



Red Hat Enterprise Linux 9

Gestion de l'authentification par carte à puce

Configuration et utilisation de l'authentification par carte à puce

Red Hat Enterprise Linux 9 Gestion de l'authentification par carte à puce

Configuration et utilisation de l'authentification par carte à puce

Notice légale

Copyright © 2023 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.

Résumé

Avec Red Hat Identity Management (IdM), vous pouvez stocker des informations d'identification sous la forme d'une clé privée et d'un certificat sur une carte à puce. Vous pouvez ensuite utiliser cette carte à puce à la place des mots de passe pour vous authentifier auprès des services. Les administrateurs peuvent configurer des règles de mappage afin de réduire la charge administrative.

Table des matières

RENDRE L'OPEN SOURCE PLUS INCLUSIF	4
FOURNIR UN RETOUR D'INFORMATION SUR LA DOCUMENTATION DE RED HAT	5
CHAPITRE 1. COMPRENDRE L'AUTHENTIFICATION PAR CARTE À PUCE	6
1.1. QU'EST-CE QU'UNE CARTE À PUCE ?	6
1.2. QU'EST-CE QUE L'AUTHENTIFICATION PAR CARTE À PUCE ?	6
1.3. OPTIONS D'AUTHENTIFICATION PAR CARTE À PUCE DANS RHEL	7
1.4. OUTILS DE GESTION DES CARTES À PUCE ET DE LEUR CONTENU	8
1.5. CERTIFICATS ET AUTHENTIFICATION PAR CARTE À PUCE	9
1.6. ÉTAPES REQUISES POUR L'AUTHENTIFICATION PAR CARTE À PUCE DANS L'IDM	9
1.7. ÉTAPES NÉCESSAIRES POUR L'AUTHENTIFICATION DE LA CARTE À PUCE AVEC DES CERTIFICATS ÉMIS PAR ACTIVE DIRECTORY	10
CHAPITRE 2. CONFIGURATION DE LA GESTION DES IDENTITÉS POUR L'AUTHENTIFICATION PAR CARTE À PUCE	11
2.1. CONFIGURATION DU SERVEUR IDM POUR L'AUTHENTIFICATION PAR CARTE À PUCE	11
2.2. UTILISER ANSIBLE POUR CONFIGURER LE SERVEUR IDM POUR L'AUTHENTIFICATION PAR CARTE À PUCE	14
2.3. CONFIGURATION DU CLIENT IDM POUR L'AUTHENTIFICATION PAR CARTE À PUCE	17
2.4. UTILISER ANSIBLE POUR CONFIGURER LES CLIENTS IDM POUR L'AUTHENTIFICATION PAR CARTE À PUCE	19
2.5. AJOUT D'UN CERTIFICAT À UNE ENTRÉE UTILISATEUR DANS L'INTERFACE WEB IDM	22
2.6. AJOUT D'UN CERTIFICAT À UNE ENTRÉE UTILISATEUR DANS LA CLI IDM	24
2.7. INSTALLATION D'OUTILS DE GESTION ET D'UTILISATION DES CARTES À PUCE	25
2.8. PRÉPARATION DE VOTRE CARTE À PUCE ET TÉLÉCHARGEMENT DE VOS CERTIFICATS ET CLÉS SUR VOTRE CARTE À PUCE	26
2.9. CONNEXION À L'IDM AVEC DES CARTES À PUCE	27
2.10. SE CONNECTER À GDM EN UTILISANT L'AUTHENTIFICATION PAR CARTE À PUCE SUR UN CLIENT IDM	29
2.11. UTILISATION DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC LA COMMANDE SU	30
CHAPITRE 3. CONFIGURATION DES CERTIFICATS ÉMIS PAR ADCS POUR L'AUTHENTIFICATION PAR CARTE À PUCE DANS IDM	31
3.1. PARAMÈTRES DU SERVEUR WINDOWS REQUIS POUR LA CONFIGURATION DE LA CONFIANCE ET L'UTILISATION DU CERTIFICAT	31
3.2. COPIER DES CERTIFICATS À PARTIR D'ACTIVE DIRECTORY À L'AIDE DE SFTP	32
3.3. CONFIGURATION DU SERVEUR IDM ET DES CLIENTS POUR L'AUTHENTIFICATION PAR CARTE À PUCE À L'AIDE DE CERTIFICATS ADCS	33
3.4. CONVERSION DU FICHIER PFX	34
3.5. INSTALLATION D'OUTILS DE GESTION ET D'UTILISATION DES CARTES À PUCE	35
3.6. PRÉPARATION DE VOTRE CARTE À PUCE ET TÉLÉCHARGEMENT DE VOS CERTIFICATS ET CLÉS SUR VOTRE CARTE À PUCE	35
3.7. CONFIGURATION DES DÉLAIS D'ATTENTE DANS SSSD.CONF	37
3.8. CRÉATION DE RÈGLES DE MAPPAGE DE CERTIFICATS POUR L'AUTHENTIFICATION PAR CARTE À PUCE	38
CHAPITRE 4. RÈGLES DE MAPPAGE DES CERTIFICATS POUR LA CONFIGURATION DE L'AUTHENTIFICATION SUR LES CARTES À PUCE	39
4.1. RÈGLES DE MAPPAGE DES CERTIFICATS POUR LES TRUSTS AVEC LES DOMAINES ACTIVE DIRECTORY	39
4.2. COMPOSANTS D'UNE RÈGLE DE MAPPAGE D'IDENTITÉ DANS IDM	40
4.3. OBTENIR L'ÉMETTEUR D'UN CERTIFICAT POUR L'UTILISER DANS UNE RÈGLE DE CORRESPONDANCE	41

4.4. RESSOURCES SUPPLÉMENTAIRES	42
CHAPITRE 5. CONFIGURATION DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC LA CONSOLE WEB POUR LES UTILISATEURS GÉRÉS DE MANIÈRE CENTRALISÉE	43
5.1. AUTHENTIFICATION PAR CARTE À PUCE POUR LES UTILISATEURS GÉRÉS DE MANIÈRE CENTRALISÉE	43
5.2. INSTALLATION D'OUTILS DE GESTION ET D'UTILISATION DES CARTES À PUCE	43
5.3. PRÉPARATION DE VOTRE CARTE À PUCE ET TÉLÉCHARGEMENT DE VOS CERTIFICATS ET CLÉS SUR VOTRE CARTE À PUCE	44
5.4. ACTIVATION DE L'AUTHENTIFICATION PAR CARTE À PUCE POUR LA CONSOLE WEB	46
5.5. SE CONNECTER À LA CONSOLE WEB AVEC DES CARTES À PUCE	46
5.6. ACTIVATION DE SUDO SANS MOT DE PASSE POUR LES UTILISATEURS DE CARTES À PUCE	47
5.7. LIMITATION DES SESSIONS D'UTILISATEURS ET DE LA MÉMOIRE POUR ÉVITER UNE ATTAQUE DOS	49
CHAPITRE 6. CONFIGURATION DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC DES CERTIFICATS LOCAUX	51
6.1. CRÉATION DE CERTIFICATS LOCAUX	51
6.2. COPIE DES CERTIFICATS DANS LE RÉPERTOIRE SSSD	54
6.3. INSTALLATION D'OUTILS DE GESTION ET D'UTILISATION DES CARTES À PUCE	55
6.4. PRÉPARATION DE VOTRE CARTE À PUCE ET TÉLÉCHARGEMENT DE VOS CERTIFICATS ET CLÉS SUR VOTRE CARTE À PUCE	56
6.5. CONFIGURATION DE L'ACCÈS SSH À L'AIDE DE L'AUTHENTIFICATION PAR CARTE À PUCE	57
CHAPITRE 7. CONFIGURATION DE L'AUTHENTIFICATION PAR CARTE À PUCE À L'AIDE DE AUTHSELECT	60
7.1. CERTIFICATS ÉLIGIBLES AUX CARTES À PUCE	60
7.2. CONFIGUREZ VOTRE SYSTÈME POUR ACTIVER L'AUTHENTIFICATION PAR CARTE À PUCE ET PAR MOT DE PASSE	60
7.3. CONFIGURATION DE VOTRE SYSTÈME POUR APPLIQUER L'AUTHENTIFICATION PAR CARTE À PUCE	61
7.4. CONFIGURATION DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC VERROUILLAGE EN CAS DE RETRAIT	62
CHAPITRE 8. S'AUTHENTIFIER À SUDO À DISTANCE À L'AIDE DE CARTES À PUCE	63
8.1. CRÉATION DE RÈGLES SUDO DANS IDM	63
8.2. MISE EN PLACE DU MODULE PAM POUR SUDO	64
8.3. SE CONNECTER À SUDO À DISTANCE À L'AIDE D'UNE CARTE À PUCE	65
CHAPITRE 9. DÉPANNAGE DE L'AUTHENTIFICATION PAR CARTE À PUCE	66
9.1. TEST DE L'ACCÈS PAR CARTE À PUCE SUR LE SYSTÈME	66
9.2. DÉPANNAGE DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC SSSD	69
9.3. VÉRIFICATION QUE LE KDC KERBEROS D'IDM PEUT UTILISER PKINIT ET QUE LES CERTIFICATS DE L'AUTORITÉ DE CERTIFICATION SONT CORRECTEMENT LOCALISÉS	71
9.4. AUGMENTATION DES DÉLAIS D'ATTENTE SSSD	73
9.5. DÉPANNAGE DES RÈGLES DE MAPPAGE ET DE CORRESPONDANCE DES CERTIFICATS	74

