Red Hat Enterprise Linux 9

管理智能卡验证

在 RHEL 中设置和管理智能卡验证

Last Updated: 2022-07-25

Red Hat Enterprise Linux 9 管理智能卡验证

在 RHEL 中设置和管理智能卡验证

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律通告

摘要

本文档集合提供了如何在 RHEL 中管理智能卡验证的说明。

目录

让开源更具包容性

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。有关更多详情，请参阅我们的首席技术官 Chris Wright 提供的消息。

在身份管理中，计划中的术语变化包括：

- 使用 ***block list*** 替换 *blacklist*
- 使用 ***allow list*** 替换 *whitelist*
- 使用 ***secondary*** 替换 *slave*
- *master* 会根据上下文被替换为其他更适当的术语:
  - 使用 ***IdM server*** 替换 *IdM master*
  - 使用 ***CA renewal server*** 替换 *CA renewal master*
  - 使用 ***CRL publisher server*** 替换 *CRL master*
  - 使用 ***multi-supplier*** 替换 *multi-master*

对红帽文档提供反馈

我们感谢您对文档提供反馈信息。请让我们了解如何改进文档。

- 关于特定内容的简单评论：
  1. 请确定您使用 *Multi-page HTML* 格式查看文档。另外，确定 **Feedback** 按钮出现在文档页的右上方。
  2. 用鼠标指针高亮显示您想评论的文本部分。
  3. 点在高亮文本上弹出的 **Add Feedback**。
  4. 按照显示的步骤操作。
- 要通过 Bugzilla 提交反馈，请创建一个新的问题单：
  1. 进入 Bugzilla 网站。
  2. 在 Component 中选择 **Documentation**。
  3. 在 **Description** 中输入您要提供的信息。包括文档相关部分的链接。
  4. 点 **Submit Bug**。

第 1 章 为智能卡验证配置身份管理

使用基于智能卡的验证是使用密码进行验证的替代选择。您可以将用户凭证以私钥和证书的形式存储在智能卡上，并使用证书、特殊的软件和硬件来访问它们。将智能卡放在读卡器或 USB 端口中，并为智能卡提供 PIN 代码，而不是提供您的密码。

身份管理(IdM)支持使用如下方式的智能卡身份验证：

- IdM 证书颁发机构发布的用户证书
- 外部证书颁发机构发布的用户证书

这个用户用例演示了如何在 IdM 中为两种类型的证书设置智能卡验证。在用户故事中，**smartcard_ca.pem** CA 证书是包含信任的外部证书颁发机构的证书的文件。

用户会包括以下模块：

- 为智能卡验证配置 IdM 服务器
- 为智能卡验证配置 IdM 客户端
- 在 IdM Web UI 的用户条目中添加证书
- 在 IdM CLI 中向用户条目中添加证书
- 安装用来管理和使用智能卡的工具
- 在智能卡中存储证书
- 使用智能卡登录到 IdM
- 使用智能卡身份验证配置 GDM 访问
- 使用智能卡验证配置 su 访问

1.1. 为智能卡验证配置 IDM 服务器

如果要为其证书是由 **EXAMPLE.ORG** 域（其 LDAP 区分名称(DN)是 **CN=Certificate Authority,DC=EXAMPLE,DC=EXAMPLE,DC=ORG**）的证书颁发机构发布的用户启用智能卡身份验证，那么您需要获取颁发机构的证书，以便您可以使用配置 IdM 服务器的脚本来运行它。例如，您可以从认证机构发布的证书的网页下载证书。详情请查看 配置浏览器来启用证书身份验证 中的步骤 1 - 4a。

要为 IdM 证书颁发机构为其发布证书的 IdM 用户启用智能卡身份验证，请从运行 IdM CA 的 IdM 服务器上的 **/etc/ipa/ca.crt** 文件中获取 CA 证书。

这部分论述了如何为智能卡验证配置 IdM 服务器。首先，获取带有 PEM 格式 CA 证书的文件，然后运行内置的 **ipa-advise** 脚本。最后，重新载入系统配置。

先决条件

- 有到 IdM 服务器的 root 访问权限。
- 您有 root CA 证书和任何子 CA 证书。

流程

1. 创建要进行配置的目录：

```
[root@server]# mkdir ~/SmartCard/
```

2. 进入该目录：

```
[root@server]# cd ~/SmartCard/
```

3. 获取存储在 PEM 格式文件中的相关 CA 证书。如果您的 CA 证书存储再不同格式的文件中，如 DER，请将其转换为 PEM 格式。IdM 证书颁发机构证书位于 **/etc/ipa/ca.crt** 文件中。
将 DER 文件转换为 PEM 文件：

```
# openssl x509 -in <filename>.der -inform DER -out <filename>.pem -outform PEM
```

4. 为方便起见，将证书复制到您要进行配置的目录中：

```
[root@server SmartCard]# cp /etc/ipa/ca.crt ~/SmartCard/
[root@server SmartCard]# cp /tmp/smartcard_ca.pem ~/SmartCard/
```

5. 另外，如果您使用外部证书颁发机构的证书，请使用 **openssl x509** 工具查看 **PEM** 格式的文件内容，来检查 **Issuer** 和 **Subject** 值是否正确：

```
[root@server SmartCard]# openssl x509 -noout -text -in smartcard_ca.pem | more
```

6. 使用管理员特权，通过内置的 **ipa-advise** 工具生成配置脚本：

```
[root@server SmartCard]# kinit admin
[root@server SmartCard]# ipa-advise config-server-for-smart-card-auth > config-server-for-smart-card-auth.sh
```

config-server-for-smart-card-auth.sh 脚本执行以下操作：

- 它配置 IdM Apache HTTP 服务器。
- 它在 KDC（Key Distribution Center）中启用 PKINIT（Public Key Cryptography for Initial Authentication in Kerberos）。
- 它将 IdM Web UI 配置为接受智能卡授权请求。

7. 执行脚本，将包含根 CA 和子 CA 证书的 PEM 文件添加为参数：

```
[root@server SmartCard]# chmod +x config-server-for-smart-card-auth.sh
[root@server SmartCard]# ./config-server-for-smart-card-auth.sh smartcard_ca.pem ca.crt
Ticket cache:KEYRING:persistent:0:0
Default principal: admin@IDM.EXAMPLE.COM
[...]
Systemwide CA database updated.
The ipa-certupdate command was successful
```

注意

在任何子 CA 证书前，确保将根 CA 的证书添加为参数，并且 CA 或子 CA 证书还没有过期。

8. 另外，如果发布用户证书的证书颁发机构不提供任何在线证书状态协议(OCSP)响应程序，您可能需要禁用对 IdM Web UI 身份验证的 OCSP 检查：

   a. 在 **/etc/httpd/conf.d/ssl.conf** 文件中将 **SSLOCSPEnable** 参数设为 **off** ：

   ```
   SSLOCSPEnable off
   ```

   b. 重启 Apache 守护进程(httpd)使更改立即生效：

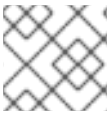
   ```
   [root@server SmartCard]# systemctl restart httpd
   ```

警告

如果您只使用 IdM CA 发出的用户证书，不要禁用 OCSP 检查。OCSP 响应器是 IdM 的一部分。

有关如何保持 OCSP 检查处于启用状态，同时防止 IdM 服务器拒绝用户证书（如果 IdM 服务器不包含有关颁发用户证书的 CA 侦听 OCSP 服务请求的位置的信息）的说明，请参阅 Apache mod_ssl 配置选项 中的 **SSLOCSPDefaultResponder** 指令。

该服务器现在被配置为智能卡验证。



注意

要在整个拓扑中启用智能卡验证，请在每个 IdM 服务器中运行操作过程。

1.2. 为智能卡验证配置 IDM 客户端

这部分描述了如何为智能卡身份验证配置 IdM 客户端。这个过程需要运行在每个 IdM 系统、客户端或服务器上，您希望在使用智能卡进行身份验证时连接到这些系统。例如，若要启用从主机 A 到主机 B 的 **ssh** 连接，需要在主机 B 上运行脚本。

作为管理员，运行这个流程来使用如下方法启用智能卡身份验证

- **ssh** 协议
  详情请参阅 使用智能卡身份验证配置 SSH 访问。
- 控制台登录
- Gnome 显示管理器(GDM)
- **su** 命令

对于向 IdM Web UI 进行身份验证，不需要此流程。向 IdM Web UI 进行身份验证涉及两个主机，它们都不必是 IdM 客户端：

- 机器可能位于运行浏览器的 IdM 域之外
- 运行 **httpd** 的 IdM 服务器

以下流程假设您在 IdM 客户端，而不是 IdM 服务器上配置智能卡身份验证。因此，您需要两台计算机：生成配置脚本的 IdM 服务器，以及运行脚本的 IdM 客户端。

先决条件

- 为智能卡验证配置了您的 IdM 服务器，如 为智能卡验证配置 IdM 服务器 所述。
- 有对 IdM 服务器和 IdM 客户端的 root 访问权限。
- 您可以访问 root CA 证书和任何子 CA 证书。
- 已使用 **--mkhomedir** 选项安装了 IdM 客户端，以确保远程用户可以成功登录。如果您没有创建主目录，则默认登录位置为 root。

步骤

1. 在 IdM 服务器上，使用管理员权限通过 **ipa-advise** 生成配置脚本：

```
[root@server SmartCard]# kinit admin
[root@server SmartCard]# ipa-advise config-client-for-smart-card-auth > config-client-for-smart-card-auth.sh
```

config-client-for-smart-card-auth.sh 脚本执行以下操作：

- 它配置智能卡守护进程。
- 它设置系统范围信任存储。
- 它将系统安全服务守护进程(SSSD)配置为允许智能卡登录到桌面。

2. 从 IdM 服务器中，将脚本复制到 IdM 客户端机器中选择的目录中：

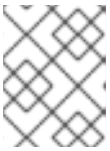
```
[root@server SmartCard]# scp config-client-for-smart-card-auth.sh root@client.idm.example.com:/root/SmartCard/
Password:
config-client-for-smart-card-auth.sh        100%  2419     3.5MB/s   00:00
```

3. 为了方便起见，将 IdM 服务器上的 PEM 格式的 CA 证书文件复制到 IdM 客户端机器上与在上一步中所使用的相同的目录中：

```
[root@server SmartCard]# scp {smartcard_ca.pem,ca.crt} root@client.idm.example.com:/root/SmartCard/
Password:
smartcard_ca.pem                 100%  1237     9.6KB/s   00:00
ca.crt                           100%  2514    19.6KB/s   00:00
```

4. 在客户端机器上执行脚本，将包含 CA 证书的 PEM 文件添加为参数：

```
[root@client SmartCard]# kinit admin
[root@client SmartCard]# chmod +x config-client-for-smart-card-auth.sh
[root@client SmartCard]# ./config-client-for-smart-card-auth.sh smartcard_ca.pem ca.crt
Ticket cache:KEYRING:persistent:0:0
Default principal: admin@IDM.EXAMPLE.COM
[...]
Systemwide CA database updated.
The ipa-certupdate command was successful
```



注意

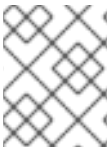
在任何子 CA 证书前，确保将根 CA 的证书添加为参数，并且 CA 或子 CA 证书还没有过期。

