Kerberos Operation Notes
DRAFT

Bill Bryant
John Kohl
Project Athena, MIT

Initial Release, January 24, 1989
(plus later patches through patchlevel 7)

These notes assume that you have used the *Kerberos Installation Notes* to build and install your Kerberos system. As in that document, we refer to the directory that contains the built Kerberos binaries as [OBJ_DIR].

This document assumes that you are a Unix system manager.

1. How Kerberos Works: A Schematic Description

This section provides a simplified description of a general user's interaction with the Kerberos system. This interaction happens transparently--users don't need to know and probably don't care about what's going on--but Kerberos administrators might find a schematic description of the process useful. The description glosses over a lot of details; for more information, see *Kerberos: An Authentication Service for Open Network Systems*, a paper presented at Winter USENIX 1988, in Dallas, Texas.

1.1 Network Services and Their Client Programs

In an environment that provides network services, you use *client* programs to request service from *server* programs that are somewhere on the network. Suppose you have logged in to a workstation and you want to *rlogin* to another machine. You use the local *rlogin* client program to contact the remote machine's *rlogin* service daemon.

1.2 Kerberos Tickets

Under Kerberos, the *rlogin* service program allows a client to login to a remote machine if it can provide a Kerberos **ticket** for the request. This ticket proves the identity of the person who has used the client program to access the server program.

1.3 The Kerberos Master Database

Kerberos will give you tickets only if you have an entry in the Kerberos server's **master database**. Your database entry includes your Kerberos username (often referred to as your Kerberos **principal** name), and your Kerberos password. Every Kerberos user must have an entry in this database.

1.4 The Ticket-Granting Ticket

The *kinit* command prompts for your Kerberos username and password, and if you enter them successfully, you will obtain a Kerberos *ticket-granting ticket*. As illustrated below, client programs use this ticket to get other Kerberos tickets as needed.

1.5 Network Services and the Master Database

The master database also contains entries for all network services that require Kerberos authentication. Suppose for instance that your site has a machine *laughter* that requires Kerberos authentication from anyone who wants to *rlogin* to it. This service must be registered in the master database. Its entry includes the service's principal name, and its **instance**.

The *instance* is the name of the service's machine; in this case, the service's instance is the name *laughter*. The instance provides a means for Kerberos to distinguish between machines that provide the same service. Your site is likely to have more than one machine that provides *rlogin* service.

1.6 The User-Kerberos Interaction

Suppose that you (in the guise of a general user) walk up to a workstation intending to login to it, and then *rlogin* to the machine *laughter*. Here's what happens.

1. You login to the workstation and use the *kinit* command to to get a ticket-granting ticket. This command prompts you for your username (your Kerberos Principal Name), and your Kerberos password [on some systems which use the new version of */bin/login*, this may be done as part of the login process, not requiring the user to run a separate program].
   a. The *kinit* command sends your request to the Kerberos master server machine. The server software looks for your principal name's entry in the Kerberos **master database**.
   b. If this entry exists, the Kerberos server creates and returns a *ticket-granting ticket*, encrypted in your password. If *kinit* can decrypt the Kerberos reply using the password you provide, it stores this ticket in a **ticket file** on your local machine for later use. The ticket file to be used can be specified in the **KRBTKFILE** environment variable. If this variable is not set, the name of the file will be */tmp/tkt**uid***, where ***uid*** is the UNIX user-id, represented in decimal.
2. Now you use the *rlogin* client to try to access the machine *laughter*.

   `host%` **rlogin laughter**

   a. The *rlogin* client checks your ticket file to see if you have a ticket for *laughter*'s *rcmd* service (the rlogin program uses the *rcmd* service name, mostly for historical reasons). You don't, so *rlogin* uses the ticket file's *ticket-granting ticket* to make a request to the master server's ticket-granting service.
   b. This ticket-granting service receives the *rcmd-laughter* request and looks in the master database for an *rcmd-laughter* entry. If that entry exists, the ticket-granting service issues you a ticket for that service. That ticket is also cached in your ticket file.
   c. The *rlogin* client now uses that ticket to request service from the *laughter rlogin* service program. The service program lets you *rlogin* if the ticket is valid.

2. Setting Up and Testing the Kerberos Server

The procedure for setting up and testing a Kerberos server is as follows:

1. Use the *kdb_init* command to create and initialize the master database.
2. Use the *kdb_edit* utility to add your username to the master database.
3. Start the Kerberos server.
4. Use the *kinit* command to obtain a Kerberos ticket-granting ticket.
5. Use the *klist* command to verify that the *kinit* command authenticated you successfully.

