Red Hat Enterprise Linux 8

Using authselect on a Red Hat Enterprise Linux host

Understanding, selecting, modifying, and creating authselect profiles
Red Hat Enterprise Linux 8 Using authselect on a Red Hat Enterprise Linux host

Understanding, selecting, modifying, and creating authselect profiles
Legal Notice

Copyright © 2020 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution-Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at http://creativecommons.org/licenses/by-sa/3.0/. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux® is the registered trademark of Linus Torvalds in the United States and other countries.

Java® is a registered trademark of Oracle and/or its affiliates.

XFS® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation’s permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This documentation collection provides instructions on how to use authselect on a Red Hat Enterprise Linux 8 host.
Table of Contents

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION .................................................. 3

CHAPTER 1. CONFIGURING USER AUTHENTICATION USING AUTHSELECT ............................ 4
  1.1. WHAT IS AUTHSELECT USED FOR ............................................................. 4
  1.2. CHOOSING AN AUTHSELECT PROFILE ..................................................... 5
  1.3. MODIFYING A READY-MADE AUTHSELECT PROFILE ................................... 6
  1.4. CREATING AND DEPLOYING YOUR OWN AUTHSELECT PROFILE .................. 7
      Example ............................................................................................................. 8
  1.5. CONVERTING YOUR SCRIPTS FROM AUTHCONFIG TO AUTHSELECT ............... 8

CHAPTER 2. CONFIGURING SMART CARDS USING AUTHSELECT ........................................ 10
  2.1. CERTIFICATES ELIGIBLE FOR SMART CARDS .............................................. 10
  2.2. ENABLING USER PASSWORD AUTHENTICATION TO CONFIGURE SMART CARD AUTHENTICATION .............................................................. 10
  2.3. CONFIGURING AUTHSELECT TO ENFORCE SMART CARD AUTHENTICATION 11
  2.4. CONFIGURING SMART CARD AUTHENTICATION WITH LOCK ON REMOVAL .... 11

CHAPTER 3. CONFIGURING AND IMPORTING LOCAL CERTIFICATES TO A SMART CARD ......... 13
  3.1. CREATING LOCAL CERTIFICATES ............................................................... 13
  3.2. COPYING CERTIFICATES TO THE SSSD DIRECTORY ................................... 16
  3.3. INSTALLING GNUTLS-UTILS ..................................................................... 17
  3.4. STORING A CERTIFICATE ON THE SMART CARD ........................................ 17
  3.5. CONFIGURING SSH ACCESS USING SMART CARD AUTHENTICATION .......... 19
PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Please let us know how we could make it better. To do so:

- For simple comments on specific passages:
  1. Make sure you are viewing the documentation in the Multi-page HTML format. In addition, ensure you see the Feedback button in the upper right corner of the document.
  2. Use your mouse cursor to highlight the part of text that you want to comment on.
  3. Click the Add Feedback pop-up that appears below the highlighted text.
  4. Follow the displayed instructions.

- For submitting more complex feedback, create a Bugzilla ticket:
  1. Go to the Bugzilla website.
  2. As the Component, use Documentation.
  3. Fill in the Description field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.
  4. Click Submit Bug.
CHAPTER 1. CONFIGURING USER AUTHENTICATION USING AUTHSELECT

1.1. WHAT IS AUTHSELECT USED FOR

Authselect is a utility that simplifies the configuration of user authentication on a Red Hat Enterprise Linux host. Authselect offers two ready-made profiles that can be universally used with all modern identity management systems:

- the sssd profile
- the winbind profile

For legacy compatibility reasons, the nis profile is also available.

Red Hat recommends using authselect in semi-centralized identity management environments, for example if your company utilizes the LDAP, winbind or nis databases to authenticate users to use services in your domain.

WARNING

Do not use authselect if your host is part of Red Hat Enterprise Linux Identity Management or Active Directory. The ipa-client-install command, called when joining your host to a Red Hat Identity Management domain, takes full care of configuring authentication on your host. Similarly the realm join command, called when joining your host to an Active Directory domain, takes full care of configuring authentication on your host.