现在为智能卡验证配置了客户端。

1.3. 在 IDM WEB UI 的用户条目中添加证书

这个步骤描述了如何在 IdM Web UI 的用户条目中添加外部证书。

也可以将证书映射数据上传到 IdM 中的用户条目，而不必上传整个证书。包含完整证书或证书映射数据的用户条目可以和相应的证书映射规则一起使用，以便于系统管理员配置智能卡身份验证。详情请查看用于在智能卡上配置身份验证的证书映射规则。



注意

如果用户的证书已由 IdM 证书颁发机构发布，且证书已存储在用户条目中，则您可以跳过本节。

先决条件

- 您有要添加到用户条目的证书。

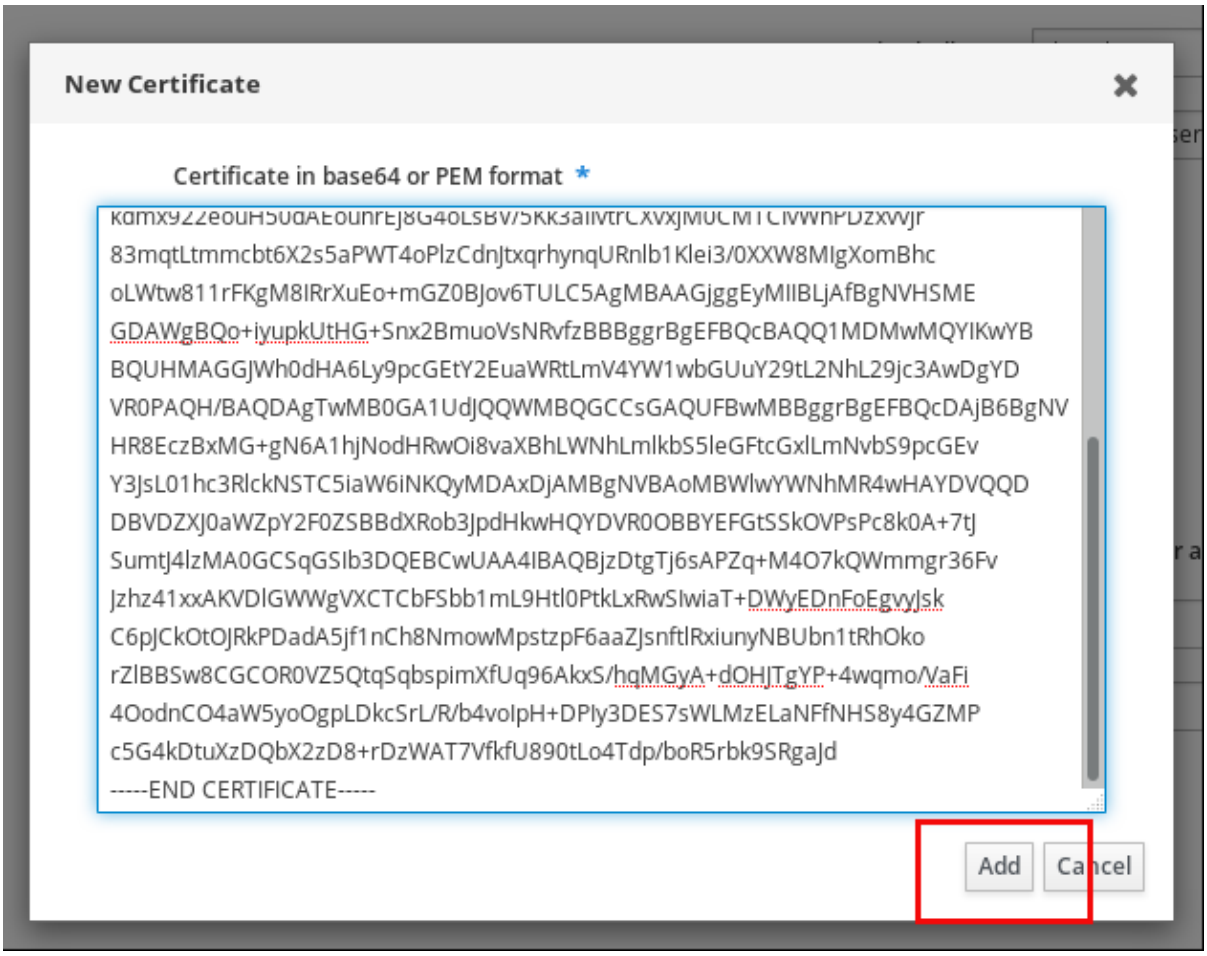
流程

1. 如果要给另一个用户添加证书，请以管理员身份登录到 IdM Web UI。要在您自己的配置文件中添加证书，您不需要管理员的凭证。
2. 导航到 **Users → Active users → sc_user**。
3. 找到 **Certificate** 选项，并单击 **Add** 。
4. 在 **命令行界面** 中，使用 **cat** 工具或文本编辑器以 **PEM** 格式显示证书：

```
[user@client SmartCard]$ cat testuser.crt
```

5. 将证书从 CLI 复制并粘贴到 Web UI 中打开的窗口中。
6. 点击 **Add**。

图 1.1. 在 IdM Web UI 中添加新证书



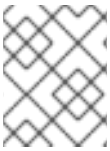
sc_user 条目现在包含一个外部证书。

1.4. 在 IDM CLI 中向用户条目中添加证书

这个步骤描述了如何在 IdM CLI 的用户条目中添加外部证书。

也可以将证书映射数据上传到 IdM 中的用户条目，而不必上传整个证书。包含完整证书或证书映射数据的用户条目可以和相应的证书映射规则一起使用，以便于系统管理员配置智能卡身份验证。详情请查看

用于在智能卡上配置身份验证的证书映射规则。



注意

如果用户的证书已由 IdM 证书颁发机构发布，且证书已存储在用户条目中，则您可以跳过本节。

先决条件

- 您有要添加到用户条目的证书。

流程

1. 如果要给另一个用户添加证书，请以管理员身份登录到 IdM CLI：

```
[user@client SmartCard]$ kinit admin
```

要在您自己的配置文件中添加证书，您不需要管理员的凭证：

```
[user@client SmartCard]$ kinit sc_user
```

2. 创建一个包含证书的环境变量，该变量移除了标头和页脚，并串联成一行，这是 **ipa user-add-cert** 命令期望的格式：

```
[user@client SmartCard]$ export CERT=`openssl x509 -outform der -in testuser.crt | base64 -w0 -`
```

请注意，**testuser.crt** 文件中的证书必须是 **PEM** 格式。

3. 使用 **ipa user-add-cert** 命令将证书添加到 sc_user 的配置文件：

```
[user@client SmartCard]$ ipa user-add-cert sc_user --certificate=$CERT
```

sc_user 条目现在包含一个外部证书。

1.5. 安装用来管理和使用智能卡的工具

要配置智能卡，您需要一些工具来生成证书并将其保存在智能卡中。

您必须：

- 安装 **gnutls-utils** 软件包，其帮助您管理证书。
- 安装 **opensc** 软件包，它提供一组用于智能卡的库和工具。
- 启动与智能卡读卡器通信的 **pcscd** 服务。

步骤

1. 安装 **opensc** 和 **gnutls-utils** 软件包：

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

2. 启动 **pcscd** 服务。

```
# systemctl start pcscd
```

验证 **pcscd** 服务是否已启动并运行。

1.6. 在智能卡中存储证书

本节描述了使用 **pkcs15-init** 工具配置智能卡，该工具可帮助您配置：

- 擦除智能卡
- 设置新的 PIN 和可选的 PIN Unblocking Keys（PUKs）
- 在智能卡上创建新插槽
- 在插槽存储证书、私钥和公钥

- 锁定智能卡设置（有些智能卡需要这种类型）

先决条件

- 已安装 **opensc** 软件包，其包含 **pkcs15-init** 工具。
  详情请查看安装用于管理和使用智能卡的工具。
- 该卡插入读卡器并连接到计算机。
- 您有可保存在智能卡中的私钥、公钥和证书。在此流程中，**testuser.key**、**testuserpublic.key** 和 **testuser.crt** 是用于私钥、公钥和证书的名称。
- 您当前的智能卡用户 PIN 和 Security Officer PIN（SO-PIN）

流程

1. 擦除智能卡并使用您的 PIN 验证自己：

```
$ pkcs15-init --erase-card --use-default-transport-keys
Using reader with a card: Reader name
PIN [Security Officer PIN] required.
Please enter PIN [Security Officer PIN]:
```

   这个卡已经被清除。

2. 初始化智能卡，设置您的用户 PIN 和 PUK，以及您的安全响应 PIN 和 PUK：

```
$ pkcs15-init --create-pkcs15 --use-default-transport-keys \
   --pin 963214 --puk 321478 --so-pin 65498714 --so-puk 784123
Using reader with a card: Reader name
```

   pcks15-init 工具在智能卡上创建一个新插槽。

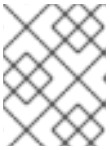
3. 为插槽设置标签和验证 ID：

```
$ pkcs15-init --store-pin --label testuser \
   --auth-id 01 --so-pin 65498714 --pin 963214 --puk 321478
Using reader with a card: Reader name
```

   标签设置为人类可读的值，在本例中为 **testuser**。**auth-id** 必须是两个十六进制值，在本例中设为 **01**。

4. 在智能卡的新插槽中存储并标记私钥：

```
$ pkcs15-init --store-private-key testuser.key --label testuser_key \
   --auth-id 01 --id 01 --pin 963214
Using reader with a card: Reader name
```



注意

在存储私钥和证书时，您为 **--id** 指定的值必须相同。如果没有为 **--id** 指定值，工具会计算一个更复杂的值，因此更容易定义您自己的值。

5. 在智能卡上的新插槽中存储并标记该证书：

```
$ pkcs15-init --store-certificate testuser.crt --label testuser_crt \
   --auth-id 01 --id 01 --format pem --pin 963214
Using reader with a card: Reader name
```

6. （可选）在智能卡上新插槽中保存并标记公钥：

```
$ pkcs15-init --store-public-key testuserpublic.key
   --label testuserpublic_key --auth-id 01 --id 01 --pin 963214
Using reader with a card: Reader name
```

注意

如果公钥与私钥和/或证书对应，您应该指定与私钥和/或证书相同的 ID。

7. （可选）有些智能卡要求您通过锁定设置来完善卡：

```
$ pkcs15-init -F
```

此时您的智能卡在新创建的插槽中包含证书、私钥和公钥。您还创建了您的用户 PIN 和 PUK，以及安全响应 PIN 和 PUK。

1.7. 使用智能卡登录到 IDM

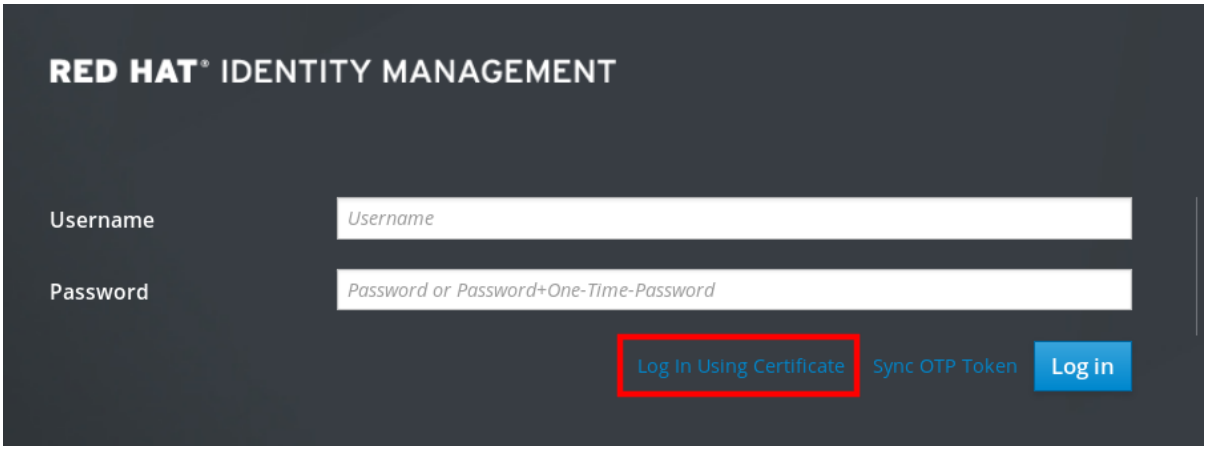
本节提供有关使用智能卡登录到 IdM Web UI 的信息。

先决条件

- web 浏览器被配置为使用智能卡验证。
- IdM 服务器已被配置为智能卡验证。
- IdM 服务器知道在智能卡中安装的证书。
- 您需要 PIN 解锁智能卡。
- 智能卡已插入到读取器中。