2.1 Creating and Initializing the Master Database

Login to the Kerberos master server machine, and use the **su** command to become root. If you installed the Kerberos administration tools with the *make install* command and the default pathnames, they should be in the */usr/etc* directory. If you installed the tools in a different directory, hopefully you know what it is. From now on, we will refer to this directory as [ADMIN_DIR].

The *kdb_init* command creates and initializes the master database. It asks you to enter the system's realm name and the database's master password. Do not forget this password. If you do, the database becomes useless. (Your realm name should be substituted for [REALMNAME] below.)

Use *kdb_init* as follows:

```
host# [ADMIN_DIR]/kdb_init
Realm name (default XXX): [REALMNAME]          <-- Enter your system's realm name.
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.

Enter Kerberos master key:     <-- Enter the master password.
```

2.2 Storing the Master Password

The *kstash* command ''stashes'' the master password in the file */.k* so that the Kerberos server can be started automatically during an unattended reboot of the master server. Other administrative programs use this hidden password so that they can access the master database without someone having to manually provide the master password. This command is an optional one; if you'd rather enter the master password each time you start the Kerberos server, don't use *kstash*.

One the one hand, if you use *kstash*, a copy of the master key will reside on disk which may not be acceptable; on the other hand, if you don't use *kstash*, the server cannot be started unless someone is around to type the password in manually.

The command prompts you twice for the master password:

```
host# [ADMIN_DIR]/kstash

Enter Kerberos master key:     <-- Enter the master password.
Current Kerberos master key version is 1.

Master key entered   BEWARE!
```

A note about the Kerberos database master key: if your master key is compromised and the database is obtained, the security of your entire authentication system is compromised. The master key must be a carefully kept secret. If you keep backups, you must guard all the master keys you use, in case someone has stolen an old backup and wants to attack users' whose passwords haven't changed since the backup was stolen. This is why we provide the option not to store it on disk.

2.3 Using *kdb_edit* to Add Users to the Master Database

The *kdb_edit* program is used to add new users and services to the master database, and to modify existing database information. The program prompts you to enter a principal's **name** and **instance**.

A principal name is typically a username or a service program's name. An instance further qualifies the principal. If the principal is a service, the instance is used to specify the name of the machine on which that service runs. If the principal is a username that has general user privileges, the instance is usually set to null.

The following example shows how to use *kdb_edit* to add the user *wave* to the Kerberos database.

```
host# [ADMIN_DIR]/kdb_edit

Opening database...

Enter Kerberos master key:
Verifying, please re-enter
Enter Kerberos master key:
Current Kerberos master key version is 1

Master key entered.  BEWARE!
Previous or default values are in [brackets] ,
enter return to leave the same, or new value.

Principal name: wave              <--  Enter the username.
Instance:                         <-- Enter a null instance.

<Not found>, Create [y] ? y       <--  The user-instance does not exist.
                                       Enter y to create the user-instance.
Principal: wave  Instance:  m_key_v: 1
New Password:                     <-- Enter the user-instance's password.
Verifying, please re-enter
New Password:
Principal's new key version = 1
Expiration date (enter dd-mm-yy) [ 12/31/99 ] ?<--  Enter newlines
Max ticket lifetime (*5 minutes) [ 255 ] ?     <--  to get the
Attributes [ 0 ] ?                  <--  default values.
Edit O.K.

Principal name:                   <-- Enter a newline to exit the program.
```

Use the *kdb_edit* utility to add your username to the master database.

2.4 Starting the Kerberos Server

Change directories to the directory in which you have installed the server program *kerberos* (the default directory is */usr/etc*), and start the program as a background process:

```
host# ./kerberos &
```

If you have used the *kstash* command to store the master database password, the server will start automatically. If you did not use *kstash*, use the following command:

```
host# ./kerberos -m
```

The server will prompt you to enter the master password before actually starting itself.

2.5 Testing the Kerberos Server

Exit the root account and use the *kinit* command obtain a Kerberos ticket-granting ticket. This command creates your ticket file and stores the ticket-granting ticket in it.

If you used the default *make install* command and directories to install the Kerberos user utilities, *kinit* will be in the */usr/athena* directory. From now on, we'll refer to the Kerberos user commands directory as [K_USER].

