Oracle Linux 7

Setting Up System Accounts and Authentication

F32251-04
September 2022

Oracle Linux 7 Setting Up System Accounts and Authentication,

F32251-04

Contents

2 Local Account Configuration

Preface

Oracle Linux 7: Setting Up System Accounts and Authentication provides information about about authenticating Oracle Linux 7 systems as well as user and group accounts to ensure the security of your systems environment.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at https://www.oracle.com/corporate/accessibility/.

For information about the accessibility of the Oracle Help Center, see the Oracle Accessibility Conformance Report at https://www.oracle.com/corporate/accessibility/templates/t2-11535.html.

Access to Oracle Support for Accessibility

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit https://www.oracle.com/corporate/accessibility/learning-support.html#support-tab.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry

standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1

Authentication Configuration

This chapter describes how to configure various authentication methods that Oracle Linux can use, including NIS, LDAP, Kerberos, and Winbind, and how you can configure the System Security Services Daemon feature to provide centralized identity and authentication management.

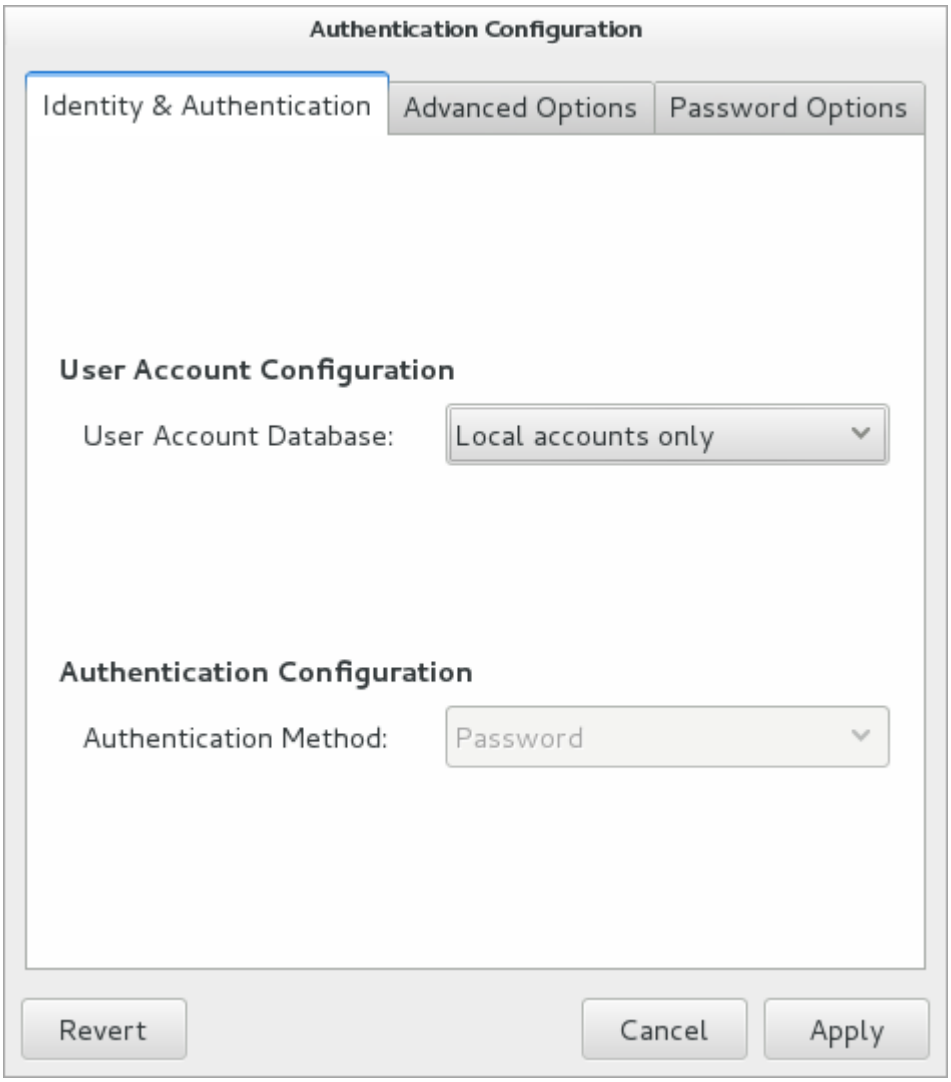
About Authentication

Authentication is the verification of the identity of an entity, such as a user, to a system. A user logs in by providing a user name and a password, and the operating system authenticates the user's identity by comparing this information to data stored on the system. If the login credentials match and the user account is active, the user is authenticated and can successfully access the system.

The information that verifies a user's identity can either be located on the local system in the `/etc/passwd` and `/etc/shadow` files, or on remote systems using Identity Policy Audit (IPA), the Lightweight Directory Access Protocol (LDAP), the Network Information Service (NIS), or Winbind. In addition, IPSv2, LDAP, and NIS data files can use the Kerberos authentication protocol, which allows nodes communicating over a non-secure network to prove their identity to one another in a secure manner.

You can use the Authentication Configuration GUI (`system-config-authentication`) to select the authentication mechanism and to configure any associated authentication options. Alternatively, you can use the `authconfig` command. Both the Authentication Configuration GUI and `authconfig` adjust settings in the PAM configuration files that are located in the `/etc/pam.d` directory. The Authentication Configuration GUI is available if you install the `authconfig-gtk` package.

Figure 1-1 Authentication Configuration of Local Accounts



About Local Oracle Linux Authentication

Unless you select a different authentication mechanism during installation or by using the Authentication Configuration GUI or the `authconfig` command, Oracle Linux verifies a user's identity by using the information that is stored in the `/etc/passwd` and `/etc/shadow` files.

The `/etc/passwd` file stores account information for each user such as his or her unique user ID (or *UID*, which is an integer), user name, home directory, and login shell. A user logs in using his or her user name, but the operating system uses the associated UID. When the user logs in, he or she is placed in his or her home directory and his or her login shell runs.

The `/etc/group` file stores information about groups of users. A user also belongs to one or more groups, and each group can contain one or more users. If you can grant access privileges to a group, all members of the group receive the same access

privileges. Each group account has a unique group ID (*GID*, again an integer) and an associated group name.

By default, Oracle Linux implements the *user private group* (*UPG*) scheme where adding a user account also creates a corresponding UPG with the same name as the user, and of which the user is the only member.

Only the `root` user can add, modify, or delete user and group accounts. By default, both users and groups use shadow passwords, which are cryptographically hashed and stored in `/etc/shadow` and `/etc/gshadow` respectively. These shadow password files are readable only by the `root` user. root can set a group password that a user must enter to become a member of the group by using the `newgrp` command. If a group does not have a password, a user can only join the group by `root` adding him or her as a member.

The `/etc/login.defs` file defines parameters for password aging and related security policies.

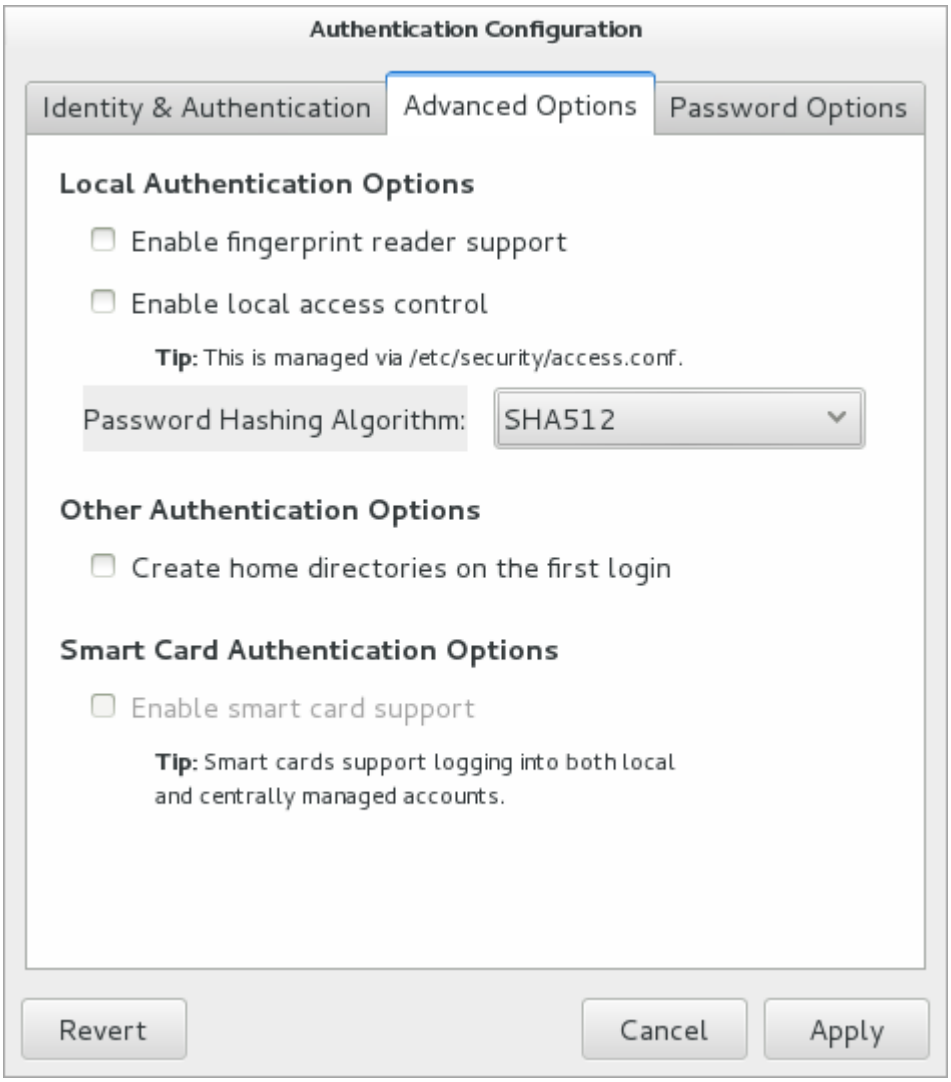
For more information about the content of these files, see the `group(5)`, `gshadow(5)`, `login.defs(5)`, `passwd(5)`, and `shadow(5)` manual pages.

Configuring Local Access

You can use the User Manager GUI (`system-config-users`) to add or delete users and groups and to modify settings such as passwords, home directories, login shells, and group membership. Alternatively, you can use commands such as `useradd` and `groupadd`. The User Manager GUI is available if you install the `system-config-users` package.

To enable local access control, select the **Enable local access control** check box on the Advanced Options tab of the Authentication Configuration GUI (`system-config-authentication`). The system can then read the `/etc/security/access.conf` file for local user authorization rules that specify login combinations that the system accepts or refuses.

Figure 1-2 Authentication Configuration Advanced Options



Alternatively, use the following command:

```
# authconfig --enablepamaccess --update
```

Each entry in `/etc/security/access.conf` takes the form:

permission : *users* : *origins* [except]

The following is a description of each entry in the file:

permission
Set to `+` or `-` to grant or deny login respectively.

users
Specifies a space-separated list of user or group names or `ALL` for any user or group. Enclose group names in parentheses to distinguish them from user names. You can use the `EXCEPT` operator to exclude a list of users from the rule.

origins
Specifies a space-separated list of host names, fully qualified domain names, network addresses, terminal device names, `ALL`, or `NONE`. You can use the `EXCEPT` operator to exclude a list of origins from the rule.

For example, the following rule denies login access by anyone except `root` from the network 192.168.2.0/24:

```
- : ALL except root : 192.168.2.0/24
```

For more information, see the `access.conf(5)` manual page and Local Account Configuration.

Configuring Fingerprint Reader Authentication

If appropriate hardware is installed and supported, the system can use fingerprint scans to authenticate users.

To enable fingerprint reader support, select the **Enable fingerprint reader support** check box on the Advanced Options tab of the Authentication Configuration GUI (`system-config-authentication`).

Alternatively, use the following command:

```
# authconfig --enablefingerprint --update
```

Configuring Smart Card Authentication

If appropriate hardware is installed and supported, the system can use smart cards to authenticate users. The `pam_pkcs11` package provides a PAM login module that enables X.509 certificate-based user authentication. The module uses the Name Service Switch (NSS) to manage and validate PKCS #11 smart cards by using locally stored root CA certificates, online or locally accessible certificate revocation lists (CRLs), and the Online Certificate Status Protocol (OCSP).

To enable smart card authentication:

1. Install the `pam_pkcs11` package:

   ```
   # yum install pam_pkcs11
   ```

2. Use the following command to install the root CA certificates in the NSS database:

   ```
   # certutil -A -d /etc/pki/nssdb -t "TC,C,C" -n "Root CA certificates" -i CACert.pem
   ```

   In the previous command, *CACert.pem* is the base-64 format root CA certificate file.

3. Run the Authentication Configuration GUI:

   ```
   # system-config-authentication
   ```

4. On the Advanced Options tab, select the **Enable smart card support** check box.
5. If you want to disable all other login authentication methods, select the **Require smart card for login** check box.

> **Caution:**
>
> Do not select this option until you have tested that can use a smart card to authenticate with the system.

6. From the **Card removal action** menu, select the system's response if a user removes a smart card while logged in to a session:

   Ignore
   The system ignores card removal for the current session.

   Lock
   The system locks the user out of the session .

You can also use the following command to configure smart card authentication:

```
# authconfig --enablesmartcard --update
```

To specify the system's response if a user removes a smart card while logged in to a session:

```
authconfig --smartcardaction=0|1 --update
```

Specify a value of 0 to `--smartcardaction` to lock the system if a card is removed. To ignore card removal, use a value of 1.

Once you have tested that you can use a smart card to authenticate with the system, you can disable all other login authentication methods.

```
# authconfig --enablerequiresmartcard --update
```

About IPA Authentication

IPA allows you to set up a domain controller for DNS, Kerberos, and authorization policies as an alternative to Active Directory Services. You can enroll client machines with an IPA domain so that they can access information for single sign-on authentication. IPA combines the capabilities of existing well-known technologies such as certificate services, DNS, LDAP, Kerberos, LDAP, and NTP.

Configuring IPA Authentication

To be able to configure IPA authentication, use `yum` to install the `ipa-client` and `ipa-admintools` packages. The `ipa-server` package is only required if you want to configure a system as an IPA server.

