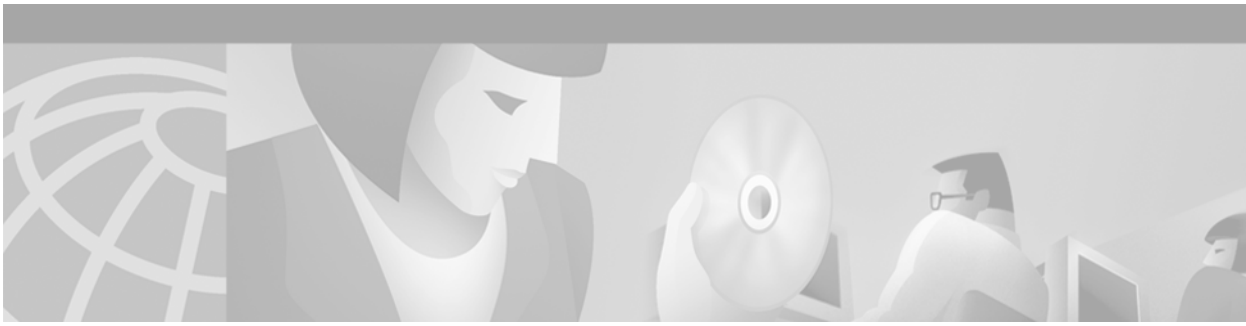

Configuring Kerberos

This chapter describes the Kerberos security system. For a complete description of the Kerberos commands used in this chapter, refer to the "Kerberos Commands" chapter in the *Cisco IOS Security Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature, or refer to the software release notes for a specific release. For more information, see the section "Identifying Supported Platforms" in the chapter "Using Cisco IOS Software."

In This Chapter

This chapter includes the following topics and tasks:

About Kerberos

Kerberos is a secret-key network authentication protocol, developed at the Massachusetts Institute of Technology (MIT), that uses the Data Encryption Standard (DES) cryptographic algorithm for encryption and authentication. Kerberos was designed to authenticate requests for network resources. Kerberos, like other secret-key systems, is based on the concept of a trusted third party that performs secure verification of users and services. In the Kerberos protocol, this trusted third party is called the key distribution center (KDC).

The primary use of Kerberos is to verify that users and the network services they use are really who and what they claim to be. To accomplish this, a trusted Kerberos server issues tickets to users. These tickets, which have a limited lifespan, are stored in a user's credential cache and can be used in place of the standard username-and-password authentication mechanism.

The Kerberos credential scheme embodies a concept called "single logon." This process requires authenticating a user once, and then allows secure authentication (without encrypting another password) wherever that user's credential is accepted.

Starting with Cisco IOS Release 11.2, Cisco IOS software includes Kerberos 5 support, which allows organizations already deploying Kerberos 5 to use the same Kerberos authentication database on their routers that they are already using on their other network hosts (such as UNIX servers and PCs).

The following network services are supported by the Kerberos authentication capabilities in Cisco IOS software:

- Telnet
- rlogin
- rsh
- rcp

Note Cisco Systems' implementation of Kerberos client support is based on code developed by CyberSafe, which was derived from the MIT code. As a result, the Cisco Kerberos implementation has successfully undergone full compatibility testing with the CyberSafe Challenger commercial Kerberos server and MIT's server code, which is freely distributed.

Table 14 lists common Kerberos-related terms and their definitions.

Table 14 Kerberos Terminology

Term	Definition
authentication	A process by which a user or service identifies itself to another service. For example, a client can authenticate to a router or a router can authenticate to another router.
authorization	A means by which the router determines what privileges you have in a network or on the router and what actions you can perform.
credential	A general term that refers to authentication tickets, such as ticket granting tickets (TGTs) and service credentials. Kerberos credentials verify the identity of a user or service. If a network service decides to trust the Kerberos server that issued a ticket, it can be used in place of retyping in a username and password. Credentials have a default lifespan of eight hours.
instance	An authorization level label for Kerberos principals. Most Kerberos principals are of the form user@REALM (for example, smith@EXAMPLE.COM). A Kerberos principal with a Kerberos instance has the form user/instance@REALM (for example, smith/admin@EXAMPLE.COM). The Kerberos instance can be used to specify the authorization level for the user if authentication is successful. It is up to the server of each network service to implement and enforce the authorization mappings of Kerberos instances. Note that the Kerberos realm name must be in uppercase characters.
Kerberized	Applications and services that have been modified to support the Kerberos credential infrastructure.
Kerberos realm	A domain consisting of users, hosts, and network services that are registered to a Kerberos server. The Kerberos server is trusted to verify the identity of a user or network service to another user or network service. Kerberos realms must always be in uppercase characters.
Kerberos server	A daemon running on a network host. Users and network services register their identity with the Kerberos server. Network services query the Kerberos server to authenticate to other network services.

Table 14 Kerberos Terminology (continued)

Term	Definition
key distribution center (KDC)	A Kerberos server and database program running on a network host.
principal	Also known as a Kerberos identity, this is who you are or what a service is according to the Kerberos server.
service credential	A credential for a network service. When issued from the KDC, this credential is encrypted with the password shared by the network service and the KDC, and with the user's TGT.
SRVTAB	A password that a network service shares with the KDC. The network service authenticates an encrypted service credential by using the SRVTAB (also known as a KEYTAB) to decrypt it.
ticket granting ticket (TGT)	A credential that the key distribution center (KDC) issues to authenticated users. When users receive a TGT, they can authenticate to network services within the Kerberos realm represented by the KDC.