The authconfig utility, used in previous Red Hat Enterprise Linux versions, created and modified many different configuration files, making troubleshooting a difficult task. Authselect makes testing and troubleshooting easy because it only modifies files in these directories:

- /etc/nsswitch.conf
- /etc/pam.d/* files
- /etc/dconf/db/distro.d/* files

The Name Service Switch (NSS) configuration file, /etc/nsswitch.conf, is used by the GNU C Library and certain other applications to determine the sources from which to obtain name-service information in a range of categories, and in what order. Each category of information is identified by a database name.

Linux-PAM (Pluggable Authentication Modules) is a system of modules that handle the authentication tasks of applications (services) on the system. The nature of the authentication is dynamically configurable: the system administrator can choose how individual service-providing applications will authenticate users. This dynamic configuration is set by the contents of the configuration files in the /etc/pam.d/ directory, which list the PAMs that will do the authentication tasks required by this service, and the appropriate behavior of the PAM-API in the event that individual PAMs fail.
CHAPTER 1. CONFIGURING USER AUTHENTICATION USING AUTHSELECT

Once an authselect profile is selected for a given host, the profile will be applied to every user logging into the host.

1.2. CHOOSING AN AUTHSELECT PROFILE

As a system administrator, you can select a profile for the authselect utility for a specific host. The profile will be applied to every user logging into the host.

Procedure

1. Select the authselect profile that is appropriate for your authentication provider. For example, for logging into the network of a company that uses LDAP, choose sssd. Run the command as root:

   # authselect select sssd

2. Optionally, review the contents of the /etc/nsswitch.conf file:

   | passwd:  sss files |
   | group:    sss files |
   | netgroup: sss files |
   | automount: sss files |
   | services: sss files |
   |

   The content of the /etc/nsswitch.conf file shows that selecting the sssd profile means that the system first uses sssd if information concerning one of the first five items is requested. Only if the requested information is not found in the sssd cache and on the server providing authentication, or if sssd is not running, the system looks at the local files, that is /etc/*.

   For example, if information is requested about a user id, the user id is first searched in the sssd cache. If it is not found there, the /etc/passwd file is consulted. Analogically, if a user’s group affiliation is requested, it is first searched in the sssd cache and only if not found there, the /etc/group file is consulted.

   In practice, the local files database does not normally get consulted at all. The only exception is the case of the root user, which is never handled by sssd but by files.

3. Optionally, review the contents of the /etc/pam.d/system-auth file:

   # Generated by authselect on Tue Sep 11 22:59:06 2018
   # Do not modify this file manually.

   auth    required    pam_env.so
   auth    required    pam_faildelay.so delay=2000000
   auth    [default=1 ignore=ignore success=ok]   pam_succeed_if.so uid >= 1000 quiet
   auth    [default=1 ignore=ignore success=ok]   pam_localuser.so
   auth    sufficient    pam_unix.so nullok try_first_pass
   auth    requisite    pam_succeed_if.so uid >= 1000 quiet_success
   auth    sufficient    pam_sss.so forward_pass
   auth    required    pam_deny.so
   account   required    pam_unix.so
   account   sufficient    pam_localuser.so
   |

...
Among other things, the `/etc/pam.d/system-auth` file contains information about:

- user password lockout condition
- the possibility to authenticate with a smart card
- the possibility to authenticate with fingerprints

You can modify the default profile settings by adding the following options to the `authselect select sssd` or `authselect select winbind` command, for example:

- `with-faillock`
- `with-smartcard`
- `with-fingerprint`

To see the full list of available options, see Section 1.5, “Converting your scripts from authconfig to authselect” or the authselect-migration(7) man page.

**NOTE**

Make sure that the configuration files that are relevant for your profile are configured properly before finishing the `authselect` select procedure. For example, if the `sssd` daemon is not configured correctly and active, running `authselect select` results in only local users being able to authenticate, using `pam_unix`.

If adjusting a ready-made profile by adding one of the `authselect select` command-line options described above is not enough for your use case, you can:

- modify a ready-made profile by changing the `/etc/authselect/user-nsswitch.conf` file. For details, see Section 1.3, “Modifying a ready-made authselect profile”.
- create your own custom profile. For details, see Section 1.4, “Creating and deploying your own authselect profile”.