You can choose between two versions of IPA in the Authentication Configuration GUI:

- **FreeIPA** (effectively, IPAv1) supports identity management and authentication of users and groups, and does not require you to join your system to an IPA realm. Enter information about the LDAP and Kerberos configuration.
- **IPAv2**, which supports identity management and authentication of machines, requires you to join your system to an IPA realm. Enter information about the IPA domain configuration, optionally choose to configure NTP, and click **Join Domain** to create a machine account on the IPA server. After your system has obtained

permission to join the IPA realm, you can select and configure the authentication method.

If you use the Authentication Configuration GUI and select IPA v2 as the user account database, you are prompted to enter the names of the IPA domain, realm, and server. You can also select to configure NTP so that the system time is consistent with the IPA server. If you have initialized Kerberos, you can click **Join Domain** to create a machine account on the IPA server and grant permission to join the domain.

For more information about configuring IPA, see http://www.freeipa.org/page/Documentation.

About LDAP Authentication

The Lightweight Directory Access Protocol (LDAP) allows client systems to access information stored on LDAP servers over a network. An LDAP directory server stores information in a directory-based database that is optimized for searching and browsing, and which also supports simple functions for accessing and updating entries in the database.

Database entries are arranged in a hierarchical tree-like structure, where each directory can store information such as names, addresses, telephone numbers, network service information, printer information, and many other types of structured data. Systems can use LDAP for authentication, which allows users to access their accounts from any machine on a network.

The smallest unit of information in an LDAP directory is an entry, which can have one or more attributes. Each attribute of an entry has a name (also known as an *attribute type* or *attribute description*) and one or more values. Examples of types are *domain component* (`dc`), *common name* (`cn`), organizational unit (`ou`) and email address (`mail`). The `objectClass` attribute allows you to specify whether an attribute is required or optional. An `objectClass` attribute's value specifies the schema rules that an entry must obey.

A *distinguished name* (`dn`) uniquely identifies an entry in LDAP. The distinguished name consists of the name of the entry (the *relative distinguished name* or RDN) concatenated with the names of its ancestor entries in the LDAP directory hierarchy. For example, the distinguished name of a user with the RDN `uid=arc815` might be `uid=arc815,ou=staff,dc=mydom,dc=com`.

The following are examples of information stored in LDAP for a user:

```
# User arc815
dn: uid=arc815,ou=People,dc=mydom,dc=com
cn: John Beck
givenName: John
sn: Beck
uid: arc815
uidNumber: 5159
gidNumber: 626
homeDirectory: /nethome/arc815
loginShell: /bin/bash
mail: johnb@mydom.com
objectClass: top
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
userPassword: {SSHA}QYrFtKkqOrifgk8H4EYf68B0JxIIaLga
```

and for a group:

```
# Group employees
dn: cn=employees,ou=Groups,dc=mydom,dc=com
```

```
cn: employees
gidNumber: 626
objectClass: top
objectClass: posixGroup
memberUid: arc815
memberUid: arc891
```

About LDAP Data Interchange Format

LDAP data itself is stored in a binary format. LDAP Data Interchange Format (LDIF) is a plain-text representation of an LDAP entry that allows the import and export LDAP data, usually to transfer the data between systems, but it also allows you to use a text editor to modify the content.

The data for an entry in an LDIF file takes the form:

```
[id] dn: distinguished_name
       attribute_type:value | [attribute=]value[, [attribute=] value]...
...
objectClass: value
...
```

The optional *id* number is determined by the application that you use to edit the entry. Each attribute type for an entry contains either a value or a comma-separated list of attribute and value pairs as defined in the LDAP directory schema.

There must be a blank line between each `dn` definition section or `include:` line. There must not be any other blank lines or any empty space at the ends of lines. An empty space at the start of a line indicates a continuation of the previous line.

Configuring an LDAP Server

OpenLDAP is an open-source implementation of LDAP that allows you configure an LDAP directory server.

To configure a system as an LDAP server:

1. Install the OpenLDAP packages:

   ```
   # yum install openldap openldap-servers openldap-clients nss-pam-ldapd
   ```

   The OpenLDAP configuration is stored in the following files below `/etc/openldap`:

   `ldap.conf`
   The configuration file for client applications.

   `slapd.d/cn=config.ldif`
   The default global configuration LDIF file for OpenLDAP.

   `slapd.d/cn=config/*.ldif`
   Configuration LDIF files for the database and schema.

   `slapd.d/cn=config/cn=schema/*.ldif`
   Schema configuration LDIF files. More information about the OpenLDAP schema is available in OpenLDAP administrator guides at https://www.openldap.org/doc/.

> **Note:**
>
> You should never need to edit any files under `/etc/openldap/slapd.d` as you can reconfigure OpenLDAP while the `slapd` service is running.

2. If you want configure `slapd` to listen on port 636 for connections over an SSL tunnel (`ldaps://`), edit `/etc/sysconfig/slapd`, and change the value of `SLAPD_LDAPS` to `yes`:

   ```
   SLAPD_LDAPS=yes
   ```

   If required, you can prevent `slapd` listening on port 389 for `ldap://` connections, by changing the value of `SLAPD_LDAP` to `no`:

   ```
   SLAPD_LDAP=no
   ```

   Ensure that you also define the correct SLAPD_URLS for the ports that are enabled. For instance, if you intend to use SSL and you wish `slapd` to listen on port 636, you must specify `ldaps://` as one of the supported URLS. For example:

   ```
   SLAPD_URLS="ldapi:/// ldap:/// ldaps:///"
   ```

3. Configure the system firewall to allow incoming TCP connections on port 389, for example:

   ```
   # firewall-cmd --zone=zone --add-port=389/tcp
   # firewall-cmd --permanent --zone=zone --add-port=389/tcp
   ```

   The primary TCP port for LDAP is 389. If you configure LDAP to use an SSL tunnel (`ldaps`), substitute the port number that the tunnel uses, which is usually 636, for example:

   ```
   # firewall-cmd --zone=zone --add-port=636/tcp
   # firewall-cmd --permanent --zone=zone --add-port=636/tcp
   ```

4. Change the user and group ownership of `/var/lib/ldap` and any files that it contains to `ldap`:

   ```
   # cd /var/lib/ldap
   # chown ldap:ldap ./*
   ```

5. Start the `slapd` service and configure it to start following system reboots:

   ```
   # systemctl start slapd
   # systemctl enable slapd
   ```

6. Generate a hash of the LDAP password that you will use with the `olcRootPW` entry in the configuration file for your domain database, for example:

   ```
   # slappasswd -h {SSHA}
   New password: password
   Re-enter new password: password
   {SSHA}lkMShz73MZBic19Q4pfOaXNxpLN3wLRy
   ```

7. Create an LDIF file with a name such as `config-mydom-com.ldif` that contains configuration entries for your domain database based on the following example:

   ```
   # Load the schema files required for accounts
   include file:///etc/openldap/schema/cosine.ldif

   include file:///etc/openldap/schema/nis.ldif
   ```

```
include file:///etc/openldap/schema/inetorgperson.ldif

# Load the HDB (hierarchical database) backend modules
dn: cn=module,cn=config
objectClass: olcModuleList
cn: module
olcModulepath: /usr/lib64/openldap
olcModuleload: back_hdb

# Configure the database settings
dn: olcDatabase=hdb,cn=config
objectClass: olcDatabaseConfig
objectClass: olcHdbConfig
olcDatabase: {1}hdb
olcSuffix: dc=mydom,dc=com
# The database directory must already exist
# and it should only be owned by ldap:ldap.
# Setting its mode to 0700 is recommended
olcDbDirectory: /var/lib/ldap
olcRootDN: cn=admin,dc=mydom,dc=com
olcRootPW: {SSHA}lkMShz73MZBic19Q4pfOaXNxpLN3wLRy
olcDbConfig: set_cachesize 0 10485760 0
olcDbConfig: set_lk_max_objects 2000
olcDbConfig: set_lk_max_locks 2000
olcDbConfig: set_lk_max_lockers 2000
olcDbIndex: objectClass eq
olcLastMod: TRUE
olcDbCheckpoint: 1024 10
# Set up access control
olcAccess: to attrs=userPassword
  by dn="cn=admin,dc=mydom,dc=com"
  write by anonymous auth
  by self write
  by * none
olcAccess: to attrs=shadowLastChange
  by self write
  by * read
olcAccess: to dn.base=""
  by * read
olcAccess: to *
  by dn="cn=admin,dc=mydom,dc=com"
  write by * read
```

> **Note:**
>
> This configuration file allows you to reconfigure `slapd` while it is running. If you use a `slapd.conf` configuration file, you can also update `slapd` dynamically, but such changes do not persist if you restart the server.

For more information, see the `slapd-config(5)` manual page.

8. Use the `ldapadd` command to add the LDIF file:

```
# ldapadd -Y EXTERNAL -H ldapi:/// -f config-mydom-com.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
```

```
adding new entry "cn=module,cn=config"

adding new entry "olcDatabase=hdb,cn=config"
```

For more information about configuring OpenLDAP, see the `slapadd(8C)`, `slapd(8C)`, `slapd-config(5)`, and `slappasswd(8C)` manual pages, the OpenLDAP Administrator's Guide (`/usr/share/doc/openldap-servers-`*`version`*`/guide.html`), and the latest OpenLDAP documentation at https://www.openldap.org/doc/.

Replacing the Default Certificates

If you configure LDAP to use Transport Layer Security (TLS) or Secure Sockets Layer (SSL) to secure the connection to the LDAP server, you need a public certificate that clients can download. You can obtain certificates from a Certification Authority (CA) or you can use the `openssl` command to create the certificate. See Creating and Distributing Self-signed CA Certificates.

Once you have a server certificate, its corresponding private key file, and a root CA certificate, you can replace the default certificates that are installed in `/etc/openldap/certs`.

To display the existing certificate entries that `slapd` uses with TLS, use the `ldapsearch` command:

```
# ldapsearch -LLL -Y EXTERNAL -H ldapi:/// -b "cn=config" \
  olcTLSCACertificatePath olcTLSCertificateFile olcTLSCertificateKeyFile
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
dn: cn=config
olcTLSCACertificatePath: /etc/openldap/certs
olcTLSCertificateFile: "OpenLDAP Server"
olcTLSCertificateKeyFile: /etc/openldap/certs/password
...
```

To replace the TLS attributes in the LDAP configuration:

1. Create an LDIF file that defines how to modify the attributes, for example:

```
dn: cn=config
changetype: modify
delete: olcTLSCACertificatePath

# Omit the following clause for olcTLSCACertificateFile
# if you do not have a separate root CA certificate
dn: cn=config
changetype: modify
add: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ssl/certsCAcert.pem

dn: cn=config
changetype: modify
replace: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ssl/certs/server-cert.pem

dn: cn=config
changetype: modify
replace: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ssl/certs/server-key.pem

dn: cn=config
```

```
changetype: modify
add: olcTLSCipherSuite
olcTLSCipherSuite: TLSv1+RSA:!NULL

dn: cn=config
changetype: modify
add: olcTLSVerifyClient
olcTLSVerifyClient: never
```

If you generate only a self-signed certificate and its corresponding key file, you do not need to specify a root CA certificate.

2. Use the `ldapmodify` command to apply the LDIF file:

```
# ldapmodify -Y EXTERNAL -H ldapi:/// -f mod-TLS.ldif
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "cn=config"

modifying entry "cn=config"

modifying entry "cn=config"
...
```

3. Verify that the entries have changed:

```
# ldapsearch -LLL -Y EXTERNAL -H ldapi:/// -b "cn=config" \
  olcTLSCACertificatePath olcTLSCertificateFile
olcTLSCertificateKeyFile
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
dn: cn=config
olcTLSCACertificateFile: /etc/ssl/certs/CAcert.pem
olcTLSCertificateFile: /etc/ssl/certs/server-cert.pem
olcTLSCertificateKeyFile: /etc/ssl/certs/server-key.pem
olcTLSCipherSuite: TLSv1+RSA:!NULL
olcTLSVerifyClient: never
...
```

4. Restart the `slapd` service to make it use the new certificates:

```
# systemctl restart slapd
```

For more information, see the `ldapmodify(1)`, `ldapsearch(1)` and `openssl(1)` manual pages.

Creating and Distributing Self-signed CA Certificates

For usage solely within an organization, you might want to create certificates that you can use with LDAP. There are a number of ways of creating suitable certificates, for example:

- Create a self-signed CA certificate together with a private key file.
- Create a self-signed root CA certificate and private key file, and use the CA certificate and its key file to sign a separate server certificate for each server.

The following procedure describes how to use `openssl` to create a self-signed CA certificate and private key file, and then use these files to sign server certificates.