RENDRE L'OPEN SOURCE PLUS INCLUSIF

Red Hat s'engage à remplacer les termes problématiques dans son code, sa documentation et ses propriétés Web. Nous commençons par ces quatre termes : *master*, *slave*, *blacklist* et *whitelist*. En raison de l'ampleur de cette entreprise, ces changements seront mis en œuvre progressivement au cours de plusieurs versions à venir. Pour plus de détails, voir le [message de notre directeur technique Chris Wright](#).

Dans le domaine de la gestion de l'identité, les remplacements terminologiques prévus sont les suivants :

- ***block list*** remplace *blacklist*
- ***allow list*** remplace *whitelist*
- ***secondary*** remplace *slave*
- Le mot *master* est remplacé par un langage plus précis, en fonction du contexte :
 - ***IdM server*** remplace *IdM master*
 - ***CA renewal server*** remplace *CA renewal master*
 - ***CRL publisher server*** remplace *CRL master*
 - ***multi-supplier*** remplace *multi-master*

FOURNIR UN RETOUR D'INFORMATION SUR LA DOCUMENTATION DE RED HAT

Nous apprécions vos commentaires sur notre documentation. Faites-nous savoir comment nous pouvons l'améliorer.

Soumettre des commentaires sur des passages spécifiques

1. Consultez la documentation au format **Multi-page HTML** et assurez-vous que le bouton **Feedback** apparaît dans le coin supérieur droit après le chargement complet de la page.
2. Utilisez votre curseur pour mettre en évidence la partie du texte que vous souhaitez commenter.
3. Cliquez sur le bouton **Add Feedback** qui apparaît près du texte en surbrillance.
4. Ajoutez vos commentaires et cliquez sur **Submit**.

Soumettre des commentaires via Bugzilla (compte requis)

1. Connectez-vous au site Web de [Bugzilla](#).
2. Sélectionnez la version correcte dans le menu **Version**.
3. Saisissez un titre descriptif dans le champ **Summary**.
4. Saisissez votre suggestion d'amélioration dans le champ **Description**. Incluez des liens vers les parties pertinentes de la documentation.
5. Cliquez sur **Submit Bug**.

CHAPITRE 1. COMPRENDRE L'AUTHENTIFICATION PAR CARTE À PUCE

L'authentification par carte à puce est une alternative aux mots de passe. Les informations d'identification de l'utilisateur peuvent être stockées sur une carte à puce sous la forme d'une clé privée et d'un certificat, et un logiciel et un matériel spéciaux sont utilisés pour y accéder. Placez la carte à puce dans un lecteur ou un port USB et fournissez le code PIN de la carte à puce au lieu de votre mot de passe.

Cette section décrit ce qu'est une carte à puce et comment fonctionne l'authentification par carte à puce. Elle décrit les outils que vous pouvez utiliser pour lire et manipuler le contenu d'une carte à puce. Elle fournit également des exemples de cas d'utilisation et décrit la configuration du serveur IdM et du client IdM pour l'authentification par carte à puce.



NOTE

Si vous souhaitez commencer à utiliser l'authentification par carte à puce, consultez la configuration matérielle requise : [Prise en charge des cartes à puce dans RHEL9](#).

1.1. QU'EST-CE QU'UNE CARTE À PUCE ?

Une carte à puce est un dispositif physique, généralement une carte en plastique dotée d'un microprocesseur, qui peut fournir une authentification personnelle à l'aide de certificats stockés sur la carte. L'authentification personnelle signifie que vous pouvez utiliser les cartes à puce de la même manière que les mots de passe des utilisateurs.

Les informations d'identification de l'utilisateur peuvent être stockées sur la carte à puce sous la forme d'une clé privée et d'un certificat, et un logiciel et un matériel spéciaux sont utilisés pour y accéder. Vous placez la carte à puce dans un lecteur ou une prise USB et vous fournissez le code PIN de la carte à puce au lieu de votre mot de passe.

1.2. QU'EST-CE QUE L'AUTHENTIFICATION PAR CARTE À PUCE ?

L'authentification par clé publique et l'authentification par certificat sont deux alternatives largement utilisées à l'authentification par mot de passe. Votre identité est confirmée en utilisant des clés publiques et privées au lieu de votre mot de passe. Un certificat est un document électronique utilisé pour identifier une personne, un serveur, une entreprise ou une autre entité et pour associer cette identité à une clé publique. Comme un permis de conduire ou un passeport, un certificat fournit une preuve généralement reconnue de l'identité d'une personne. La cryptographie à clé publique utilise les certificats pour résoudre le problème de l'usurpation d'identité.

Dans le cas de l'authentification par carte à puce, les informations d'identification de l'utilisateur, c'est-à-dire les clés publique et privée et le certificat, sont stockées sur une carte à puce et ne peuvent être utilisées que lorsque la carte à puce est insérée dans le lecteur et qu'un code PIN est fourni. Comme vous devez posséder un dispositif physique, la carte à puce, et connaître son code PIN, l'authentification par carte à puce est considérée comme un type d'authentification à deux facteurs.

1.2.1. Exemples d'authentification par carte à puce dans l'IdM

Les exemples suivants décrivent deux scénarios simples d'utilisation des cartes à puce dans l'IdM.

1.2.1.1. Connexion à votre système à l'aide d'une carte à puce

Vous pouvez utiliser une carte à puce pour vous authentifier sur un système RHEL en tant qu'utilisateur local. Si votre système est configuré pour appliquer l'authentification par carte à puce, vous êtes invité à insérer votre carte à puce et à saisir son code PIN et, en cas d'échec, vous ne pouvez pas vous connecter à votre système. Vous pouvez également configurer votre système pour qu'il s'authentifie à l'aide de la carte à puce ou de votre nom d'utilisateur et de votre mot de passe. Dans ce cas, si votre carte à puce n'est pas insérée, vous êtes invité à saisir votre nom d'utilisateur et votre mot de passe.

1.2.1.2. Se connecter à GDM avec un verrou sur le retrait

Vous pouvez activer la fonction de verrouillage lors du retrait si vous avez configuré l'authentification par carte à puce sur votre système RHEL. Si vous êtes connecté au gestionnaire d'affichage GNOME (GDM) et que vous retirez votre carte à puce, le verrouillage de l'écran est activé et vous devez réinsérer votre carte à puce et vous authentifier avec le code PIN pour déverrouiller l'écran. Vous ne pouvez pas utiliser votre nom d'utilisateur et votre mot de passe pour vous authentifier.