### 1.3. MODIFYING A READY-MADE AUTHSELECT PROFILE

As a system administrator, you can modify one of the default profiles, the `sssd`, `winbind`, or the `nis` profile, to suit your needs. You can modify any of the items in the `/etc/authselect/user-nsswitch.conf` file with the exception of:

- `passwd`
- `group`
- `netgroup`
- `automount`
- `services`

Running `authselect select profile_name` afterwards will result in permissible changes to the profile being transferred from `/etc/authselect/user-nsswitch.conf` to the `/etc/nsswitch.conf` file but unacceptable changes being overwritten by the default profile configuration.
IMPORTANT

Do not modify the `/etc/nsswitch.conf` file directly.

Procedure

1. Select an `authselect` profile, for example:
   
   ```
   # authselect select sssd
   ```

2. Edit the `/etc/authselect/user-nsswitch.conf` file.

3. Apply the changes from the `/etc/authselect/user-nsswitch.conf` file:
   
   ```
   # authselect apply-changes
   ```

4. Optionally, review the `/etc/nsswitch.conf` file to verify that the changes from `/etc/authselect/user-nsswitch.conf` have been propagated there.

1.4. CREATING AND DEPLOYING YOUR OWN AUTHSELECT PROFILE

As a system administrator, you can create and deploy a custom profile by customizing one of the default profiles, the `sssd`, `winbind`, or the `nis` profile. This is particularly useful if Section 1.3, “Modifying a ready-made authselect profile” is not enough for your needs. When you deploy a custom profile, the profile is applied to every user logging into the given host.

Procedure

1. Create your custom profile by using the `authselect create-profile` command. For example, to create a custom profile called `user-profile` based on the ready-made `sssd` profile but one in which you can configure the items in the `/etc/nsswitch.conf` file yourself:
   
   ```
   # authselect create-profile user-profile -b sssd --symlink-meta --symlink-pam
   New profile was created at /etc/authselect/custom/user-profile
   ```

   Including the `--symlink-pam` option in the command means that PAM templates will be symbolic links to the origin profile files instead of their copy; including the `--symlink-meta` option means that meta files, such as README and REQUIREMENTS will be symbolic links to the origin profile files instead of their copy. This ensures that all future updates to the PAM templates and meta files in the original profile will be reflected in your custom profile, too.

   The command has created a copy of the `/etc/nsswitch.conf` file in the `/etc/authselect/custom/user-profile/` directory.

2. Configure the `/etc/authselect/custom/user-profile/nsswitch.conf` file.

3. Select the custom profile by running the `authselect select` command, and adding `custom/name_of_the_profile` as a parameter. For example, to select the `user-profile` profile:
   
   ```
   # authselect select custom/user-profile
   ```

   Selecting the `user-profile` profile for your machine means that if the `sssd` profile is subsequently updated by Red Hat, you will benefit from all the updates with the exception of updates made to the `/etc/nsswitch.conf` file.
Example
The following procedure shows how to create a profile based on the `sssd` profile which only consults the local static table lookup for hostnames in the `/etc/hosts` file, not in the `dns` or `myhostname` databases.

1. Edit the `/etc/nsswitch.conf` file by editing the following line:

   ```
   hosts: files
   ```

2. Create a custom profile based on `sssd` that excludes changes to `/etc/nsswitch.conf`:

   ```
   # authselect create-profile user-profile -b sssd --symlink-meta --symlink-pam
   ```

3. Select the profile:

   ```
   # authselect select custom/user-profile
   ```

4. Optionally, check that selecting the custom profile has
   - created the `/etc/pam.d/system-auth` file according to the chosen `sssd` profile
   - left the configuration in the `/etc/nsswitch.conf` unchanged:

   ```
   hosts: files
   ```

   **NOTE**
   Running `authselect select sssd` would, in contrast, result in

   ```
   hosts: files dns myhostname
   ```

1.5. CONVERTING YOUR SCRIPTS FROM AUTHCONFIG TO AUTHSELECT

If you use `ipa-client-install` or `realm join` to join a domain, you can safely remove any `authconfig` call in your scripts. If this is not possible, replace each `authconfig` call with its equivalent `authselect` call. In doing that, select the correct profile and the appropriate options. In addition, edit the necessary configuration files:

- `/etc/krb5.conf`

- `/etc/sssd/sssd.conf` (for the `sssd` profile) or `/etc/samba/smb.conf` (for the `winbind` profile)

Table 1.1, “Relation of authconfig options to authselect profiles” and Table 1.2, “Authselect profile option equivalents of authconfig options” show the `authselect` equivalents of `authconfig` options.