Use *kinit* as follows:

```
host% [K_USER]/kinit
MIT Project Athena, (ariadne)
Kerberos Initialization
Kerberos name: yourusername        <--  Enter your Kerberos username.
Password:                          <--  Enter your Kerberos password.
```

Use the *klist* program to list the contents of your ticket file.

```
host% [K_USER]/klist
```

The command should display something like the following:

```
Ticket file:    /tmp/tkt5555
Principal:      yourusername@REALMNAME

  Issued           Expires          Principal
May  6 10:15:23  May  6 18:15:23  krbtgt.REALMNAME@REALMNAME
```

If you have any problems, you can examine the log file */kerberos/kerberos.log* on the Kerberos server machine to see if there was some sort of error.

3. Setting up and testing the Administration server

The procedure for setting up and testing the Kerberos administration server is as follows:

1. Use the *kdb_edit* utility to add your username with an administration instance to the master database.
2. Edit the access control lists for the administration server
3. Start the Kerberos administration server.
4. Use the *kpasswd* command to change your password.
5. Use the *kadmin* command to add new entries to the database.
6. Use the *kinit* command to verify that the *kadmin* command correctly added new entries to the database.

3.1 Adding an administration instance for the administrator

Login to the Kerberos master server machine, and use the **su** command to become root. Use the *kdb_edit* program to create an entry for each administrator with the instance ‘‘***admin***’’.

```
host# [ADMIN_DIR]/kdb_edit

Opening database...

Enter Kerberos master key:
Verifying, please re-enter
Enter Kerberos master key:
Current Kerberos master key version is 1

Master key entered.  BEWARE!
Previous or default values are in [brackets] ,
enter return to leave the same, or new value.

Principal name: wave            <--  Enter the username.
Instance:admin                  <--  Enter ‘‘admin’’.

<Not found>, Create [y] ? y     <--  The user-instance does not exist.
                                     Enter y to create the user-instance.
Principal: wave  Instance: admin m_key_v: 1
New Password:                   <-- Enter the user-instance’s password.
Verifying, please re-enter
New Password:
Principal’s new key version = 1
Expiration date (enter dd-mm-yy) [ 12/31/99 ] ?<--  Enter newlines
Max ticket lifetime (*5 minutes) [ 255 ] ?     <--  to get the
Attributes [ 0 ] ?                <--  default values.
Edit O.K.

Principal name:                 <-- Enter a newline to exit the program.
```

3.2 The Access Control Lists

The Kerberos administration server uses three access control lists to determine who is authorized to make certain requests. The access control lists are stored on the master Kerberos server in the same directory as the principal database, */kerberos*. The access control lists are simple ASCII text files, with each line specifying the name of one principal who is allowed the particular function. To allow several people to perform the same function, put their principal names on separate lines in the same file.

The first list, */kerberos/admin_acl.mod*, is a list of principals which are authorized to change entries in the database. To allow the administrator ‘**wave**’ to modify entries in the database for the realm ‘**TIM.EDU**’, you would put the following line into the file */kerberos/admin_acl.mod*:

```
wave.admin@TIM.EDU
```

The second list, */kerberos/admin_acl.get*, is a list of principals which are authorized to retrieve entries from the database.

The third list, */kerberos/admin_acl.add*, is a list of principals which are authorized to add new entries to the database.

3.3 Starting the administration server

Change directories to the directory in which you have installed the administration server program *kadmind* (the default directory is */usr/etc*), and start the program as a background process:

```
host# ./kadmind -n&
```

If you have used the *kstash* command to store the master database password, the server will start automatically. If you did not use *kstash*, use the following command:

```
host# ./kadmind
```

The server will prompt you to enter the master password before actually starting itself; after it starts, you should suspend it and put it in the background (usually this is done by typing control-Z and then **bg**).

3.4 Testing *kpasswd*

To test the administration server, you should try changing your password with the *kpasswd* command, and you should try adding new users with the *kadmin* command (both commands are installed into */usr/athena* by default).

Before testing, you should exit the root account.

To change your password, run the *kpasswd* command:

```
host% [K_USER]/kpasswd
Old password for wave@TIM.EDU: <--Enter your password
New Password for wave@TIM.EDU: <--Enter a new password
Verifying, please re-enter New Password for wave@TIM.EDU:
                               <--Enter new password again
Password changed.
```

Once you have changed your password, use the *kinit* program as shown above to verify that the password was properly changed.