NOTE

Si vous êtes connecté à GDM et que vous retirez votre carte à puce, le verrouillage de l'écran est activé et vous devez réinsérer votre carte à puce et vous authentifier avec le code PIN pour déverrouiller l'écran.

1.3. OPTIONS D'AUTHENTIFICATION PAR CARTE À PUCE DANS RHEL

Vous pouvez configurer le fonctionnement de l'authentification par carte à puce dans un client Identity Management (IdM) particulier à l'aide de la commande **authselect**, **authselect enable-feature <smartcardoption>**. Les options de carte à puce suivantes sont disponibles :

- **with-smartcard**: Les utilisateurs peuvent s'authentifier avec leur nom d'utilisateur et leur mot de passe ou avec leur carte à puce.
- **with-smartcard-required**: Les utilisateurs peuvent s'authentifier avec leur carte à puce et l'authentification par mot de passe est désactivée. Vous ne pouvez pas accéder au système sans votre carte à puce. Une fois que vous vous êtes authentifié avec votre carte à puce, vous pouvez rester connecté même si votre carte à puce est retirée de son lecteur.



NOTE

L'option **with-smartcard-required** n'impose l'authentification exclusive par carte à puce que pour les services de connexion, tels que **login**, **gdm**, **xdm**, **xscreensaver** et **gnome-screensaver**. Pour les autres services, tels que **su** ou **sudo** pour les utilisateurs qui changent de service, l'authentification par carte à puce n'est pas appliquée et si votre carte à puce n'est pas insérée, un mot de passe vous est demandé.

- **with-smartcard-lock-on-removal**: Les utilisateurs peuvent s'authentifier avec leur carte à puce. Cependant, si vous retirez votre carte à puce de son lecteur, vous êtes automatiquement bloqué dans le système. Vous ne pouvez pas utiliser l'authentification par mot de passe.



NOTE

L'option **with-smartcard-lock-on-removal** ne fonctionne que sur les systèmes dotés de l'environnement de bureau GNOME. Si vous utilisez un système basé sur **tty** ou une console et que vous retirez votre carte à puce de son lecteur, vous n'êtes pas automatiquement verrouillé.

Pour plus d'informations, voir [Configuration des cartes à puce à l'aide de authselect](#) .

1.4. OUTILS DE GESTION DES CARTES À PUCE ET DE LEUR CONTENU

Vous pouvez utiliser différents outils pour gérer les clés et les certificats stockés sur vos cartes à puce. Vous pouvez utiliser ces outils pour effectuer les opérations suivantes :

- Liste des lecteurs de cartes à puce disponibles connectés à un système.
- Liste des cartes à puce disponibles et visualisation de leur contenu.
- Manipuler le contenu de la carte à puce, c'est-à-dire les clés et les certificats.

Il existe de nombreux outils qui offrent des fonctionnalités similaires, mais certains fonctionnent à des niveaux différents de votre système. Les cartes à puce sont gérées à plusieurs niveaux par plusieurs composants. Au niveau inférieur, le système d'exploitation communique avec le lecteur de cartes à puce en utilisant le protocole PC/SC, et cette communication est gérée par le démon pcsc-lite. Le démon transmet les commandes reçues au lecteur de cartes à puce, généralement via USB, qui est géré par le pilote CCID de bas niveau. La communication de bas niveau PC/SC est rarement vue au niveau de l'application. Dans RHEL, les applications accèdent principalement aux cartes à puce via une interface de programmation d'applications (API) de niveau supérieur, l'API OASIS PKCS#11, qui réduit la communication avec la carte à des commandes spécifiques qui opèrent sur des objets cryptographiques, par exemple des clés privées. Les vendeurs de cartes à puce fournissent un module partagé, tel qu'un fichier **.so**, qui suit l'API PKCS#11 et sert de pilote pour la carte à puce.

Vous pouvez utiliser les outils suivants pour gérer vos cartes à puce et leur contenu :

- Outils OpenSC : travailler avec les pilotes implémentés dans **opensc**.
 - `opensc-tool` : effectuer des opérations sur les cartes à puce.
 - `pkcs15-tool` : gère les structures de données PKCS#15 sur les cartes à puce, telles que la liste et la lecture des codes PIN, des clés et des certificats stockés sur le jeton.
 - `pkcs11-tool` : gère les objets de données PKCS#11 sur les cartes à puce, tels que la liste et la lecture des codes PIN, des clés et des certificats stockés sur le jeton.
- GnuTLS utils : une API pour les applications afin de permettre une communication sécurisée sur la couche de transport du réseau, ainsi que des interfaces pour accéder à X.509, PKCS#12, OpenPGP, et d'autres structures.
 - `p11tool` : permet d'effectuer des opérations sur les cartes à puce et les modules de sécurité PKCS#11.
 - `certtool` : analyse et génère des certificats X.509, des requêtes et des clés privées.
- Outils NSS (Network Security Services) : ensemble de bibliothèques conçues pour soutenir le développement multiplateforme d'applications client et serveur sécurisées. Les applications construites avec NSS peuvent prendre en charge SSL v3, TLS, PKCS #5, PKCS #7, PKCS #11, PKCS #12, S/MIME, les certificats X.509 v3 et d'autres normes de sécurité.
 - `modutil` : gestion des informations relatives aux modules PKCS#11 avec la base de données des modules de sécurité.
 - `certutil` : gère les clés et les certificats dans les bases de données NSS et les autres jetons NSS.

Pour plus d'informations sur l'utilisation de ces outils pour résoudre les problèmes d'authentification à l'aide d'une carte à puce, voir [Dépannage de l'authentification par carte à puce](#) .

Ressources supplémentaires

- **opensc-tool** page de manuel
- **pkcs15-tool** page de manuel
- **pkcs11-tool** page de manuel
- **p11tool** page de manuel
- **certtool** page de manuel
- **modutil** page de manuel
- **certutil** page de manuel

1.5. CERTIFICATS ET AUTHENTIFICATION PAR CARTE À PUCE

Si vous utilisez Identity Management (IdM) ou Active Directory (AD) pour gérer les magasins d'identité, l'authentification, les stratégies et les politiques d'autorisation dans votre domaine, les certificats utilisés pour l'authentification sont générés par IdM ou AD, respectivement. Vous pouvez également utiliser des certificats fournis par une autorité de certification externe et, dans ce cas, vous devez configurer Active Directory ou IdM pour qu'il accepte les certificats du fournisseur externe. Si l'utilisateur ne fait pas partie d'un domaine, vous pouvez utiliser un certificat généré par une autorité de certification locale. Pour plus de détails, reportez-vous aux sections suivantes :

- [Configuration de la gestion des identités pour l'authentification par carte à puce](#)
- [Configuration des certificats émis par ADCS pour l'authentification par carte à puce dans IdM](#)
- [Gestion des certificats signés en externe pour les utilisateurs, les hôtes et les services IdM](#)
- [Configuration et importation de certificats locaux sur une carte à puce](#)

Pour une liste complète des certificats éligibles pour l'authentification par carte à puce, voir [Certificats éligibles pour les cartes à puce](#).

