Linux Academy

Live! Lab

Configure a System to Authenticate Using Kerberos

Contents

Lab Connection Information

- Labs may take up to five minutes to build
- Labs may take up to five minutes to build
- The IP address of your server is located on the Live! Lab page
- Username: linuxacademy
- Password: 123456

Related Courses

Linux Academy Red Hat Certified Engineer Prep

Related Videos

Configure a System to Authenticate Using Kerberos - KDC Server Setup

Configure a System to Authenticate Using Kerberos - Client Setup

Need Help?

Linux Academy Community

... and you can always send in a support ticket on our website to talk to an instructor!

Kerberos is a network authentication protocol that uses secret-key cryptography to allow for secure authentication between clients and servers; it tends to be a significant portion of the RHCE exam.

This lab provides two servers with RHEL 7; it can also be completed with two CentOS 7 machines.

Note

Kerberos depends on the use of fully-qualified domain names:

Server 1

FQDN: kdc-server.mylabserver.com
PRIVATE IP: 10.0.0.100

Server 2

FQDN: kerb-client.mylabserver.com
PRIVATE IP: 10.0.0.101

Due to our lab environment, we want to associate our private IPs with the FQDN. Add the appropriate configuration to your server's `/etc/hosts` file:

```
127.0.0.1   localhost localhost.localdomain localhost4
localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.
localdomain6
10.0.0.100  kdc-server.mylabserver.com
```

Create the KDC Server

To authenticate using Kerberos, we first need to set up a Kerberos authentication server. Install the necessary tools:

```
[root@kdc-server ~]# yum install -y krb5-server krb5-workstation pam_
krb5
```

We install the `krb5-workstation` package to validate the connection between the server and localhost.

In this lab, we leave Kerberos 4 utilities intact to allow for backward compatibility, but these utilities can be removed if you require only Kerberos 5 or wish to enhance your security profile. Instructions for disabling backward compatibility are noted, when applicable.

Navigate to `/var/kerberos/krb5kdc`; this is our server configuration directory and includes files for the Kerberos configuration and ACLs.

Open *kdc.conf* in your chosen text editor. The realm names need to be updated to match the domain name; configuration is case sensitive, so when the example domain is in uppercase, the actual domain needs to be, as well, upon replacing it:

```
[kdcdefaults]
 kdc_ports = 88
 kdc_tcp_ports = 88

[realms]
 MYLABSERVER.COM = {
  #master_key_type = aes256-cts
  acl_file = /var/kerberos/krb5kdc/kadm5.acl
  dict_file = /usr/share/dict/words
  admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
  supported_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-
sha1:normal arcfour-hmac:normal camellia256-cts:normal camellia128-
cts:normal des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal
 }
```

Additionally, if you wish to remove backwards compatibility from your Kerberos install, uncomment the `#master_key_type = aes256-cts` line, then add `default_principal_flags = +preauth` to the line below it.

Save and exit.

We now need to edit the */etc/krb5.conf* file to replace any instances of the example FQDN; note that lines need to be uncommented:

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 dns_lookup_realm = false
 ticket_lifetime = 24h
 renew_lifetime = 7d
 forwardable = true
 rdns = false
 default_realm = MYLABSERVER.COM
 default_ccache_name = KEYRING:persistent:%{uid}

[realms]
 MYLABSERVER.COM = {
  kdc = kdc-server.mylabserver.com
  admin_server = kdc-server.mylabserver.com
 }
```

```
[domain_realm]
 .mylabserver.com = MYLABSERVER.COM
 mylabserver.com = MYLABSERVER.COM
```

The last code block denotes that anything on `.mylabserver.com` or the associated top-level domain is served as the primary realm.

Save and exit.

We now need to make changes to the ACL file, *kadm4.acl*, located in `/var/kerberos/kdb5kdc/` directory. As before, we are updating the domain information:

```
*/admin@MYLABSERVER.COM *
```

Finally, we can create the actual Kerberos data using the installed Kerberos utility:

```
[root@kdc-server]# kdb5_util create -s -r MYLABSERVER.COM
```

This takes a few minutes, as it creates data and uses `/dev/random` to generate entropy and create secure keys.

Input the database master password when prompted. On production servers, you want to select a secure password.

Enable and start the service:

```
[root@kdc-server etc]# systemctl enable krb5kdc kadmin
Created symlink from /etc/systemd/system/multi-user.target.wants/
krb5kdc.service to /usr/lib/systemd/system/krb5kdc.service.
Created symlink from /etc/systemd/system/multi-user.target.wants/kadmin.
service to /usr/lib/systemd/system/kadmin.service.
[root@kdc-server etc]# systemctl start krb5kdc kadmin
```

Add KDC Principals

Next, we use the Kerberos Administration tool to add principals to our KDC configuration. This gives us a prompt where we can create principals. We want to create a principal to for the root/admin account for the actual system (not the database, which we created earlier).