Kerberos Client Support Operation

This section describes how the Kerberos security system works with a Cisco router functioning as the security server. Although (for convenience or technical reasons) you can customize Kerberos in a number of ways, remote users attempting to access network services must pass through three layers of security before they can access network services.

This section includes the following sections:

- Authenticating to the Boundary Router
- Obtaining a TGT from a KDC
- Authenticating to Network Services

Authenticating to the Boundary Router

This section describes the first layer of security that remote users must pass through when they attempt to access a network. The first step in the Kerberos authentication process is for users to authenticate themselves to the boundary router. The following process describes how users authenticate to a boundary router:

1. The remote user opens a PPP connection to the corporate site router.
2. The router prompts the user for a username and password.
3. The router requests a TGT from the KDC for this particular user.
4. The KDC sends an encrypted TGT to the router that includes (among other things) the user's identity.
5. The router attempts to decrypt the TGT using the password the user entered. If the decryption is successful, the remote user is authenticated to the router.

A remote user who successfully initiates a PPP session and authenticates to the boundary router is inside the firewall but still must authenticate to the KDC directly before being allowed to access network services. This is because the TGT issued by the KDC is stored on the router and is not useful for additional authentication unless the user physically logs on to the router.

Obtaining a TGT from a KDC

This section describes how remote users who are authenticated to the boundary router authenticate themselves to a KDC.

When a remote user authenticates to a boundary router, that user technically becomes part of the network; that is, the network is extended to include the remote user and the user's machine or network. To gain access to network services, however, the remote user must obtain a TGT from the KDC. The following process describes how remote users authenticate to the KDC:

1. The remote user, at a workstation on a remote site, launches the KINIT program (part of the client software provided with the Kerberos protocol).
2. The KINIT program finds the user's identity and requests a TGT from the KDC.
3. The KDC creates a TGT, which contains the identity of the user, the identity of the KDC, and the expiration time of the TGT.
4. Using the user's password as a key, the KDC encrypts the TGT and sends the TGT to the workstation.
5. When the KINIT program receives the encrypted TGT, it prompts the user for a password (this is the password that is defined for the user in the KDC).
6. If the KINIT program can decrypt the TGT with the password the user enters, the user is authenticated to the KDC, and the KINIT program stores the TGT in the user's credential cache.

At this point, the user has a TGT and can communicate securely with the KDC. In turn, the TGT allows the user to authenticate to other network services.

Authenticating to Network Services

The following process describes how a remote user with a TGT authenticates to network services within a given Kerberos realm. Assume the user is on a remote workstation (Host A) and wants to log in to Host B.

1. The user on Host A initiates a Kerberized application (such as Telnet) to Host B.
2. The Kerberized application builds a service credential request and sends it to the KDC. The service credential request includes (among other things) the user's identity and the identity of the desired network service. The TGT is used to encrypt the service credential request.
3. The KDC tries to decrypt the service credential request with the TGT it issued to the user on Host A. If the KDC can decrypt the packet, it is assured that the authenticated user on Host A sent the request.
4. The KDC notes the network service identity in the service credential request.
5. The KDC builds a service credential for the appropriate network service on Host B on behalf of the user on Host A. The service credential contains the client's identity and the desired network service's identity.

6. The KDC then encrypts the service credential twice. It first encrypts the credential with the SRVTAB that it shares with the network service identified in the credential. It then encrypts the resulting packet with the TGT of the user (who, in this case, is on Host A).
7. The KDC sends the twice-encrypted credential to Host A.
8. Host A attempts to decrypt the service credential with the user's TGT. If Host A can decrypt the service credential, it is assured the credential came from the real KDC.
9. Host A sends the service credential to the desired network service. Note that the credential is still encrypted with the SRVTAB shared by the KDC and the network service.
10. The network service attempts to decrypt the service credential using its SRVTAB.
11. If the network service can decrypt the credential, it is assured the credential was in fact issued from the KDC. Note that the network service trusts anything it can decrypt from the KDC, even if it receives it indirectly from a user. This is because the user first authenticated with the KDC.

At this point, the user is authenticated to the network service on Host B. This process is repeated each time a user wants to access a network service in the Kerberos realm.

Kerberos Configuration Task List

For hosts and the KDC in your Kerberos realm to communicate and mutually authenticate, you must identify them to each other. To do this, you add entries for the hosts to the Kerberos database on the KDC and add SRVTAB files generated by the KDC to all hosts in the Kerberos realm. You also make entries for users in the KDC database.

This section describes how to set up a Kerberos-authenticated server-client system and contains the following topics:

- Configuring the KDC Using Kerberos Commands
- Configuring the Router to Use the Kerberos Protocol

This section assumes that you have installed the Kerberos administrative programs on a UNIX host, known as the KDC, initialized the database, and selected a Kerberos realm name and password. For instructions about completing these tasks, refer to documentation that came with your Kerberos software.