流程

1. 在浏览器中打开 IdM Web UI。
2. 点 **Log Insing Certificate**。

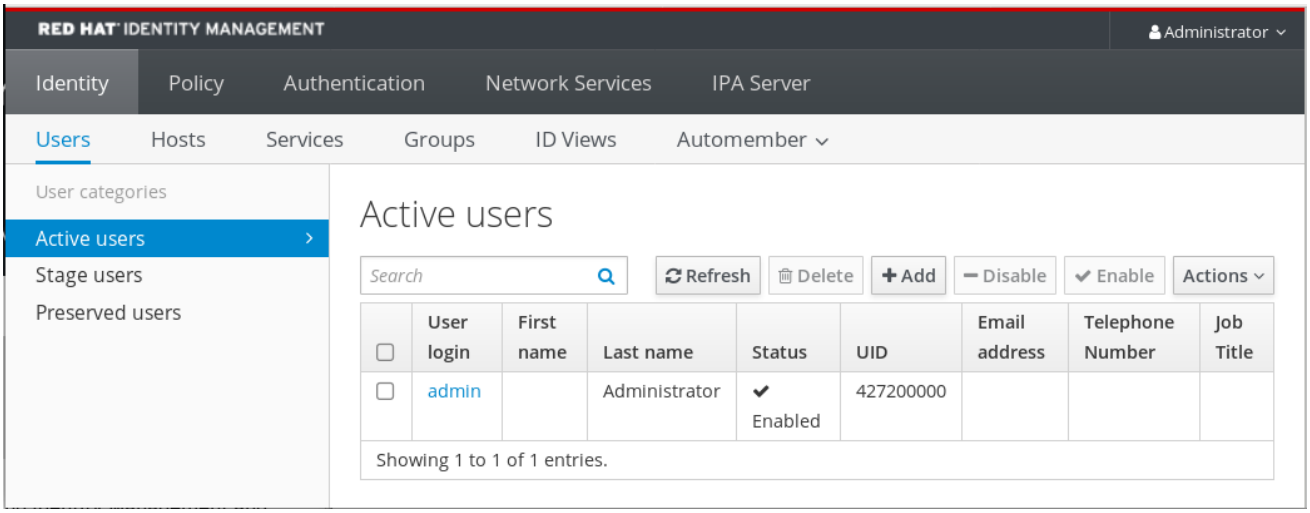


3. 如果 **Password Required** 对话框打开，请添加 PIN 来解锁智能卡，然后单击 **OK** 按钮。此时会打开 **User Identification Request** 对话框。

   如果智能卡包含多个证书，请在 **选择用于验证的证书** 下方的下拉列表中选择您要用于身份验证的证书。

4. 单击 **OK** 按钮。

现在，您已成功登录到 IdM Web UI。



1.8. 使用智能卡身份验证配置 GDM 访问

Gnome 桌面管理器(GDM)需要身份验证。您可以使用您的密码，但是，您也可以使用智能卡进行身份验证。

这部分描述了访问 GDM 的智能卡身份验证。

使用智能卡身份验证的好处在于，如果用户帐户是身份管理域的一部分，您还会获得一张票据授予票据(TGT)。

先决条件

- 该智能卡包含您的证书和私钥。
- 该用户帐户是 IdM 域的成员。
- 智能卡上的证书通过以下方式映射到用户条目：

- 为特定用户条目分配证书。详情请参阅 Adding a certificate to a user entry in the IdM Web UI 或 Adding a certificate to a user entry in the IdM CLI 。
- 应用到该帐户的证书映射数据。详情请查看用于 在智能卡上配置身份验证的证书映射规则。

流程

1. 在读取器中插入智能卡。
2. 输入智能卡 PIN。
3. 点 **Sign In**。

您成功登录到 RHEL 系统，并且您有一张由 IdM 服务器提供的 TGT。

验证步骤

- 在 **Terminal** 中输入 **klist**，并检查结果：

```
$ klist
Ticket cache: KEYRING:persistent:1358900015:krb_cache_TObtNMd
Default principal: example.user@REDHAT.COM

Valid starting       Expires              Service principal
04/20/2020 13:58:24  04/20/2020 23:58:24  krbtgt/EXAMPLE.COM@EXAMPLE.COM
 renew until 04/27/2020 08:58:15
```

1.9. 使用智能卡验证配置 SU 访问

切换到其他用户需要身份验证。您可以使用密码或证书。这部分描述了通过 **su** 命令使用智能卡。这意味着输入 **su** 命令后，系统会提示您输入智能卡 PIN。

先决条件

- 该智能卡包含您的证书和私钥。
- 该卡插入读卡器并连接到计算机。

步骤

- 在终端窗口中，使用 **su** 命令切换到其他用户：

```
$ su - example.user
PIN for smart_card
```

如果配置成功，会提示您输入智能卡 PIN。

第 2 章 为 IDM 中智能卡验证配置 ADCS 发布的证书

这种情境描述了以下情况：

- 您的部署是基于身份管理(IdM)和活动目录(AD)之间的跨林信任。
- 您希望允许智能卡验证存储在 AD 中的帐户的用户。
- 证书创建并存储在活动目录证书服务(ADCS)中。

配置将按以下步骤完成：

- 将 CA 和用户证书从活动目录复制到 IdM 服务器和客户端
- 使用 ADCS 证书为智能卡身份验证配置 IdM 服务器和客户端
- 转换 PFX(PKCS#12)文件，以便能够将证书和私钥存储到智能卡中
- 在 sssd.conf 文件中配置超时
- 为智能卡身份验证创建证书映射规则

先决条件

- 身份管理(IdM)和活动目录(AD)信任已安装
  详情请参阅在 IdM 和 AD 之间安装信任。
- 活动目录证书服务(ADCS)已安装，并且用户证书已生成

2.1. 智能卡验证

智能卡是一个物理设备，它可使用保存在卡中的证书提供个人验证。个人验证意味着，您可以象使用用户密码一样使用智能卡。

您可以将用户凭证以私钥和证书的形式存储在智能卡上，并使用特殊的软件和硬件来访问它们。您可以将智能卡放在读卡器或 USB 插座中，并为智能卡提供 PIN 代码，而不是提供您的密码。

您可以配置智能卡身份验证如何在特定的 IdM 客户端中工作：

- 用户可以使用用户名和密码进行身份验证，或者使用他们的智能卡进行身份验证
- 用户可以使用智能卡进行验证，并不允许使用密码进行验证
- 用户可以使用智能卡退出登录，并设置删除时的锁定功能，不允许密码

身份管理(IdM)支持使用如下方式的智能卡身份验证：

- IdM 证书颁发机构发布的用户证书。详情请参阅 为智能卡身份验证配置身份管理。
- ADCS 证书颁发机构发布的用户证书。详情请查看 为 IdM 中的智能卡身份验证配置 ADCS 发布的证书。
- 由本地证书颁发机构发布的用户证书是在 RHEL 系统上生成的。详情请参阅 配置并将本地证书导入到智能卡。
- 由外部证书颁发机构发布的用户证书。

注意

如果要开始使用智能卡验证，请参阅硬件要求：RHEL9 中的智能卡支持。

2.2. 信任配置和证书使用量所需的 WINDOWS 服务器设置

本节总结在 Windows 服务器上必须配置的内容：

- 已安装活动目录证书服务(ADCS)
- 创建证书颁发机构
- [可选] 如果您正在使用证书颁发机构 Web 注册，则必须配置互联网信息服务(IIS)

导出证书：

- 密钥必须有 **2048** 位或更多
- 包括一个私钥
- 您需要一个以下格式的证书：个人信息交换 - **PKCS #12(.PFX)**
  - 启用证书隐私

2.3. 使用 SFTP 从 ACTIVE DIRECTORY 复制证书

要能够使用智能卡身份验证，您需要复制以下证书文件：

- **CER** 格式的根 CA 证书：IdM 服务器上的 **adcs-winserver-ca.cer**。
- 具有 **PFX** 格式私钥的用户证书：IdM 客户端上的 **aduser1.pfx**。

注意

这个过程预期 SSH 访问是允许的。如果 SSH 不可用，用户必须将 文件从 AD 服务器复制到 IdM 服务器和客户端。

步骤

1. 从 **IdM 服务器** 连接，并将 **adcs-winserver-ca.cer** 根证书复制到 IdM 服务器：

```
root@idmserver ~]# sftp Administrator@winserver.ad.example.com
Administrator@winserver.ad.example.com's password:
Connected to Administrator@winserver.ad.example.com.
sftp> cd <Path to certificates>
sftp> ls
adcs-winserver-ca.cer    aduser1.pfx
sftp>
sftp> get adcs-winserver-ca.cer
Fetching <Path to certificates>/adcs-winserver-ca.cer to adcs-winserver-ca.cer
<Path to certificates>/adcs-winserver-ca.cer                100%  1254    15KB/s 00:00
sftp quit
```

2. 从 **IdM 客户端** 连接，并将 **aduser1.pfx** 用户证书复制到客户端：

```
[root@client1 ~]# sftp Administrator@winserver.ad.example.com
Administrator@winserver.ad.example.com's password:
Connected to Administrator@winserver.ad.example.com.
sftp> cd /<Path to certificates>
sftp> get aduser1.pfx
Fetching <Path to certificates>/aduser1.pfx to aduser1.pfx
<Path to certificates>/aduser1.pfx                100%  1254    15KB/s 00:00
sftp quit
```

现在，CA 证书保存在 IdM 服务器上，用户证书存储在客户端机器上。

2.4. 使用 ADCS 证书为智能卡身份验证配置 IDM 服务器和客户端

您必须配置 IdM（身份管理）服务器和客户端，以便能够在 IdM 环境中使用智能卡身份验证。IdM 包含进行了所有必要更改的 **ipa-advise** 脚本：

- 安装所需的软件包
- 它配置 IdM 服务器和客户端
- 将 CA 证书复制到预期的位置

您可以在 IdM 服务器中运行 **ipa-advise**。

这个步骤描述了：

- 在 IdM 服务器中：准备 **ipa-advise** 脚本，以配置您的 IdM 服务器进行智能卡验证。
- 在 IdM 服务器中：准备 **ipa-advise** 脚本，以配置您的 IdM 客户端进行智能卡验证。
- 在 IdM 服务器中：使用 AD 证书在 IdM 服务器中应用 **ipa-advise** 服务器脚本。
- 将客户端脚本移动到 IdM 客户端机器中。
- 在 IdM 客户端中：使用 AD 证书在 IdM 客户端中应用 **ipa-advise** 客户端脚本。