Table 1.1. Relation of authconfig options to authselect profiles

<table>
<thead>
<tr>
<th>Authconfig options</th>
<th>Authselect profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>--enableldap --enableldapauth</td>
<td>sssd</td>
</tr>
</tbody>
</table>
Table 1.2. Authselect profile option equivalents of authconfig options

<table>
<thead>
<tr>
<th>Authconfig option</th>
<th>Authselect profile feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>--enablesmartcard</td>
<td>with-smartcard</td>
</tr>
<tr>
<td>--enablefingerprint</td>
<td>with-fingerprint</td>
</tr>
<tr>
<td>--enableecryptfs</td>
<td>with-ecryptfs</td>
</tr>
<tr>
<td>--enablemkhomedir</td>
<td>with-mkhomedir</td>
</tr>
<tr>
<td>--enablefaillock</td>
<td>with-faillock</td>
</tr>
<tr>
<td>--enablepamaccess</td>
<td>with-pamaccess</td>
</tr>
<tr>
<td>--enablewinbindkrb5</td>
<td>with-krb5</td>
</tr>
</tbody>
</table>

Table 1.3, “Examples of authselect commands equivalents to authconfig commands” shows example transformations of Kickstart calls to authconfig into Kickstart calls to authselect.

Table 1.3. Examples of authselect commands equivalents to authconfig commands

<table>
<thead>
<tr>
<th>authconfig command</th>
<th>authselect equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>authconfig --enableldap --enableldapauth --enablefaillock --updateall</td>
<td>authselect select sssd with-faillock</td>
</tr>
<tr>
<td>authconfig --enablessssd --enablessssdauth --enablesmartcard --smartcardmodule=sssd --updateall</td>
<td>authselect select sssd with-smartcard</td>
</tr>
<tr>
<td>authconfig --enableecryptfs --enablepamaccess --updateall</td>
<td>authselect select sssd with-ecryptfs with-pamaccess</td>
</tr>
<tr>
<td>authconfig --enablewinbind --enablewinbindauth --winbindjoin=Administrator --updateall</td>
<td>realm join -U Administrator --client-software=winbind WINBINDDOMAIN</td>
</tr>
</tbody>
</table>
CHAPTER 2. CONFIGURING SMART CARDS USING AUTHSELECT

This chapter describes how to configure your smart card to achieve one of the following aims:

- Enable both password and smart card authentication
- Disable password and enable smart card authentication
- Enable lock on removal

Prerequisites

- Authselect installed
  The authselect tool configures user authentication on Linux hosts and you can use it to configure smart card authentication parameters. For details about authselect, see Explaining authselect.

- Smart Card or USB device supported by RHEL 8
  For details, see Smart Card support in RHEL8.

2.1. CERTIFICATES ELIGIBLE FOR SMART CARDS

Before you can configure a smart card with authselect, you must import a certificate into your card. You can use the following tools to generate the certificate:

- Active Directory (AD)
- Identity Management (IdM)
  For details about how to create IdM certificates, see Requesting a new user certificate and exporting it to the client.
- Red Hat Certificate System (RHCS)
  For details, see Managing Smart Cards with the Enterprise Security Client.
- Local Certification Authority. You can use a certificate generated by the Local Certification Authority if the user is not part of a domain or for testing purposes.
  For details about how to create and import local certificates into a smart card, Configuring and importing local certificates to a smart card.

2.2. ENABLING USER PASSWORD AUTHENTICATION TO CONFIGURE SMART CARD AUTHENTICATION

This section describes how to enable both smart card and password authentication on your system.