Note Write down the host name or IP address of the KDC, the port number you want the KDC to monitor for queries, and the name of the Kerberos realm it will serve. You need this information to configure the router.

Configuring the KDC Using Kerberos Commands

After you set up a host to function as the KDC in your Kerberos realm, you must make entries to the KDC database for all principals in the realm. Principals can be network services on Cisco routers and hosts or they can be users.

To use Kerberos commands to add services to the KDC database (and to modify existing database information), complete the tasks in the following sections:

- Adding Users to the KDC Database
- Creating SRVTABs on the KDC
- Extracting SRVTABs

Note All Kerberos command examples are based on Kerberos 5 Beta 5 of the original MIT implementation. Later versions use a slightly different interface.

Adding Users to the KDC Database

To add users to the KDC and create privileged instances of those users, use the **su** command to become root on the host running the KDC and use the kdb5_edit program to use the following commands in privileged EXEC mode:

	Command	Purpose
Step 1	`Router# ank username@REALM`	Use the **ank** (add new key) command to add a user to the KDC. This command prompts for a password, which the user must enter to authenticate to the router.
Step 2	`Router# ank username/instance@REALM`	Use the **ank** command to add a privileged instance of a user.

For example, to add user *loki* of Kerberos realm CISCO.COM, enter the following Kerberos command:

```
ank loki@CISCO.COM
```

Note The Kerberos realm name must be in uppercase characters.

You might want to create privileged instances to allow network administrators to connect to the router at the enable level, for example, so that they need not enter a clear text password (and compromise security) to enter enable mode.

To add an instance of *loki* with additional privileges (in this case, *enable*, although it could be anything) enter the following Kerberos command:

```
ank loki/enable@CISCO.COM
```

In each of these examples, you are prompted to enter a password, which you must give to user *loki* to use at login.

The "Enabling Kerberos Instance Mapping" section describes how to map Kerberos instances to various Cisco IOS privilege levels.

Creating SRVTABs on the KDC

All routers that you want to authenticate to use the Kerberos protocol must have an SRVTAB. This section and the "Extracting SRVTABs" section describe how to create and extract SRVTABs for a router called *router1*. The section "Copying SRVTAB Files" describes how to copy SRVTAB files to the router.

To make SRVTAB entries on the KDC, use the following command in privileged EXEC mode:

Command	Purpose
`Router# ark SERVICE/HOSTNAME@REALM`	Use the **ark** (add random key) command to add a network service supported by a host or router to the KDC.

For example, to add a Kerberized authentication service for a Cisco router called *router1* to the Kerberos realm CISCO.COM, enter the following Kerberos command:

```
ark host/router1.cisco.com@CISCO.COM
```

Make entries for all network services on all Kerberized hosts that use this KDC for authentication.

Extracting SRVTABs

SRVTABs contain (among other things) the passwords or randomly generated keys for the service principals you entered into the KDC database. Service principal keys must be shared with the host running that service. To do this, you must save the SRVTAB entries to a file, then copy the file to the router and all hosts in the Kerberos realm. Saving SRVTAB entries to a file is called *extracting* SRVTABs. To extract SRVTABs, use the following command in privileged EXEC mode:

Command	Purpose
`Router# xst router-name host`	Use the kdb5_edit command **xst** to write an SRVTAB entry to a file.

For example, to write the host/router1.cisco.com@CISCO.COM SRVTAB to a file, enter the following Kerberos command:

```
xst router1.cisco.com@CISCO.COM host
```

Use the **quit** command to exit the kdb5_edit program.

Configuring the Router to Use the Kerberos Protocol

To configure a Cisco router to function as a network security server and authenticate users using the Kerberos protocol, complete the tasks in the following sections:

- Defining a Kerberos Realm
- Copying SRVTAB Files
- Specifying Kerberos Authentication
- Enabling Credentials Forwarding
- Opening a Telnet Session to the Router
- Establishing an Encrypted Kerberized Telnet Session
- Enabling Mandatory Kerberos Authentication
- Enabling Kerberos Instance Mapping
- Monitoring and Maintaining Kerberos

Defining a Kerberos Realm

For a router to authenticate a user defined in the Kerberos database, it must know the host name or IP address of the host running the KDC, the name of the Kerberos realm and, optionally, be able to map the host name or Domain Name System (DNS) domain to the Kerberos realm.

To configure the router to authenticate to a specified KDC in a specified Kerberos realm, use the following commands in global configuration mode. Note that DNS domain names must begin with a leading dot (.):

	Command	Purpose
Step 1	`Router(config)# kerberos local-realm kerberos-realm`	Defines the default realm for the router.
Step 2	`Router(config)# kerberos server kerberos-realm {hostname | ip-address} [port-number]`	Specifies to the router which KDC to use in a given Kerberos realm and, optionally, the port number that the KDC is monitoring. (The default is 88.)
Step 3	`Router(config)# kerberos realm {dns-domain | host} kerberos-realm`	(Optional) Maps a host name or DNS domain to a Kerberos realm.

Note Because the machine running the KDC and all Kerberized hosts must interact within a 5-minute window or authentication fails, all Kerberized machines, and especially the KDC, should be running the Network Time Protocol (NTP).