先决条件

- 证书已复制到 IdM 服务器。
- 获取 Kerberos 票据。
- 以具有管理权限的用户身份登录。

步骤

1. 在 IdM 服务器中，使用 **ipa-advise** 脚本来配置客户端：

```
[root@idmserver ~]# ipa-advise config-client-for-smart-card-auth > sc_client.sh
```

2. 在 IdM 服务器中，使用 **ipa-advise** 脚本来配置服务器：

```
[root@idmserver ~]# ipa-advise config-server-for-smart-card-auth > sc_server.sh
```

3. 在 IdM 服务器中执行脚本：

```
[root@idmserver ~]# sh -x sc_server.sh adcs-winserver-ca.cer
```

- 它配置 IdM Apache HTTP 服务器。
- 它在 KDC（Key Distribution Center）中启用 PKINIT（Public Key Cryptography for Initial Authentication in Kerberos）。
- 它将 IdM Web UI 配置为接受智能卡授权请求。

4. 将 **sc_client.sh** 脚本复制到客户端系统中：

```
[root@idmserver ~]# scp sc_client.sh root@client1.idm.example.com:/root
Password:
sc_client.sh                  100%  2857   1.6MB/s   00:00
```

5. 将 Windows 证书复制到客户端系统中：

```
[root@idmserver ~]# scp adcs-winserver-ca.cer root@client1.idm.example.com:/root
Password:
adcs-winserver-ca.cer                 100%  1254   952.0KB/s   00:00
```

6. 在客户端系统中运行客户端脚本：

```
[root@idmclient1 ~]# sh -x sc_client.sh adcs-winserver-ca.cer
```

CA 证书以正确格式安装在 IdM 服务器和客户端系统中，下一步是将用户证书复制到智能卡本身。

2.5. 转换 PFX 文件

在将 PFX(PKCS#12)文件存储到智能卡之前，您必须：

- 将文件转换为 PEM 格式
- 将私钥和证书提取到两个不同的文件

先决条件

- PFX 文件被复制到 IdM 客户端机器中。

流程

1. 在 IdM 客户端中，采用 PEM 格式：

```
[root@idmclient1 ~]# openssl pkcs12 -in aduser1.pfx -out aduser1_cert_only.pem -clcerts -
nodes
Enter Import Password:
```

2. 将密钥提取到单独的文件中：

```
[root@idmclient1 ~]# openssl pkcs12 -in adduser1.pfx -nocerts -out adduser1.pem >
aduser1.key
```

3. 将公共证书提取到单独的文件中：

```
[root@idmclient1 ~]# openssl pkcs12 -in adduser1.pfx -clcerts -nokeys -out
aduser1_cert_only.pem > aduser1.crt
```

此时，您可以将 **aduser1.key** 和 **aduser1.crt** 存储到智能卡。

2.6. 安装用来管理和使用智能卡的工具

要配置智能卡，您需要一些工具来生成证书并将其保存在智能卡中。

您必须：

- 安装 **gnutls-utils** 软件包，其帮助您管理证书。
- 安装 **opensc** 软件包，它提供一组用于智能卡的库和工具。
- 启动与智能卡读卡器通信的 **pcscd** 服务。

步骤

1. 安装 **opensc** 和 **gnutls-utils** 软件包：

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

2. 启动 **pcscd** 服务。

```
# systemctl start pcscd
```

验证 **pcscd** 服务是否已启动并运行。

2.7. 在智能卡中存储证书

本节描述了使用 **pkcs15-init** 工具配置智能卡，该工具可帮助您配置：

- 擦除智能卡
- 设置新的 PIN 和可选的 PIN Unblocking Keys（PUKs）
- 在智能卡上创建新插槽
- 在插槽存储证书、私钥和公钥
- 锁定智能卡设置（有些智能卡需要这种类型）

先决条件

- 已安装 **opensc** 软件包，其包含 **pkcs15-init** 工具。
  详情请查看安装用于管理和使用智能卡的工具。
- 该卡插入读卡器并连接到计算机。
- 您有可保存在智能卡中的私钥、公钥和证书。在此流程中，**testuser.key**、**testuserpublic.key** 和 **testuser.crt** 是用于私钥、公钥和证书的名称。
- 您当前的智能卡用户 PIN 和 Security Officer PIN（SO-PIN）

流程

1. 擦除智能卡并使用您的 PIN 验证自己：

```
$ pkcs15-init --erase-card --use-default-transport-keys
Using reader with a card: Reader name
PIN [Security Officer PIN] required.
Please enter PIN [Security Officer PIN]:
```

这个卡已经被清除。

2. 初始化智能卡，设置您的用户 PIN 和 PUK，以及您的安全响应 PIN 和 PUK：

```
$ pkcs15-init --create-pkcs15 --use-default-transport-keys \
   --pin 963214 --puk 321478 --so-pin 65498714 --so-puk 784123
Using reader with a card: Reader name
```

pcks15-init 工具在智能卡上创建一个新插槽。

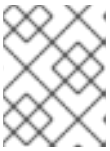
3. 为插槽设置标签和验证 ID：

```
$ pkcs15-init --store-pin --label testuser \
   --auth-id 01 --so-pin 65498714 --pin 963214 --puk 321478
Using reader with a card: Reader name
```

标签设置为人类可读的值，在本例中为 **testuser**。**auth-id** 必须是两个十六进制值，在本例中设为 **01**。

4. 在智能卡的新插槽中存储并标记私钥：

```
$ pkcs15-init --store-private-key testuser.key --label testuser_key \
   --auth-id 01 --id 01 --pin 963214
Using reader with a card: Reader name
```



注意

在存储私钥和证书时，您为 **--id** 指定的值必须相同。如果没有为 **--id** 指定值，工具会计算一个更复杂的值，因此更容易定义您自己的值。

5. 在智能卡上的新插槽中存储并标记该证书：

```
$ pkcs15-init --store-certificate testuser.crt --label testuser_crt \
   --auth-id 01 --id 01 --format pem --pin 963214
Using reader with a card: Reader name
```

6. （可选）在智能卡上新插槽中保存并标记公钥：

```
$ pkcs15-init --store-public-key testuserpublic.key
   --label testuserpublic_key --auth-id 01 --id 01 --pin 963214
Using reader with a card: Reader name
```

注意

如果公钥与私钥和/或证书对应，您应该指定与私钥和/或证书相同的 ID。

7. （可选）有些智能卡要求您通过锁定设置来完善卡：

```
$ pkcs15-init -F
```

此时您的智能卡在新创建的插槽中包含证书、私钥和公钥。您还创建了您的用户 PIN 和 PUK，以及安全响应 PIN 和 PUK。

2.8. 在 SSSD.CONF 中配置超时

使用智能卡证书进行身份验证的时间可能比 SSSD 使用的默认超时时间更长。超时时间可能是由以下原因造成的：

- 阅读速度较慢
- 一个转发将物理设备转发到到虚拟环境中
- 保存在智能卡中的证书太多
- 如果使用 OCSP 验证证书，对 OCSP（在线证书状态协议）的响应较慢

在这种情况下，您可以在 **sssd.conf** 文件中将以下超时时间延长到 60 秒：

- **p11_child_timeout**
- **krb5_auth_timeout**

先决条件

- 您必须以 root 身份登录。

步骤

1. 打开 **sssd.conf** 文件：

```
[root@idmclient1 ~]# vim /etc/sssd/sssd.conf
```

2. 更改 **p11_child_timeout** 的值：

```
[pam]
p11_child_timeout = 60
```

3. 更改 **krb5_auth_timeout** 的值：

```
[domain/IDM.EXAMPLE.COM]
krb5_auth_timeout = 60
```

4. 保存设置。

现在，在验证被认为出现超时故障前，与智能卡的交互可以运行 1 分钟（60 秒）。

2.9. 为智能卡身份验证创建证书映射规则

如果要将一个证书用于 AD(Active Directory)和 IdM 中拥有帐户的用户，您可以在 IdM 服务器上创建证书映射规则。

创建这样的规则后，用户可以在这两个域中使用智能卡进行验证。

有关证书映射规则的详情，请参阅用于在智能卡上配置身份验证的证书映射规则。

第 3 章 用于在智能卡中配置身份验证的证书映射规则

证书映射规则是允许用户在 Identity Management(IdM)管理员无法访问某些用户证书时使用证书进行身份验证的方法。不足的访问权限通常是由证书由外部证书颁发机构发布的事实造成的。一个特殊的用例由 IdM 域在信任关系中的 Active Directory(AD)的证书系统发布。

如果 IdM 环境较大且有大量使用智能卡的用户，使用证书映射规则就会比较方便。在这种情况下，添加完整证书可能会比较复杂。在大多数场景中，主题和发行者都是可预测的，因此提前添加的时间比完整证书更容易。作为系统管理员，您可以创建证书映射规则，并在向特定用户签发证书前向用户条目添加证书映射数据。签发证书后，用户就可以使用证书登录，即使证书还没有上传到用户条目。

另外，因为必须定期续订证书，证书映射规则减少了管理开销。用户的证书续订时，管理员必须更新用户条目。例如，如果映射基于 **Subject** 和 **Issuer** 的值，如果新的证书具有与旧证书相同的主题和签发者，则映射仍适用。如果使用完整证书，则管理员必须将新证书上传到用户条目以替换旧证书。

设置证书映射：

1. 管理员必须将证书映射数据（通常是签发者和主体）或完整证书加载到用户帐户中。
2. 管理员必须创建证书映射规则，允许用户成功登录到 IdM
   a. 其帐户包含证书映射数据条目
   b. 哪个证书映射数据条目与证书的信息匹配

   有关构成映射规则的单独组件，以及如何获取和使用它们的详细信息，请参阅 IdM 中的身份映射规则组件，以及获取证书中的签发者，以便在匹配规则中使用。

之后，当最终用户提供存储在 文件系统 或 智能卡 上的证书时，身份验证成功。

3.1. 使用 ACTIVE DIRECTORY 域信任的证书映射规则

本节概述了在与 Active Directory(AD)域的信任关系时可能出现的不同证书映射用例。

证书映射规则是为具有可信 AD 证书系统发布的智能卡证书的用户启用 IdM 资源的便捷方法。根据 AD 配置，可能会出现以下情况：

- 如果证书由 AD 发出，但用户和证书存储在 IdM 中，那么身份验证请求的映射和整个处理都位于 IdM 端。有关配置此情境的详情，请参阅为存储在 IdM 中的用户配置证书映射
- 如果用户存储在 AD 中，则身份验证请求的处理发生在 AD 中。有三个不同的子案例：
  - AD 用户条目包含整个证书。有关在这种情况下配置 IdM 的详情，请参考为 AD 用户条目包含整个证书的用户配置证书映射。
  - AD 配置为将用户证书映射到用户帐户。在这种情况下，AD 用户条目不包含整个证书，而是包含名为 **altSecurityIdentities** 的属性。有关如何在这种场景中配置 IdM 的详情，请参阅在将 AD 配置为将用户证书映射到用户帐户时配置证书 映射。
  - AD 用户条目既不包含整个证书也不包括映射数据。在这种情况下，唯一解决方案是使用 **ipa idoverrideuser-add** 命令将整个证书添加到 IdM 中的 AD 用户 ID 覆盖中。详情请参阅 在 AD 用户条目不包含证书或映射数据时配置证书映射。