1.6. ÉTAPES REQUISES POUR L'AUTHENTIFICATION PAR CARTE À PUCE DANS L'IDM

Vous devez vous assurer que les étapes suivantes ont été suivies avant de pouvoir vous authentifier avec une carte à puce dans la gestion des identités (IdM) :

- Configurez votre serveur IdM pour l'authentification par carte à puce. Voir [Configuration du serveur IdM pour l'authentification par carte à puce](#)
- Configurez votre client IdM pour l'authentification par carte à puce. Voir [Configuration du client IdM pour l'authentification par carte à puce](#)
- Ajouter le certificat à l'entrée de l'utilisateur dans IdM. Voir [Ajouter un certificat à une entrée utilisateur dans l'interface Web IdM](#)
- Stockez vos clés et certificats sur la carte à puce. Voir [Stocker un certificat sur une carte à puce](#)

1.7. ÉTAPES NÉCESSAIRES POUR L'AUTHENTIFICATION DE LA CARTE À PUCE AVEC DES CERTIFICATS ÉMIS PAR ACTIVE DIRECTORY

Vous devez vous assurer que les étapes suivantes ont été suivies avant de pouvoir vous authentifier avec une carte à puce avec des certificats émis par Active Directory (AD) :

- Copier les certificats de l'autorité de certification et de l'utilisateur depuis Active Directory vers le serveur et le client IdM.
- Configurer le serveur IdM et les clients pour l'authentification par carte à puce à l'aide de certificats ADCS.
- Convertissez le fichier PFX (PKCS#12) pour pouvoir stocker le certificat et la clé privée sur la carte à puce.
- Configurez les délais d'attente dans le fichier `sssd.conf`.
- Créer des règles de mappage de certificats pour l'authentification par carte à puce .

CHAPITRE 2. CONFIGURATION DE LA GESTION DES IDENTITÉS POUR L'AUTHENTIFICATION PAR CARTE À PUCE

La gestion des identités (IdM) prend en charge l'authentification par carte à puce avec :

- Certificats d'utilisateur délivrés par l'autorité de certification IdM
- Certificats d'utilisateur délivrés par une autorité de certification externe

Cette histoire d'utilisateur montre comment configurer l'authentification par carte à puce dans IdM pour les deux types de certificats. Dans l'histoire de l'utilisateur, le certificat CA **rootca.pem** est le fichier contenant le certificat d'une autorité de certification externe de confiance.

Pour plus d'informations sur l'authentification par carte à puce dans IdM, voir [Comprendre l'authentification par carte à puce](#).

L'histoire de l'utilisateur contient les modules suivants :

- [Configuration du serveur IdM pour l'authentification par carte à puce](#)
- [Configuration du client IdM pour l'authentification par carte à puce](#)
- [Ajout d'un certificat à une entrée utilisateur dans l'interface Web IdM](#)
- [Ajout d'un certificat à une entrée utilisateur dans la CLI IdM](#)
- [Installation d'outils de gestion et d'utilisation des cartes à puce](#)
- [Stocker un certificat sur une carte à puce](#)
- [Connexion à l'IdM avec des cartes à puce](#)
- [Configuration de l'accès au GDM à l'aide de l'authentification par carte à puce](#)
- [Configuration de l'accès su à l'aide de l'authentification par carte à puce](#)

2.1. CONFIGURATION DU SERVEUR IDM POUR L'AUTHENTIFICATION PAR CARTE À PUCE

Si vous souhaitez activer l'authentification par carte à puce pour les utilisateurs dont les certificats ont été émis par l'autorité de certification (AC) du domaine <EXAMPLE.ORG> auquel votre AC de gestion des identités (IdM) fait confiance, vous devez obtenir les certificats suivants afin de pouvoir les ajouter lors de l'exécution du script **ipa-advise** qui configure le serveur IdM :

- Le certificat de l'autorité de certification racine qui a délivré le certificat pour l'autorité de certification <EXAMPLE.ORG> directement ou par l'intermédiaire d'une ou de plusieurs de ses autorités de certification secondaires. Vous pouvez télécharger la chaîne de certificats à partir d'une page web dont le certificat a été délivré par l'autorité. Pour plus de détails, voir les étapes 1 à 4a de la section [Configuration d'un navigateur pour activer l'authentification par certificat](#) .
- Le certificat de l'autorité de certification IdM. Vous pouvez obtenir le certificat de l'autorité de certification à partir du fichier **/etc/ipa/ca.crt** du serveur IdM sur lequel tourne une instance de l'autorité de certification IdM.

- Les certificats de toutes les autorités de certification intermédiaires, c'est-à-dire intermédiaires entre l'autorité de certification <EXAMPLE.ORG> et l'autorité de certification IdM.

Pour configurer un serveur IdM pour l'authentification par carte à puce :

1. Obtenir les fichiers contenant les certificats de l'autorité de certification au format PEM.
2. Exécutez le script intégré **ipa-advise**.
3. Recharger la configuration du système.

Conditions préalables

- Vous avez un accès root au serveur IdM.
- Vous disposez du certificat de l'autorité de certification racine et de tous les certificats des autorités de certification intermédiaires.

Procédure

1. Créez un répertoire dans lequel vous effectuerez la configuration :

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

2. Naviguez jusqu'au répertoire :

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

3. Obtenez les certificats d'autorité de certification pertinents stockés dans des fichiers au format PEM. Si votre certificat d'autorité de certification est stocké dans un fichier d'un format différent, tel que DER, convertissez-le au format PEM. Le certificat de l'autorité de certification IdM est au format PEM et se trouve dans le fichier **/etc/ipa/ca.crt**.
Convertit un fichier DER en fichier PEM :

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

4. Pour plus de commodité, copiez les certificats dans le répertoire dans lequel vous souhaitez effectuer la configuration :

```
[root@server SmartCard]# cp /tmp/rootca.pem ~/SmartCard/  
[root@server SmartCard]# cp /tmp/subca.pem ~/SmartCard/  
[root@server SmartCard]# cp /tmp/issuingca.pem ~/SmartCard/
```

5. En option, si vous utilisez des certificats d'autorités de certification externes, utilisez l'utilitaire **openssl x509** pour visualiser le contenu des fichiers au format **PEM** et vérifier que les valeurs **Issuer** et **Subject** sont correctes :

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

6. Générer un script de configuration avec l'utilitaire intégré **ipa-advise**, en utilisant les privilèges de l'administrateur :

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

Le script **config-server-for-smart-card-auth.sh** effectue les actions suivantes :

- Il configure le serveur HTTP Apache de l'IdM.
 - Il active la cryptographie à clé publique pour l'authentification initiale dans Kerberos (PKINIT) sur le centre de distribution de clés (KDC).
 - Il configure l'interface Web IdM pour qu'elle accepte les demandes d'autorisation de carte à puce.
7. Exécutez le script en ajoutant les fichiers PEM contenant les certificats de l'autorité de certification racine et de l'autorité de certification secondaire en tant qu'arguments :

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



NOTE

Assurez-vous que vous ajoutez le certificat de l'autorité de certification racine en tant qu'argument avant tout certificat d'autorité de certification secondaire et que les certificats de l'autorité de certification ou de l'autorité de certification secondaire n'ont pas expiré.

8. En option, si l'autorité de certification qui a émis le certificat utilisateur ne fournit pas de répondeur OCSP (Online Certificate Status Protocol), il peut être nécessaire de désactiver la vérification OCSP pour l'authentification à l'IdM Web UI :
- a. Définissez le paramètre **SSLOCSPEnable** à **off** dans le fichier **/etc/httpd/conf.d/ssl.conf**:

```
SSLOCSPEnable off
```

- b. Redémarrez le démon Apache (httpd) pour que les modifications prennent effet immédiatement :

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



AVERTISSEMENT

Ne désactivez pas le contrôle OCSP si vous n'utilisez que des certificats d'utilisateur émis par l'autorité de certification IdM. Les répondeurs OCSP font partie de l'IdM.

Pour savoir comment maintenir la vérification OCSP activée, tout en empêchant un certificat d'utilisateur d'être rejeté par le serveur IdM s'il ne contient pas les informations relatives à l'emplacement où l'autorité de certification qui a délivré le certificat d'utilisateur écoute les demandes de service OCSP, voir la directive **SSL**`OCSPDefaultResponder` dans les [options de configuration de Apache mod_ssl](#).

Le serveur est maintenant configuré pour l'authentification par carte à puce.



NOTE

Pour activer l'authentification par carte à puce dans l'ensemble de la topologie, exécutez la procédure sur chaque serveur IdM.