To create the CA certificate and use it to sign a server certificate:

1. Change directory to `/etc/openldap/certs` on the LDAP server:

```
# cd /etc/openldap/certs
```

2. Create the private key file `CAcert-key.pem` for the CA certificate:

```
# openssl genrsa -out CAcert-key.pem 1024
Generating RSA private key, 1024 bit long modulus
......++++++
....++++++
e is 65537 (0x10001)
```

3. Change the mode on the key file to 0400:

```
# chmod 0400 CAcert-key.pem
```

4. Create the certificate request `CAcert.csr`:

```
# openssl req -new -key CAcert-key.pem -out CAcert.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:US
State or Province Name (full name) []:California
Locality Name (eg, city) [Default City]:Redwood City
Organization Name (eg, company) [Default Company Ltd]:Mydom Inc
Organizational Unit Name (eg, section) []:Org
Common Name (eg, your name or your server's hostname) []:www.mydom.org
Email Address []:root@mydom.org

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:<Enter>
An optional company name []:<Enter>
```

5. Create a CA certificate that is valid for approximately three years:

```
# openssl x509 -req -days 1095 -in CAcert.csr -signkey CAcert-key.pem -out
CAcert.pem
rt-key.pem -out CAcert.pem
Signature ok
subject=/C=US/ST=California/L=Redwood City/O=Mydom
Inc/OU=Org/CN=www.mydom.org/emailAddress=root@mydom.org
Getting Private key
```

6. For each server certificate that you want to create:

   a. Create the private key for the server certificate:

```
# openssl genrsa -out server-key.pem 1024
Generating RSA private key, 1024 bit long modulus
.............++++++
.............................++++++
e is 65537 (0x10001)
```

> **Note:**
>
> If you intend to generate server certificates for several servers, name the certificate, its key file, and the certificate request so that you can easily identify both the server and the service, for example, `ldap_host02-cert.pem`, `ldap_host02-key.pem`, and `ldap_host02-cert.csr`.

b. Change the mode on the key file to 0400, and change its user and group ownership to `ldap`:

```
# chmod 0400 server-key.pem
# chown ldap:ldap server-key.pem
```

c. Create the certificate request `server-cert.csr`:

```
# openssl req -new -key server-key.pem -out server-cert.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a
DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:US
State or Province Name (full name) []:California
Locality Name (eg, city) [Default City]:Redwood City
Organization Name (eg, company) [Default Company Ltd]:Mydom Inc
Organizational Unit Name (eg, section) []:Org
Common Name (eg, your name or your server's hostname) []:ldap.mydom.com
Email Address []:root@mydom.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:<Enter>
An optional company name []:<Enter>
```

> **Note:**
>
> For the Common Name, specify the Fully Qualified Domain Name (FQDN) of the server. If the FQDN of the server does not match the common name specified in the certificate, clients cannot obtain a connection to the server.

d. Use the CA certificate and its corresponding key file to sign the certificate request and generate the server certificate:

```
# openssl x509 -req -days 1095 -CAcreateserial \
  -in server-cert.csr -CA CAcert.pem -CAkey CAcert-key.pem \
  -out server-cert.pem
Signature ok
subject=/C=US/ST=California/L=Redwood City/O=Mydom
Inc/OU=Org/CN=ldap.mydom.com/emailAddress=root@mydom.com
Getting CA Private Key
```

7. If you generate server certificates for other LDAP servers, copy the appropriate server certificate, its corresponding key file, and the CA certificate to `/etc/openldap/certs` on those servers.
8. Set up a web server to host the CA certificate for access by clients. The following steps assume that the LDAP server performs this function. You can use any suitable, alternative server instead.
   a. Install the Apache HTTP server.

      ```
      # yum install httpd
      ```

   b. Create a directory for the CA certificate under `/var/www/html`, for example:

      ```
      # mkdir /var/www/html/certs
      ```

   c. Copy the CA certificate to `/var/www/html/certs`.

      ```
      # cp CAcert.pem /var/www/html/certs
      ```

      > **Caution:**
      >
      > Do not copy the key files.

   d. Edit the HTTP server configuration file, `/etc/httpd/conf/httpd.conf`, and specify the resolvable domain name of the server in the argument to `ServerName`.

      ```
      ServerName server_addr:80
      ```

      If the server does not have a resolvable domain name, enter its IP address instead.

      Verify that the setting of the `Options` directive in the `<Directory "/var/www/html">` section specifies `Indexes` and `FollowSymLinks` to allow you to browse the directory hierarchy, for example:

      ```
      Options Indexes FollowSymLinks
      ```

   e. Start the Apache HTTP server, and configure it to start after a reboot.

      ```
      # systemctl start httpd
      # systemctl enable httpd
      ```

   f. If you have enabled the firewall on your system, configure it to allow incoming HTTP connection requests on TCP port 80, for example:

      ```
      # firewall-cmd --zone=zone --add-port=80/tcp
      # firewall-cmd --permanent --zone=zone --add-port=80/tcp
      ```

Initializing an Organization in LDAP

Before you can define people, groups, servers, printers, and other entitles for your organization, you must first set up information in LDAP for the organization itself.

To define an organization in LDAP:

1. Create an LDIF file that defines the organization, for example `mydom-com-organization.ldif`:

   ```
   # Organization mydom.com
   dn: dc=mydom,dc=com
   ```

```
dc: mydom
objectclass: dcObject
objectclass: organizationalUnit
ou: mydom.com

# Users
dn: ou=People,dc=mydom,dc=com
objectClass: organizationalUnit
ou: people

# Groups
dn: ou=Groups,dc=mydom,dc=com
objectClass: organizationalUnit
ou: groups
```

2. If you have configured LDAP authentication, use the `ldapadd` command to add the organization to LDAP:

   ```
   # ldapadd -cxWD "cn=admin,dc=mydom,dc=com" -f mydom-com-
   organization.ldif
   Enter LDAP Password: admin_password
   adding new entry "dc=mydom,dc=com"

   adding new entry "ou=People,dc=mydom,dc=com"

   adding new entry "ou=Groups,dc=mydom,dc=com"
   ```

   If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the `ldapadd` command:

   ```
   # ldapadd -f mydom-com-organization.ldif
   ```

For more information, see the `ldapadd(1)` manual page.

Adding an Automount Map to LDAP

You can make an automount map such as `auto.home` available in LDAP so that the automounter mounts a user's home directory on demand.

To add the `auto.home` map to LDAP:

1. Create an LDIF file that defines entries for the map's name and its contents, for example `auto-home.ldif`:

   ```
   dn: nisMapName=auto.home,dc=mydom,dc=com
   objectClass: top
   objectClass: nisMap
   nisMapName: auto.home

   dn: cn=*,nisMapName=auto.home,dc=mydom,dc=com
   objectClass: nisObject
   cn: *
   nisMapEntry: -rw,sync nfssvr:/nethome/&
   nisMapName: auto.home
   ```

   In the previous example, *nfssvr* is the host name or IP address of the NFS server that exports the home directories of the users.

2. If you have configured LDAP authentication, use the following command to add the map to LDAP:

```
# ldapadd -xcWD "cn=admin,dc=mydom,dc=com" \
  -f auto-home.ldif
Enter LDAP Password: user_password
adding new entry "nisMapName=auto.home,dc=mydom,dc=com"

adding new entry "cn=*,nisMapName=auto.home,dc=mydom,dc=com"
```

If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the command:

```
# ldapmodify -f auto-home.ldif
```

3. Verify that the map appears in LDAP:

```
# ldapsearch -LLL -x -b "dc=mydom,dc=com" nisMapName=auto.home
dn: nisMapName=auto.home,dc=mydom,dc=com
objectClass: top
objectClass: nisMap
nisMapName: auto.home

dn: cn=*,nisMapName=auto.home,dc=mydom,dc=com
objectClass: nisObject
cn: *
nisMapEntry: -rw,sync nfssvr.mydom.com:/nethome/&
nisMapName: auto.home
```

Adding a Group to LDAP

If you configure users in user private groups (UPGs), define that group along with the user. See Adding a User to LDAP.

To add a group to LDAP:

1. Create an LDIF file that defines the group, for example `employees-group.ldif`:

```
# Group employees
dn: cn=employees,ou=Groups,dc=mydom,dc=com
cn: employees
gidNumber: 626
objectClass: top
objectclass: posixGroup
```

2. If you have configured LDAP authentication, use the following command to add the group to LDAP:

```
# ldapadd -cxWD "cn=admin,dc=mydom,dc=com" -f employees-group.ldif
Enter LDAP Password: admin_password
adding new entry "cn=employees,ou=Groups,dc=mydom,dc=com"
```

If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the `ldapadd` command:

```
# ldapadd -f employees-group.ldif
```

3. Verify that you can locate the group in LDAP:

```
# ldapsearch -LLL -x -b "dc=mydom,dc=com" gidNumber=626
dn: cn=employees,ou=Groups,dc=mydom,dc=com
cn: employees
gidNumber: 626
objectClass: top
objectClass: posixGroup
```

For more information, see the `ldapadd(1)` and `ldapsearch(1)` manual pages.

Adding a User to LDAP

> **Note:**
>
> This procedure assumes that:
>
> - LDAP provides information for `ou=People`, `ou=Groups`, and `nisMapName=auto.home`.
> - The LDAP server uses NFS to export the users' home directories. See Mounting an NFS Filesystem in Oracle Linux 7: Managing File Systems.

To create an account for a user on the LDAP server:

1. If the LDAP server does not already export the base directory of the users' home directories, perform the following steps on the LDAP server:
   a. Create the base directory for user directories, for example `/nethome`:

      ```
      # mkdir /nethome
      ```

   b. Add an entry such as the following to `/etc/exports`:

      ```
      /nethome    *(rw,sync)
      ```

      You might prefer to restrict which clients can mount the file system. For example, the following entry allows only clients in the 192.168.1.0/24 subnet to mount `/nethome`:

      ```
      /nethome    192.168.1.0/24(rw,sync)
      ```

   c. Use the following command to export the file system:

      ```
      # exportfs -i -o ro,sync *:/nethome
      ```

2. Create the user account, but do not allow local logins:

   ```
   # useradd -b base_dir -s /sbin/nologin -u UID -U
   username
   ```

   For example:

   ```
   # useradd -b /nethome -s /sbin/nologin -u 5159 -U arc815
   ```

   The command updates the `/etc/passwd` file and creates a home directory under `/nethome` on the LDAP server.

   The user's login shell will be overridden by the `LoginShell` value set in LDAP.

3. Use the `id` command to list the user and group IDs that have been assigned to the user, for example:

   ```
   # id arc815
   uid=5159(arc815) gid=5159(arc815) groups=5159(arc815)
   ```

4. Create an LDIF file that defines the user, for example `arc815-user.ldif`:

```
# UPG arc815
dn: cn=arc815,ou=Groups,dc=mydom,dc=com
cn: arc815
gidNumber: 5159
objectclass: top
objectclass: posixGroup

# User arc815
dn: uid=arc815,ou=People,dc=mydom,dc=com
cn: John Beck
givenName: John
sn: Beck
uid: arc815
uidNumber: 5159
gidNumber: 5159
homeDirectory: /nethome/arc815
loginShell: /bin/bash
mail: johnb@mydom.com
objectClass: top
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
userPassword: {SSHA}x
```

In this example, the user belongs to a user private group (UPG), which is defined in the same file. The user's login shell attribute `LoginShell` is set to `/bin/bash`. The user's password attribute `userPassword` is set to a placeholder value. If you use Kerberos authentication with LDAP, this attribute is not used.

5. If you have configured LDAP authentication, use the following command to add the user to LDAP:

```
# ldapadd -cxWD cn=admin,dc=mydom,dc=com -f arc815-user.ldif
Enter LDAP Password: admin_password
adding new entry "cn=arc815,ou=Groups,dc=mydom,dc=com"

adding new entry "uid=arc815,ou=People,dc=mydom,dc=com"
```

If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the `ldapadd` command:

```
# ldapadd -f arc815-user.ldif
```

6. Verify that you can locate the user and his or her UPG in LDAP:

```
# ldapsearch -LLL -x -b "dc=mydom,dc=com" '(|(uid=arc815)(cn=arc815))'
dn: cn=arc815,ou=Groups,dc=mydom,dc=com
cn: arc815
gidNumber: 5159
objectClass: top
objectClass: posixGroup

dn: uid=arc815,ou=People,dc=mydom,dc=com
cn: John Beck
givenName: John
sn: Beck
uid: arc815
uidNumber: 5159
gidNumber: 5159
homeDirectory: /home/arc815
loginShell: /bin/bash
```

```
mail: johnb@mydom.com
objectClass: top
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
```

7. If you have configured LDAP authentication, set the user password in LDAP:

```
# ldappasswd -xWD "cn=admin,dc=mydom,dc=com" \
  -S "uid=arc815,ou=people,dc=mydom,dc=com"
New password: user_password
Re-enter new password: user_password
Enter LDAP Password: admin_password
```

   If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use the `kadmin` command to add the user (principal) and password to the database for the Kerberos domain, for example:

```
# kadmin -q "addprinc alice@MYDOM.COM"
```

For more information, see the `kadmin(1)`, `ldapadd(1)`, `ldappasswd(1)`, and `ldapsearch(1)` manual pages.

Adding Users to a Group in LDAP

To add users to an existing group in LDAP:

1. Create an LDIF file that defines the users that should be added to the `memberuid` attribute for the group, for example `employees-add-users.ldif`:

```
dn: cn=employees,ou=Groups,dc=mydom,dc=com
changetype: modify
add: memberUid
memberUid: arc815

dn: cn=employees,ou=Groups,dc=mydom,dc=com
changetype: modify
add: memberUid
memberUid: arc891

...
```

2. If you have configured LDAP authentication, use the following command to add the group to LDAP:

```
# ldapmodify -xcWD "cn=admin,dc=mydom,dc=com" \
  -f employees-add-users.ldif
Enter LDAP Password: user_password
modifying entry "cn=employees,ou=Groups,dc=mydom,dc=com"
...
```

   If you have configured Kerberos authentication, use `kinit` to obtain a ticket granting ticket (TGT) for the `admin` principal, and use this form of the command:

```
# ldapmodify -f employees-add-users.ldif
```

3. Verify that the group has been updated in LDAP:

```
# ldapsearch -LLL -x -b "dc=mydom,dc=com" gidNumber=626
dn: cn=employees,ou=Groups,dc=mydom,dc=com
```

```
cn: employees
gidNumber: 626
objectClass: top
objectClass: posixGroup
memberUid: arc815
memberUid: arc891
...
```

Enabling LDAP Authentication

To enable LDAP authentication for an LDAP client by using the Authentication Configuration GUI:

1. Install the `openldap-clients` package:

   ```
   # yum install openldap-clients
   ```

2. Run the Authentication Configuration GUI:

   ```
   # system-config-authentication
   ```

3. Select **LDAP** as the user account database and enter values for:

   LDAP Search Base DN
   The LDAP Search Base DN for the database. For example: `dc=mydom,dc=com`.

   LDAP Server
   The URL of the LDAP server including the port number. For example, `ldap://ldap.mydom.com:389` or `ldaps://ldap.mydom.com:636`.

   LDAP authentication requires that you use either LDAP over SSL (`ldaps`) or Transport Layer Security (TLS) to secure the connection to the LDAP server.

4. If you use TLS, click **Download CA Certificate** and enter the URL from which to download the CA certificate that provides the basis for authentication within the domain.

5. Select either **LDAP password** or **Kerberos password** for authentication.

6. If you select Kerberos authentication, enter values for:

   Realm
   The name of the Kerberos realm.

   KDCs
   A comma-separated list of Key Distribution Center (KDC) servers that can issue Kerberos ticket granting tickets and service tickets.

   Admin Servers
   A comma-separated list of Kerberos administration servers.

