17:05:52  kadmin/admin@MYREALM
$

```

*Example: Execute 'date' command on remote host tx3d using Kerberized rsh command*

---

```

$ rsh tx3d date
Thu Apr 28 14:24:16 CDT 2005

```

---

*Example: Login to remote host tx3d using Kerberized rlogin command*

---

```

$ hostname
kdcsrv.austin.ibm.com
$ rlogin tx3d -l foo
*****
*
*
*   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.
*
*
*****

$ hostname
tx3d.austin.ibm.com
$ id
uid=234(foo) gid=1(staff)

```

---

*Example: Transfer a file to remote host tx3d using Kerberized rcp command*

---

```

[kdcsrv] $ rsh tx3d "ls -l /home/foo"
total 0
[kdcsrv] $ echo "Testing Kerberize-d rcp" >> xfile
[kdcsrv] $ rcp xfile tx3d:/home/foo
[kdcsrv] $ rsh tx3d "ls -l /home/foo"
total 0
-rw-r--r--  1 foo      staff          0 Apr 28 14:30 xfile
[kdcsrv] $ rsh tx3d "more /home/foo/xfile"

```

```
Testing Kerberize-d rcp
$
```

---

*Example: Telnet to the remote host tx3d using Kerberos credentials*

---

```
[kdcsrv] $ telnet tx3d
Trying...
Connected to tx3d.austin.ibm.com.
Escape character is '^]'.
[ Kerberos V5 accepts you as ``foo@MYREALM'' ]

telnet (tx3d.austin.ibm.com)

*****
*
*
* 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.
*
*
*****

$ hostname
tx3d.austin.ibm.com
$ id
uid=234(foo) gid=1(staff)
$
```

---

Notice that in the above example no password information is asked.

Note: Remove **.klogin**, **.rhost** or **hosts.equiv** files before running the examples above.

Before using Kerberized ftp, ftp service principal **ftp/tx3d.austin.ibm.com** needs to be created and extracted into the keytab file **/etc/krb5/krb5.keytab** file. This can be done by invoking **kadmin** command from *tx3d.austin.ibm.com*.

```
kadmin: addprinc -randkey ftp/tx3d.austin.ibm.com@MYREALM
WARNING: no policy specified for ftp/tx3d.austin.ibm.com@MYREALM;
defaulting to no policy. Note that policy may be overridden by
ACL restrictions.
Principal "ftp/tx3d.austin.ibm.com@MYREALM" created.

kadmin: ktadd -k /etc/krb5/krb5.keytab ftp/tx3d.austin.ibm.com@MYREALM
Entry for principal ftp/tx3d.austin.ibm.com@MYREALM with kvno 3, encryption
type Triple DES cbc mode with HMAC/sha1 added to keytab
WRFILE:/etc/krb5/krb5.keytab.
Entry for principal ftp/tx3d.austin.ibm.com@MYREALM with kvno 3, encryption
type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
Entry for principal ftp/tx3d.austin.ibm.com@MYREALM with kvno 3, encryption
type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab
WRFILE:/etc/krb5/krb5.keytab.
Entry for principal ftp/tx3d.austin.ibm.com@MYREALM with kvno 3, encryption
type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/krb5/krb5.keytab.
kadmin:
```

### Example: ftp to the remote host tx3d using Kerberos credentials

---