Prerequisites

- The Smart card contains your certificate and private key.
- The card is inserted into the reader and connected to the computer.
- The authselect tool is installed on your system.
CHAPTER 2. CONFIGURING SMART CARDS USING AUTHSELECT

2.3. CONFIGURING AUTHSELECT TO ENFORCE SMART CARD AUTHENTICATION

The authselect tool enables you to configure smart card authentication on your system and to disable the default password authentication. The authselect command must include the following options:

- `with-smartcard` — enabling smart card authentication
- `with-smartcard-required` — enabling exclusive smart card authentication (authentication with a password is disabled)

Prerequisites

- Smart card contains your certificate and private key.
- The card is inserted into the reader and connected to the computer.
- The authselect tool is installed on your local system.

Procedure

Enter the following command to enforce smart card authentication:

```
# authselect select sssd with-smartcard with-smartcard-required --force
```

At this point, you can only log in with a smart card. Password authentication will not be working any more.

2.4. CONFIGURING SMART CARD AUTHENTICATION WITH LOCK ON REMOVAL

The authselect service enables you to configure your smart card authentication to lock your screen instantly after removing the smart card from the reader. The authselect command must include the following variables:

- `with-smartcard` — enabling smart card authentication
- `with-smartcard-required` — enabling exclusive smart card authentication (authentication with a password is disabled)
- `with-smartcard-lock-on-removal` — enforcing log out after the smart card removal

Prerequisites
- Smart card contains your certificate and private key.
- The card is inserted into the reader and connected to the computer.
- The `authselect` tool is installed on your local system.

**Procedure**

- Enter the following command to enable smart card authentication, disable password authentication, and enforce lock on removal:

```bash
# authselect select sssd with-smartcard with-smartcard-required with-smartcard-lock-on-removal --force
```

Now, when you remove the card, the screen locks. You must re-insert your smart card to unlock it.
CHAPTER 3. CONFIGURING AND IMPORTING LOCAL CERTIFICATES TO A SMART CARD

This chapter describes a scenario where:

- The host is not connected to a domain.
- You want to authenticate with a smart card on this host.
- You want to configure SSH access using smart card authentication.
- You want to configure the smart card with **authselect**

Use the following configuration to accomplish this scenario:

- Obtain a user certificate for the user who wants to authenticate with a smart card. The certificate should be generated by a trustworthy Certification Authority used in the domain. If you cannot get the certificate, you can generate a user certificate signed by a local certificate authority for testing purposes.
- Store the certificate and private key in a smart card.
- Configure the smart card authentication for SSH access.

**IMPORTANT**

If a host can be part of the domain, add the host to the domain and use certificates generated by Active Directory or Identity Management Certification Authority.

For details about how to create IdM certificates for a smart card, see Configuring Identity Management for smart card authentication.

Prerequisites

- Authselect installed
  The authselect tool configures user authentication on Linux hosts and you can use it to configure smart card authentication parameters. For details about authselect, see Explaining authselect.
- Smart Card or USB device supported by RHEL 8
  For details, see Smart Card support in RHEL 8.

3.1. CREATING LOCAL CERTIFICATES

This section describes how to perform these tasks:

- Generate the OpenSSL certificate authority
- Create a certificate signing request
WARNING

The following steps are intended for testing purpose only. Certificates generated by a local self-signed Certificate Authority are not as secure as using AD, IdM, or RHCS Certification Authority. You should use a certificate generated by your enterprise Certification Authority even if the host is not part of the domain.

Procedure

1. Create a directory where you can generate the certificate, for example:

```
# mkdir /tmp/ca
# cd /tmp/ca
```

2. Set up the certificate (copy this text to your command line in the `ca` directory):

```
cat > ca.cnf <<EOF
[ ca ]
default_ca = CA_default

[ CA_default ]
dir = .
database = $dir/index.txt
new_certs_dir = $dir/newcerts
certificate = $dir/rootCA.crt
serial = $dir/serial
private_key = $dir/rootCA.key
RANDFILE = $dir/rand

default_days = 365
default_crl_days = 30
default_md = sha256

policy = policy_any
default_crl_days = 30
default_md = sha256

name_opt = ca_default
cert_opt = ca_default
copy_extensions = copy

[ usr_cert ]
authorityKeyIdentifier = keyid, issuer

[ v3_ca ]
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer:always
basicConstraints = CA:true
keyUsage = critical, digitalSignature, cRLSign, keyCertSign

[ policy_any ]
```
organizationName = supplied
organizationalUnitName = supplied
commonName = supplied
emailAddress = optional