The **kerberos local-realm**, **kerberos realm**, and **kerberos server** commands are equivalent to the UNIX *krb.conf* file. Table 15 identifies mappings from the Cisco IOS configuration commands to a Kerberos 5 configuration file (krb5.conf).

Table 15 Kerberos 5 Configuration File and Commands

krb5.conf File	Cisco IOS Configuration Command
[libdefaults] `default_realm = DOMAIN.COM`	(in configuration mode) **kerberos local-realm** *DOMAIN.COM*

[domain_realm]
`.domain.com = DOMAIN.COM`
`domain.com = DOMAIN.COM`
	(in configuration mode)
	kerberos realm.*domain.com DOMAIN.COM*
	kerberos realm *domain.com DOMAIN.COM*

[realms]
`kdc = DOMAIN.PIL.COM:750`
`admin_server = DOMAIN.PIL.COM`
`default_domain = DOMAIN.COM`
	(in configuration mode)
	kerberos server *DOMAIN.COM 172.65.44.2*
	(*172.65.44.2* is the example IP address for *DOMAIN.PIL.COM*)

For an example of defining a Kerberos realm, see the section "Defining a Kerberos Realm" later in this chapter.

Copying SRVTAB Files

To make it possible for remote users to authenticate to the router using Kerberos credentials, the router must share a secret key with the KDC. To do this, you must give the router a copy of the SRVTAB you extracted on the KDC.

The most secure method to copy SRVTAB files to the hosts in your Kerberos realm is to copy them onto physical media and go to each host in turn and manually copy the files onto the system. To copy SRVTAB files to the router, which does not have a physical media drive, you must transfer them via the network using TFTP.

To remotely copy SRVTAB files to the router from the KDC, use the following command in global configuration mode:

Command	Purpose
`Router(config)# kerberos srvtab remote {hostname | ip-address} {filename}`	Retrieves an SRVTAB file from the KDC.

When you copy the SRVTAB file from the router to the KDC, the **kerberos srvtab remote** command parses the information in this file and stores it in the router's running configuration in the **kerberos srvtab entry** format. To ensure that the SRVTAB is available (does not need to be acquired from the KDC) when you reboot the router, use the **write memory** configuration command to write your running configuration (which contains the parsed SRVTAB file) to NVRAM.

For an example of copying SRVTAB files, see the section "SRVTAB File Copying Example" later in this chapter.

Specifying Kerberos Authentication

You have now configured Kerberos on your router. This makes it possible for the router to authenticate using Kerberos. The next step is to tell it to do so. Because Kerberos authentication is facilitated through AAA, you need to enter the **aaa authentication** command, specifying Kerberos as the authentication method. For more information, refer to the chapter "Configuring Authentication".

Enabling Credentials Forwarding

With Kerberos configured thus far, a user authenticated to a Kerberized router has a TGT and can use it to authenticate to a host on the network. However, if the user tries to list credentials after authenticating to a host, the output will show no Kerberos credentials present.

You can optionally configure the router to forward users' TGTs with them as they authenticate from the router to Kerberized remote hosts on the network when using Kerberized Telnet, rcp, rsh, and rlogin (with the appropriate flags).

To force all clients to forward users' credentials as they connect to other hosts in the Kerberos realm, use the following command in global configuration mode:

Command	Purpose
`Router(config)# kerberos credentials forward`	Forces all clients to forward user credentials upon successful Kerberos authentication.

With credentials forwarding enabled, users' TGTs are automatically forwarded to the next host they authenticate to. In this way, users can connect to multiple hosts in the Kerberos realm without running the KINIT program each time to get a new TGT.

Opening a Telnet Session to the Router

To use Kerberos to authenticate users opening a Telnet session to the router from within the network, use the following command in global configuration mode:

Command	Purpose
`Router(config)# aaa authentication login {default | list-name} krb5_telnet`	Sets login authentication to use the Kerberos 5 Telnet authentication protocol when using Telnet to connect to the router.

Although Telnet sessions to the router are authenticated, users must still enter a clear text password if they want to enter enable mode. The **kerberos instance map** command, discussed in a later section, allows them to authenticate to the router at a predefined privilege level.

Establishing an Encrypted Kerberized Telnet Session

Another way for users to open a secure Telnet session is to use Encrypted Kerberized Telnet. With Encrypted Kerberized Telnet, users are authenticated by their Kerberos credentials before a Telnet session is established. The Telnet session is encrypted using 56-bit Data Encryption Standard (DES) encryption with 64-bit Cipher Feedback (CFB). Because data sent or received is encrypted, not clear text, the integrity of the dialed router or access server can be more easily controlled.

Note This feature is available only if you have the 56-bit encryption image. 56-bit DES encryption is subject to U.S. Government export control regulations.

To establish an encrypted Kerberized Telnet session from a router to a remote host, use either of the following commands in EXEC command mode:

Command	Purpose

`Router(config)# connect host [port] /encrypt kerberos`
or
`Router(config)# telnet host [port] /encrypt kerberos`
	Establishes an encrypted Telnet session.