```
$ ftp tx3d
Connected to tx3d.austin.ibm.com.
220 tx3d.austin.ibm.com FTP server (Version 4.2 Fri Feb 25 12:30:44 CST 2005)
ready.
334 Using authentication type GSSAPI; ADAT must follow
GSSAPI accepted as authentication type
2005-05-19-18:01:19.403-05:00I----- PID#675852 ERROR gss authentication
gssapi.c 4003 0x00000001 msgID=0x128620FF
Could not retrieve default login context (171220ec)
GSSAPI authentication succeeded
Name (tx3d:foo): foo
232 GSSAPI user foo@MYREALM is authorized as foo
230-Last login: Thu May 19 17:58:57 CDT 2005 on ftp from kdcsrv.austin.ibm.com
230 User foo logged in.
ftp> ftp> ls -la
200 PORT command successful.
150 Opening data connection for /bin/ls.
total 5
drwxr-xr-x  2 foo      staff      512 May 19 17:46 .
drwxr-xr-x  9 bin      bin        512 May 19 17:04 ..
-rwxr----- 1 foo      staff      254 May 19 17:04 .profile
-rw-----  1 foo      staff      46  May 19 17:47 .sh_history
-rw-r--r--  1 foo      staff      24  May 19 17:46 xfile
226 Transfer complete.
ftp>
```

---

## 8 References

1. Steiner, J. G., Neuman, C. and Jeffrey I. Schiller, “Kerberos: An Authentication Service for Open Network Systems.” in Usenix Conference Proceedings, Dallas, Texas, February 1998.
2. Kohl, J. and C. Neuman, “The Kerberos Network Authentication Service (V5)”, RFC 1510.
3. IBM Network Authentication Service Administrator’s and User’s Guide. Network Authentication Service documentation is provided in the **krb5.doc.lang.[pdf/html]** package, where *lang* represents the supported language (i.e. krb5.doc.en\_US.pdf). After the installation one can locate the documents in /usr/lpp/krb5/doc/pdf/en\_US.
4. IBM Tivoli Directory Server Installation and Configuration Guide, Version 5.2, SC32-1338-00
5. AIX 5L Version 5.3 Security Guide: LDAP exploitation of the Security Subsystem. <http://publib.boulder.ibm.com/infocenter/pseries/topic/com.ibm.aix.doc/aixbman/securit/security.pdf>
6. AIX client setup white paper for LDAP user/group management. “Configuring an AIX Client System for User Authentication and Management Through LDAP” [http://www.ibm.com/servers/aix/whitepapers/ldap\\_client.html](http://www.ibm.com/servers/aix/whitepapers/ldap_client.html)
7. “Configuring an IBM Directory Server for User Authentication and Management in AIX 5L V5.2” [http://www.ibm.com/servers/aix/whitepapers/ldap\\_server.html](http://www.ibm.com/servers/aix/whitepapers/ldap_server.html)

8. AIX 5L Version 5.3 Commands Reference, Volume 1
9. AIX 5L Version 5.3 Commands Reference, Volume 2
10. AIX 5L Version 5.3 Commands Reference, Volume 3
11. AIX 5L Version 5.3 Commands Reference, Volume 4
12. AIX 5L Version 5.3 Files Reference.
13. AIX KRB5A setup white paper. "Configuring AIX 5L for Kerberos Based Authentication using Windows Kerberos Service". To be published.

## APPENDIX A

The syntax of the grammar used for the **SYSTEM** attribute value is given below in BNF notation:

```

system_attr ::= SYSTEM = expression
expression ::= primitive |
              ( expression ) |
              expression op expression
primitive   ::= method |
              method [ result ]
result      ::= SUCCESS | FAILURE | NOTFOUND | UNAVAIL | *
op          ::= AND | OR
method      ::= compat | files | NONE | [a-z,A-Z,0-9]*

```

Each method callable from the **SYSTEM** attribute must return one of the following:

**SUCCESS**: indicates that the user successfully authenticated to that method

**FAILURE**: indicates that the user did not authenticate correctly

**UNAVAIL**: indicates the method can not determine the authentication status, probably due to unavailability of other services.

**NOTFOUND**: indicates this user is not known to the method

One can view this grammar as an **if** clause with short circuit semantics. The return code from a method invocation is compared against the expected result specified in the grammar. The result of this comparison determines if the evaluation of the grammar has to continue or stop with respect to short circuit rules. If the grammar evaluates to true then the user authentication succeeds.

When no expected result is specified for a method the implied behavior of the method is to return **SUCCESS**. For instance if the **SYSTEM** attribute has the value of **SYSTEM=LDAP**, then this implies **SYSTEM=LDAP[SUCCESS]** .

The methods are combined using the **AND** or **OR** logical connectors. When two methods are combined with **AND**, this indicates that both methods must evaluate to **true** for the clause to be true. If the left method evaluates to **false** then the right method is not invoked. In the case of **OR**, it suffices that one method to evaluate true for the whole

clause to be true. If the left method evaluates to **true** then the second method is not invoked

Following are some examples of SYSTEM attribute:

```
SYSTEM=DCE OR (DCE[UNAVAIL] AND compat[SUCCESS])
```

This grammar is interpreted as follows: First invoke the DCE method. If it returns success then the authentication process is completed successfully. Otherwise if the DCE is system is unavailable say due to a registry problem then try compat (this means AIX uses crypt()). If DCE method fails for any other reasons then the authentication fails.