3.5 Testing *kadmin*

You should also test the function of the *kadmin* program, by adding a new user (here named ‘‘username’’):

```
host% [K_USER]/kadmin
Welcome to the Kerberos Administration Program, version 2
Type "help" if you need it.
admin:  ank username           'ank' stands for Add New Key
Admin password:                <--enter the password
                               you chose above for wave.admin
Password for username:         <--Enter the user's initial password
Verifying, please re-enter Password for username:<--enter it again
username added to database.

admin:  quit
Cleaning up and exiting.
```

3.6 Verifying with *kinit*

Once you've added a new user, you should test to make sure it was added properly by using *kinit*, and trying to get tickets for that user:

```
host% [K_USER]/kinit username
MIT Project Athena (ariadne)
Kerberos Initialization for "username@TIM.EDU"
Password: <--Enter the user's password you used above
host% [K_USER]/klist
Ticket file:    /tmp/tkt_5509_spare1
Principal:      username@TIM.MIT.EDU

  Issued            Expires           Principal
Nov 20 15:58:52  Nov 20 23:58:52  krbtgt.TIM.EDU@TIM.EDU
```

If you have any problems, you can examine the log files */kerberos/kerberos.log* and */kerberos/admin_server.syslog* on the Kerberos server machine to see if there was some sort of error.

4. Setting up and testing slave server(s)

[Unfortunately, this chapter is not yet ready. Sorry. -ed]

5. A Sample Application

This release of Kerberos comes with a sample application server and a corresponding client program. You will find this software in the [OBJ_DIR]*/appl/sample* directory. The file *sample_client* contains the client program's executable code, the file *sample_server* contains the server's executable.

The programs are rudimentary. When they have been installed (the installation procedure is described in detail later), they work as follows:

- The user starts *sample_client* and provides as arguments to the command the name of the server machine and a checksum. For instance:

  host% **sample_client** ***servername 43***

- *Sample_client* contacts the server machine and authenticates the user to *sample_server*.
- *Sample_server* authenticates itself to *sample_client*, then returns a message to the client program. This message contains diagnostic information that includes the user's username, the Kerberos realm, and the user's workstation address.
- *Sample_client* displays the server's message on the user's terminal screen.

5.1 The Installation Process

In general, you use the following procedure to install a Kerberos-authenticated server-client system.

1. Add the appropriate entry to the Kerberos database using *kdb_edit* or *kadmin* (described below).

2. Create a */etc/srvtab* file for the server machine.

3. Install the service program and the */etc/srvtab* file on the server machine.

4. Install the client program on the client machine.

5. Update the */etc/services* file on the client and server machines.

We will use the sample application as an example, although the procedure used to install *sample_server* differs slightly from the general case because the *sample_server* takes requests via the *inetd* program. *Inetd* starts *sample_server* each time a client process contacts the server machine. *Sample_server* processes the request, terminiates, then is restarted when *inetd* receives another *sample_client* request. When you install the program on the server, you must add a *sample* entry to the server machine's */etc/inetd.conf* file.

The following description assumes that you are installing *sample_server* on the machine *ariadne.tim.edu*. Here's the process, step by step:

1. Login as or *su* to root on the Kerberos server machine. Use the *kdb_edit* or *kadmin* program to create an entry for *sample* in the Kerberos database:

```
host# [ADMIN_DIR]/kdb_edit

Opening database...

Enter Kerberos master key:
Verifying, please re-enter
master key entered.  BEWARE!
Previous or default values are in [brackets] ,
enter return to leave the same, or new value.

Principal name: sample <-- Enter the principal name.
Instance: ariadne      <-- Instances cannot have periods in them.

<Not found>, Create [y] ? y

Principal: sample_server  Instance: ariadne m_key_v: 1
New Password:          <-- Enter ''RANDOM'' to get random password.
Verifying, please re-enter
New Password:          <-- Enter ''RANDOM'' again.
Random password [y] ? y

Principal's new key version = 1
Expiration date (enter dd-mm-yy) [ 12/31/99 ] ?
Max ticket lifetime (*5 minutes) [ 255 ] ?
Attributes [ 0 ] ?
Edit O.K.

Principal name:        <-- Enter newline to exit kdb_edit.
```

2. Use the *ext_srvtab* program to create a *srvtab* file for *sample_server*'s host machine:

```
host# [ADMIN_DIR]/ext_srvtab  ariadne

Enter Kerberos master key:
Current Kerberos master key version is 1.

Generating 'ariadne-new-srvtab'....
```

Transfer the *ariadne-new-srvtab* file to *ariadne* and install it as */etc/srvtab*. Note that this file is equivalent to the service's password and should be treated with care. For example, it could be transferred by removable media, but should not be sent over an open network in the clear. Once installed, this file should be readable only by root.