When a user opens a Telnet session from a Cisco router to a remote host, the router and remote host negotiate to authenticate the user using Kerberos credentials. If this authentication is successful, the router and remote host then negotiate whether or not to use encryption. If this negotiation is successful, both inbound and outbound traffic is encrypted using 56-bit DES encryption with 64-bit CFB.

When a user dials in from a remote host to a Cisco router configured for Kerberos authentication, the host and router will attempt to negotiate whether or not to use encryption for the Telnet session. If this negotiation is successful, the router will encrypt all outbound data during the Telnet session.

If encryption is not successfully negotiated, the session will be terminated and the user will receive a message stating that the encrypted Telnet session was not successfully established.

For information about enabling bidirectional encryption from a remote host, refer to the documentation specific to the remote host device.

For an example of using encrypted Kerberized Telnet to open a secure Telnet session, see the section "Encrypted Telnet Session Example" later in this chapter.

Enabling Mandatory Kerberos Authentication

As an added layer of security, you can optionally configure the router so that, after remote users authenticate to it, these users can authenticate to other services on the network only with Kerberized Telnet, rlogin, rsh, and rcp. If you do not make Kerberos authentication mandatory and Kerberos authentication fails, the application attempts to authenticate users using the default method of authentication for that network service; for example, Telnet and rlogin prompt for a password, and rsh attempts to authenticate using the local rhost file.

To make Kerberos authentication mandatory, use the following command in global configuration mode:

Command	Purpose
`Router(config)# kerberos clients mandatory`	Sets Telnet, rlogin, rsh, and rcp to fail if they cannot negotiate the Kerberos protocol with the remote server.

Enabling Kerberos Instance Mapping

As mentioned in the section "Creating SRVTABs on the KDC," you can create administrative instances of users in the KDC database. The **kerberos instance map** command allows you to map those instances to Cisco IOS privilege levels so that users can open secure Telnet sessions to the router at a predefined privilege level, obviating the need to enter a clear text password to enter enable mode.

To map a Kerberos instance to a Cisco IOS privilege level, use the following command in global configuration mode:

Command	Purpose
`Router(config)# kerberos instance map` *`instance privilege-level`*	Maps a Kerberos instance to a Cisco IOS privilege level.

If there is a Kerberos instance for user *loki* in the KDC database (for example, *loki/admin*), user *loki* can now open a Telnet session to the router as loki/admin and authenticate automatically at privilege level 15, assuming instance "admin" is mapped to privilege level 15. (See the section "Adding Users to the KDC Database" earlier in this chapter.)

Cisco IOS commands can be set to various privilege levels using the **privilege level** command.

After you map a Kerberos instance to a Cisco IOS privilege level, you must configure the router to check for Kerberos instances each time a user logs in. To run authorization to determine if a user is allowed to run an EXEC shell based on a mapped Kerberos instance, use the **aaa authorization** command with the **krb5-instance** keyword. For more information, refer to the chapter "Configuring Authorization."

Monitoring and Maintaining Kerberos

To display or remove a current user's credentials, use the following commands in EXEC mode:

	Command	Purpose
Step 1	`Router# show kerberos creds`	Lists the credentials in a current user's credentials cache.
Step 2	`Router# clear kerberos creds`	Destroys all credentials in a current user's credentials cache, including those forwarded.

For an example of Kerberos configuration, see the section "Kerberos Configuration Examples".

Kerberos Configuration Examples

The following sections provide Kerberos configuration examples:

- Kerberos Realm Definition Examples
- SRVTAB File Copying Example
- Kerberos Configuration Examples
- Encrypted Telnet Session Example

Kerberos Realm Definition Examples

To define CISCO.COM as the default Kerberos realm, use the following command:

```
kerberos local-realm CISCO.COM
```

To tell the router that the CISCO.COM KDC is running on host 10.2.3.4 at port number 170, use the following Kerberos command:

```
kerberos server CISCO.COM 10.2.3.4 170
```

To map the DNS domain cisco.com to the Kerberos realm CISCO.COM, use the following command:

```
kerberos realm.cisco.com CISCO.COM
```

SRVTAB File Copying Example

To copy over the SRVTAB file on a host named host123.cisco.com for a router named router1.cisco.com, the command would look like this:

```
kerberos srvtab remote host123.cisco.com router1.cisco.com-new-srvtab
```

Kerberos Configuration Examples

This section provides a typical non-Kerberos router configuration and shows output for this configuration from the **write term** command, then builds on this configuration by adding optional Kerberos functionality. Output for each configuration is presented for comparison against the previous configuration.