   Alternatively, you can use DNS to configure these settings:

   - Select the **Use DNS to resolve hosts to realms** check box to look up the name of the realm defined as a `TXT` record in DNS, for example:

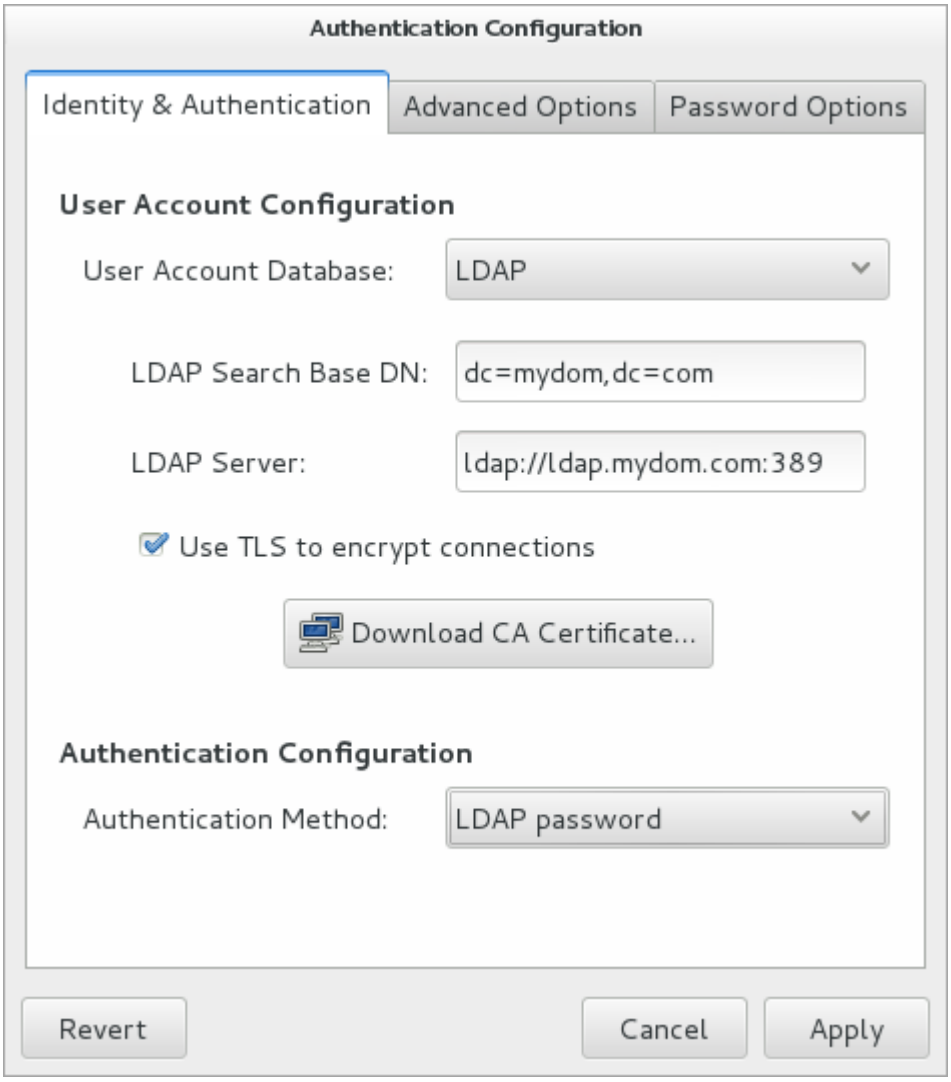
     ```
     _kerberos.mydom.com     IN TXT "MYDOM.COM"
     ```

   - Select the **Use DNS to locate KDCs for realms** check box to look up the KDCs and administration servers defined as `SVR` records in DNS, for example:

```
_kerberos._tcp.mydom.com      IN SVR 1  0 88  krbsvr.mydom.com
_kerberos._udp.mydom.com      IN SVR 1  0 88  krbsvr.mydom.com
_kpasswd._udp.mydom.com       IN SVR 1  0 464 krbsvr.mydom.com
_kerberos-adm._tcp.mydom.com  IN SVR 1  0 749 krbsvr.mydom.com
```

7. Click **Apply** to save your changes.

Figure 1-3 Authentication Configuration Using LDAP



You can also enable LDAP by using the `authconfig` command.

To use LDAP as the authentication source, specify the `--enableldapauth` option together with the full LDAP server URL including the port number and the LDAP Search Base DN, as shown in the following example:.

```
# authconfig --enableldap --enableldapauth \
  --ldapserver=ldaps://ldap.mydom.com:636 \
  --ldapbasedn="ou=people,dc=mydom,dc=com" \
  --update
```

If you want to use TLS, additionally specify the `--enableldaptls` option and the download URL of the CA certificate, for example:

```
# authconfig --enableldap --enableldapauth \
  --ldapserver=ldap://ldap.mydom.com:389 \
  --ldapbasedn="ou=people,dc=mydom,dc=com" \
  --enableldaptls \
  --ldaploadcacert=https://ca-server.mydom.com/CAcert.pem \
  --update
```

The `--enableldap` option configures `/etc/nsswitch.conf` to enable the system to use LDAP and SSSD for information services. The `--enableldapauth` option enables LDAP authentication by modifying the PAM configuration files in `/etc/pam.d` to use the `pam_ldap.so` module.

For more information, see the `authconfig(8)`, `pam_ldap(5)`, and `nsswitch.conf(5)` manual pages.

For information about using Kerberos authentication with LDAP, see Enabling Kerberos Authentication.

> **Note:**
>
> You must also configure SSSD to be able to access information in LDAP. See Configuring an LDAP Client to use SSSD.
>
> If your client uses automount maps stored in LDAP, you must configure `autofs` to work with LDAP. See Configuring an LDAP Client to Use Automount Maps.

Configuring an LDAP Client to use SSSD

The Authentication Configuration GUI and `authconfig` configure access to LDAP via `sss` entries in `/etc/nsswitch.conf` so you must configure the System Security Services Daemon (SSSD) on the LDAP client.

To configure an LDAP client to use SSSD:

1. Install the `sssd` and `sssd-client` packages:

   ```
   # yum install sssd sssd-client
   ```

2. Edit the `/etc/sssd/sssd.conf` configuration file and configure the sections to support the required services, for example:

   ```
   [sssd]
   config_file_version = 2
   domains = default
   services = nss, pam

   [domain/default]
   id_provider = ldap
   ldap_uri = ldap://ldap.mydom.com
   ldap_id_use_start_tls = true
   ldap_search_base = dc=mydom,dc=com
   ldap_tls_cacertdir = /etc/openldap/cacerts
   ```

```
auth_provider = krb5
chpass_provider = krb5
krb5_realm = MYDOM.COM
krb5_server = krbsvr.mydom.com
krb5_kpasswd = krbsvr.mydom.com
cache_credentials = true

[domain/LDAP]
id_provider = ldap
ldap_uri = ldap://ldap.mydom.com
ldap_search_base = dc=mydom,dc=com

auth_provider = krb5
krb5_realm = MYDOM.COM
krb5_server = kdcsvr.mydom.com
cache_credentials = true

min_id = 5000
max_id = 25000
enumerate = false

[nss]
filter_groups = root
filter_users = root
reconnection_retries = 3
entry_cache_timeout = 300

[pam]
reconnection_retries = 3
offline_credentials_expiration = 2
offline_failed_login_attempts = 3
offline_failed_login_delay = 5
```

3. Change the mode of `/etc/sssd/sssd.conf` to 0600:

```
# chmod 0600 /etc/sssd/sssd.conf
```

4. Enable the SSSD service:

```
# authconfig --update --enablesssd --enablesssdauth
```

For more information, see the `sssd.conf(5)` manual page and About the System Security Services Daemon.

Configuring an LDAP Client to Use Automount Maps

If you have configured an automount map for `auto.home` in LDAP, you can configure an LDAP client to mount the users' home directories when they log in.

To configure an LDAP client to automount users' home directories:

1. Install the `autofs` package:

```
# yum install autofs
```

2. Verify that the `auto.home` map is available :

```
# ldapsearch -LLL -x -b "dc=mydom,dc=com" nisMapName=auto.home
dn: nisMapName=auto.home,dc=mydom,dc=com
objectClass: top
objectClass: nisMap
nisMapName: auto.home
```

```
dn: cn=*,nisMapName=auto.home,dc=mydom,dc=com
objectClass: nisObject
cn: *
nisMapEntry: -rw,sync nfssvr.mydom.com:/nethome/&
nisMapName: auto.home
```

In this example, the map is available. For details of how to make this map available, see Adding an Automount Map to LDAP.

3. If the `auto.home` map is available, edit `/etc/auto.master` and create an entry that tells `autofs` where to find the `auto.home` map in LDAP, for example:

```
/nethome    ldap:nisMapName=auto.home,dc=mydom,dc=com
```

If you use LDAP over SSL, specify `ldaps:` instead of `ldap:`.

4. Edit `/etc/autofs_ldap_auth.conf` and configure the authentication settings for `autofs` with LDAP, for example:

```
<autofs_ldap_sasl_conf
     usetls="yes"
     tlsrequired="no"
     authrequired="autodetect"
     authtype="GSSAPI"
     clientprinc="host/ldapclient.mydom.com@MYDOM.COM"
     />
```

This example assumes that Kerberos authentication with the LDAP server uses TLS for the connection. The principal for the client system must exist in the Kerberos database. You can use the `klist -k` command to verify this. If the principal for the client does not exist, use `kadmin` to add the principal.

5. If you use Kerberos Authentication, use `kadmin` to add a principal for the LDAP service on the LDAP server, for example:

```
# kadmin -q "addprinc ldap/ldap.mydom.com@MYDOM.COM
```

6. Restart the `autofs` service, and configure the service to start following a system reboot:

```
# systemctl restart autofs
# systemctl enable autofs
```

The `autofs` service creates the directory `/nethome`. When a user logs in, the automounter mounts his or her home directory under `/nethome`.

If the owner and group for the user's files are unexpectedly listed as the anonymous user or group (`nobody` or `nogroup`) and `all_squash` has not been specified as a mount option, verify that the `Domain` setting in `/etc/idmapd.conf` on the NFS server is set to the DNS domain name. Restart the NFS services on the NFS server if you change this file.

For more information, see the `auto.master(5)` and `autofs_ldap_auth.conf(5)` manual pages.

About NIS Authentication

NIS stores administrative information such as user names, passwords, and host names on a centralized server. Client systems on the network can access this common data. This configuration allows to move from machine to machine without having to remember different passwords and copy data from one machine to another. Storing administrative information

centrally, and providing a means of accessing it from networked systems, also ensures the consistency of that data. NIS also reduces the overhead of maintaining administration files such as `/etc/passwd` on each system.

A network of NIS systems is an *NIS domain*. Each system within the domain has the same NIS domain name, which is different from a DNS domain name. The DNS domain is used throughout the Internet to refer to a group of systems. an NIS domain is used to identify systems that use files on an NIS server. an NIS domain must have exactly one primary (master) server but can have multiple worker or secondary (slave) servers.

> **NOT_SUPPORTED:**
>
> NIS authentication is deprecated as it has security issues, including a lack of protection of authentication data.

About NIS Maps

The administrative files within an NIS domain are NIS maps, which are `dbm`-format files that you generate from existing configuration files such as `/etc/passwd`, `/etc/shadow`, and `/etc/groups`. Each map is indexed on one field, and records are retrieved by specifying a value from that field. Some source files such as `/etc/passwd` have two maps:

`passwd.byname`
Indexed on user name.

`passwd.byuid`
Indexed on user ID.

The `/var/yp/nicknames` file contains a list of commonly used short names for maps such as `passwd` for `passwd.byname` and `group` for `group.byname`.

You can use the `ypcat` command to display the contents of an NIS map, for example:

```
# ypcat - passwd | grep 1500
guest:$6$gMIxsr3W$LaAo...6EE6sdsFPI2mdm7/NEm0:1500:1500::/nethome/guest:/bin/bash
```

> **Note:**
>
> As the `ypcat` command displays password hashes to any user, this example demonstrates that NIS authentication is inherently insecure against password-hash cracking programs. If you use Kerberos authentication, you can configure password hashes not to appear in NIS maps, although other information that `ypcat` displays could also be useful to an attacker.

For more information, see the `ypcat(1)` manual page.

Configuring an NIS Server

NIS primary servers act as a central, authoritative repository for NIS information. NIS worker servers act as mirrors of this information. There must be only one NIS primary server in an NIS domain. The number of NIS worker servers is optional, but creating at least one worker server provides a degree of redundancy should the primary server be unavailable.

To configure an NIS primary or worker server:

1. Install the `ypserv` package:

   ```
   # yum install ypserv
   ```

2. Edit `/etc/sysconfig/network` and add an entry to define the NIS domain, for example:

   ```
   NISDOMAIN=mynisdom
   ```

3. Edit `/etc/ypserv.conf` to configure NIS options and to add rules for which hosts and domains can access which NIS maps.

   For example, the following entries allow access only to NIS clients in the `mynisdom` domain on the 192.168.1 subnet:

   ```
   192.168.1.0/24: mynisdom : * : none
   * : * : * : deny
   ```

   For more information, see the `ypserv.conf(5)` manual page and the comments in `/etc/ypserv.conf`.

4. Create the file `/var/yp/securenets` and add entries for the networks for which the server should respond to requests, for example:

   ```
   # cat > /var/yp/securenets <<!
   255.255.255.255 127.0.0.1
   255.255.255.0   192.168.1.0
   !
   # cat /var/yp/securenets
   255.255.255.255 127.0.0.1
   255.255.255.0   192.168.1.0
   ```

   In this example, the server accepts requests from the local loopback interface and the 192.168.1 subnet.

5. Edit `/var/yp/Makefile`:

   a. Set any required map options and specify which NIS maps to create using the `all` target, for example:

      ```
      all:
      passwd group auto.home
      # hosts rpc services netid protocols mail \
      # netgrp shadow publickey networks ethers bootparams printcap \
      # amd.home auto.local. passwd.adjunct \
      # timezone locale netmasks
      ```

      This example allows NIS to create maps for the `/etc/passwd`, `/etc/group`, and `/etc/auto.home` files. By default, the information from the `/etc/shadow` file is merged with the `passwd` maps, and the information from the `/etc/gshadow` file is merged with the `group` maps.

      For more information, see the comments in `/var/yp/Makefile`.

b. If you intend to use Kerberos authentication instead of NIS authentication, change the values of `MERGE_PASSWD` and `MERGE_GROUP` to `false`:

```
MERGE_PASSWD=false
MERGE_GROUP=false
```

> **Note:**
>
> These settings prevent password hashes from appearing in the NIS maps.

c. If you configure any NIS worker servers in the domain, set the value of `NOPUSH` to `false`:

```
NOPUSH=false
```

If you update the maps, this setting allows the primary server to automatically push the maps to the worker servers.