2.2. UTILISER ANSIBLE POUR CONFIGURER LE SERVEUR IDM POUR L'AUTHEMIFICATION PAR CARTE À PUCE

Vous pouvez utiliser Ansible pour activer l'authentification par carte à puce pour les utilisateurs dont les certificats ont été émis par l'autorité de certification (AC) du domaine <EXAMPLE.ORG> auquel votre AC de gestion des identités (IdM) fait confiance. Pour ce faire, vous devez obtenir les certificats suivants afin de pouvoir les utiliser lors de l'exécution d'un playbook Ansible avec le script de rôle

ipasmartcard_server ansible-freeipa :

- Le certificat de l'autorité de certification racine qui a délivré le certificat pour l'autorité de certification <EXAMPLE.ORG> directement ou par l'intermédiaire d'une ou de plusieurs de ses autorités de certification secondaires. Vous pouvez télécharger la chaîne de certificats à partir d'une page web dont le certificat a été émis par l'autorité. Pour plus d'informations, voir l'étape 4 de la section [Configuration d'un navigateur pour activer l'authentification par certificat](#) .
- Le certificat de l'autorité de certification IdM. Vous pouvez obtenir le certificat de l'autorité de certification à partir du fichier `/etc/ipa/ca.crt` sur n'importe quel serveur de l'autorité de certification IdM.
- Les certificats de toutes les AC intermédiaires entre l'AC <EXAMPLE.ORG> et l'AC IdM.

Conditions préalables

- Vous avez un accès **root** au serveur IdM.
- Vous connaissez le mot de passe de l'IdM **admin**.
- Vous disposez du certificat de l'autorité de certification racine, du certificat de l'autorité de certification IdM et de tous les certificats des autorités de certification intermédiaires.

- Vous avez configuré votre nœud de contrôle Ansible pour qu'il réponde aux exigences suivantes :
 - Vous utilisez la version 2.8 ou ultérieure d'Ansible.
 - Vous avez installé le paquetage **ansible-freeipa** sur le contrôleur Ansible.
 - L'exemple suppose que dans le répertoire `~/MyPlaybooks/` vous avez créé un [fichier d'inventaire Ansible](#) avec le nom de domaine complet (FQDN) du serveur IdM.
 - L'exemple suppose que le coffre-fort **secret.yml** Ansible stocke votre **ipadmin_password**.

Procédure

1. Si vos certificats d'autorité de certification sont stockés dans des fichiers d'un format différent, tel que **DER**, convertissez-les au format **PEM**:

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

Le certificat de l'autorité de certification IdM est au format **PEM** et se trouve dans le fichier **/etc/ipa/ca.crt**.

2. En option, utilisez l'utilitaire **openssl x509** pour visualiser le contenu des fichiers au format **PEM** et vérifier que les valeurs **Issuer** et **Subject** sont correctes :

```
# openssl x509 -noout -text -in root-ca.pem | more
```

3. Naviguez jusqu'à votre répertoire `~/MyPlaybooks/` répertoire :

```
$ cd ~/MyPlaybooks/
```

4. Créez un sous-répertoire dédié aux certificats d'autorité de certification :

```
$ mkdir SmartCard/
```

5. Pour plus de commodité, copiez tous les certificats requis dans le répertoire `~/MyPlaybooks/SmartCard/`:

```
# cp /tmp/root-ca.pem ~/MyPlaybooks/SmartCard/
# cp /tmp/intermediate-ca.pem ~/MyPlaybooks/SmartCard/
# cp /etc/ipa/ca.crt ~/MyPlaybooks/SmartCard/ipa-ca.crt
```

6. Dans votre fichier d'inventaire Ansible, spécifiez ce qui suit :
 - Les serveurs IdM que vous souhaitez configurer pour l'authentification par carte à puce.
 - Le mot de passe de l'administrateur de l'IdM.
 - Les chemins d'accès aux certificats des autorités de certification dans l'ordre suivant :
 - Le fichier du certificat de l'autorité de certification racine
 - Les fichiers des certificats de l'autorité de certification intermédiaire

- Le fichier du certificat de l'autorité de certification IdM

Le fichier peut se présenter comme suit :

```
[ipaserver]
ipaserver.idm.example.com

[ipareplicas]
ipareplica1.idm.example.com
ipareplica2.idm.example.com

[ipacluster:children]
ipaserver
ipareplicas

[ipacluster:vars]
ipaadmin_password=SomeADMINpassword
ipasmartcard_server_ca_certs=/home/<user_name>/MyPlaybooks/SmartCard/root-
ca.pem,/home/<user_name>/MyPlaybooks/SmartCard/intermediate-
ca.pem,/home/<user_name>/MyPlaybooks/SmartCard/ipa-ca.crt
```

7. Créez un playbook **install-smartcard-server.yml** avec le contenu suivant :

```
---
- name: Playbook to set up smart card authentication for an IdM server
  hosts: ipaserver
  become: true

  roles:
  - role: ipasmartcard_server
    state: present
```

8. Enregistrer le fichier.
9. Exécutez le playbook Ansible. Spécifiez le fichier du livre de jeu, le fichier contenant le mot de passe protégeant le fichier **secret.yml** et le fichier d'inventaire :

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory install-
smartcard-server.yml
```

Le rôle **ipasmartcard_server** Ansible effectue les actions suivantes :

- Il configure le serveur HTTP Apache de l'IdM.
 - Il active la cryptographie à clé publique pour l'authentification initiale dans Kerberos (PKINIT) sur le centre de distribution de clés (KDC).
 - Il configure l'interface Web IdM pour qu'elle accepte les demandes d'autorisation de carte à puce.
10. En option, si l'autorité de certification qui a émis le certificat utilisateur ne fournit pas de répondeur OCSP (Online Certificate Status Protocol), il peut être nécessaire de désactiver la vérification OCSP pour l'authentification à l'IdM Web UI :
 - a. Se connecter au serveur IdM en tant que **root**:
 -

```
ssh root@ipaserver.idm.example.com
```

- b. Définissez le paramètre **SSLOCSPEnable** à **off** dans le fichier `/etc/httpd/conf.d/ssl.conf`:

```
SSLOCSPEnable off
```

- c. Redémarrez le démon Apache (httpd) pour que les modifications prennent effet immédiatement :

```
# systemctl restart httpd
```



AVERTISSEMENT

Ne désactivez pas le contrôle OCSP si vous n'utilisez que des certificats d'utilisateur émis par l'autorité de certification IdM. Les répondeurs OCSP font partie de l'IdM.

Pour savoir comment maintenir la vérification OCSP activée, tout en empêchant un certificat d'utilisateur d'être rejeté par le serveur IdM s'il ne contient pas les informations relatives à l'emplacement où l'autorité de certification qui a délivré le certificat d'utilisateur écoute les demandes de service OCSP, voir la directive **SSLOCSPDefaultResponder** dans les [options de configuration de Apache mod_ssl](#).

Le serveur figurant dans le fichier d'inventaire est maintenant configuré pour l'authentification par carte à puce.



NOTE

Pour activer l'authentification par carte à puce dans l'ensemble de la topologie, définissez la variable **hosts** dans le playbook Ansible à **ipacluster**:

```
---
- name: Playbook to setup smartcard for IPA server and replicas
  hosts: ipacluster
  [...]
```

Ressources supplémentaires

- Exemples de playbooks utilisant le rôle **ipasmartcard_server** dans le répertoire `/usr/share/doc/ansible-freeipa/playbooks/`

2.3. CONFIGURATION DU CLIENT IDM POUR L'AUTHENTIFICATION PAR CARTE À PUCE

Cette section décrit comment configurer les clients IdM pour l'authentification par carte à puce. La procédure doit être exécutée sur chaque système IdM, client ou serveur, auquel vous souhaitez vous connecter en utilisant une carte à puce pour l'authentification. Par exemple, pour activer une connexion

ssh de l'hôte A à l'hôte B, le script doit être exécuté sur l'hôte B.

En tant qu'administrateur, exécutez cette procédure pour activer l'authentification par carte à puce à l'aide de

- Le protocole **ssh**
Pour plus d'informations, voir [Configuration de l'accès SSH à l'aide de l'authentification par carte à puce](#).
- Le login de la console
- Le gestionnaire d'affichage Gnome (GDM)
- La commande **su**

Cette procédure n'est pas nécessaire pour s'authentifier auprès de l'interface Web IdM.

L'authentification à l'interface Web IdM implique deux hôtes, dont aucun ne doit être un client IdM :

- La machine sur laquelle le navigateur s'exécute. La machine peut être en dehors du domaine IdM.
- Le serveur IdM sur lequel **httpd** est exécuté.

La procédure suivante suppose que vous configurez l'authentification par carte à puce sur un client IdM et non sur un serveur IdM. C'est pourquoi vous avez besoin de deux ordinateurs : un serveur IdM pour générer le script de configuration et le client IdM sur lequel le script sera exécuté.

Conditions préalables

- Votre serveur IdM a été configuré pour l'authentification par carte à puce, comme décrit dans la section [Configuration du serveur IdM pour l'authentification par carte à puce](#) .
- Vous disposez d'un accès root au serveur IdM et au client IdM.
- Vous disposez du certificat de l'autorité de certification racine et de tous les certificats des autorités de certification intermédiaires.
- Vous avez installé le client IdM avec l'option **--mkhomedir** pour vous assurer que les utilisateurs distants peuvent se connecter avec succès. Si vous ne créez pas de répertoire personnel, l'emplacement de connexion par défaut est la racine de la structure de répertoires, /.

Procédure

1. Sur un serveur IdM, générez un script de configuration avec **ipa-advise** en utilisant les privilèges de l'administrateur :

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

Le script **config-client-for-smart-card-auth.sh** effectue les actions suivantes :

- Il configure le démon de la carte à puce.
- Il définit la réserve de confiance du système.

- Il configure le System Security Services Daemon (SSSD) pour permettre aux utilisateurs de s'authentifier soit avec leur nom d'utilisateur et leur mot de passe, soit avec leur carte à puce. Pour plus de détails sur les options du profil SSSD pour l'authentification par carte à puce, voir [Options d'authentification par carte à puce dans RHEL](#) .
2. A partir du serveur IdM, copiez le script dans un répertoire de votre choix sur la machine du client IdM :

```
[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. À partir du serveur IdM, copiez les fichiers de certificats d'autorité de certification au format PEM, pour plus de commodité, dans le même répertoire de la machine du client IdM que celui utilisé à l'étape précédente :

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

4. Sur l'ordinateur client, exécutez le script en ajoutant les fichiers PEM contenant les certificats d'autorité de certification en tant qu'arguments :

```
[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 rootca.pem subca.pem
issuingca.pem
Ticket cache:KEYRING:persistent:0:0
Default principal: admin@IDM.EXAMPLE.COM
[...]
Systemwide CA database updated.
The ipa-certupdate command was successful
```



NOTE

Assurez-vous que vous ajoutez le certificat de l'autorité de certification racine en tant qu'argument avant tout certificat d'autorité de certification secondaire et que les certificats de l'autorité de certification ou de l'autorité de certification secondaire n'ont pas expiré.

Le client est maintenant configuré pour l'authentification par carte à puce.

2.4. UTILISER ANSIBLE POUR CONFIGURER LES CLIENTS IDM POUR L'AUTHENTIFICATION PAR CARTE À PUCE

Cette section explique comment utiliser le module `ansible-freeipa ipasmartcard_client` pour configurer des clients Identity Management (IdM) spécifiques afin de permettre aux utilisateurs IdM de s'authentifier à l'aide d'une carte à puce. Exécutez cette procédure pour activer l'authentification par carte à puce pour les utilisateurs IdM qui utilisent l'un des éléments suivants pour accéder à IdM :

- Le protocole **ssh**
Pour plus d'informations, voir [Configuration de l'accès SSH à l'aide de l'authentification par carte à puce](#).
- Le login de la console
- Le gestionnaire d'affichage Gnome (GDM)
- La commande **su**



NOTE

Cette procédure n'est pas nécessaire pour s'authentifier auprès de l'interface Web IdM. L'authentification à l'interface Web IdM implique deux hôtes, dont aucun ne doit être un client IdM :

- La machine sur laquelle le navigateur s'exécute. La machine peut être en dehors du domaine IdM.
- Le serveur IdM sur lequel **httpd** est exécuté.

Conditions préalables

- Votre serveur IdM a été configuré pour l'authentification par carte à puce, comme décrit dans la section [Utilisation d'Ansible pour configurer le serveur IdM pour l'authentification par carte à puce](#).
- Vous disposez d'un accès root au serveur IdM et au client IdM.
- Vous disposez du certificat de l'autorité de certification racine, du certificat de l'autorité de certification IdM et de tous les certificats des autorités de certification intermédiaires.
- Vous avez configuré votre nœud de contrôle Ansible pour qu'il réponde aux exigences suivantes :
 - Vous utilisez la version 2.8 ou ultérieure d'Ansible.
 - Vous avez installé le paquetage **ansible-freeipa** sur le contrôleur Ansible.
 - L'exemple suppose que dans le répertoire `~/MyPlaybooks/` vous avez créé un [fichier d'inventaire Ansible](#) avec le nom de domaine complet (FQDN) du serveur IdM.
 - L'exemple suppose que le coffre-fort **secret.yml** Ansible stocke votre **ipadmin_password**.

Procédure

1. Si vos certificats d'autorité de certification sont stockés dans des fichiers d'un format différent, tel que **DER**, convertissez-les au format **PEM**:

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

Le certificat de l'autorité de certification IdM est au format **PEM** et se trouve dans le fichier **/etc/ipa/ca.crt**.

- En option, utilisez l'utilitaire **openssl x509** pour visualiser le contenu des fichiers au format **PEM** et vérifier que les valeurs **Issuer** et **Subject** sont correctes :

```
# openssl x509 -noout -text -in root-ca.pem | more
```

- Sur votre nœud de contrôle Ansible, naviguez jusqu'à votre répertoire `~/MyPlaybooks/` répertoire :

```
$ cd ~/MyPlaybooks/
```

- Créez un sous-répertoire dédié aux certificats d'autorité de certification :

```
$ mkdir SmartCard/
```

- Pour plus de commodité, copiez tous les certificats requis dans le répertoire `~/MyPlaybooks/SmartCard/`, par exemple :

```
# cp /tmp/root-ca.pem ~/MyPlaybooks/SmartCard/
# cp /tmp/intermediate-ca.pem ~/MyPlaybooks/SmartCard/
# cp /etc/ipa/ca.crt ~/MyPlaybooks/SmartCard/ipa-ca.crt
```

- Dans votre fichier d'inventaire Ansible, spécifiez ce qui suit :

- Les clients IdM que vous souhaitez configurer pour l'authentification par carte à puce.
- Le mot de passe de l'administrateur de l'IdM.
- Les chemins d'accès aux certificats des autorités de certification dans l'ordre suivant :
 - Le fichier du certificat de l'autorité de certification racine
 - Les fichiers des certificats de l'autorité de certification intermédiaire
 - Le fichier du certificat de l'autorité de certification IdM

Le fichier peut se présenter comme suit :

```
[ipaclients]
ipaclient1.example.com
ipaclient2.example.com

[ipaclients:vars]
ipaadmin_password=SomeADMINpassword
ipasmartcard_client_ca_certs=/home/<user_name>/MyPlaybooks/SmartCard/root-
ca.pem,/home/<user_name>/MyPlaybooks/SmartCard/intermediate-
ca.pem,/home/<user_name>/MyPlaybooks/SmartCard/ipa-ca.crt
```