This example shows how to use the kdb5_edit program to perform the following configuration tasks:

- Adding user chet to the Kerberos database
- Adding a privileged Kerberos instance of user chet (chet/admin) to the Kerberos database
- Adding a restricted instance of chet (chet/restricted) to the Kerberos database
- Adding workstation chet-ss20.cisco.com
- Adding router chet-2500.cisco.com to the Kerberos database
- Adding workstation chet-ss20.cisco.com to the Kerberos database
- Extracting SRVTABs for the router and workstations
- Listing the contents of the KDC database (with the **ldb** command)

Note that, in this sample configuration, host chet-ss20 is also the KDC:

```
chet-ss20# sbin/kdb5_edit
kdb5_edit:  ank chet
Enter password:
Re-enter password for verification:
kdb5_edit:  ank chet/admin
Enter password:
Re-enter password for verification:
kdb5_edit:  ank chet/restricted
Enter password:
Re-enter password for verification:
kdb5_edit:  ark host/chet-ss20.cisco.com
kdb5_edit:  ark host/chet-2500.cisco.com
kdb5_edit:  xst chet-ss20.cisco.com host
'host/chet-ss20.cisco.com@CISCO.COM' added to keytab
'WRFILE:chet-ss20.cisco.com-new-srvtab'
kdb5_edit:  xst chet-2500.cisco.com host
'host/chet-2500.cisco.com@CISCO.COM' added to keytab
'WRFILE:chet-2500.cisco.com-new-srvtab'
kdb5_edit:  ldb
entry: host/chet-2500.cisco.com@CISCO.COM
entry: chet/restricted@CISCO.COM
entry: chet@CISCO.COM
entry: K/M@CISCO.COM
entry: host/chet-ss20.cisco.com@CISCO.COM
entry: krbtgt/CISCO.COM@CISCO.COM
entry: chet/admin@CISCO.COM
kdb5_edit:  q
chet-ss20#
```

The following example shows output from a **write term** command, which displays the configuration of router chet-2500. This is a typical configuration with no Kerberos authentication.

```
chet-2500# write term
Building configuration...

Current configuration:
!
! Last configuration
change at 14:03:55 PDT Mon May 13 1996
!
version 11.2
service udp-small-servers
service tcp-small-servers
!
hostname chet-2500
!
clock timezone PST -8
```

```
clock summer-time PDT recurring
aaa new-model
aaa authentication login console none
aaa authentication ppp local local
enable password sMudgKin
!
username chet-2500 password 7 sMudgkin
username chet-3000 password 7 sMudgkin
username chetin password 7 sMudgkin
!
interface Ethernet0
 ip address 172.16.0.0 255.255.255.0
!
interface Serial0
 no ip address
 shutdown
 no fair-queue
!
interface Serial1
 no ip address
 shutdown
 no fair-queue
!
interface Async2
 ip unnumbered Ethernet0
 encapsulation ppp
 shutdown
 async dynamic routing
 async mode dedicated
 no cdp enable
 ppp authentication pap local
 no tarp propagate
!
interface Async3
 ip unnumbered Ethernet0
 encapsulation ppp
 shutdown
 async dynamic address
 async dynamic routing
 async mode dedicated
 no cdp enable
 ppp authentication pap local
 no tarp propagate
!
router eigrp 109
 network 172.17.0.0
 no auto-summary
!
ip default-gateway 172.30.55.64
ip domain-name cisco.com
ip name-server 192.168.0.0
ip classless
!
!

line con 0
 exec-timeout 0 0
 login authentication console
line 1 16
 transport input all
line aux 0
 transport input all
line vty 0 4
 password sMudgKin
```

```
!
ntp clock-period 17179703
ntp peer 172.19.10.0
ntp peer 172.19.0.0
end
```

The following example shows how to enable user authentication on the router via the Kerberos database. To enable user authentication via the Kerberos database, you would perform the following tasks:

- Entering configuration mode
- Defining the Kerberos local realm
- Identifying the machine hosting the KDC
- Enabling credentials forwarding
- Specifying Kerberos as the method of authentication for login
- Exiting configuration mode (CTL-Z)
- Writing the new configuration to the terminal

```
chet-2500# configure term
Enter configuration commands, one per line. End with CNTL/Z.
chet-2500(config)# kerberos local-realm CISCO.COM
chet-2500(config)# kerberos server CISCO.COM chet-ss20
Translating "chet-ss20"...domain server (192.168.0.0) [OK]

chet-2500(config)# kerberos credentials forward
chet-2500(config)# aaa authentication login default krb5
chet-2500(config)#
chet-2500#
%SYS-5-CONFIG_I: Configured from console by console
chet-2500# write term
```

Compare the following configuration with the previous one. In particular, look at the lines beginning with the words “aaa,” “username,” and “kerberos” (lines 10 through 20) in this new configuration.

```
Building configuration...

Current configuration:
!
! Last configuration change at 14:05:54 PDT Mon May 13 1996
!
version 11.2
service udp-small-servers
service tcp-small-servers
!
hostname chet-2500
!
clock timezone PST -8
clock summer-time PDT recurring
aaa new-model
aaa authentication login default krb5
aaa authentication login console none
aaa authentication ppp local local
enable password sMudgKin
!
username chet-2500 password 7 sMudgkin
username chet-3000 password 7 sMudgkin
username chetin password 7 sMudgkin
kerberos local-realm CISCO.COM
kerberos server CISCO.COM 172.71.54.14
kerberos credentials forward
!
```

```
interface Ethernet0
 ip address 172.16.0.0 255.255.255.0
!
interface Serial0
 no ip address
 shutdown
 no fair-queue
!
interface Serial1
 no ip address
 shutdown
 no fair-queue
!
interface Async2
 ip unnumbered Ethernet0
 encapsulation ppp
 shutdown
 async dynamic routing
 async mode dedicated
 no cdp enable
 ppp authentication pap local
 no tarp propagate
!
interface Async3
 ip unnumbered Ethernet0
 encapsulation ppp
 shutdown
 async dynamic address
 async dynamic routing
 async mode dedicated
 no cdp enable
 ppp authentication pap local
 no tarp propagate
!
router eigrp 109
 network 172.17.0.0
 no auto-summary
!
ip default-gateway 172.30.55.64
ip domain-name cisco.com
ip name-server 192.168.0.0
ip classless
!
!
line con 0
 exec-timeout 0 0
 login authentication console
line 1 16
 transport input all
line aux 0
 transport input all
line vty 0 4
 password sMudgKin
!
ntp clock-period 17179703
ntp peer 172.19.10.0
ntp peer 172.19.0.0
end
```