3.2. IDM 中身份映射规则的组件

本节论述了 IdM 中 *身份映射规则* 的组件以及如何配置它们。每个组件都有一个可覆盖的默认值。您可以在 Web UI 或 CLI 中定义这些组件。在 CLI 中，身份映射规则是使用 **ipa certmaprule-add** 命令创建的。

映射规则

映射规则组件将（或 *映射*）证书与一个或多个用户帐户相关联。规则定义了一个 LDAP 搜索过滤器，用于将证书与预期用户帐户相关联。
由不同证书颁发机构(CA)发布的证书可能具有不同的属性，可能在不同的域中使用。因此，IdM 不会以无条件的方式应用映射规则，而是只应用于适当的证书。适当的证书是使用 *匹配规则* 定义的。

请注意，如果您将映射规则选项留空，则会在 **userCertificate** 属性中搜索证书作为编码的二进制文件。

在 CLI 中使用 **--maprule** 选项定义映射规则。

匹配规则

匹配的规则组件选择您要应用映射规则的证书。默认匹配规则与带有 **digitalSignature key** 使用和 **clientAuth extended key** 使用的证书匹配。
使用 **--matchrule** 选项在 CLI 中定义匹配的规则。

域列表

域列表指定您希望 IdM 在处理身份映射规则时搜索用户的身份域。如果您未指定选项，IdM 将仅在 IdM 客户端所属的本地域中搜索用户。
使用 **--domain** 选项在 CLI 中定义域。

优先级

当多个规则适用于证书时，具有最高优先级的规则将具有优先权。所有其他规则将被忽略。

- 数字值越低，身份映射规则的优先级越高。例如，具有优先级 1 的规则的优先级高于优先级 2 的规则。
- 如果规则没有定义优先级值，它具有最低的优先级。

使用 **--priority** 选项在 CLI 中定义映射规则优先级。

证书映射规则示例

要定义，使用 CLI，一个被称为 **simple_rule** 的证书映射规则，允许使用 **EXAMPLE.ORG** 机构的 **Smart Card CA** 签发的证书进行验证，只要该证书上的 **Subject** 与 IdM 中用户帐户中的 **certmapdata** 条目匹配：

```
# ipa certmaprule-add simple_rule --matchrule '<ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG' --maprule '(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})'
```

3.3. 从匹配规则中使用的证书获取签发者

此流程描述了如何从证书获取签发者信息，以便将其复制并粘贴到证书映射规则的匹配规则中。要获得匹配的规则所需的签发者格式，请使用 **openssl x509** 实用程序。

先决条件

- 您有 **.pem** 或 **.crt** 格式的用户证书

步骤

1. 从证书获取用户信息。使用 **openssl x509** 证书显示和签名工具：

   - 用于阻止编码版本的请求输出的 **-noout** 选项
   - 输出签发者名称的 **-issuer** 选项
   - 用来读取证书的输入文件名的 **-in** 选项
   - 带有 **RFC2253** 值的 **-nameopt** 选项会首先显示使用最具体的相对可分辨名称(RDN)。
     如果输入文件包含 Identity Management 证书，命令的输出将使用 **Organisation** 信息定义 Issuer：

     ```
     # openssl x509 -noout -issuer -in idm_user.crt -nameopt RFC2253
     issuer=CN=Certificate Authority,O=REALM.EXAMPLE.COM
     ```

     如果输入文件包含 Active Directory 证书，命令的输出将使用 **Domain** 组件信息来定义 Issuer：

     ```
     # openssl x509 -noout -issuer -in ad_user.crt -nameopt RFC2253
     issuer=CN=AD-WIN2012R2-CA,DC=AD,DC=EXAMPLE,DC=COM
     ```

2. 另外，要根据匹配规则在 CLI 中创建一个新的映射规则，该规则指定证书签发者必须是 **ad.example.com** 域的 **AD-WIN2012R2-CA**，证书上的主题必须与 IdM 中用户帐户中的 **certmapdata** 条目匹配：

   ```
   # ipa certmaprule-add simple_rule --matchrule '<ISSUER>CN=AD-WIN2012R2-
   CA,DC=AD,DC=EXAMPLE,DC=COM' --maprule '(ipacertmapdata=X509:<I>
   {issuer_dn!nss_x500}<S>{subject_dn!nss_x500})'
   ```

3.4. 其他资源

- 请参阅 **sss-certmap(5)** man page。

第 4 章 配置和导入本地证书到智能卡

本章描述了以下情况：

- 主机没有连接到某个域。
- 您需要在这个主机上使用智能卡进行验证。
- 您需要使用智能卡验证配置 SSH 访问。
- 您需要使用 **authselect** 配置智能卡。

使用以下配置来实现这种情况：

- 为希望使用智能卡进行身份验证的用户获取用户证书。证书应该由在域中使用的可信认证认证机构生成。
  如果您无法获得证书，您可以生成由本地证书颁发机构签名的用户证书用于测试。
- 在智能卡中保存证书和私钥。
- 为 SSH 访问配置智能卡验证。

重要

如果主机可以作为域的一部分，将主机添加到域中，并使用活动目录或者身份管理认证机构生成的证书。

有关如何为智能卡创建 IdM 证书的详情，请参考为智能卡验证配置身份管理。

先决条件

- 已安装 authselect
  authselect 工具在 Linux 主机中配置用户验证，您可以使用它配置智能卡验证参数。有关 authselect 的详情，请参考浏览 authselect。
- RHEL 9 支持的智能卡或者 USB 设备
  详情请查看 RHEL9 中的智能卡支持。

4.1. 创建本地证书

本节论述了如何执行这些任务：

- 生成 OpenSSL 证书颁发机构
- 创建证书签名请求

警告

以下步骤仅用于测试目的。由本地自签名证书颁发机构生成的证书不如使用 AD、IdM 或 RHCS 认证认证机构的安全。即使主机不是域的一部分，您仍应使用企业认证认证机构生成的证书。

流程

1. 创建可生成证书的目录，例如：

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

2. 设置证书（将此文本复制到 **ca** 目录中的命令行）：

```
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
email_in_dn     = no

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. 创建以下目录：

```
# mkdir certs crl newcerts
```

4. 创建以下文件：

```
# touch index.txt crlnumber index.txt.attr
```

5. 在串行文件中写入数字 01:

```
# echo 01 > serial
```

该命令在串行文件中写入数字 01。它是证书的序列号。当这个 CA 发布一个新证书时这个数字会加一。

6. 创建一个 OpenSSL root CA 密钥：

```
# openssl genrsa -out rootCA.key 2048
```

7. 创建自签名 root 认证认证机构证书：

```
# 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. 为您的用户名创建密钥：

```
# openssl genrsa -out example.user.key 2048
```

这个密钥是在本地系统中生成的，因此当密钥保存在卡中时，从系统中删除密钥。

您还可以直接在智能卡中创建密钥。要做到这一点，请遵循智能卡生产商生成的说明。

9. 创建证书签名请求配置文件（将这个文本复制到 ca 目录中的命令行中）：

```
cat > req.cnf <<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"
subjectKeyIdentifier = hash
keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = clientAuth, emailProtection, msSmartcardLogin
subjectAltName = otherName:msUPN;UTF8:testuser@EXAMPLE.COM,
email:testuser@example.com
EOF
```

10. 为 example.user 证书创建证书签名请求：

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

11. 配置新证书。过期期限设定为 1 年：

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

此时，认证颁发机构和证书被成功生成并准备好导入到智能卡。

4.2. 将证书复制到 SSSD 目录中

GNOME 桌面管理器(GDM)需要 SSSD。如果使用 GDM，则需要将 PEM 证书复制到 **/etc/sssd/pki** 目录。

先决条件

- 已生成本地 CA 颁发机构和证书

流程

1. 确保已在系统中安装了 SSSD。

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

2. 创建 **/etc/sssd/pki** 目录：

```
# file /etc/sssd/pki
/etc/sssd/pki/: directory
```

3. 将 **rootCA.crt** 作为 PEM 文件复制到 **/etc/sssd/pki/** 目录中：

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

现在，您已成功生成了证书颁发机构和证书，并将其保存在 **/etc/sssd/pki** 目录中。

注意

如果要与另一个应用程序共享证书颁发机构证书，您可以在 sssd.conf 中更改位置：

- SSSD PAM 响应器： **[pam]** 部分中的 **pam_cert_db_path**
- SSSD ssh 响应器：**[ssh]** 部分中的 **ca_db**

详情请查看 **sssd.conf** 的 man page。

红帽建议保留默认路径，并为 SSSD 使用专用证书颁发机构证书文件来确保此处仅列出可信的证书颁发机构。

4.3. 安装用来管理和使用智能卡的工具

要配置智能卡，您需要一些工具来生成证书并将其保存在智能卡中。

您必须：

- 安装 **gnutls-utils** 软件包，其帮助您管理证书。
- 安装 **opensc** 软件包，它提供一组用于智能卡的库和工具。
- 启动与智能卡读卡器通信的 **pcscd** 服务。

步骤

1. 安装 **opensc** 和 **gnutls-utils** 软件包：

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

2. 启动 **pcscd** 服务。

```
# systemctl start pcscd
```

验证 **pcscd** 服务是否已启动并运行。

4.4. 在智能卡中存储证书

本节描述了使用 **pkcs15-init** 工具配置智能卡，该工具可帮助您配置：

- 擦除智能卡
- 设置新的 PIN 和可选的 PIN Unblocking Keys（PUKs）
- 在智能卡上创建新插槽
- 在插槽存储证书、私钥和公钥
- 锁定智能卡设置（有些智能卡需要这种类型）

先决条件

- 已安装 **opensc** 软件包，其包含 **pkcs15-init** 工具。
  详情请查看安装用于管理和使用智能卡的工具。
- 该卡插入读卡器并连接到计算机。
- 您有可保存在智能卡中的私钥、公钥和证书。在此流程中，**testuser.key**、**testuserpublic.key** 和 **testuser.crt** 是用于私钥、公钥和证书的名称。
- 您当前的智能卡用户 PIN 和 Security Officer PIN（SO-PIN）

流程

1. 擦除智能卡并使用您的 PIN 验证自己：

```
$ pkcs15-init --erase-card --use-default-transport-keys
Using reader with a card: Reader name
PIN [Security Officer PIN] required.
Please enter PIN [Security Officer PIN]:
```

   这个卡已经被清除。

2. 初始化智能卡，设置您的用户 PIN 和 PUK，以及您的安全响应 PIN 和 PUK：

```
$ pkcs15-init --create-pkcs15 --use-default-transport-keys \
    --pin 963214 --puk 321478 --so-pin 65498714 --so-puk 784123
Using reader with a card: Reader name
```

   pcks15-init 工具在智能卡上创建一个新插槽。

3. 为插槽设置标签和验证 ID：

```
$ pkcs15-init --store-pin --label testuser \
    --auth-id 01 --so-pin 65498714 --pin 963214 --puk 321478
Using reader with a card: Reader name
```

   标签设置为人类可读的值，在本例中为 **testuser**。**auth-id** 必须是两个十六进制值，在本例中设为 **01**。

4. 在智能卡的新插槽中存储并标记私钥：

```
$ pkcs15-init --store-private-key testuser.key --label testuser_key \
    --auth-id 01 --id 01 --pin 963214
Using reader with a card: Reader name
```