6. Configure the NIS services:

a. Start the `ypserv` service and configure it to start after system reboots:

```
# systemctl start ypserv
# systemctl enable ypserv
```

The `ypserv` service runs on the NIS primary server and any worker servers.

b. If the server will act as the primary NIS server and there will be at least one worker NIS server, start the `ypxfrd` service and configure it to start after system reboots:

```
# systemctl start ypxfrd
# systemctl enable ypxfrd
```

The `ypxfrd` service speeds up the distribution of very large NIS maps from an NIS primary to any NIS worker servers. The service runs on the primary server only, and not on any worker servers. You do not need to start this service if there are no worker servers.

c. Start the `yppasswdd` service and configure it to start after system reboots:

```
# systemctl start yppasswdd
# systemctl enable yppasswdd
```

The `yppasswdd` service allows NIS users to change their password in the `shadow` map. The service runs on the NIS primary server and any worker servers.

7. Configure the firewall settings:

a. Edit `/etc/sysconfig/network` and add the following entries that define the ports on which the `ypserv` and `ypxfrd` services listen:

```
YPSERV_ARGS="-p 834"
YPXFRD_ARGS="-p 835"
```

These entries fix the ports on which `ypserv` and `ypxfrd` listen.

b. Allow incoming TCP connections to ports 111 and 834 and incoming UDP datagrams on ports 111 and 834:

```
# firewall-cmd --zone=zone --add-port=111/tcp --add-port=111/udp \
  --add-port=834/tcp --add-port=834/udp
# firewall-cmd --permanent --zone=zone --add-port=111/tcp --add-
port=111/udp \
  --add-port=834/tcp --add-port=834/udp
```

`portmapper` services requests on TCP port 111 and UDP port 111, and `ypserv` services requests on TCP port 834 and UDP port 834.

c. On the primary server, if you run the `ypxfrd` service to support transfers to worker servers, allow incoming TCP connections to port 835 and incoming UDP datagrams on port 835:

```
# firewall-cmd --zone=zone --add-port=835/tcp --add-port=835/udp
# firewall-cmd --permanent --zone=zone --add-port=835/tcp --add-
port=835/udp
```

d. Allow incoming UDP datagrams on the port on which `yppasswdd` listens:

```
# firewall-cmd --zone=zone \
  --add-port=`rpcinfo -p | gawk '/yppasswdd/ {print $4}'`/udp
```

> **Note:**
>
> Do not make this rule permanent. The UDP port number that `yppasswdd` uses is different every time that it restarts.

e. Edit `/etc/rc.local` and add the following line:

```
firewall-cmd --zone=zone \
  --add-port=`rpcinfo -p | gawk '/yppasswdd/ {print $4}'`/udp
```

This entry creates a firewall rule for the `yppasswdd` service when the system reboots. If you restart `yppasswdd`, you must correct the firewall rules manually unless you modify the `/etc/init.d/yppasswdd` script.

8. After you have started all the servers, create the NIS maps on the primary NIS server:

```
# /usr/lib64/yp/ypinit -m

At this point, we have to construct a list of the hosts which will run NIS
servers.  nisprimary is in the list of NIS server hosts.  Please continue to add
the names for the other hosts, one per line.  When you are done with the
list, type a <control D>."
      next host to add:  nisprimary
      next host to add:  nisworker1
      next host to add:  nisworker2
      next host to add:  ^D

The current list of NIS servers looks like this:

nisprimary nisworker1 nisworker2

Is this correct?  [y/n: y]  y
We need a few minutes to build the databases...
```

```
...
localhost has been set up as a NIS master server.

Now you can run ypinit -s nisprimary on all slave server.
```

Enter the host names of the NIS worker servers (if any), type `Ctrl-D` to finish, and enter `y` to confirm the list of NIS servers. The host names must be resolvable to IP addresses in DNS or by entries in `/etc/hosts`.

The `ypinit` utility builds the domain subdirectory in `/var/yp` and makes the NIS maps that are defined for the `all` target in `/var/yp/Makefile`. If you have configured `NOPUSH=false` in `/var/yp/Makefile` and the names of the worker servers in `/var/yp/ypservers`, the command also pushes the updated maps to the worker servers.

9. On each NIS worker server, run the following command to initialize the server:

   ```
   # /usr/lib64/yp/ypinit -s nisprimary
   ```

   In the previous command, *nisprimary* is the host name or IP address of the NIS primary server.

   For more information, see the `ypinit(8)` manual page.

> **Note:**
>
> If you update any of the source files on the primary NIS server that are used to build the maps, use the following command on the primary NIS server to remake the map and push the changes out to the worker servers:
>
> ```
> # make -C /var/yp
> ```

Adding User Accounts to NIS

> **Note:**
>
> This procedure assumes that:
>
> - NIS provides maps for `passwd`, `group`, and `auto.home`.
> - The NIS primary server uses NFS to export the users' home directories. See Mounting an NFS File System in Oracle Linux 7: Managing File Systems.

To create an account for an NIS user on the NIS primary server:

1. If the NIS primary server does not already export the base directory of the users' home directories, perform the following steps on the NIS primary server:

   a. Create the base directory for user directories, for example `/nethome`:

      ```
      # mkdir /nethome
      ```

   b. Add an entry such as the following to `/etc/exports`:

```
/nethome    *(rw,sync)
```

You might prefer to restrict which clients can mount the file system. For example, the following entry allows only clients in the 192.168.1.0/24 subnet to mount `/nethome`:

```
/nethome    192.168.1.0/24(rw,sync)
```

c. Use the following command to export the file system:

```
# exportfs -i -o ro,sync *:/nethome
```

d. If you have configured `/var/yp/Makfile` to make the `auto.home` map available to NIS clients, create the following entry in `/etc/auto.home`:

```
*    -rw,sync    nissvr:/nethome/&
```

In the previous example, *nissvr* is the host name or IP address of the NIS server.

2. Create the user account:

```
# useradd -b /nethome username
```

The command updates the `/etc/passwd` file and creates a home directory on the NIS server.

3. Depending on the type of authentication that you have configured:

- For Kerberos authentication, on the Kerberos server or a client system with `kadmin` access, use `kadmin` to create a principal for the user in the Kerberos domain, for example:

```
# kadmin -q "addprinc username@KRBDOMAIN"
```

The command prompts you to set a password for the user, and adds the principal to the Kerberos database.

- For NIS authentication, use the `passwd` command:

```
# passwd username
```

The command updates the `/etc/shadow` file with the hashed password.

4. Update the NIS maps:

```
# make -C /var/yp
```

This command makes the NIS maps that are defined for the `all` target in `/var/yp/Makefile`. If you have configured `NOPUSH=false` in `/var/yp/Makefile` and the names of the worker servers in `/var/yp/ypservers`, the command also pushes the updated maps to the worker servers.

> **Note:**
>
> A Kerberos-authenticated user can use either `kpasswd` or `passwd` to change his or her password. An NIS-authenticated user must use the `yppasswd` command rather than `passwd` to change his or her password.

Enabling NIS Authentication

To enable NIS authentication for an NIS client by using the Authentication Configuration GUI:

1. Install the `yp-tools` and `ypbind` packages:

   ```
   # yum install yp-tools ypbind
   ```

2. Run the Authentication Configuration GUI:

   ```
   # system-config-authentication
   ```

3. Select **NIS** as the user account database and enter values for:

   NIS Domain
   The name of the NIS domain. For example: `mynisdom`.

   NIS Server
   The domain name or IP address of the NIS server. For example, `nissvr.mydom.com`.

4. Select either **Kerberos password** or **NIS password** for authentication.

5. If you select Kerberos authentication, enter values for:

   Realm
   The name of the Kerberos realm.

   KDCs
   A comma-separated list of Key Distribution Center (KDC) servers that can issue Kerberos ticket granting tickets and service tickets.

   Admin Servers
   A comma-separated list of Kerberos administration servers.

   Alternatively, you can use DNS to configure these settings:

   - Select the **Use DNS to resolve hosts to realms** check box to look up the name of the realm defined as a `TXT` record in DNS, for example:

     ```
     _kerberos.mydom.com    IN TXT "MYDOM.COM"
     ```

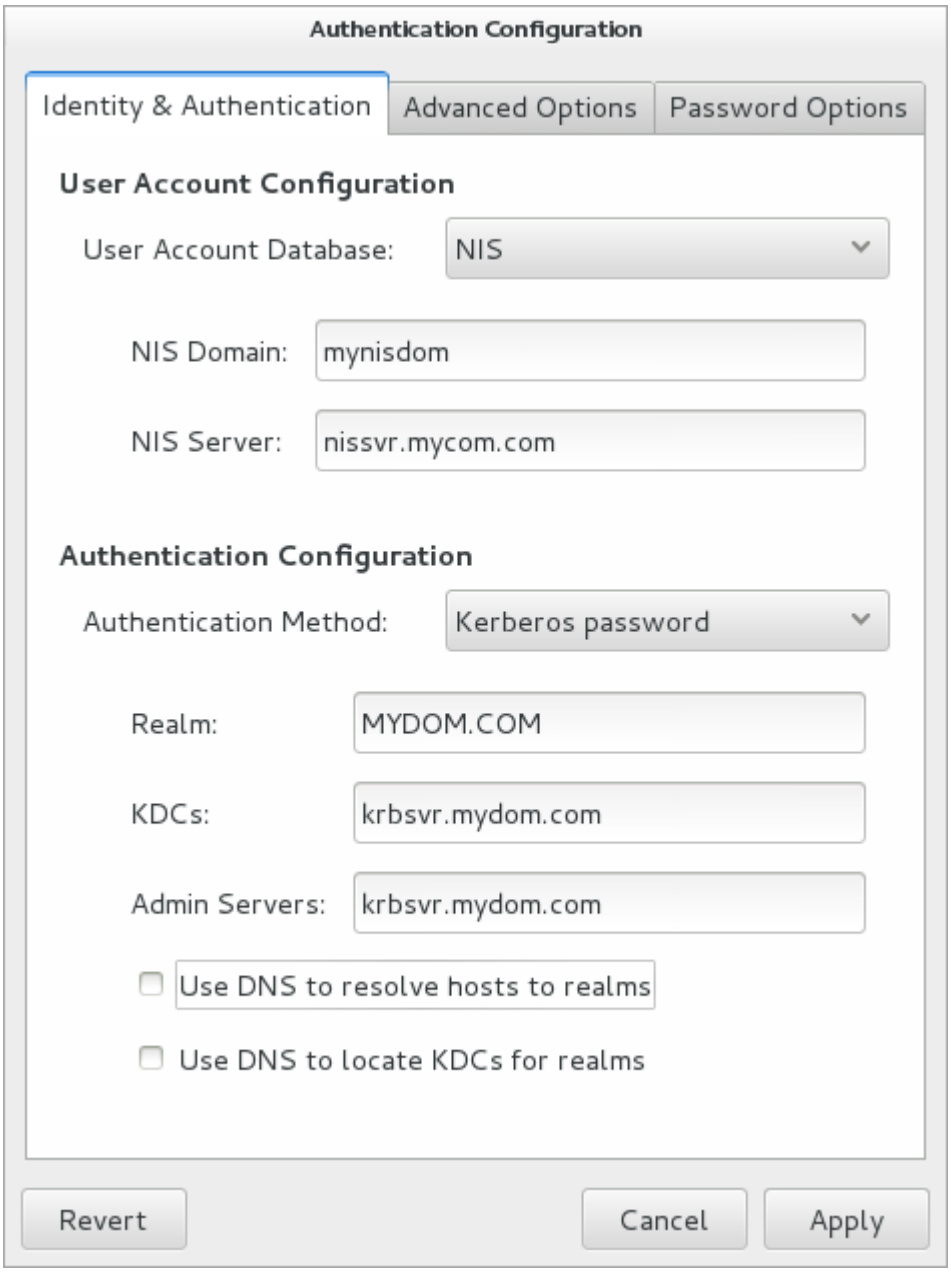
   - Select the **Use DNS to locate KDCs for realms** check box to look up the KDCs and administration servers defined as `SVR` records in DNS, for example:

     ```
     _kerberos._tcp.mydom.com      IN SVR 1  0 88  krbsvr.mydom.com
     _kerberos._udp.mydom.com      IN SVR 1  0 88  krbsvr.mydom.com
     _kpasswd._udp.mydom.com       IN SVR 1  0 464 krbsvr.mydom.com
     _kerberos-adm._tcp.mydom.com  IN SVR 1  0 749 krbsvr.mydom.com
     ```

6. Click **Apply** to save your changes.

> **NOT_SUPPORTED:**
>
> NIS authentication is deprecated as it has security issues, including a lack of protection of authentication data.

Figure 1-4 Authentication Configuration of NIS with Kerberos Authentication



You can also enable and configure NIS or Kerberos authentication by using the `authconfig` command.

For example, to use NIS authentication, specify the `--enablenis` option together with the NIS domain name and the host name or IP address of the primary server, as shown in the following example:.

```
# authconfig --enablenis --nisdomain mynisdom \
  --nisserver nissvr.mydom.com --update
```

The `--enablenis` option configures `/etc/nsswitch.conf` to enable the system to use NIS for information services. The `--nisdomain` and `--nisserver` settings are added to `/etc/yp.conf`.

For more information, see the `authconfig(8)`, `nsswitch.conf(5)`, and `yp.conf(5)` manual pages.

For information about using Kerberos authentication with NIS, see Enabling Kerberos Authentication.

Configuring an NIS Client to Use Automount Maps

If you have configured an automount map for `auto.home` in NIS, you can configure an NIS client to mount the users' home directories when they log in.

To configure an NIS client to automount users' home directories:

1. Install the `autofs` package:

   ```
   # yum install autofs
   ```

2. Create an `/etc/auto.master` file that contains the following entry:

   ```
   /nethome      /etc/auto.home
   ```

3. Verify that the `auto.home` map is available:

   ```
   # ypcat -k auto.home
   *    -rw,sync    nfssvr:/nethome/&
   ```

   In this example, the map is available. For details of how to make this map available, see Adding User Accounts to NIS.