With the router configured thus far, user chet can log in to the router with a username and password and automatically obtain a TGT, as illustrated in the next example. With possession of a credential, user chet successfully authenticates to host chet-ss20 without entering a username/password.

```
chet-ss20% telnet chet-2500
Trying 172.16.0.0 ...
Connected to chet-2500.cisco.com.
Escape character is '^]'.


User Access Verification

Username: chet
Password:

chet-2500> show kerberos creds
Default Principal:  chet@CISCO.COM
Valid Starting          Expires                 Service Principal
13-May-1996 14:05:39    13-May-1996 22:06:40    krbtgt/CISCO.COM@CISCO.COM

chet-2500> telnet chet-ss20
Trying chet-ss20.cisco.com (172.71.54.14)... Open
Kerberos:       Successfully forwarded credentials


SunOS UNIX (chet-ss20) (pts/7)

Last login: Mon May 13 13:47:35 from chet-ss20.cisco.c
Sun Microsystems Inc.   SunOS 5.4       Generic July 1994
unknown mode: new
chet-ss20%
```

The following example shows how to authenticate to the router using Kerberos credentials. To authenticate using Kerberos credentials, you would perform the following tasks:

- Entering configuration mode
- Remotely copying over the SRVTAB file from the KDC
- Setting authentication at login to use the Kerberos 5 Telnet authentication protocol when using Telnet to connect to the router
- Writing the configuration to the terminal

Note that the new configuration contains a **kerberos srvtab entry** line. This line is created by the **kerberos srvtab remote** command.

```
chet-2500# configure term
Enter configuration commands, one per line.  End with CNTL/Z.
chet-2500(config)# kerberos srvtab remote earth chet/chet-2500.cisco.com-new-srvtab
Translating "earth"...domain server (192.168.0.0) [OK]

Loading chet/chet-2500.cisco.com-new-srvtab from 172.68.1.123 (via Ethernet0): !
[OK - 66/1000 bytes]

chet-2500(config)# aaa authentication login default krb5-telnet krb5
chet-2500(config)#
chet-2500#
%SYS-5-CONFIG_I: Configured from console by console
chet-2500# write term
Building configuration...

Current configuration:
!
```

```
! Last configuration change at 14:08:32 PDT Mon May 13 1996
!
version 11.2
service udp-small-servers
service tcp-small-servers
!
hostname chet-2500
!
clock timezone PST -8
clock summer-time PDT recurring
aaa new-model
aaa authentication login default krb5-telnet krb5
aaa authentication login console none
aaa authentication ppp local local
enable password sMudgKin
!
username chet-2500 password 7 sMudgkin
username chet-3000 password 7 sMudgkin
username chetin password 7 sMudgkin
kerberos local-realm CISCO.COM
kerberos srvtab entry host/chet-2500.cisco.com@CISCO.COM 0 832015393 1 1 8 7 sMudgkin
kerberos server CISCO.COM 172.71.54.14
kerberos credentials forward
!
interface Ethernet0
 ip address 172.16.0.0 255.255.255.0
!
interface Serial0
 no ip address
 shutdown
 no fair-queue
!

interface Serial1
 no ip address
 shutdown
 no fair-queue
!
interface Async2
 ip unnumbered Ethernet0
 encapsulation ppp
 shutdown
 async dynamic routing
 async mode dedicated
 no cdp enable
 ppp authentication pap local
 no tarp propagate
!
interface Async3
 ip unnumbered Ethernet0
 encapsulation ppp
 shutdown
 async dynamic address
 async dynamic routing
 async mode dedicated
 no cdp enable
 ppp authentication pap local
 no tarp propagate
!
router eigrp 109
 network 172.17.0.0
 no auto-summary
!
ip default-gateway 172.30.55.64
```

```
ip domain-name cisco.com
ip name-server 192.168.0.0
ip classless
!
!
line con 0
 exec-timeout 0 0
 login authentication console
line 1 16
 transport input all
line aux 0
 transport input all
line vty 0 4
 password sMudgKin
!
ntp clock-period 17179703
ntp peer 172.19.10.0
ntp peer 172.19.0.0
end

chet-2500#
```

With this configuration, the user can Telnet in to the router using Kerberos credentials, as illustrated in the next example:

```
chet-ss20% bin/telnet -a -F chet-2500
Trying 172.16.0.0...
Connected to chet-2500.cisco.com.
Escape character is '^]'.
[ Kerberos V5 accepts you as "chet@CISCO.COM" ]

User Access Verification

chet-2500>[ Kerberos V5 accepted forwarded credentials ]

chet-2500> show kerberos creds
Default Principal:  chet@CISCO.COM
Valid Starting          Expires                 Service Principal
13-May-1996 15:06:25    14-May-1996 00:08:29    krbtgt/CISCO.COM@CISCO.COM

chet-2500>q
Connection closed by foreign host.
chet-ss20%
```