- Créez un playbook **install-smartcard-clients.yml** avec le contenu suivant :

```
---
- name: Playbook to set up smart card authentication for an IdM client
  hosts: ipaclients
  become: true
```

```
roles:
- role: ipasmartcard_client
  state: present
```

8. Enregistrer le fichier.
9. Exécutez le playbook Ansible. Spécifiez le playbook et les fichiers d'inventaire :

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory install-smartcard-clients.yml
```

Le rôle **ipasmartcard_client** Ansible effectue les actions suivantes :

- Il configure le démon de la carte à puce.
- Il définit la réserve de confiance du système.
- Il configure le System Security Services Daemon (SSSD) pour permettre aux utilisateurs de s'authentifier soit avec leur nom d'utilisateur et leur mot de passe, soit avec leur carte à puce. Pour plus de détails sur les options de profil SSSD pour l'authentification par carte à puce, voir [Options d'authentification par carte à puce dans RHEL](#) .

Les clients répertoriés dans la section **ipaclients** du fichier d'inventaire sont maintenant configurés pour l'authentification par carte à puce.



NOTE

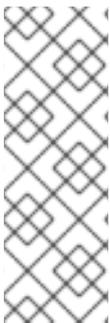
Si vous avez installé les clients IdM avec l'option **--mkhomedir**, les utilisateurs distants pourront se connecter à leur répertoire personnel. Sinon, l'emplacement de connexion par défaut est la racine de la structure de répertoires, `/`.

Ressources supplémentaires

- Exemples de playbooks utilisant le rôle **ipasmartcard_server** dans le répertoire `/usr/share/doc/ansible-freeipa/playbooks/`

2.5. AJOUT D'UN CERTIFICAT À UNE ENTRÉE UTILISATEUR DANS L'INTERFACE WEB IDM

Cette procédure décrit comment ajouter un certificat externe à une entrée utilisateur dans l'interface Web IdM.



NOTE

Au lieu de télécharger le certificat complet, il est également possible de télécharger des données de mappage de certificat vers une entrée d'utilisateur dans IdM. Les entrées utilisateur contenant des certificats complets ou des données de mappage de certificats peuvent être utilisées conjointement avec les règles de mappage de certificats correspondantes pour faciliter la configuration de l'authentification par carte à puce pour les administrateurs système. Pour plus de détails, voir [Règles de mappage de certificats pour la configuration de l'authentification par carte à puce](#).



NOTE

Si le certificat de l'utilisateur a été délivré par l'autorité de certification IdM, le certificat est déjà stocké dans l'entrée de l'utilisateur et vous pouvez ignorer cette section.

Conditions préalables

- Vous disposez du certificat que vous souhaitez ajouter à l'entrée de l'utilisateur.

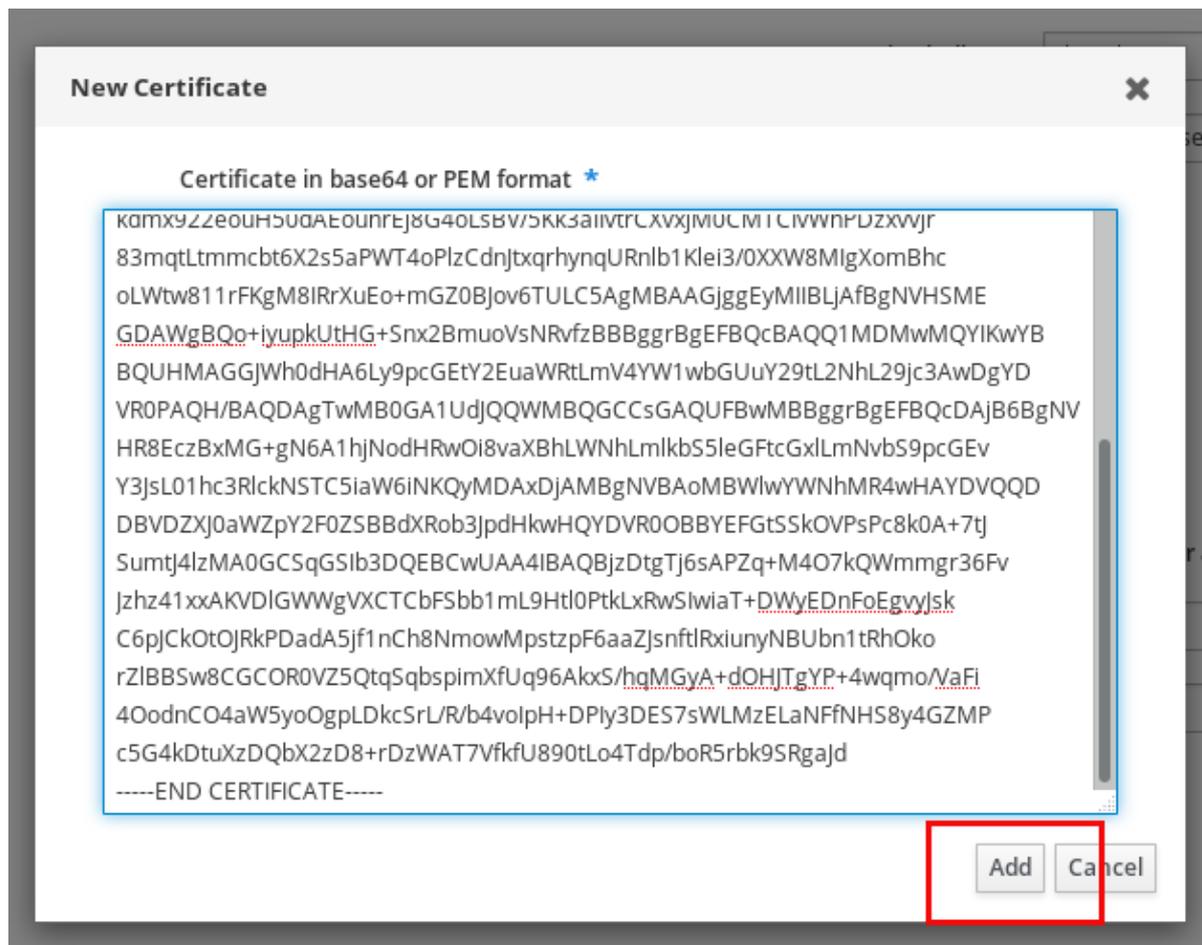
Procédure

1. Connectez-vous à l'interface Web IdM en tant qu'administrateur si vous souhaitez ajouter un certificat à un autre utilisateur. Pour ajouter un certificat à votre propre profil, vous n'avez pas besoin des informations d'identification de l'administrateur.
2. Naviguez vers **Users** → **Active users** → **sc_user**.
3. Recherchez l'option **Certificate** et cliquez sur **Add**.
4. Sur le site **Command-Line Interface**, affichez le certificat au format **PEM** à l'aide de l'utilitaire **cat** ou d'un éditeur de texte :

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

5. Copiez et collez le certificat de l'interface de gestion dans la fenêtre qui s'est ouverte dans l'interface Web.
6. Cliquez sur **Add**.

Figure 2.1. Ajout d'un nouveau certificat dans l'interface Web IdM



L'entrée **sc_user** contient maintenant un certificat externe.

2.6. AJOUT D'UN CERTIFICAT À UNE ENTRÉE UTILISATEUR DANS LA CLI IDM

Cette procédure décrit comment ajouter un certificat externe à une entrée utilisateur dans IdM CLI.



NOTE

Au lieu de télécharger le certificat complet, il est également possible de télécharger des données de mappage de certificat vers une entrée d'utilisateur dans IdM. Les entrées utilisateur contenant des certificats complets ou des données de mappage de certificats peuvent être utilisées conjointement avec les règles de mappage de certificats correspondantes pour faciliter la configuration de l'authentification par carte à puce pour les administrateurs système. Pour plus de détails, voir [Règles de mappage de certificats pour la configuration de l'authentification par carte à puce](#).



NOTE

Si le certificat de l'utilisateur a été délivré par l'autorité de certification IdM, le certificat est déjà stocké dans l'entrée de l'utilisateur et vous pouvez ignorer cette section.

Conditions préalables

- Vous disposez du certificat que vous souhaitez ajouter à l'entrée de l'utilisateur.

Procédure

1. Connectez-vous au CLI IdM en tant qu'administrateur si vous souhaitez ajouter un certificat à un autre utilisateur :

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

Pour ajouter un certificat à votre propre profil, vous n'avez pas besoin des informations d'identification de l'administrateur :

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

2. Créez une variable d'environnement contenant le certificat dont l'en-tête et le pied de page ont été supprimés et concaténés en une seule ligne, ce qui correspond au format attendu par la commande **ipa user-add-cert**:

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

Notez que le certificat dans le fichier **testuser.crt** doit être au format **PEM**.

3. Ajoutez le certificat au profil de l'utilisateur **sc_user** à l'aide de la commande **ipa user-add-cert**:

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

L'entrée **sc_user** contient maintenant un certificat externe.

2.7. INSTALLATION D'OUTILS DE GESTION ET D'UTILISATION DES CARTES À PUCE

Pour configurer votre carte à puce, vous avez besoin d'outils qui peuvent générer des certificats et les stocker sur une carte à puce.

Vous devez :

- Installez le paquet **gnutls-utils**, qui vous aide à gérer les certificats.
- Installez le paquetage **opensc**, qui fournit un ensemble de bibliothèques et d'utilitaires pour travailler avec des cartes à puce.
- Démarrez le service **pcscd**, qui communique avec le lecteur de cartes à puce.

Procédure

1. Installez les paquets **opensc** et **gnutls-utils**:

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

2. Démarrez le service **pcscd**.