4. If the `auto.home` map is available, edit the file `/etc/auto.home` to contain the following entry:

   ```
   +auto.home
   ```

   This entry causes the automounter to use the `auto.home` map.

5. Restart the `autofs` service, and configure the service to start following a system reboot:

   ```
   # systemctl restart autofs
   # systemctl enable autofs
   ```

   The `autofs` service creates the directory `/nethome`. When a user logs in, the automounter mounts his or her home directory under `/nethome`.

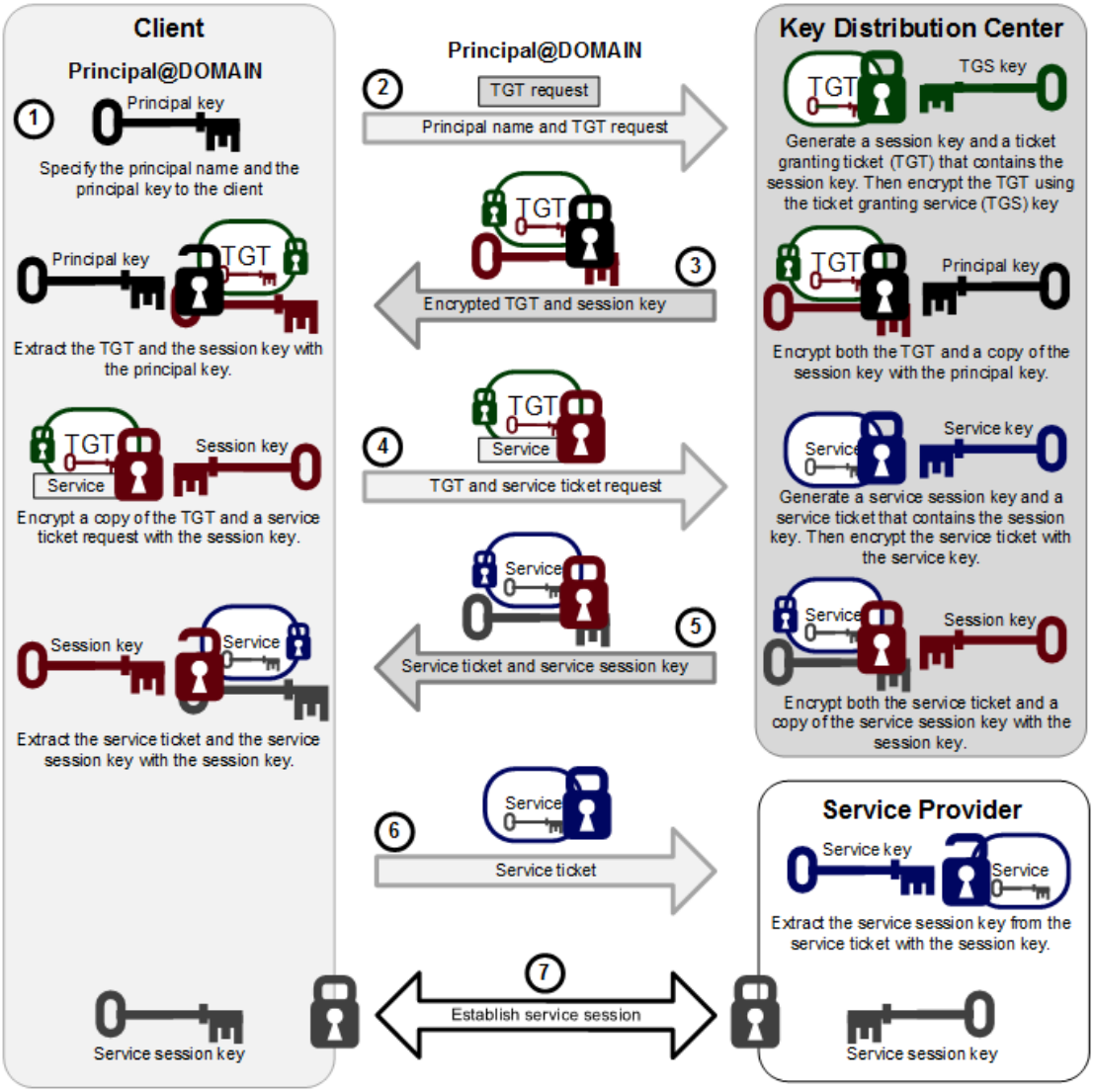
   If the owner and group for the user's files are unexpectedly listed as the anonymous user or group (`nobody` or `nogroup`) and `all_squash` has not been specified as a mount option, verify that the `Domain` setting in `/etc/idmapd.conf` on the NFS server is set to the DNS domain name. Restart the NFS services on the NFS server if you change this file.

About Kerberos Authentication

Both LDAP and NIS authentication optionally support Kerberos authentication. In the case of IPA, Kerberos is fully integrated. Kerberos provides a secure connection over

standard ports, and it also allows offline logins if you enable credential caching in SSSD.

Figure 1-5 Kerberos Authentication



The steps in the process are:

1. A principal name and key are specified to the client.
2. The client sends the principal name and a request for a TGT to the KDC.

   The KDC generates a session key and a TGT that contains a copy of the session key, and uses the Ticket Granting Service (TGS) key to encrypt the TGT. It then uses the principal's key to encrypt both the already encrypted TGT and another copy of the session key.
3. The KDC sends the encrypted combination of the session key and the encrypted TGT to the client.

   The client uses the principal's key to extract the session key and the encrypted TGT.

4. When the client want to use a service, usually to obtain access to a local or remote host system, it uses the session key to encrypt a copy of the encrypted TGT, the client's IP address, a time stamp, and a service ticket request, and it sends this item to the KDC.

   The KDC uses its copies of the session key and the TGS key to extract the TGT, IP address, and time stamp, which allow it to validate the client. Provided that both the client and its service request are valid, the KDC generates a service session key and a service ticket that contains the client's IP address, a time stamp, and a copy of the service session key, and it uses the service key to encrypt the service ticket. It then uses the session key to encrypt both the service ticket and another copy of the service session key.

   The service key is usually the host principal's key for the system on which the service provider runs.

5. The KDC sends the encrypted combination of the service session key and the encrypted service ticket to the client.

   The client uses its copy of the session key to extract the encrypted service ticket and the service session key.

6. The client sends the encrypted service ticket to the service provider together with the principal name and a time stamp encrypted with the service session key.

   The service provider uses the service key to extract the data in the service session ticket, including the service session key.

7. The service provider enables the service for the client, which is usually to grant access to its host system.

   If the client and service provider are hosted on different systems, they can each use their own copy of the service session key to secure network communication for the service session.

Note the following points about the authentication handshake:

- Steps 1 through 3 correspond to using the `kinit` command to obtain and cache a TGT.
- Steps 4 through 7 correspond to using a TGT to gain access to a Kerberos-aware service.
- Authentication relies on pre-shared keys.
- Keys are never sent in the clear over any communications channel between the client, the KDC, and the service provider.
- At the start of the authentication process, the client and the KDC share the principal's key, and the KDC and the service provider share the service key. Neither the principal nor the service provider know the TGS key.
- At the end of the process, both the client and the service provider share a service session key that they can use to secure the service session. The client does not know the service key and the service provider does not know the principal's key.
- The client can use the TGT to request access to other service providers for the lifetime of the ticket, which is usually one day. The session manager renews the TGT if it expires while the session is active.

Configuring a Kerberos Server

If you want to configure any client systems to use Kerberos authentication, it is recommended that you first configure a Kerberos server. You can then configure any clients that you require.

> **Note:**
>
> Keep any system that you configure as a Kerberos server very secure, and do not configure it to perform any other service function.

To configure a Kerberos server that can act as a key distribution center (KDC) and a Kerberos administration server:

1. Configure the server to use DNS and that both direct and reverse name lookups of the server's domain name and IP address work.

   For more information about configuring DNS, see Name Service Configuration in Oracle Linux 7: Setting Up Networking.

2. Configure the server to use network time synchronization mechanism such as the Network Time Protocol (NTP), Precision Time Protocol (PTP), or chrony. Kerberos requires that the system time on Kerberos servers and clients are synchronized as closely as possible. If the system times of the server and a client differ by more than 300 seconds (by default), authentication fails.

   For more information, see Network Time Configuration in Oracle Linux 7: Setting Up Networking

3. Install the `krb5-libs`, `krb5-server`, and `krb5-workstation` packages:

   ```
   # yum install krb5-libs krb5-server krb5-workstation
   ```

4. Edit `/etc/krb5.conf` and configure settings for the Kerberos realm, for example:

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

   [libdefaults]
    default_realm = MYDOM.COM
    dns_lookup_realm = false
    dns_lookup_kdc = false
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = true

   [realms]
    MYDOM.COM = {
     kdc = krbsvr.mydom.com
     admin_server = krbsvr.mydom.com
    }

   [domain_realm]
    .mydom.com = MYDOM.COM mydom.com = MYDOM.COM

   [appdefaults]
   ```

```
 pam = {
   debug = true
   validate = false
 }
```

In this example, the Kerberos realm is `MYDOM.COM` in the DNS domain mydom.com and `krbsvr.mydom.com` (the local system) acts as both a KDC and an administration server. The `[appdefaults]` section configures options for the `pam_krb5.so` module.

For more information, see the `krb5.conf(5)` and `pam_krb5(5)` manual pages.

5. Edit `/var/kerberos/krb5kdc/kdc.conf` and configure settings for the key distribution center, for example:

```
kdcdefaults]
 kdc_ports = 88
 kdc_tcp_ports = 88

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

For more information, see the `kdc.conf(5)` manual page.

6. Create the Kerberos database and store the database password in a stash file:

```
# /usr/sbin/kdb5_util create -s
```

7. Edit `/var/kerberos/krb5kdc/kadm5.acl` and define the principals who have administrative access to the Kerberos database, for example:

```
*/admin@EXAMPLE.COM     *
```

In this example, any principal who has an instance of `admin`, such as `alice/admin@MYDOM.COM`, has full administrative control of the Kerberos database for the `MYDOM.COM` domain. Ordinary users in the database usually have an empty instance, for example `bob@MYDOM.COM`. These users have no administrative control other than being able to change their password, which is stored in the database.

8. Create a principal for each user who should have the `admin` instance, for example:

```
# kadmin.local -q "addprinc alice/admin"
```

9. Cache the keys that `kadmind` uses to decrypt administration Kerberos tickets in `/etc/kadm5.keytab`:

```
# kadmin.local -q "ktadd -k /etc/kadm5.keytab kadmin/admin"
# kadmin.local -q "ktadd -k /etc/kadm5.keytab kadmin/changepw"
```

10. Start the KDC and administration services and configure them to start following system reboots:

```
# systemctl start krb5kdc
# systemctl start kadmin
# systemctl enable krb5kdc
# systemctl enable kadmin
```

11. Add principals for users and the Kerberos server and cache the key for the server's host principal in `/etc/kadm5.keytab` by using either `kadmin.local` or `kadmin`, for example:

    ```
    # kadmin.local -q "addprinc bob"
    # kadmin.local -q "addprinc -randkey host/krbsvr.mydom.com"
    # kadmin.local -q "ktadd -k /etc/kadm5.keytab host/krbsvr.mydom.com"
    ```

12. Allow incoming TCP connections to ports 88, 464, and 749 and UDP datagrams on UDP port 88, 464, and 749:

    ```
    # firewall-cmd --zone=zone --add-port=88/tcp --add-port=88/udp \
      --add-port=464/tcp --add-port=464/udp \
      --add-port=749/tcp --add-port=749/udp
    # firewall-cmd --permanent --zone=zone --add-port=88/tcp --add-port=88/udp
    \
      --add-port=464/tcp --add-port=464/udp \
      --add-port=749/tcp --add-port=749/udp
    ```

    `krb5kdc` services requests on TCP port 88 and UDP port 88, and `kadmind` services requests on TCP ports 464 and 749 and UDP ports 464 and 749.

    In addition, you might need to allow TCP and UDP access on different ports for other applications.

For more information, see the `kadmin(1)` manual page.

Configuring a Kerberos Client

Setting up a Kerberos client on a system allows it to use Kerberos to authenticate users who are defined in NIS or LDAP, and to provide secure remote access by using commands such as `ssh` with GSS-API enabled or the Kerberos implementation of `telnet`.

To set up a system as a Kerberos client:

1. Configure the client system to use DNS and that both direct and reverse name lookups of the domain name and IP address for both the client and the Kerberos server work.

   For more information about configuring DNS, see Name Service Configuration in Oracle Linux 7: Setting Up Networking

2. Configure the system to use a network time synchronization protocol such as the Network Time Protocol (NTP). Kerberos requires that the system time on Kerberos servers and clients are synchronized as closely as possible. If the system times of the server and a client differ by more than 300 seconds (by default), authentication fails.

   To configure the server as an NTP client:

   a. Install the `ntp` package:

      ```
      # yum install ntp
      ```

   b. Edit `/etc/ntp.conf` and configure the settings as required. See the `ntp.conf(5)` manual page and http://www.ntp.org.

   c. Start the `ntpd` service and configure it to start following system reboots.

```
# systemctl start ntpd
# systemctl enable ntpd
```

3. Install the `krb5-libs` and `krb5-workstation` packages:

   ```
   # yum install krb5-libs krb5-workstation
   ```

4. Copy the `/etc/krb5.conf` file to the system from the Kerberos server.

5. Use the Authentication Configuration GUI or `authconfig` to set up the system to use Kerberos with either NIS or LDAP, for example:

   ```
   # authconfig --enablenis --enablekrb5 --krb5realm=MYDOM.COM \
     --krb5adminserver=krbsvr.mydom.com --krb5kdc=krbsvr.mydom.com \
     --update
   ```

   See Enabling Kerberos Authentication.