The following example shows how to map Kerberos instances to Cisco's privilege levels. To map Kerberos instances to privilege levels, you would perform the following tasks:

- Entering configuration mode
- Mapping the Kerberos instance admin to privilege level 15
- Mapping the Kerberos instance restricted to privilege level 3
- Specifying that the instance defined by the **kerberos instance map** command be used for AAA Authorization
- Writing the configuration to the terminal

```
chet-2500# configure term
Enter configuration commands, one per line.  End with CNTL/Z.
chet-2500(config)# kerberos instance map admin 15
chet-2500(config)# kerberos instance map restricted 3
chet-2500(config)# aaa authorization exec default krb5-instance
chet-2500(config)#
chet-2500#
```

```
%SYS-5-CONFIG_I: Configured from console by console
chet-2500# write term
Building configuration...

Current configuration:
!
! Last configuration change at 14:59:05 PDT Mon May 13 1996
!
version 11.2
service udp-small-servers
service tcp-small-servers
!
hostname chet-2500
!
aaa new-model
aaa authentication login default krb5-telnet krb5
aaa authentication login console none
aaa authentication ppp default krb5 local
aaa authorization exec default krb5-instance
enable password sMudgKin
!
username chet-2500 password 7 sMudgkin
username chet-3000 password 7 sMudgkin
username chetin password 7 sMudgkin
ip domain-name cisco.com
ip name-server 192.168.0.0
kerberos local-realm CISCO.COM
kerberos srvtab entry host/chet-2500.cisco.com@CISCO.COM 0 832015393 1 1 8 7 sMudgkin
kerberos server CISCO.COM 172.71.54.14
kerberos instance map admin 15
kerberos instance map restricted 3
kerberos credentials forward
clock timezone PST -8
clock summer-time PDT recurring
!
interface Ethernet0
 ip address 172.16.0.0 255.255.255.0
!
interface Serial0
 no ip address
 shutdown
 no fair-queue
!
interface Serial1
 no ip address
 shutdown
 no fair-queue
!
interface Async2
 ip unnumbered Ethernet0
 encapsulation ppp
 shutdown
 async dynamic routing
 async mode dedicated
 no cdp enable
 ppp authentication pap local
 no tarp propagate
!
interface Async3
 ip unnumbered Ethernet0
 encapsulation ppp
 shutdown
 async dynamic address
 async dynamic routing
```

```
 async mode dedicated
 no cdp enable
 ppp authentication pap local
 no tarp propagate
!
router eigrp 109
 network 172.17.0.0
no auto-summary
!
ip default-gateway 172.30.55.64
ip classless
!
!
line con 0
 exec-timeout 0 0
 login authentication console
line 1 16
 transport input all
line aux 0
 transport input all
line vty 0 4
 password sMudgKin
!
ntp clock-period 17179703
ntp peer 172.19.10.0
ntp peer 172.19.0.0
end

chet-2500#
```

The following example shows output from the three types of sessions now possible for user chet with Kerberos instances turned on:

```
chet-ss20% telnet chet-2500
Trying 172.16.0.0 ...
Connected to chet-2500.cisco.com.
Escape character is '^]'.

User Access Verification

Username: chet
Password:

chet-2500> show kerberos creds
Default Principal:  chet@CISCO.COM
Valid Starting          Expires                 Service Principal
13-May-1996 14:58:28    13-May-1996 22:59:29    krbtgt/CISCO.COM@CISCO.COM

chet-2500> show privilege
Current privilege level is 1
chet-2500> q
Connection closed by foreign host.
chet-ss20% telnet chet-2500
Trying 172.16.0.0 ...
Connected to chet-2500.cisco.com.
Escape character is '^]'.


User Access Verification

Username: chet/admin
Password:

chet-2500# show kerberos creds
```

```
Default Principal:  chet/admin@CISCO.COM
Valid Starting          Expires                 Service Principal
13-May-1996 14:59:44    13-May-1996 23:00:45    krbtgt/CISCO.COM@CISCO.COM
chet-2500# show privilege
Current privilege level is 15
chet-2500# q
Connection closed by foreign host.
chet-ss20% telnet chet-2500
Trying 172.16.0.0 ...
Connected to chet-2500.cisco.com.
Escape character is '^]'.

User Access Verification

Username: chet/restricted
Password:

chet-2500# show kerberos creds
Default Principal:  chet/restricted@CISCO.COM
Valid Starting          Expires                 Service Principal
13-May-1996 15:00:32    13-May-1996 23:01:33    krbtgt/CISCO.COM@CISCO.COM

chet-2500# show privilege
Current privilege level is 3
chet-2500# q
Connection closed by foreign host.
chet-ss20%
```

Encrypted Telnet Session Example

The following example shows how to establish an encrypted Telnet session from a router to a remote host named “host1”:

```
Router> telnet host1 /encrypt kerberos
```