[ req ]
distinguished_name = req_distinguished_name
prompt = no

[ req_distinguished_name ]
O = Example
OU = Example Test
CN = Example Test CA
EOF

3. Create the following directories:

   # mkdir certs crl newcerts

4. Create the following files:

   # touch index.txt crlnumber index.txt.attr

5. Write the number 01 in the serial file:

   # echo 01 > serial

   This command writes a number 01 in the serial file. It is a serial number of the certificate. With each new certificate released by this CA the number increases by one.

6. Create an OpenSSL root CA key:

   # openssl genrsa -out rootCA.key 2048

7. Create a self-signed root Certification Authority certificate:

   # openssl req -batch -config ca.cnf
      -x509 -new -nodes -key rootCA.key -sha256 -days 10000
      -set_serial 0 -extensions v3_ca -out rootCA.crt

8. Create the key for your username:

   # openssl genrsa -out example.user.key 2048

   This key is generated in the local system which is not secure, therefore, remove the key from the system when the key is stored in the card.

   You can create a key directly in the smart card as well. For doing this, follow instructions created by the manufacturer of your smart card.

9. Create the certificate signing request configuration file (copy this text to your command line in the ca directory):

   cat > req.cnf <<EOF

   CHAPTER 3. CONFIGURING AND IMPORTING LOCAL CERTIFICATES TO A SMART CARD

   EOF
[ req ]
distinguished_name = req_distinguished_name
prompt = no

[ req_distinguished_name ]
O = Example
OU = Example Test
CN = testuser

[ req_exts ]
basicConstraints = CA:FALSE
nsCertType = client, email
nsComment = "testuser"
sObjectKeyIdentifier = hash
keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = clientAuth, emailProtection, msSmartcardLogin
subjectAltName = otherName:msUPN;UTF8:testuser@example.com,
email:testuser@example.com
EOF

10. Create a certificate signing request for your example.user certificate:

    # openssl req -new -nodes -key example.user.key
    -reqexts req_exts -config req.cnf -out testuser.csr

11. Configure the new certificate. Expiration period is set to 1 year:

    # openssl ca -config ca.cnf -batch -notext
    # -keyfile rootCA.key -in example.user.csr -days 365
    # -extensions usr_cert -out example.user.crt

At this point, the certification authority and certificates are successfully generated and prepared for
import into a smart card.

3.2. COPYING CERTIFICATES TO THE SSSD DIRECTORY

Gnome Desktop Manager (GDM) requires SSSD. If you use GDM, you need to copy the PEM certificate
to the /etc/sssd/pki directory.

Prerequisites

- The local CA authority and certificates have been generated

Procedure

1. Ensure that you have SSSD installed on the system.

    # rpm -q sssd
    sssd-2.0.0.43.el8_0.3.x86_64

2. Create a /etc/sssd/pki directory:
3. Copy the `rootCA.crt` as a PEM file in the `/etc/sssd/pki/` directory:

```bash
# cp /tmp/ca/rootCA.crt /etc/sssd/pki/sssd_auth_ca_db.pem
```

Now you have successfully generated the certificate authority and certificates, and you have saved them in the `/etc/sssd/pki` directory.

**NOTE**

If you want to share the Certificate Authority certificates with another application, you can change a location in sssd.conf:

- SSSD PAM responder: `pam_cert_db_path` in the `[pam]` section
- SSSD ssh responder: `ca_db` in the `[ssh]` section

For details, see `man sssd.conf`.

Red Hat recommends to keep the default path and use a dedicated Certificate Authority certificate file for SSSD to make sure that only Certificate Authorities trusted for authentication are listed here.

### 3.3. INSTALLING GNUTLS-UTILS

To configure your smart card, you need a tool that can generate certificates and store them in the smart card.