6. On the Kerberos KDC, use either `kadmin` or `kadmin.local` to add a host principal for the client, for example:

   ```
   # kadmin.local -q "addprinc -randkey host/client.mydom.com"
   ```

7. On the client system, use `kadmin` to cache the key for its host principal in `/etc/kadm5.keytab`, for example:

   ```
   # kadmin -q "ktadd -k /etc/kadm5.keytab host/client.mydom.com"
   ```

8. To use `ssh` and related OpenSSH commands to connect from Kerberos client system to another Kerberos client system:

   a. On the remote Kerberos client system, verify that `GSSAPIAuthentication` is enabled in `/etc/ssh/sshd_config`:

      ```
      GSSAPIAuthentication yes
      ```

   b. On the local Kerberos client system, enable `GSSAPIAuthentication` and `GSSAPIDelegateCredentials` in the user's `.ssh/config` file:

      ```
      GSSAPIAuthentication yes
      GSSAPIDelegateCredentials yes
      ```

      Alternatively, the user can specify the `-K` option to `ssh`.

   c. Test that the principal can obtain a ticket and connect to the remote system, for example:

      ```
      $ kinit principal_name@MYDOM.COM

      $ ssh username@remote.mydom.com
      ```

   To allow use of the Kerberos versions of `rlogin`, `rsh`, and `telnet`, which are provided in the `krb5-appl-clients` package, you must enable the corresponding services on the remote client.

For more information, see the `kadmin(1)` manual page.

Enabling Kerberos Authentication

To be able to use Kerberos authentication with an LDAP or NIS client, use `yum` to install the `krb5-libs` and `krb5-workstation` packages.

If you use the Authentication Configuration GUI (`system-config-authentication`) and select LDAP or NIS as the user account database, select Kerberos password as the authentication method and enter values for:

Realm
The name of the Kerberos realm.

KDCs
A comma-separated list of Key Distribution Center (KDC) servers that can issue Kerberos ticket granting tickets and service tickets.

Admin Servers
A comma-separated list of Kerberos administration servers.

Alternatively, you can use DNS to configure these settings:

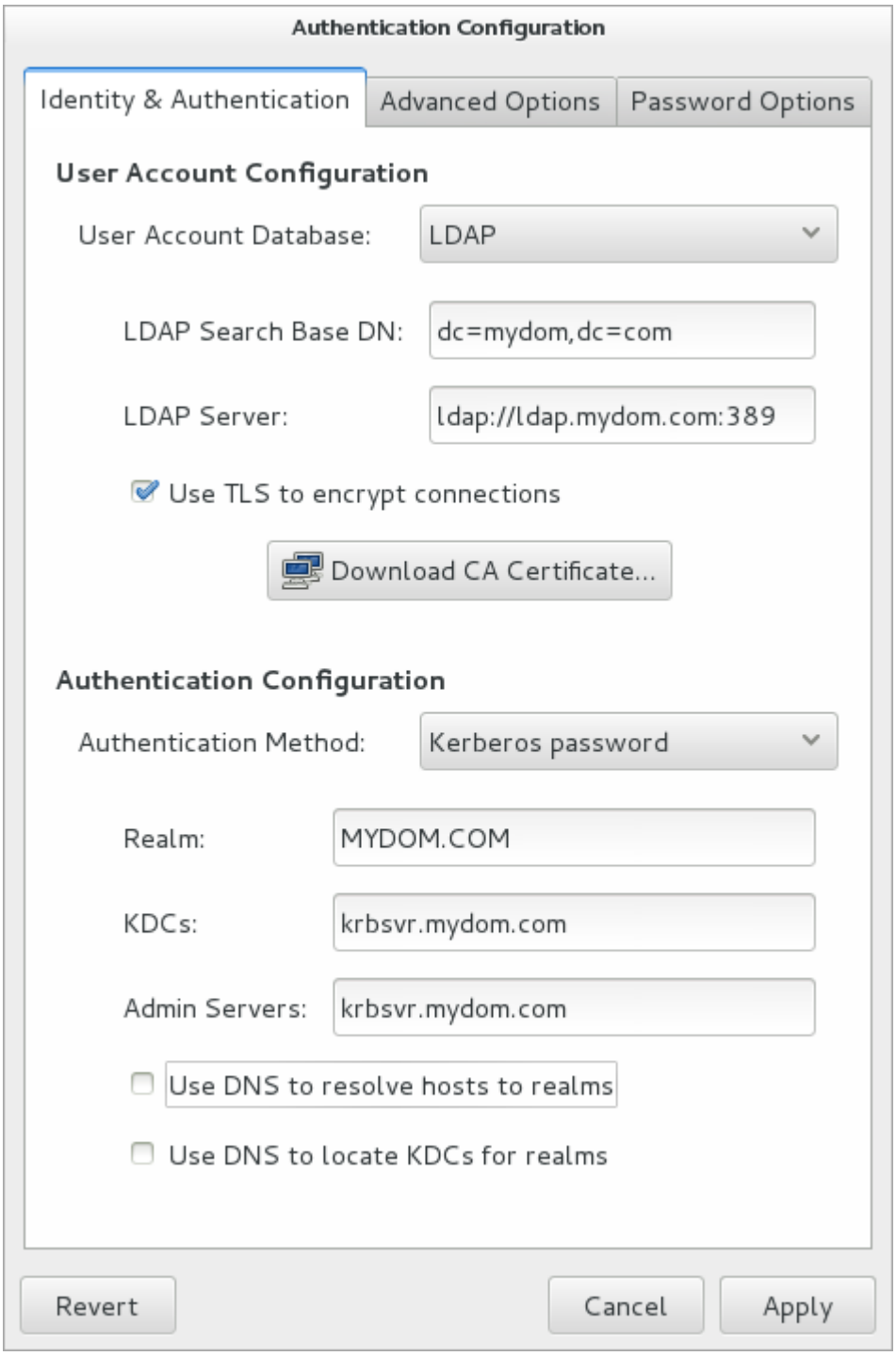
- Select the **Use DNS to resolve hosts to realms** check box to look up the name of the realm defined as a `TXT` record in DNS, for example:

```
_kerberos.mydom.com    IN TXT "MYDOM.COM"
```

- Select the **Use DNS to locate KDCs for realms** check box to look up the KDCs and administration servers defined as `SVR` records in DNS, for example:

```
_kerberos._tcp.mydom.com      IN SVR 1  0 88  krbsvr.mydom.com
_kerberos._udp.mydom.com      IN SVR 1  0 88  krbsvr.mydom.com
_kpasswd._udp.mydom.com       IN SVR 1  0 464 krbsvr.mydom.com
_kerberos-adm._tcp.mydom.com  IN SVR 1  0 749 krbsvr.mydom.com
```

Figure 1-6 Authentication Configuration of LDAP with Kerberos Authentication



Alternatively, you can use the `authconfig` command to configure Kerberos authentication with LDAP, for example:

```
# authconfig --enableldap \
  --ldapbasedn="dc=mydom,dc=com" --ldapserver=ldap://ldap.mydom.com:389 \
  [--enableldaptls --ldaploadcacert=https://ca-server.mydom.com/
```

```
CAcert.pem] \
  --enablekrb5  \
  --krb5realm=MYDOM.COM | --enablekrb5realmdns \
  --krb5kdc=krbsvr.mydom.com --krb5adminserver=krbsvr.mydom.com | --
enablekrb5kdcdns \
  --update
```

or with NIS:

```
# authconfig --enablenis \
  --enablekrb5 \
  --krb5realm=MYDOM.COM | --enablekrb5realmdns \
  --krb5kdc=krbsvr.mydom.com --krb5adminserver=krbsvr.mydom.com | --
enablekrb5kdcdns \
  --update
```

The `--enablekrb5` option enables Kerberos authentication by modifying the PAM configuration files in `/etc/pam.d` to use the `pam_krb5.so` module. The `--enableldap` and `--enablenis` options configure `/etc/nsswitch.conf` to enable the system to use LDAP or NIS for information services.

For more information, see the `authconfig(8)`, `nsswitch.conf(5)`, and `pam_krb5(5)` manual pages.

About Pluggable Authentication Modules

The Pluggable Authentication Modules (PAM) feature is an authentication mechanism that allows you to configure how applications use authentication to verify the identity of a user. The PAM configuration files, which are located in the `/etc/pam.d` directory, describe the authentication procedure for an application. The name of each configuration file is the same as, or is similar to, the name of the application for which the module provides authentication. For example, the configuration files for `passwd` and `sudo` are named `passwd` and `sudo`.

Each PAM configuration file contains a list (*stack*) of calls to authentication modules. For example, the following listing shows the default content of the `login` configuration file:

```
#%PAM-1.0
auth [user_unknown=ignore success=ok ignore=ignore default=bad] pam_securetty.so
auth       include      system-auth
auth       include      postlogin
account    required     pam_nologin.so
account    include      system-auth
password   include      system-auth
# pam_selinux.so close should be the first session rule
session    required     pam_selinux.so close
session    required     pam_loginuid.so
session    optional     pam_console.so
# pam_selinux.so open should only be followed by sessions to be executed in the user
context
session    required     pam_selinux.so open
session    required     pam_namespace.so
session    optional     pam_keyinit.so force revoke
session    include      system-auth
session    include      postlogin
-session   optional     pam_ck_connector.so
```

Comments in the file start with a `#` character. The remaining lines each define an operation type, a control flag, the name of a module such as `pam_rootok.so` or the name of an included configuration file such as `system-auth`, and any arguments to the module. PAM provides authentication modules as shared libraries in `/usr/lib64/security`.

For a particular operation type, PAM reads the stack from top to bottom and calls the modules listed in the configuration file. Each module generates a success or failure result when called.

The following operation types are defined for use:

`auth`
The module tests whether a user is authenticated or authorized to use a service or application. For example, the module might request and verify a password. Such modules can also set credentials, such as a group membership or a Kerberos ticket.

`account`
The module tests whether an authenticated user is allowed access to a service or application. For example, the module might check if a user account has expired or if a user is allowed to use a service at a given time.

`password`
The module handles updates to an authentication token.

`session`
The module configures and manages user sessions, performing tasks such as mounting or unmounting a user's home directory.

If the operation type is preceded with a dash (`-`), PAM does not add an create a system log entry if the module is missing.

With the exception of `include`, the control flags tell PAM what to do with the result of running a module. The following control flags are defined for use:

`optional`
The module is required for authentication if it is the only module listed for a service.

`required`
The module must succeed for access to be granted. PAM continues to execute the remaining modules in the stack whether the module succeeds or fails. PAM does not immediately inform the user of the failure.

`requisite`
The module must succeed for access to be granted. If the module succeeds, PAM continues to execute the remaining modules in the stack. However, if the module fails, PAM notifies the user immediately and does not continue to execute the remaining modules in the stack.

`sufficient`
If the module succeeds, PAM does not process any remaining modules of the same operation type. If the module fails, PAM processes the remaining modules of the same operation type to determine overall success or failure.

The control flag field can also define one or more rules that specify the action that PAM should take depending on the value that a module returns. Each rule takes the form *`value=action`*, and the rules are enclosed in square brackets, for example:

```
[user_unknown=ignore success=ok ignore=ignore default=bad]
```

If the result returned by a module matches a value, PAM uses the corresponding action, or, if there is no match, it uses the default action.

The `include` flag specifies that PAM must also consult the PAM configuration file specified as the argument.

Most authentication modules and PAM configuration files have their own manual pages. In addition, the `/usr/share/doc/pam-`*`version`* directory contains the PAM System Administrator's Guide (`html/Linux-PAM_SAG.html` or `Linux-PAM_SAG.txt`) and a copy of the PAM standard (`rfc86.0.txt`).

For more information, see the `pam(8)` manual page. In addition, each PAM module has its own manual page, for example `pam_unix(8)`, `postlogin(5)`, and `system-auth(5)`.

About the System Security Services Daemon

The System Security Services Daemon (SSSD) feature provides access on a client system to remote identity and authentication providers. The SSSD acts as an intermediary between local clients and any back-end provider that you configure.

The benefits of configuring SSSD include:

- Reduced system load

  Clients do not have to contact the identification or authentication servers directly.
- Offline authentication

  You can configure SSSD to maintain a cache of user identities and credentials.
- Single sign-on access

  If you configure SSSD to store network credentials, users need only authenticate once per session with the local system to access network resources.

For more information, see the `authconfig(8)`, `pam_sss(8)`, `sssd(8)`, and `sssd.conf(5)` manual pages.

Configuring an SSSD Server

To configure an SSSD server:

1. Install the `sssd` and `sssd-client` packages:

   ```
   # yum install sssd sssd-client
   ```
2. Edit the `/etc/sssd/sssd.conf` configuration file and configure the sections to support the required services, for example:

   ```
   [sssd]
   config_file_version = 2
   domains = LDAP
   services = nss, pam

   [domain/LDAP]
   id_provider = ldap
   ldap_uri = ldap://ldap.mydom.com
   ldap_search_base = dc=mydom,dc=com
   ```

```
auth_provider = krb5
krb5_server = krbsvr.mydom.com
krb5_realm = MYDOM.COM
cache_credentials = true

min_id = 5000
max_id = 25000
enumerate = false

[nss]
filter_groups = root
filter_users = root
reconnection_retries = 3
entry_cache_timeout = 300

[pam]
reconnection_retries = 3
offline_credentials_expiration = 2
offline_failed_login_attempts = 3
offline_failed_login_delay = 5
```

The `[sssd]` section contains configuration settings for SSSD monitor options, domains, and services. The SSSD monitor service manages the services that SSSD provides.

The `services` entry defines the supported services, which should include `nss` for the Name Service Switch and `pam` for Pluggable Authentication Modules.

The `domains` entry specifies the name of the sections that define authentication domains.

The `[domain/LDAP]` section defines a domain for an LDAP identity provider that uses Kerberos authentication. Each domain defines where user information is stored, the authentication method, and any configuration options. SSSD can work with LDAP identity providers such as OpenLDAP, Red Hat Directory Server, IPA, and Microsoft Active Directory, and it can use either native LDAP or Kerberos authentication.

The `id_provider` entry specifies the type of provider (in this example, LDAP). `ldap_uri` specifies a comma-separated list of the Universal Resource Identifiers (URIs) of the LDAP servers, in order of preference, to which SSSD can connect. `ldap_search_base` specifies the base distinguished name (`dn`) that SSSD should use when performing LDAP user operations on a relative distinguished name (RDN) such as a common name (`cn`).

The `auth_provider` entry specifies the authentication provider (in this example, Kerberos). `krb5_server` specifies a comma-separated list of Kerberos servers, in order of preference, to which SSSD can connect. `krb5_realm` specifies the Kerberos realm. `cache_credentials` specifies if SSSD caches user credentials such as tickets, session keys, and other identifying information to support offline authentication and single sign-on.

> **Note:**
>
> To allow SSSD to use Kerberos authentication with an LDAP server, you must configure the LDAP server to use both Simple Authentication and Security Layer (SASL) and the Generic Security Services API (GSSAPI). For more information about configuring SASL and GSSAPI for OpenLDAP, see https://www.openldap.org/doc/admin24/sasl.html.