注意

在存储私钥和证书时，您为 **--id** 指定的值必须相同。如果没有为 **--id** 指定值，工具会计算一个更复杂的值，因此更容易定义您自己的值。

5. 在智能卡上的新插槽中存储并标记该证书：

```
$ pkcs15-init --store-certificate testuser.crt --label testuser_crt \
    --auth-id 01 --id 01 --format pem --pin 963214
Using reader with a card: Reader name
```

6. （可选）在智能卡上新插槽中保存并标记公钥：

```
$ pkcs15-init --store-public-key testuserpublic.key
   --label testuserpublic_key --auth-id 01 --id 01 --pin 963214
Using reader with a card: Reader name
```

注意

如果公钥与私钥和/或证书对应，您应该指定与私钥和/或证书相同的 ID。

7. （可选）有些智能卡要求您通过锁定设置来完善卡：

```
$ pkcs15-init -F
```

此时您的智能卡在新创建的插槽中包含证书、私钥和公钥。您还创建了您的用户 PIN 和 PUK，以及安全响应 PIN 和 PUK。

4.5. 使用智能卡验证配置 SSH 访问

SSH 连接需要身份验证。您可以使用密码或证书。本节描述：

- 使用保存在智能卡中的证书启用验证所需的配置
- 使用 **authselect** 工具在移除配置时进行锁定

取出卡时进行锁定会强制在智能卡被取出后注销用户的登陆。

有关使用 **authselect** 配置智能卡的详情，请参考使用 authselect 配置智能卡。

先决条件

- 该智能卡包含您的证书和私钥。
- 该卡插入读卡器并连接到计算机。
- 已安装并配置了 SSSD。
- 您的用户名与证书的 SUBJECT 中的通用名称(CN)或用户 ID(UID)匹配。
- **pcscd** 服务正在您的本地计算机上运行。
  详情请查看安装用于管理和使用智能卡的工具。

流程

1. 在使用智能卡验证的用户主目录中为 SSH 密钥创建新目录：

```
# mkdir /home/example.user/.ssh
```

2. 使用 **opensc** 库运行 **ssh-keygen -D** 命令以使用智能卡中的私钥检索现有公钥对，并将其添加到用户的 SSH 密钥目录的 **authorized_keys** 列表中，以便通过智能卡验证启用 SSH 访问。

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

3. SSH 需要访问 **/.ssh** 目录和 **authorized_keys** 文件的适当配置。要设置或更改访问权限，请输入：

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

4. 另外，显示密钥：

```
# cat ~example.user/.ssh/authorized_keys
```

终端会显示密钥。

5. 验证 **/etc/sssd/sssd.conf** 文件中是否启用了智能卡验证：
在 **[pam]** 部分中，启用 pam 证书验证模块： **pam_cert_auth = True**

如果尚未创建 **sssd.conf** 文件，您可以通过将以下脚本复制到命令行来创建最小功能配置：

```
# cat > /etc/sssd/sssd.conf <<EOF
[sssd]
services = nss, pam
domains = shadowutils

[nss]

[pam]
pam_cert_auth = True

[domain/shadowutils]
id_provider = files
EOF
```

6. 要使用 SSH 密钥，请使用 **authselect** 命令配置身份验证：

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

现在，您可以使用以下命令验证 SSH 访问：

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

如果配置成功，会提示您输入智能卡 PIN。

这个配置现在可以在本地运行。现在，您可以复制公钥并将其分发到您要使用 SSH 的所有服务器中的 **authorized_keys** 文件。

第 5 章 使用 AUTHSELECT 配置智能卡

这部分论述了如何配置智能卡以达到以下目的之一：

- 启用密码和智能卡验证
- 禁用密码并启用智能卡验证
- 在删除时启用锁定

先决条件

- 已安装 authselect
  authselect 工具在 Linux 主机中配置用户验证，您可以使用它配置智能卡验证参数。有关 authselect 的详情，请参考使用 authselect 配置用户身份验证。
- RHEL 9 支持的智能卡或者 USB 设备
  详情请查看 RHEL9 中的智能卡支持。

5.1. 适用于智能卡的证书

在使用 **authselect** 配置智能卡前，您必须将证书导入您的卡中。您可以使用以下工具生成证书：

- Active Directory(AD)
- 身份管理(IdM)
  有关如何创建 IdM 证书的详情，请参阅请求新用户证书并将其导出到客户端。
- 红帽认证系统(RHCS)
  详情请参阅使用企业安全客户端管理智能卡。
- 本地认证认证机构。如果用户不是某个域的一部分或用于测试,您可以使用本地认证认证机构生成的证书。
  有关如何创建并导入本地证书到智能卡的详情，请参阅配置和导入本地证书到智能卡。

5.2. 启用用户密码验证来配置智能卡验证

这部分论述了如何在您的系统中启用智能卡和密码验证。

先决条件

- 智能卡包含您的证书和私钥。
- 在读卡器中插入卡并连接到计算机。
- **authselect** 工具已安装在您的系统中。

步骤

- 输入以下命令允许智能卡和密码验证：

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

此时，智能卡验证会被启用。但是如果您忘记携带了智能卡，密码验证仍可以正常工作。

5.3. 配置 AUTHSELECT 以强制智能卡验证

authselect 工具可让您在系统中配置智能卡验证，并禁用默认密码验证。**authselect** 命令包括以下选项：

- **with-smartcard** -- 除了密码验证外，启用智能卡验证
- **with-smartcard-required** – 启用智能卡验证，禁用密码验证

注意

with-smartcard-required 选项只对使用智能卡进行特定登录服务时才强制执行，如 **login**、**gdm**、**xdm**、**kdm**、**xscreensaver**、**gnome-screensaver** 和 **kscreensaver**。其他服务（如 **su** 或 **sudo**）默认情况下不使用智能卡验证，并将继续提示您输入密码。

先决条件

- 智能卡包含您的证书和私钥。
- 在读卡器中插入卡并连接到计算机。
- **authselect** 工具安装在您的本地系统中。

步骤

- 输入以下命令强制智能卡验证：

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

注意

运行此命令后，密码验证将无法正常工作，您只能使用智能卡登录。在运行此命令之前，确保智能卡验证正在运行，或者您会被锁定于您的系统。

5.4. 配置智能卡认证，使它在取出智能卡时进行锁定

authselect 服务可让您配置智能卡验证，以便在从读取器中删除智能卡后立即锁定屏幕。**authselect** 命令必须包括以下变量：

- **with-smartcard** – 启用智能卡验证
- **with-smartcard-required** - 启用专用智能卡验证（禁用密码验证）
- **with-smartcard-lock-on-removal** – 在智能卡被取出后会强制登出

先决条件

- 智能卡包含您的证书和私钥。
- 在读卡器中插入卡并连接到计算机。
- **authselect** 工具安装在您的本地系统中。

步骤

- 输入以下命令启用智能卡验证、禁用密码验证并在删除时强制锁定：

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

现在，当您取出卡时，屏幕会锁定。您必须重新插入智能卡来解锁它。

第 6 章 使用智能卡进行远程向 SUDO 进行身份验证

这部分论述了如何使用智能卡远程对 sudo 进行身份验证。在 **ssh-agent** 服务本地运行且无法将 **ssh-agent** 套接字转发到远程机器后，您可以使用 sudo PAM 模块中的 SSH 身份验证协议进行远程验证。

在使用智能卡进行本地登录后，您可以通过 SSH 登录到远程机器并运行 **sudo** 命令，而无需使用智能卡验证的 SSH 转发来提示输入密码。

在本示例中，客户端通过 SSH 连接到 IPA 服务器，并在 IPA 服务器中使用保存在智能卡中的凭证运行 sudo 命令。

- 在 IdM 中创建 sudo 规则
- 为 sudo 设置 PAM 模块
- 使用智能卡远程连接到 sudo

6.1. 在 IDM 中创建 SUDO 规则

此流程描述了如何在 IdM 中创建 sudo 规则，以便 **ipauser1** 权限在远程主机上运行 sudo。

在本示例中，**less** 和 **whoami** 命令被添加为 sudo 命令来测试该流程。

先决条件

- IdM 用户已创建。在本例中，用户名为 **ipauser1**。
- 您有远程运行 sudo 的系统的主机名。在本示例中，主机是 **server.ipa.test**。

步骤

1. 创建名为 **adminrule** 的 **sudo** 规则，允许用户运行命令。

```
ipa sudorule-add adminrule
```

2. 添加 **less** 和 **whoami** 作为 **sudo** 命令：

```
ipa sudocmd-add /usr/bin/less
ipa sudocmd-add /usr/bin/whoami
```

3. 将 **less** 和 **whoami** 命令添加到 **adminrule** 中：

```
ipa sudorule-add-allow-command adminrule --sudocmds /usr/bin/less
ipa sudorule-add-allow-command adminrule --sudocmds /usr/bin/whoami
```

4. 将 **ipauser1** 用户添加到 **adminrule** 中：

```
ipa sudorule-add-user adminrule --users ipauser1
```

5. 将运行 **sudo** 的主机添加到 **adminrule** 中：

```
ipa sudorule-add-host adminrule --hosts server.ipa.test
```

其他资源

- 请参阅 **ipa sudorule-add --help**。
- 请参阅 **ipa sudocmd-add --help**。

6.2. 为 SUDO 设置 PAM 模块

这个步骤描述了如何在运行 sudo 的任何主机上安装和设置 **pam_ssh_agent_auth.so** PAM 模块进行 sudo 身份验证。

步骤

1. 安装 PAM SSH 代理：

```
dnf -y install pam_ssh_agent_auth
```

2. 在任何其他 **auth** 条目之前，将 **pam_ssh_agent_auth.so** 添加到 **/etc/pam.d/sudo** 文件的 **authorized_keys_command**:

```
#%PAM-1.0
auth sufficient pam_ssh_agent_auth.so
authorized_keys_command=/usr/bin/sss_ssh_authorizedkeys
auth       include      system-auth
account    include      system-auth
password   include      system-auth
session    include      system-auth
```

3. 要在运行 sudo 命令时启用 SSH 代理转发功能，请将以下内容添加到 **/etc/sudoers** 文件中：

```
Defaults env_keep += "SSH_AUTH_SOCK"
```

这允许从存储在 IPA/SSSD 中的智能卡中的公钥在无需输入密码的情况下向 sudo 进行身份验证。

4. 重启 **sssd** 服务：

```
systemctl restart sssd
```

其他资源

- 请参阅 **pam** man page。

6.3. 使用智能卡远程连接到 SUDO

这个步骤描述了如何配置 SSH 代理和客户端，以便使用智能卡远程连接到 sudo。

先决条件

- 您已在 IdM 中创建了 sudo 规则。
- 已安装并设置 **pam_ssh_agent_auth** PAM 模块，用于在您要运行 sudo 的远程系统中进行 sudo 身份验证。

步骤

1. 启动 SSH 代理（如果尚未运行）。

```
eval `ssh-agent`
```

2. 将您的智能卡添加到 SSH 代理。提示时输入您的 PIN：

```
ssh-add -s /usr/lib64/opensc-pkcs11.so
```

3. 通过启用了 ssh-agent 转发的 SSH 连接（使用 **-A** 选项）连接到您要远程运行 **sudo** 的系统：

```
ssh -A ipauser1@server.ipa.test
```