3. Add the following line to the */etc/services* file on *ariadne*, and on all machines that will run the *sample_client* program:

   ```
   sample     906/tcp       # Kerberos sample app server
   ```

4. Add a line similar to the following line to the */etc/inetd.conf* file on *sample_server*'s machine:

   ```
   sample   stream   tcp   nowait   switched   root
       [PATH]/sample_server sample_server
   ```

   where [PATH] should be substituted with the path to the *sample_server* program. (This *inetd.conf* information should be placed on one line.) You should examine existing lines in */etc/inetd.conf* and use the same format used by other entries (e.g. for telnet). Most systems do not have a column for the 'switched' keyword, and some do not have a column for the username (usually 'root', as above).

5. Restart *inetd* by sending the current *inetd* process a hangup signal:

   ```
   host# kill -HUP  process_id_number
   ```

6. The *sample_server* is now ready to take *sample_client* requests.

5.2 Testing the Sample Server

Assume that you have installed *sample_server* on *ariadne*.

Login to your workstation and use the *kinit* command to obtain a Kerberos ticket-granting ticket:

```
host% [K_USER]/kinit
MIT Project Athena, (your_workstation)
Kerberos Initialization
Kerberos name: yourusername        <--  Enter your Kerberos username.
Password:                          <--  Enter your Kerberos password.
```

Now use the *sample_client* program as follows:

```
host% [PATH]/sample_client  ariadne
```

The command should display something like the following:

```
The server says:
You are yourusername.@REALMNAME (local name yourusername),
 at address yournetaddress, version VERSION9, cksum 997
```

6. Service names and other services

6.1 rlogin, rsh, rcp, tftp, and others

Many services use a common principal name for authentication purposes. *rlogin*, *rsh*, *rcp*, *tftp* and others use the principal name ``rcmd''. For example, to set up the machine *ariadne* to support Kerberos rlogin, it needs to have a service key for principal ``rcmd'', instance ``ariadne''. You create this key in the same way as shown above for the sample service.

After creating this key, you need to run the *ext_srvtab* program again to generate a new srvtab file for ariadne.

6.2 NFS modifications

The NFS modifications distributed separately use the service name ``rvdsrv'' with the instance set to the machine name (as for the sample server and the rlogin, rsh, rcp and tftp services).

6.3 inetd.conf entries

The following are the */etc/inetd.conf* entries necessary to support rlogin, encrypted rlogin, rsh, and rcp services on a server machine. As above, your *inetd.conf* may not support all the fields shown here.

```
eklogin  stream   tcp   nowait   unswitched   root
    [PATH]/klogind   eklogind
kshell   stream   tcp   nowait   unswitched   root
    [PATH]/kshd   kshd
klogin   stream   tcp   nowait   unswitched   root
    [PATH]/klogind   klogind
```

Table of Contents

1. How Kerberos Works: A Schematic Description **1**
1.1 Network Services and Their Client Programs 1
1.2 Kerberos Tickets 1
1.3 The Kerberos Master Database 1
1.4 The Ticket-Granting Ticket 1
1.5 Network Services and the Master Database 1
1.6 The User-Kerberos Interaction 2
2. Setting Up and Testing the Kerberos Server **2**
2.1 Creating and Initializing the Master Database 3
2.2 Storing the Master Password 3
2.3 Using ***kdb_edit*** to Add Users to the Master Database 4
2.4 Starting the Kerberos Server 4
2.5 Testing the Kerberos Server 5
3. Setting up and testing the Administration server **5**
3.1 Adding an administration instance for the administrator 6
3.2 The Access Control Lists 6
3.3 Starting the administration server 7
3.4 Testing ***kpasswd*** 7
3.5 Testing ***kadmin*** 7
3.6 Verifying with ***kinit*** 8
4. Setting up and testing slave server(s) **8**
5. A Sample Application **8**
5.1 The Installation Process 8
5.2 Testing the Sample Server 10
6. Service names and other services **11**
6.1 rlogin, rsh, rcp, tftp, and others 11
6.2 NFS modifications 11
6.3 inetd.conf entries 11