```
SYSTEM=KRB5files OR compat
```

This grammar indicates that if the user authenticates successfully using Kerberos then the overall authentication succeeds. In case of failure, AIX standard authentication is tried and the result of this determines the overall authentication status.

Certain methods are used for non-authentication purposes. For instance a method may be used only for getting user credentials thus does not need to be involved in the authentication decision. For instance in the grammar below the sole purpose of GETCRED method is to acquire some type of credentials.

```
SYSTEM = compat AND GETCRED[*]
```

Therefore if the compat method successfully authenticates the user then the result of GETCRED is ignored and the user authentication succeeds. Note that it is very dangerous to use \* with **OR** operator as this could potentially allow access a user to the system who had failed the valid authentication methods. The return value from a method that has \* as its expected result is treated as SUCCESS.

The methods **compat**, **files** and **NONE** are known to the system. Any other method needs to be defined in `/usr/lib/security/methods.cfg` in order to be usable by AIX's I & A subsystem. For example when the method name is LDAP, the following entry needs to be added into methods.cfg file

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

Example:

```
SYSTEM = LDAP or compat
```

The description of methods.cfg file is given in [11].

## APPENDIX B

The Kerberos integrated login client configuration command syntax is given below. For details on storing Kerberos information on LDAP consult [3].

```
mkkrb5clnt -h | -r <realm> { -c <KDC> -s <server> | -l {ldapservice | ldapservice:port} [-c <KDC> -s <server>] } [-a <admin> ] -d <domain> [-A] [-i <data base>] [-K] [-T] | -i <data base> | -U [-a <admin>]
```

- h** Specifies that the command is only to display the valid command syntax.
- r *realm*** Specifies the full realm name for which the Kerberos client is to be configured
- a *admin*** Specifies the principal name of the Kerberos server admin.
- c *KDC*** Specifies the KDC server.
- s *server*** Specifies the fully qualified host name for Kerberos admin server.
- d *domain*** Specifies the complete domain name for the Kerberos client.
- l *ldapservice / ldapservice:port*** Specifies the LDAP directory server to use for Administration server and KDC discovery using LDAP. If the **-l** flag is used, then the KDC and server flags are optional. If the **-l** option is not used, the KDC and server flags must be specified. The port number can optionally be specified. If the port number is not specified, the client connects to the default LDAP server port 389 or 636 for SSL connections.
- i *database*** Configures integrated Kerberos authentication. This creates the compound load module KRB5<database> entry in **methods.cfg** file. If local file system is used to store AIX user and group information specify this as “**files**”. In this case the user/group identification is performed by lookups to local file system.
- K** Specifies Kerberos to be configured as the default authentication scheme. When this option is specified, the **SYSTEM** line in **/etc/security/user** default stanza would be updated.
- A** Specifies root to be added as a Network Authentication Service administrative user. Root account on AIX does not have any special privileges on the Network Authentication Service servers. When this option is specified root/<fully qualified host name> would be created

and this principal would be given administrative privileges.

- T** Specifies the flag to acquire server admin TGT based admin ticket.
- U** Undo the setup from the previous configuration command. It only undoes the Kerberos integrated login related configuration. The system still remains a Kerberos client.

## Error Messages and Recovery Actions

Errors that can occur when using the **mkkrb5clnt** command include the following:

- You will receive a warning message if **krb5.conf** file exists.
- Incorrect values for **krb5.conf** can be fixed by editing the **/etc/krb5/krb5.conf** file.
- Incorrect values for the **-i** flag can be fixed by editing the **/usr/lib/security/methods.cfg** file.
- Incorrect values for the **-K** flag can be fixed running **chsec** command or modifying the default value for SYSTEM attribute by editing **/etc/security/user** file. Example: **chsec -f /etc/security/user -s default -a SYSTEM="KRB5files OR compat"** changes the default SYSTEM value to **KRB5files OR compat**.
- To perform a cleanup and start over use **mkkrb5clnt -U -a admin/admin** command.

The Kerberos integrated login server configuration command syntax is given below. For details on storing Kerberos information on LDAP and LDAP bind types consult [3].