验证步骤

- 使用 **sudo** 运行 **whoami** 命令：

```
sudo /usr/bin/whoami
```

不应该提示您输入 PIN 或密码。

第 7 章 使用智能卡对身份验证进行故障排除

下面的部分描述了如何在设置智能卡验证时解决您可能遇到的一些问题。

- 测试智能卡验证
- 使用 SSSD 对智能卡验证进行故障排除
- 验证 IdM Kerberos KDC 可以使用 PKINIT 和 CA 证书正确位置
- 增加 SSSD 超时
- 证书映射和匹配规则故障排除

7.1. 测试系统中的智能卡访问

这个步骤描述了如何测试是否可以访问智能卡。

先决条件

- 已安装并配置了用于智能卡的 IdM 服务器和客户端。
- 您已从 **nss-tools** 软件包安装了 **certutil** 工具。
- 您有智能卡的 PIN 或密码。

步骤

1. 使用 **lsusb** 命令，验证智能卡读取器是否在操作系统中看到：

```
$ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 072f:b100 Advanced Card Systems, Ltd ACR39U
Bus 001 Device 002: ID 0627:0001 Adomax Technology Co., Ltd
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

   有关 RHEL 中经过测试和支持的智能卡和读卡的更多信息，请参阅 RHEL 9 中的智能卡支持。

2. 确保 **pcscd** 服务和套接字已启用并正在运行：

```
$ systemctl status pcscd.service pcscd.socket

● pcscd.service - PC/SC Smart Card Daemon
     Loaded: loaded (/usr/lib/systemd/system/pcscd.service; indirect;
vendor preset: disabled)
     Active: active (running) since Fri 2021-09-24 11:05:04 CEST; 2
weeks 6 days ago
TriggeredBy: ● pcscd.socket
       Docs: man:pcscd(8)
   Main PID: 3772184 (pcscd)
      Tasks: 12 (limit: 38201)
     Memory: 8.2M
        CPU: 1min 8.067s
     CGroup: /system.slice/pcscd.service
             └─3772184 /usr/sbin/pcscd --foreground --auto-exit
```

```
● pcscd.socket - PC/SC Smart Card Daemon Activation Socket
     Loaded: loaded (/usr/lib/systemd/system/pcscd.socket; enabled;
vendor preset: enabled)
     Active: active (running) since Fri 2021-09-24 11:05:04 CEST; 2
weeks 6 days ago
   Triggers: ● pcscd.service
     Listen: /run/pcscd/pcscd.comm (Stream)
     CGroup: /system.slice/pcscd.socket
```

3. 使用 **p11-kit list-modules** 命令，显示有关配置的智能卡和智能卡中的令牌的信息：

```
$ p11-kit list-modules
p11-kit-trust: p11-kit-trust.so
[...]
opensc: opensc-pkcs11.so
    library-description: OpenSC smartcard framework
    library-manufacturer: OpenSC Project
    library-version: 0.20
    token: MyEID (sctest)
        manufacturer: Aventra Ltd.
        model: PKCS#15
        serial-number: 8185043840990797
        firmware-version: 40.1
        flags:
              rng
              login-required
              user-pin-initialized
              token-initialized
```

4. 验证您可以访问智能卡的内容：

```
$ pkcs11-tool --list-objects --login
Using slot 0 with a present token (0x0)
Logging in to "MyEID (sctest)".
Please enter User PIN:
Private Key Object; RSA
  label:      Certificate
  ID:         01
  Usage:      sign
  Access:     sensitive
Public Key Object; RSA 2048 bits
  label:      Public Key
  ID:         01
  Usage:      verify
  Access:     none
Certificate Object; type = X.509 cert
  label:      Certificate
  subject:    DN: O=IDM.EXAMPLE.COM, CN=idmuser1
  ID:         01
```

5. 使用 **certutil** 命令显示智能卡中的证书内容：

    a. 运行以下命令来确定证书的正确名称：

```
$ certutil -d /etc/pki/nssdb -L -h all

Certificate Nickname                                         Trust Attributes
                                                             SSL,S/MIME,JAR/XPI

Enter Password or Pin for "MyEID (sctest)":
Smart Card CA 0f5019a8-7e65-46a1-afe5-8e17c256ae00           CT,C,C
MyEID (sctest):Certificate                                   u,u,u
```

b. 在智能卡中显示证书内容：

注意

确保证书的名称是与上一步中显示的输出完全匹配，在本例中为 **MyEID(sctest):Certificate**。

```
$ certutil -d /etc/pki/nssdb -L -n "MyEID (sctest):Certificate"

Enter Password or Pin for "MyEID (sctest)":
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 15 (0xf)
        Signature Algorithm: PKCS #1 SHA-256 With RSA Encryption
        Issuer: "CN=Certificate Authority,O=IDM.EXAMPLE.COM"
        Validity:
            Not Before: Thu Sep 30 14:01:41 2021
            Not After : Sun Oct 01 14:01:41 2023
        Subject: "CN=idmuser1,O=IDM.EXAMPLE.COM"
        Subject Public Key Info:
            Public Key Algorithm: PKCS #1 RSA Encryption
            RSA Public Key:
                Modulus:
                    [...]
                Exponent: 65537 (0x10001)
        Signed Extensions:
            Name: Certificate Authority Key Identifier
            Key ID:
                e2:27:56:0d:2f:f5:f2:72:ce:de:37:20:44:8f:18:7f:
                2f:56:f9:1a

            Name: Authority Information Access
            Method: PKIX Online Certificate Status Protocol
            Location:
                URI: "http://ipa-ca.idm.example.com/ca/ocsp"

            Name: Certificate Key Usage
            Critical: True
            Usages: Digital Signature
                    Non-Repudiation
                    Key Encipherment
                    Data Encipherment

            Name: Extended Key Usage
```

```
            TLS Web Server Authentication Certificate
            TLS Web Client Authentication Certificate

        Name: CRL Distribution Points
        Distribution point:
            URI: "http://ipa-ca.idm.example.com/ipa/crl/MasterCRL.bin"
            CRL issuer:
                Directory Name: "CN=Certificate Authority,O=ipaca"

        Name: Certificate Subject Key ID
        Data:
            43:23:9f:c1:cf:b1:9f:51:18:be:05:b5:44:dc:e6:ab:
            be:07:1f:36

    Signature Algorithm: PKCS #1 SHA-256 With RSA Encryption
    Signature:
        [...]
    Fingerprint (SHA-256):

6A:F9:64:F7:F2:A2:B5:04:88:27:6E:B8:53:3E:44:3E:F5:75:85:91:34:ED:48:A8:0D:F0:31:5
D:7B:C9:E0:EC
    Fingerprint (SHA1):
        B4:9A:59:9F:1C:A8:5D:0E:C1:A2:41:EC:FD:43:E0:80:5F:63:DF:29

    Mozilla-CA-Policy: false (attribute missing)
    Certificate Trust Flags:
        SSL Flags:
            User
        Email Flags:
            User
        Object Signing Flags:
            User
```

其他资源

- 请参阅 **certutil(1)** man page。

7.2. 使用 SSSD 对智能卡验证进行故障排除

这个步骤描述了如何使用智能卡使用 SSSD 对身份验证进行故障排除。

先决条件

- 已安装并配置了用于智能卡的 IdM 服务器和客户端。
- 已安装 **sssd-tools** 软件包。
- 您可以检测智能卡读取器并显示智能卡的内容。请参阅在系统上测试智能卡访问。

步骤

1. 使用 **su** 验证您可以使用智能卡进行验证：

```
$ su - idmuser1 -c 'su - idmuser1 -c whoami'
PIN for MyEID (sctest):
idmuser1
```

如果没有提示输入智能卡 PIN，且返回一个密码提示或者返回授权错误，请检查 SSSD 日志。有关在 SSSD 中登录的信息，请参阅 IdM 中的 SSSD 对身份验证进行故障排除 。以下是身份验证失败的示例：

```
$ su - idmuser1 -c 'su - idmuser1 -c whoami'
PIN for MyEID (sctest):
su: Authentication failure
```

如果 SSSD 日志指明了 **krb5_child** 的问题，类似于以下内容，则可能对您的 CA 证书有问题。要排除与证书相关的问题，请参阅 验证 IdM Kerberos KDC 可以使用 Pkinit 以及 CA 证书正确位于。

```
[Pre-authentication failed: Failed to verify own certificate (depth 0): unable to get local issuer certificate: could not load the shared library]
```

如果 SSSD 日志表示来自 **p11_child** 或 **krb5_child** 的超时，您可能需要提高 SSSD 超时，并尝试使用智能卡再次进行身份验证。有关如何增加超时的详情，请参阅增加 SSSD 超时。

2. 验证您的 GDM 智能卡验证配置是否正确。应返回 PAM 验证的成功消息，如下所示：

```
# sssctl user-checks -s gdm-smartcard "idmuser1" -a auth
user: idmuser1
action: auth
service: gdm-smartcard

SSSD nss user lookup result:
 - user name: idmuser1
 - user id: 603200210
 - group id: 603200210
 - gecos: idm user1
 - home directory: /home/idmuser1
 - shell: /bin/sh

SSSD InfoPipe user lookup result:
 - name: idmuser1
 - uidNumber: 603200210
 - gidNumber: 603200210
 - gecos: idm user1
 - homeDirectory: /home/idmuser1
 - loginShell: /bin/sh

testing pam_authenticate

PIN for MyEID (sctest)
pam_authenticate for user [idmuser1]: Success

PAM Environment:
 - PKCS11_LOGIN_TOKEN_NAME=MyEID (sctest)
 - KRB5CCNAME=KCM:
```

如果身份验证错误（类似于以下内容）被返回，请检查 SSSD 日志尝试并确定导致这个问题的原因。有关在 SSSD 中登录的信息，请参阅 IdM 中的 SSSD 对身份验证进行故障排除 。

```
pam_authenticate for user [idmuser1]: Authentication failure

PAM Environment:
 - no env -
```

如果 PAM 验证仍失败，请清除您的缓存并再次运行命令。

```
# sssctl cache-remove
SSSD must not be running. Stop SSSD now? (yes/no) [yes] yes
Creating backup of local data…
Removing cache files…
SSSD needs to be running. Start SSSD now? (yes/no) [yes] yes
```

7.3. 验证 IDM KERBEROS KDC 可以使用 PKINIT 和 CA 证书正确位置

这个步骤描述了如何验证 IdM Kerberos KDC 可以使用 PKINIT，并描述了如何正确验证您的 CA 证书。

先决条件

- 已安装并配置了用于智能卡的 IdM 服务器和客户端。
- 您可以检测智能卡读取器并显示智能卡的内容。请参阅在系统上测试智能卡访问。

步骤

1. 运行 **kinit** 工具，以 **idmuser1** 使用保存在智能卡中的证书进行验证：

```
$ kinit -X X509_user_identity=PKCS11: idmuser1
MyEID (sctest)                  PIN:
```

2. 输入您的智能卡 PIN。如果没有提示输入 PIN，请检查您是否可检测智能卡读取器并显示智能卡的内容。请参阅测试智能卡验证。

3. 如果您的 PIN 被接受，系统会提示您输入密码，可能会缺少您的 CA 签名证书。

   a. 使用 **openssl** 命令，验证默认证书捆绑包文件中列出的 CA 链：