You must:

- Install the `gnutls-utils` program which helps you to manage certificates.
- Start the `pcscd` service which communicates with the smart card reader.

**Procedure**

1. Install the the `gnutls-utils` package which allows you to manage smart card settings for reading and writing from the smart card:

   ```bash
   # dnf -y install opensc gnutls-utils
   ```

2. Start the `pcscd` service.

   ```bash
   # systemctl start pcscd
   ```

Verify that the `pcscd` service is up and running.

### 3.4. STORING A CERTIFICATE ON THE SMART CARD

This section describes smart card configuration with `gnutls-utils` which helps you to configure:
- Erasing your smart card
- Setting a new PIN and PUK
- Creating a new slot in the smart card
- Storing the certificate and private key in the slot
- Locking the smart card settings (some smart cards need this type of finalization)

Prerequisites

- The `gnutls-utils` package is installed. For details, see [Installing gnutls-utils](#).
- The card is inserted in the reader and connected to the computer.

Procedure

1. Erase your smart card and authenticate yourself with PIN:

   ```bash
   # spawn pkcs15-init --erase-card --use-default-transport-keys
   Using reader with a card: Smart Card name
   PIN [Security Officer PIN] required.
   Please enter PIN [Security Officer PIN]:
   
   The card has been erased.
   ```

2. Set your PIN and PUK — both twice for verification:

   ```bash
   # pkcs15-init --create-pkcs15 --use-default-transport-keys \
   --pin redhat --puk redhat --so-pin redhat --so-puk redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   
   The `pcks15-init` creates a new slot in the smart card and marks it with a label and an authentication ID.
   ```

3. Add a label that is the name of the slot and the authentication ID:

   ```bash
   # pkcs15-init --store-pin --label testuser \n   --auth-id 01 --so-pin redhat --pin redhat --puk redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   
   The label auth-id is set to 01.
   ```

4. Store the private key in the new slot of the smart card:

   ```bash
   # pkcs15-init --store-private-key testuser.key --label testuser_key \n   --auth-id 01 --id 01 --so-pin redhat --pin redhat
   Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface]
   (53311657131456) 00 00
   ```

5. Store the certificate in the new slot of the smart card:
pkcs15-init --store-certificate testuser.crt --label testuser_crt --auth-id 01 --id 01 --format pem --so-pin redhat --pin redhat

Using reader with a card: SCM Microsystems Inc. SCR 3320 [CCID Interface] (53311657131456) 00 00

6. Optionally finalize the card settings with locking the settings:

# pkcs15-init -F

**IMPORTANT**

Some smart cards require this step.

At this stage, your smart card includes the certificate and private key in the newly created slot. PIN and PUK has been successfully created.

### 3.5. CONFIGURING SSH ACCESS USING SMART CARD AUTHENTICATION

If your smart card is configured, you can set up SSH access to this host using smart card authentication.

Basically you need to run the `ssh-keygen` command to generate a key using the opensc library to write to the SSH keys.

**Prerequisites**

- The smart card contains your certificate and private key.
- The card is inserted in the reader and connected to the computer.

**Procedure**

1. Create a new directory for SSH keys in the home directory of the user who uses smart card authentication:

   # mkdir /home/example.user/.ssh

2. Generate the SSH keys:

   # ssh-keygen -D /usr/lib64/pkcs11/opensc-pkcs11.so >> ~example.user/.ssh/authorized_keys

3. Change access rights:

   # chown -R example.user:example.user ~example.user/.ssh/
   # chmod 700 ~example.user/.ssh/
   # chmod 600 ~example.user/.ssh/authorized_keys

4. Optionally, display the keys:

   # cat ~example.user/.ssh/authorized_keys
The terminal displays the keys.

5. To use the ssh keys, configure authentication with the **authselect** command:

   ```bash
   # authselect select sssd with-smartcard with-smartcard-lock-on-removal --force
   ```

   For details about configuring smart cards with authselect, see [Configuring smart cards using authselect](#).

Now, you can verify ssh access with the following command:

```bash
# ssh -l /usr/lib64/opensc-pkcs11.so -l example.user localhost hostname
```

If the configuration is successful, you should not be prompted for entering your credentials. You just need to add the smart card PIN.