```
mkkrb5srv -h | {-r <realm> -s <server> -d <domain> [-a <admin name>] [[-l {  
ldapsrv | ldapsrv:port }]] [-u ldap_DN -p ldap_DN_pw] [-f {keyring |  
keyring:entry_dn} -k keyring_pw] [-m masterkey_location] [-b bind_type]] | -U }
```

- a** *admin name* Specifies the Kerberos Principal name for the administrator.
- b** *bind\_type* Specifies the LDAP bind type. Supported values are simple, cram-md5 and external. These bind types can be specified in either upper case or lower case.
- d** *domain* Specifies the domain name for the Kerberos realm.
- f** {*keyring* / *keyring:entry\_dn*} Specifies the LDAP keyring database file name if you are using SSL communication.
- h** Specifies that the command is only to display the valid command

syntax.

**-k** *keyring\_pw* Specifies the password for the LDAP keyring database file. If not specified, SSL uses the password that is encrypted in the appropriate password stash file.

**-l** *ldapservers / ldapservers:port* For servers, specifies the LDAP directory used to store the Network Authentication Service principal and policy information.

The port number can optionally be specified. If the port number is not specified, the client connects to the default LDAP server port 389 or 636 for SSL connections.

**-m** *masterkey\_location* Specifies the fully qualified file name for storing the master key in the local file system when using LDAP to store data.

**Note:**

This flag is only for use with the LDAP directory.

**-r** *realm* Specifies the realm for which the Kerberos server is to be configured.

**-s** *server* Specifies the fully qualified name of Kerberos Admin Server.

**-u** *ldap\_DN* Specifies the LDAP entry to be used as the LDAP bind DN.

**Note:**

With external bind, the **-u** and **-p** flags are not required, and the values come from the certificate. In this method the certificates are used for the Kerberos servers to authenticate their identities to LDAP server and vice versa.

**-p** *ldap\_DN\_pw* Specifies the password for the LDAP bind DN.

**-U** Undo the setup from the previous configuration command

## Error Messages and Recovery Actions

Potential errors that can occur when using the **mkkrb5srv** command are described below:

- If a previous server configuration exists, you receive a message asking to remove the existing configuration. Respond 'n' if you want to keep the existing configuration.
- If the **krb5.conf** file already exists, you receive a message that a previous configuration exists and asks to remove it. Respond 'n' if you want to keep the existing configuration.
- Editing the **krb5.conf**, **kdc.conf**, or **kadm5.acl** files can change configuration values. If you modify these files, stop and start **kadmind** and **krb5kdc** for the new values to take affect.
- If you mistype something and no database is created, remove the configuration files and the database created and rerun the command. This can be done using **mkkrb5srv -U** command.
- Make sure the **kadmind** and the **krb5kdc** daemons are started on your machine. Use the **ps** command to verify that the daemons are running. If these daemons have not started, check the log file. You can manually start these daemons by typing `/usr/krb5/sbin/krb5kdc` and `/usr/krb5/sbin/kadmind`



© IBM Corporation 2006  
IBM Corporation  
Systems and Technology Group  
Route 100  
Somers, New York 10589

Produced in the United States of America  
February 2006  
All Rights Reserved

This document was developed for products and/or services offered in the United States. IBM may not offer the products, features, or services discussed in this document in other countries.

The information may be subject to change without notice. Consult your local IBM business contact for information on the products, features and services available in your area.

All statements regarding IBM future directions and intent are subject to change or withdrawal without notice and represent goals and objectives only.

IBM, the IBM logo, AIX, AIX 5L, DB2, Tivoli are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both. A full list of U.S. trademarks owned by IBM may be found at: <http://www.ibm.com/legal/copytrade.shtml>.

UNIX is a registered trademark of The Open Group in the United States, other countries or both.

POSIX is a registered trademark of IEEE.

Microsoft and Windows are registered trademarks of the Microsoft Corporation.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. In the United States and/or other countries.

Kerberos and Project Athena are trademarks of Massachusetts Institute of Technology.

Other company, product, and service names may be trademarks or service marks of others.

Copying or downloading the images contained in this document is expressly prohibited without the written consent of IBM.

Information concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of the non-IBM products should be addressed with those suppliers.

The IBM home page on the Internet can be found at: <http://www.ibm.com>.

The IBM System p5 and @server p5 home page on the Internet can be found at: <http://www.ibm.com/systems/p>.