The `min_id` and `max_id` entries specify upper and lower limits on the values of user and group IDs. `enumerate` specifies whether SSSD caches the complete list of users and groups that are available on the provider. The recommended setting is `False` unless a domain contains relatively few users or groups.

The `[nss]` section configures the Name Service Switch (NSS) module that integrates the SSS database with NSS. The `filter_users` and `filter_groups` entries prevent NSS retrieving information about the specified users and groups being retrieved from SSS. `reconnection_retries` specifies the number of times that SSSD should attempt to reconnect if a data provider crashes. `enum_cache_timeout` specifies the number of seconds for which SSSD caches user information requests.

The `[pam]` section configures the PAM module that integrates SSS with PAM. The `offline_credentials_expiration` entry specifies the number of days for which to allow cached logins if the authentication provider is offline. `offline_failed_login_attempts` specifies how many failed login attempts are allowed if the authentication provider is offline. `offline_failed_login_delay` specifies how many minutes after `offline_failed_login_attempts` failed login attempts that a new login attempt is permitted.

3. Change the mode of `/etc/sssd/sssd.conf` to 0600:

```
# chmod 0600 /etc/sssd/sssd.conf
```

4. Enable the SSSD service:

```
# authconfig --update --enablesssd --enablesssdauth
```

> **Note:**
>
> If you edit `/etc/sssd/sssd.conf`, use this command to update the service.

The `--enablesssd` option updates `/etc/nsswitch.conf` to support SSS.

The `--enablesssdauth` option updates `/etc/pam.d/system-auth` to include the required `pam_sss.so` entries to support SSSD.

About Winbind Authentication

Winbind is a client-side service that resolves user and group information on a Windows server, and allows Oracle Linux to understand Windows users and groups. To be able to configure Winbind authentication, use `yum` to install the `samba-winbind` package. This package includes the `winbindd` daemon that implements the `winbind` service.

Enabling Winbind Authentication

If you use the Authentication Configuration GUI and select Winbind as the user account database, you are prompted for the information that is required to connect to a Microsoft workgroup, Active Directory, or Windows NT domain controller. Enter the name of the Winbind domain and select the security model for the Samba server:

`ads`
In the Activity Directory Server (ADS) security model, Samba acts as a domain member in an ADS realm, and clients use Kerberos tickets for Active Directory authentication. You must configure Kerberos and join the server to the domain, which creates a machine account for your server on the domain controller.

`domain`
In the domain security model, the local Samba server has a machine account (a domain security trust account) and Samba authenticates user names and passwords with a domain controller in a domain that implements Windows NT4 security.

> **NOT_SUPPORTED:**
>
> If the local machine acts as a Primary or Backup Domain Controller, do not use the domain security model. Use the user security model instead.

`server`
In the server security model, the local Samba server authenticates user names and passwords with another server, such as a Windows NT server.

> **NOT_SUPPORTED:**
>
> The server security model is deprecated as it has numerous security issues.

`user`
In the user security model, a client must log in with a valid user name and password. This model supports encrypted passwords. If the server successfully validates the client's user name and password, the client can mount multiple shares without being required to specify a password.

Depending on the security model that you choose, you might also need to specify the following information:

- The name of the ADS realm that the Samba server is to join (ADS security model only).
- The names of the domain controllers. If there are several domain controllers, separate the names with spaces.
- The login template shell to use for the Windows NT user account (ADS and domain security models only).
- Whether to allow user authentication using information that has been cached by the System Security Services Daemon (SSSD) if the domain controllers are offline.

Your selection updates the security directive in the `[global]` section of the `/etc/samba/smb.conf` configuration file.

If you have initialized Kerberos, you can click **Join Domain** to create a machine account on the Active Directory server and grant permission for the Samba domain member server to join the domain.

You can also use the `authconfig` command to configure Winbind authentication. To use the user-level security models, specify the name of the domain or workgroup and the host names of the domain controllers. for example:

```
# authconfig --enablewinbind --enablewinbindauth --smbsecurity user \
  [--enablewinbindoffline] --smbservers="ad1.mydomain.com ad2.mydomain.com" \
  --smbworkgroup=MYDOMAIN --update
```

To allow user authentication using information that has been cached by the System Security Services Daemon (SSSD) if the domain controllers are offline, specify the `--enablewinbindoffline` option.

For the domain security model, additionally specify the template shell, for example:

```
# authconfig --enablewinbind --enablewinbindauth --smbsecurity domain \
  [--enablewinbindoffline] --smbservers="ad1.mydomain.com ad2.mydomain.com" \
  --smbworkgroup=MYDOMAIN --update --winbindtemplateshell=/bin/bash --update
```

For the ADS security model, additionally specify the ADS realm and template shell, for example:

```
# authconfig --enablewinbind --enablewinbindauth --smbsecurity ads \
  [--enablewinbindoffline] --smbservers="ad1.mydomain.com ad2.mydomain.com" \
  --smbworkgroup=MYDOMAIN --update --smbrealm MYDOMAIN.COM \
  --winbindtemplateshell=/bin/bash --update
```

For more information, see the `authconfig(8)` manual page.

2

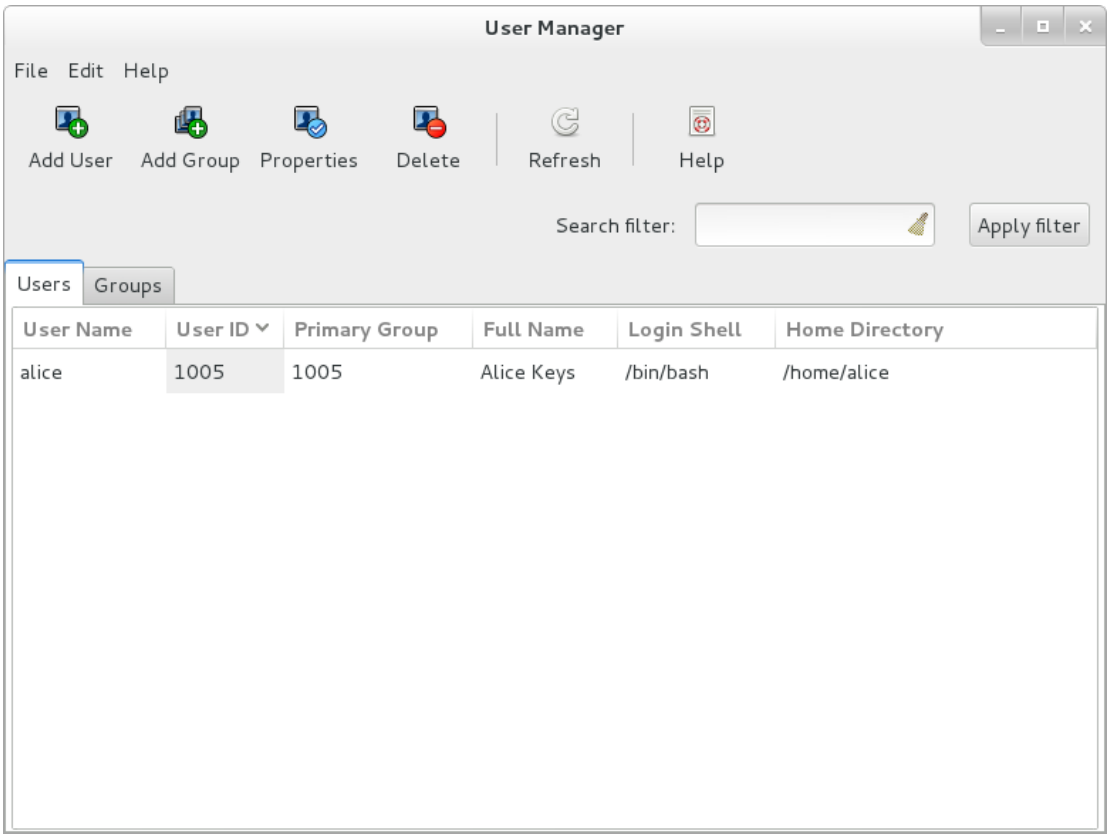
Local Account Configuration

This chapter describes how to configure and manage local user and group accounts.

About User and Group Configuration

You can use the User Manager GUI (`system-config-users`) to add or delete users and groups and to modify settings such as passwords, home directories, login shells, and group membership. Alternatively, you can use commands such as `useradd` and `groupadd`.

Figure 2-1 User Manager



In an enterprise environment that might have hundreds of servers and thousands of users, user and group account information is more likely to be held in a central repository rather than in files on individual servers. You can configure user and group information on a central server and retrieve this information by using services such as Lightweight Directory Access Protocol (LDAP) or Network Information Service (NIS). You can also create users' home directories on a central server and automatically mount, or access, these remote file systems when a user logs in to a system.

Changing Default Settings for User Accounts

To display the default settings for an account use the following command:

```
# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=yes
```

`INACTIVE` specifies after how many days the system locks an account if a user's password expires. If set to 0, the system locks the account immediately. If set to -1, the system does not lock the account.

`SKEL` defines a template directory, whose contents are copied to a newly created user's home directory. The contents of this directory should match the default shell defined by `SHELL`.

You can specify options to `useradd -D` to change the default settings for user accounts. For example, to change the defaults for `INACTIVE`, `HOME` and `SHELL`:

```
# useradd -D -f 3 -b /home2 -s /bin/sh
```

> **Note:**
>
> If you change the default login shell, you would usually also create a new `SKEL` template directory with contents that are appropriate to the new shell.
>
> If you specify `/sbin/nologin` for a user's `SHELL`, that user cannot log into the system directly but processes can run with that user's ID. This setting is typically used for services that run as users other than `root`.

The default settings are stored in the `/etc/default/useradd` file.

For more information, see Configuring Password Aging and the `useradd(8)` manual page.

Creating User Accounts

To create a user account by using the `useradd` command:

1. Enter the following command to create a user account:

   ```
   # useradd [options] username
   ```

   You can specify options to change the account's settings from the default ones.

   By default, if you specify a user name argument but do not specify any options, `useradd` creates a locked user account using the next available UID and assigns

a user private group (UPG) rather than the value defined for `GROUP` as the user's group.

2. Assign a password to the account to unlock it:

   ```
   # passwd username
   ```

   The command prompts you to enter a password for the account.

   If you want to change the password non-interactively (for example, from a script), use the `chpasswd` command instead:

   ```
   echo "username:password" | chpasswd
   ```

Alternatively, you can use the `newusers` command to create a number of user accounts at the same time.

For more information, see the `chpasswd(8)`, `newusers(8)`, `passwd(1)`, and `useradd(8)` manual pages.

About umask and the setgid and Restricted Deletion Bits

Users whose primary group is not a UPG have a `umask` of 0022 set by `/etc/profile` or `/etc/bashrc`, which prevents other users, including other members of the primary group, from modifying any file that the user owns.

A user whose primary group is a UPG has a `umask` of 0002. It is assumed that no other user has the same group.

To grant users in the same group write access to files within the same directory, change the group ownership on the directory to the group, and set the `setgid` bit on the directory:

```
# chgrp groupname directory
# chmod g+s directory
```

Files created in such a directory have their group set to that of the directory rather than the primary group of the user who creates the file.

The restricted deletion bit prevents unprivileged users from removing or renaming a file in the directory unless they own either the file or the directory.

To set the restricted deletion bit on a directory:

```
# chmod a+t directory
```

For more information, see the `chmod(1)` manual page.

Locking an Account

To lock a user's account, enter:

```
# passwd -l username
```

To unlock the account:

```
# passwd -u username
```

For more information, see the `passwd(1)` manual page.

Modifying or Deleting User Accounts

To modify a user account, use the `usermod` command:

```
# usermod [options] username
```

For example, to add a user to a supplementary group (other than his or her login group):

```
# usermod -aG groupname username
```

You can use the `groups` command to display the groups to which a user belongs, for example:

```
# groups root
root : root bin daemon sys adm disk wheel
```

To delete a user's account, use the `userdel` command:

```
# userdel username
```

For more information, see the `groups(1)`, `userdel(8)` and `usermod(8)` manual pages.

Creating Groups

To create a group by using the `groupadd` command:

```
# groupadd [options] groupname
```

Typically, you might want to use the `-g` option to specify the group ID (GID). For example:

```
# groupadd -g 1000 devgrp
```

For more information, see the `groupadd(8)` manual page.

Modifying or Deleting Groups

To modify a group, use the `groupmod` command:

```
# groupmod [options] username
```

To delete a user's account, use the `groupdel` command:

```
# groupdel username
```

For more information, see the `groupdel(8)` and `groupmod(8)` manual pages.

Configuring Password Aging

To specify how users' passwords are aged, edit the following settings in the `/etc/login.defs` file:

Setting	Description
PASS_MAX_DAYS	Maximum number of days for which a password can be used before it must be changed. The default value is 99,999 days.
PASS_MIN_DAYS	Minimum number of days that is allowed between password changes. The default value is 0 days.
PASS_WARN_AGE	Number of days warning that is given before a password expires. The default value is 7 days.

For more information, see the `login.defs(5)` manual page.

To change how long a user's account can be inactive before it is locked, use the `usermod` command. For example, to set the inactivity period to 30 days:

```
# usermod -f 30 username
```

To change the default inactivity period for new user accounts, use the `useradd` command:

```
# useradd -D -f 30
```

A value of -1 specifies that user accounts are not locked due to inactivity.

For more information, see the `useradd(8)` and `usermod(8)` manual pages.

Granting sudo Access to Users

By default, an Oracle Linux system is configured so that you cannot log in directly as `root`. You must log in as a named user before using either `su` or `sudo` to perform tasks as `root`. This configuration allows system accounting to trace the original login name of any user who performs a privileged administrative action. If you want to grant certain users authority to be able to perform specific administrative tasks via `sudo`, use the `visudo` command to modify the `/etc/sudoers` file.

For example, the following entry grants the user `erin` the same privileges as `root` when using `sudo`, but defines a limited set of privileges to `frank` so that he can run commands such as `systemctl`, `rpm`, and `yum`:

```
erin            ALL=(ALL)       ALL
frank           ALL= SERVICES, SOFTWARE
```

For more information, see the `su(1)`, `sudo(8)`, `sudoers(5)`, and `visudo(8)` manual pages.