```
$ openssl crl2pkcs7 -nocrl -certfile /var/lib/ipa-client/pki/ca-bundle.pem | openssl pkcs7 -print_certs -noout
subject=O = IDM.EXAMPLE.COM, CN = Certificate Authority

issuer=O = IDM.EXAMPLE.COM, CN = Certificate Authority
```

   b. 验证您的证书的有效性：

      i. 查找 **idmuser1** 的用户身份验证证书 ID：

```
$ pkcs11-tool --list-objects --login
[...]
Certificate Object; type = X.509 cert
```

```
  label:      Certificate
  subject:    DN: O=IDM.EXAMPLE.COM, CN=idmuser1
ID: 01
```

ii. 从智能卡以 DER 格式读取用户证书信息：

```
$ pkcs11-tool --read-object --id 01 --type cert --output-file cert.der
Using slot 0 with a present token (0x0)
```

iii. 将 DER 证书转换为 PEM 格式：

```
$ openssl x509 -in cert.der -inform DER -out cert.pem -outform PEM
```

iv. 验证证书是否有有效的签发者签名到 CA：

```
$ openssl verify -CAfile /var/lib/ipa-client/pki/ca-bundle.pem <path>/cert.pem
cert.pem: OK
```

4. 如果您的智能卡包含多个证书，**kinit** 可能无法选择正确的证书进行验证。在这种情况下，您需要使用 **certid=<ID>** 选项将证书 ID 指定为 **kinit** 命令的参数。

a. 检查保存在智能卡中的证书数量，并获取您要使用的证书 ID：

```
$ pkcs11-tool --list-objects --type cert --login
Using slot 0 with a present token (0x0)
Logging in to "MyEID (sctest)".
Please enter User PIN:
Certificate Object; type = X.509 cert
  label:      Certificate
  subject:    DN: O=IDM.EXAMPLE.COM, CN=idmuser1
  ID:         01
Certificate Object; type = X.509 cert
  label:      Second certificate
  subject:    DN: O=IDM.EXAMPLE.COM, CN=ipauser1
  ID:         02
```

b. 使用证书 ID 01 运行 **kinit** ：

```
$ kinit -X kinit -X X509_user_identity=PKCS11:certid=01 idmuser1
MyEID (sctest)                  PIN:
```

5. 运行 **klist** 查看 Kerberos 凭证缓存的内容：

```
$ klist
Ticket cache: KCM:0:11485
Default principal: idmuser1@EXAMPLE.COM

Valid starting       Expires              Service principal
10/04/2021 10:50:04  10/05/2021 10:49:55  krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

6. 完成后销毁您的活跃 Kerberos 票据：

```
$ kdestroy -A
```

其他资源

- 请参阅 **kinit** man page。
- 请参阅 **kdestroy** man page。

7.4. 增加 SSSD 超时

如果您在使用智能卡进行身份验证时遇到问题，请检查 **krb5_child.log** 和 **p11_child.log** 文件以查找类似如下的超时条目：

krb5_child:Timeout for child [9607] reached…..consider increasing value of krb5_auth_timeout.

如果日志文件中有一个超时条目，请尝试增加 SSSD 超时，如此过程中所述。

先决条件

- 您已为智能卡验证配置了 IdM 服务器和客户端。

步骤

1. 在 IdM 客户端中打开 **sssd.conf** 文件：

```
# vim /etc/sssd/sssd.conf
```

2. 在您的 domain 部分中，如 **[domain/idm.example.com]**，添加以下选项：

```
krb5_auth_timeout = 60
```

3. 在 **[pam]** 部分中，添加以下内容：

```
p11_child_timeout = 60
```

4. 清除 SSSD 缓存：

```
# sssctl cache-remove
SSSD must not be running. Stop SSSD now? (yes/no) [yes] yes
Creating backup of local data…
Removing cache files…
SSSD needs to be running. Start SSSD now? (yes/no) [yes] yes
```

增加超时后，请尝试使用智能卡再次进行身份验证。如需了解更多详细信息，请参阅测试智能卡验证。

7.5. 证书映射和匹配规则故障排除

如果您在使用智能卡验证时遇到问题，请检查您已将智能卡证书正确链接到用户。默认情况下，当用户条目包含完整证书作为 **usercertificate** 属性的一部分时，会关联一个证书。但是，如果您定义了证书映射规则，您可能已经更改了与用户关联的证书的方式。要排除证书映射和匹配规则的问题，请参阅以下部分：

- 检查证书如何映射到用户
- 检查与智能卡证书关联的用户

注意

如果您通过 SSH 使用智能卡进行验证，则需要将完整证书添加到 Identity Management(IdM)的用户条目中。如果您不使用 SSH 验证智能卡，您可以使用 **ipa user-add-certmapdata** 命令添加证书映射数据。

7.5.1. 检查证书如何映射到用户

默认情况下，当用户条目包含完整证书作为 **usercertificate** 属性的一部分时，会关联一个证书。但是，如果您定义了证书映射规则，您可能已经更改了与用户关联的证书的方式。这个步骤描述了如何检查证书映射规则。

先决条件

- 已安装并配置了 Identity Management(IdM)服务器和客户端，用于智能卡。
- 您可以检测智能卡读取器并显示智能卡的内容。请参阅在系统上测试智能卡访问。
- 您已将智能卡证书映射到 IdM 用户。请参阅在智能卡 上配置身份验证的证书映射规则。

步骤

1. 验证当前为 IdM 配置的证书映射规则：

```
# ipa certmaprule-find
----------------------------------------
1 Certificate Identity Mapping Rule matched
----------------------------------------
  Rule name: smartcardrule
  Mapping rule: (ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})
  Matching rule: <ISSUER>CN=Certificate Authority,O=IDM.EXAMPLE.COM
  Enabled: TRUE
----------------------------
Number of entries returned 1
----------------------------
```

   您可以看到定义的以下映射规则之一：

   - **ipacertmapdata** 表示使用 IdM 用户条目 **certmapdata** 属性。
   - **altSecurityIdentities** 指定使用 Active Directory 的用户条目名称映射属性。
   - **userCertificate;binary=** 表示使用 IdM 或 AD 中的整个证书。

   您可以定义许多匹配选项，但一些通常配置的选项如下：

   - **<ISSUER>CN=[...]** 指定被检查的证书的 issuer 属性，以确保它与此匹配。
   - **<SUBJECT>.*,DC=MY,DC=DOMAIN** 表示是否已检查证书的主题。

2. 通过在 IdM 服务器上的 **/etc/sssd/sssd.conf** 文件中添加 **debug_level = 9** 来启用系统安全服务守护进程(SSSD)日志：

```
[domain/idm.example.com]
...
debug_level = 9
```

3. 重启 SSSD：

```
# systemctl restart sssd
```

4. 如果正确读取映射，您应该在 **/var/log/sssd/sssd_idm.example.com.log** 文件中看到以下条目：

```
[be[idm.example.com]] [sdap_setup_certmap] (0x4000): Trying to add rule [smartcardrule][-1]
[<ISSUER>CN=Certificate Authority,O=IDM.EXAMPLE.COM][(|(userCertificate;binary=
{cert!bin})(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500}))].
```

5. 如果您的映射规则包含无效的语法，则日志文件中可以看到类似如下的条目：

```
[be[idm.example.com]]] [sss_certmap_init] (0x0040): sss_certmap initialized.
[be[idm.example.com]]] [ipa_certmap_parse_results] (0x4000): Trying to add rule
[smartcardrule][-1][<ISSUER>CN=Certificate Authority,O=IDM.EXAMPLE.COM]
[(ipacertmapdata=X509:<I>{issuer_dn!x509}<S>{subject_dn})].
[be[idm.example.com]]] [parse_template] (0x0040): Parse template invalid.
[be[idm.example.com]]] [parse_ldap_mapping_rule] (0x0040): Failed to add template.
[be[idm.example.com]]] [parse_mapping_rule] (0x0040): Failed to parse LDAP mapping rule.
[be[idm.example.com]]] [ipa_certmap_parse_results] (0x0020): sss_certmap_add_rule failed
for rule [smartcardrule], skipping. Please check for typos and if rule syntax is supported.
[be[idm.example.com]]] [ipa_subdomains_certmap_done] (0x0040): Unable to parse certmap
results [22]: Invalid argument
[be[idm.example.com]]] [ipa_subdomains_refresh_certmap_done] (0x0020): Failed to read
certificate mapping rules [22]: Invalid argument
```

6. 检查您的映射规则语法。

```
# ipa certmaprule-show smartcardrule
  Rule name: smartcardrule
  Mapping rule: (|(userCertificate;binary={cert!bin})(ipacertmapdata=X509:<I>
{issuer_dn!nss_x500}<S>{subject_dn!nss_x500}))
  Matching rule: <ISSUER>CN=Certificate Authority,O=IDM.EXAMPLE.COM
  Domain name: ipa.test
  Enabled: TRUE
```

7. 如果需要，修改您的证书映射规则：

```
# ipa certmaprule-mod smartcardrule --maprule '(ipacertmapdata=X509:<I>
{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})'
```

其他资源

- 请参阅 **sss-certmap** man page。

7.5.2. 检查与智能卡证书关联的用户

如果您在使用智能卡进行身份验证时遇到问题，请验证正确的用户是否与您的智能卡证书关联。

先决条件

- 已安装并配置了 Identity Management(IdM)服务器和客户端，用于智能卡。
- 您可以检测智能卡读取器并显示智能卡的内容。请参阅在系统上测试智能卡访问。
- 您已将智能卡证书映射到 IdM 用户。请参阅在智能卡 上配置身份验证的证书映射规则。
- 您有 PEM 格式的智能卡中的证书副本，例如 **cert.pem**。

步骤

1. 验证用户是否与您的智能卡证书关联：

```
# ipa certmap-match cert.pem
--------------
1 user matched
--------------
  Domain: IDM.EXAMPLE.COM
  User logins: idmuser1
----------------------------
Number of entries returned 1
----------------------------
```

如果用户或域不正确，请检查您的证书如何映射到用户。请参阅检查如何将证书映射到用户。

2. 检查用户条目是否包含证书：

```
# ipa user-show idmuser1
  User login: idmuser1
[...]
Certificate:MIIEejCCAuKgAwIBAgIBCzANBgkqhkiG9w0BAQsFADAzMREwDwYDVQQKDAhJ
UEEuVEVTVDEeMBwGA1UEAwwVQ2VydGlmaWNhdGUgQXV0aG9yaXR5MB4XD
```

3. 如果您的用户条目不包含证书，请将您的 base-64 编码证书添加到用户条目中：

   a. 创建一个包含证书的环境变量，该变量移除了标头和页脚，并串联成一行，这是 **ipa user-add-cert** 命令期望的格式：

   ```
   $ export CERT=`openssl x509 -outform der -in idmuser1.crt | base64 -w0 -`
   ```

   请注意，**idmuser1.crt** 文件中的证书必须采用 PEM 格式。

   b. 使用 **ipa user-add-cert** 命令将证书添加到 **idmuser1** 配置集中：

   ```
   $ ipa user-add-cert idmuser1 --certificate=$CERT
   ```

   c. 清除系统安全服务守护进程(SSSD)缓存。

   ```
   # sssctl cache-remove
   SSSD must not be running. Stop SSSD now? (yes/no) [yes] yes
   Creating backup of local data…
   Removing cache files…
   SSSD needs to be running. Start SSSD now? (yes/no) [yes] yes
   ```

4. 再次运行 **ipa certmap-match** 来确认用户与您的智能卡证书关联。