Kerberos Operation Notes

DRAFT

1. Preparing the Master Database and Server

Assuming you have installed the Kerberos adminstrative programs in an appropriate place, use the *kdb_init* program to create and initialize the principal database. This program asks you to supply the system's realm name and the database's master password. Do not forget this password. If you do, the database becomes useless.

Use *kdb_init* as follows:

```
host# kdb_init
Realm name: 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 Password:                <-- Enter the master password
Verifying, please re-enter Enter Password:
```

1.1 Storing the Master Password

The *kstash* command "stashes" the master password in a secret place so that the *kerberos* and *admin_server* servers can be started automatically during an automatic 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. The command prompts you twice for the master password:

```
host# kstash

Current Kerberos master key version is 1
Enter current Kerberos master key:
Verifying, please re-enter
Enter current Kerberos master key: host#
```

2. 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 either a username or a service program's name. An instance further qualifies the principal. If the principal is a service, the instance specifies 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. Eventually you will be able to create users with **admin** instances. These users will then be able to use the *kadmin* program to modify the Kerberos master database over the network.

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

```
host# kdb_edit

Opening database...
Current Kerberos master key version is 1
Enter current Kerberos master key:
Verifying, please re-enter
Enter current Kerberos master key:
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 ] ?
Max ticket lifetime (*5 minutes) [ 255 ] ?
Attributes [ 0 ] ?
Edit O.K.

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

3. Starting the Kerberos Server

Change directories to the directory in which you have installed the server program *kerberos*, and start the program as a background process:

```
host# kerberos &
```

4. Installing the Sample Application Server and Clinet

This release of Kerberos comes with a sample application server and an corresponding client program. You will find this software in the **$OBJDIR/***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.

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

  `host%` **sample_client** ***servername***

- *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, the user's workstation address.
- *Sample_client* displays the server's message on the user's terminal screen.

Suppose you install *sample_server* on the machine *ARIADNE.OCEAN.EDU*. Suppose the local Kerberos realm is PACIFIC.OCEAN.EDU. Your username is *francis*, and you are logged in to a workstation with the network address 18.72.1.3. Here's how you enter the *sample_client* command, and here's what the program displays:

```
host% sample_client ariadne
The server says:
You are francis.@PACIFIC.OCEAN.EDU (local name francis),
 at address 18.72.1.3, version VERSION9, cksum 997
```

Of course none of this works until you install the sample programs and add the appropriate entries to the Kerberos database.

4.1 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.
2. Create a */etc/srvtab* file for the server machine.
3. Install the service program and the */etc/srvtab* file on the service machine.
4. Install the client progam on the system's workstations and update each workstation's */etc/services* file appropriately.

The procedure used to install *sample_server* differs slightly from that listed above because *sample_server* does not start on its own and run continuously. Instead, *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, halts, then starts again when *inetd* receives another *sample_client* request. When you install the on the server, you must add a *sample_server* entry to the server machine's */etc/inetd.conf* file.

The following description assumes that you are installing *sample_server* on the machine *ARIADNE.OCEAN.EDU*. Here's the process, step by step:

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

```
host# kdb_edit

Opening database...
Current Kerberos master key version is 1
Enter current Kerberos master key: Enter Master Password.
Verifying, please re-enter
Enter current Kerberos master key:
master key entered.  BEWARE!
Previous or default values are in [brackets] ,
enter return to leave the same, or new value.

Principal name: sample_server
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.than there were space for

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# ext_srvtab ariadne

Enter current Kerberos master key:
Verifying, please re-enter
Enter current Kerberos master key:
Master key verified.

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

Transfer this file to *ariadne* and install it as */etc/srvtab*. 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 the following line to the */etc/inetd.conf* file on *sample_server*'s machine:

```
sample  stream  tcp    nowait  switched  root
/site/kerberos/appl/sample/sample_server   sample_server
```

(This information should be placed on one line.)

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.

Table of Contents