```
[root@kdc-server etc]# kadmin.local
Authenticating as principal root/admin@MYLABSERVER.COM with password.
kadmin.local:  addprinc root/admin
WARNING: no policy specified for root/admin@MYLABSERVER.COM; defaulting
to no policy
Enter password for principal “root/admin@MYLABSERVER.COM”:
Re-enter password for principal “root/admin@MYLABSERVER.COM”:
```

```
Principal "root/admin@MYLABSERVER.COM" created.
```

We now need to add a test user account we can use to confirm authentication:

```
kadmin.local:  addprinc krbtest
WARNING: no policy specified for krbtest@MYLABSERVER.COM; defaulting to
no policy
Enter password for principal "krbtest@MYLABSERVER.COM":
Re-enter password for principal "krbtest@MYLABSERVER.COM":
Principal "krbtest@MYLABSERVER.COM" created.
```

The hostname of the KDC server needs to be added, so Kerberos views it as an authenticated server during testing:

```
kadmin.local:  addprinc -randkey host/kdc-server.mylabserver.com
WARNING: no policy specified for host/kdc-server.mylabserver.com@
MYLABSERVER.COM; defaulting to no policy
Principal "host/kdc-server.mylabserver.com@MYLABSERVER.COM" created.
```

All of this now needs to be stored to a local keytab file in the */etc/* directory:

```
kadmin.local:  ktadd host/kdc-server.mylabserver.com
Entry for principal host/kdc-server.mylabserver.com with kvno 2,
encryption type aes256-cts-hmac-sha1-96 added to keytab FILE:/etc/krb5.
keytab.
Entry for principal host/kdc-server.mylabserver.com with kvno 2,
encryption type aes128-cts-hmac-sha1-96 added to keytab FILE:/etc/krb5.
keytab.
Entry for principal host/kdc-server.mylabserver.com with kvno 2,
encryption type des3-cbc-sha1 added to keytab FILE:/etc/krb5.keytab.
Entry for principal host/kdc-server.mylabserver.com with kvno 2,
encryption type arcfour-hmac added to keytab FILE:/etc/krb5.keytab.
Entry for principal host/kdc-server.mylabserver.com with kvno 2,
encryption type camellia256-cts-cmac added to keytab FILE:/etc/krb5.
keytab.
Entry for principal host/kdc-server.mylabserver.com with kvno 2,
encryption type camellia128-cts-cmac added to keytab FILE:/etc/krb5.
keytab.
Entry for principal host/kdc-server.mylabserver.com with kvno 2,
encryption type des-hmac-sha1 added to keytab FILE:/etc/krb5.keytab.
Entry for principal host/kdc-server.mylabserver.com with kvno 2,
encryption type des-cbc-md5 added to keytab FILE:/etc/krb5.keytab.
```

Quit the `kadmin` tool:

```
kadmin.local:  quit
```

Check for the keytab file:

```
[root@kdc-server ~]# cd /etc/
[root@kdc-server etc]# ls -al *keytab
-rw-------. 1 root root 714 Aug 18 13:52 krb5.keytab
```

Test Authentication

Before we test our configuration, we need to alter some of our SSH configuration, at */etc/ssh/ssh_config*. Find the lines `#   GSSAPIAuthentication no` and `#   GSSAPIDelegateCredentials no`. Uncomment the lines and change `no` to `yes` for each.

Reload SSH:

```
[root@kdc-server etc]# systemctl reload sshd
```

Update the Kerberos authentication configuration:

```
[root@kdc-server etc]# authconfig --enablekrb5 --update
```

If on an environment where a firewall is installed, open ports TCP 88 and 749 and UDP 88. Because your exam environment *will* have a firewall, it is suggested you install and enable Firewalld now, if you do not have it. We suggest adding the rules using an XML file at */etc/firewalld/services/kerberos.xml*:

```
<?xml version="1.0" encoding="utf-8"?>
<service>
<short>Kerberos</short>
<description>Kerberos network authentication protocol server</
description>
<port protocol="tcp" port="88"/>
<port protocol="udp" port="88"/>
<port protocol="tcp" port="749"/>
</service>
```

Apply the changes with:

```
[root@kdc-server etc]# firewall-cmd --permanent --add-service=kerberos
[root@kdc-server etc]# firewall-cmd --reload
```

We can now test our configuration with the *krbtest* user. Add the user and initialize Kerberos:

```
[root@kdc-server etc]# useradd krbtest
[root@kdc-server etc]# su - krbtest
[krbtest@kdc-server ~]$ kinit
Password for krbtest@MYLABSERVER.COM:
```

If we now run `klist` we can see that the test ran successfully:

```
[krbtest@kdc-server ~]$ klist
Ticket cache: KEYRING:persistent:1002:1002
Default principal: krbtest@MYLABSERVER.COM
Valid starting   Expires  Service principal
08/18/2016 14:11:43  08/19/2016 14:11:43  krbtgt/MYLABSERVER.COM@
MYLABSERVER.COM
```

Now, we should be able to SSH into `kdc-server.mylabserver.com`:

```
[krbtest@kdc-server ~]$ ssh kdc-server.mylabserver.com
```

Client Authentication

Knowing that we can authenticate with the local Kerberos server, we want to configure our setup to work with a remote client.

Before we begin, SSH into the client server, and update the */etc/hosts* file to reflect the private IP and FQDN, as we did before.

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.
localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.0101   kerb-client.mylabserver.com
```

Because this is only a client server, we only need to install the workstation and PAM authentication packages:

```
[root@kerb-client ~]# yum install -y krb5-workstation pam_krb5
```

As with the server, we need to edit the */etc/krb5.conf* file, replacing **EXAMPLE.COM** appropriately; this file should be identical to the one on the KDC server.

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 dns_lookup_realm = false
 ticket_lifetime = 24h
 renew_lifetime = 7d
 forwardable = true
 rdns = false
 default_realm = MYLABSERVER.COM
 default_ccache_name = KEYRING:persistent:%{uid}

[realms]
```

```
 MYLABSERVER.COM = {
  kdc = kdc-server.mylabserver.com
  admin_server = kdc-server.mylabserver.com
 }

[domain_realm]
 .mylabserver.com = MYLABSERVER.COM
 mylabserver.com = MYLABSERVER.COM
```

Add the *krbtest* user:

```
[root@kerb-client ~]# useradd krbtest
```

Where we configured the system with `kadmin.local` before, we instead use `kadmin` because we are on a remote server:

```
[root@kerb-client ~]# kadmin
```

Pass in the root password created in `kadmin.local`, and add the host:

```
kadmin:  addprinc -randkey host/kerb-client.mylabserver.com
WARNING: no policy specified for host/kerb-client.mylabserver.com@
MYLABSERVER.COM; defaulting to no policy
Principal “host/kerb-client.mylabserver.com@MYLABSERVER.COM” created.
```

And a local keytab file:

```
kadmin:  ktadd host/kerb-client.mylabserver.com
Entry for principal host/kerb-client.mylabserver.com with kvno 2,
encryption type aes256-cts-hmac-sha1-96 added to keytab FILE:/etc/krb5.
keytab.
Entry for principal host/kerb-client.mylabserver.com with kvno 2,
encryption type aes128-cts-hmac-sha1-96 added to keytab FILE:/etc/krb5.
keytab.
Entry for principal host/kerb-client.mylabserver.com with kvno 2,
encryption type des3-cbc-sha1 added to keytab FILE:/etc/krb5.keytab.
Entry for principal host/kerb-client.mylabserver.com with kvno 2,
encryption type arcfour-hmac added to keytab FILE:/etc/krb5.keytab.
Entry for principal host/kerb-client.mylabserver.com with kvno 2,
encryption type camellia256-cts-cmac added to keytab FILE:/etc/krb5.
keytab.
Entry for principal host/kerb-client.mylabserver.com with kvno 2,
encryption type camellia128-cts-cmac added to keytab FILE:/etc/krb5.
keytab.
Entry for principal host/kerb-client.mylabserver.com with kvno 2,
encryption type des-hmac-sha1 added to keytab FILE:/etc/krb5.keytab.
Entry for principal host/kerb-client.mylabserver.com with kvno 2,
encryption type des-cbc-md5 added to keytab FILE:/etc/krb5.keytab.
```

Quit `kadmin`:

```
kadmin: quit
```

As before, we need to edit the SSH configuration on our client server to enable `GSSAPIAuthentication` and `GSSAPIDelegateCredentials;` these can be found in the `/etc/ssh/ssh_config` file. The lines also need to be uncommented.

Reload SSH:

```
[root@kerb-client ~]# systemctl reload sshd
```

Configure PAM for authentication:

```
[root@kerb-client ~]# authconfig --enablekrb5 --update
```

We can now `su` into the *krbtest* user, and log into our `krb-server.mylabserver.com` server.

```
[root@kerb-client ~]# su - krbtest
[krbtest@kerb-client ~]# ssh kdc-server.mylabserver.com
```

The KDC server is now set to allow users to authenticate on known hosts.