```
# systemctl start pcscd
```

Vérifiez que le service **pcscd** est opérationnel.

2.8. PRÉPARATION DE VOTRE CARTE À PUCE ET TÉLÉCHARGEMENT DE VOS CERTIFICATS ET CLÉS SUR VOTRE CARTE À PUCE

Cette section décrit la configuration de la carte à puce avec l'outil **pkcs15-init**, qui vous aide à configurer :

- Effacer votre carte à puce
- Définition de nouveaux codes PIN et de clés de déblocage de code PIN (PUK) en option
- Création d'un nouvel emplacement sur la carte à puce
- Stockage du certificat, de la clé privée et de la clé publique dans la fente
- Si nécessaire, verrouiller les paramètres de la carte à puce, car certaines cartes à puce nécessitent ce type de finalisation



NOTE

L'outil **pkcs15-init** peut ne pas fonctionner avec toutes les cartes à puce. Vous devez utiliser les outils qui fonctionnent avec la carte à puce que vous utilisez.

Conditions préalables

- Le paquet **opensc**, qui comprend l'outil **pkcs15-init**, est installé.
Pour plus de détails, voir [Installation des outils de gestion et d'utilisation des cartes à puce](#).
- La carte est insérée dans le lecteur et connectée à l'ordinateur.
- Vous disposez de la clé privée, de la clé publique et du certificat à stocker sur la carte à puce. Dans cette procédure, **testuser.key**, **testuserpublic.key**, et **testuser.crt** sont les noms utilisés pour la clé privée, la clé publique et le certificat.
- Vous disposez du code PIN de l'utilisateur de votre carte à puce actuelle et du code PIN de l'agent de sécurité (SO-PIN).

Procédure

1. Effacez votre carte à puce et authentifiez-vous avec votre code PIN :

```
$ 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]:
```

La carte a été effacée.

2. Initialisez votre carte à puce, définissez votre code PIN et PUK d'utilisateur, ainsi que le code PIN et PUK de votre responsable de la sécurité :

```
$ 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
```

L'outil **pkcs15-init** crée un nouvel emplacement sur la carte à puce.

3. Définir l'étiquette et l'ID d'authentification pour l'emplacement :

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

L'étiquette est définie sur une valeur lisible par l'homme, dans ce cas, **testuser**. L'adresse **auth-id** doit être composée de deux valeurs hexadécimales ; dans ce cas, elle est fixée à **01**.

4. Stockez et étiquetez la clé privée dans le nouvel emplacement de la carte à puce :

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



NOTE

La valeur que vous indiquez pour **--id** doit être la même lorsque vous stockez votre clé privée et votre certificat à l'étape suivante. Il est recommandé de spécifier votre propre valeur pour **--id**, sinon l'outil calculera une valeur plus complexe.

5. Stockez et étiquetez le certificat dans le nouvel emplacement de la carte à puce :

```
$ 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. (Facultatif) Stockez et étiquetez la clé publique dans le nouvel emplacement de la carte à puce :

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



NOTE

Si la clé publique correspond à une clé privée ou à un certificat, indiquez le même ID que celui de la clé privée ou du certificat.

7. (Facultatif) Certaines cartes à puce exigent que vous finalisiez la carte en verrouillant les paramètres :

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

À ce stade, votre carte à puce comprend le certificat, la clé privée et la clé publique dans l'emplacement nouvellement créé. Vous avez également créé votre code PIN et PUK d'utilisateur ainsi que le code PIN et PUK de l'agent de sécurité.

2.9. CONNEXION À L'IDM AVEC DES CARTES À PUCE

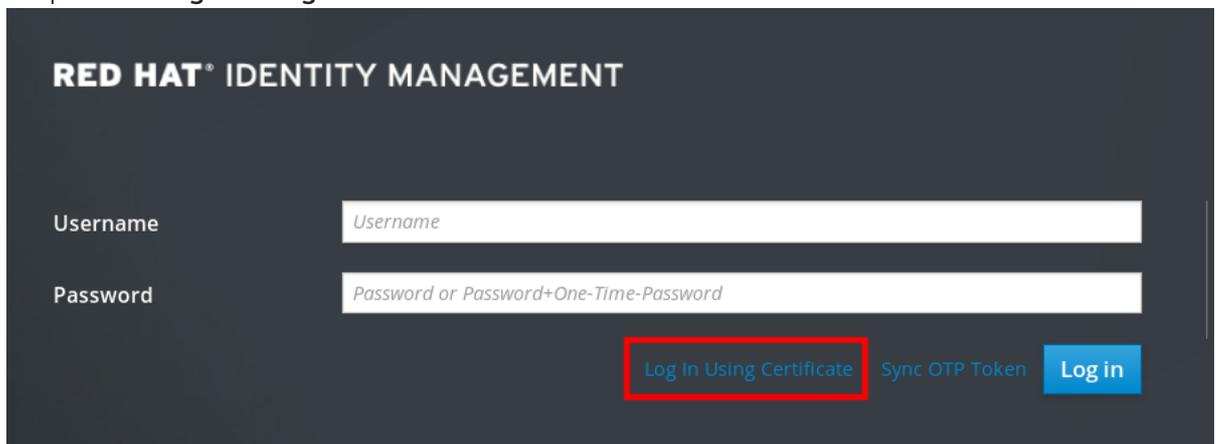
Cette section décrit l'utilisation des cartes à puce pour se connecter à l'interface Web IdM.

Conditions préalables

- Le navigateur web est configuré pour utiliser l'authentification par carte à puce.
- Le serveur IdM est configuré pour l'authentification par carte à puce.
- Le certificat installé sur votre carte à puce est soit émis par le serveur IdM, soit ajouté à l'entrée de l'utilisateur dans IdM.
- Vous connaissez le code PIN requis pour déverrouiller la carte à puce.
- La carte à puce a été insérée dans le lecteur.

Procédure

1. Ouvrez l'interface Web IdM dans le navigateur.
2. Cliquez sur **Log In Using Certificate**



3. Si la boîte de dialogue **Password Required** s'ouvre, ajoutez le code PIN pour déverrouiller la carte à puce et cliquez sur le bouton **OK**.
La boîte de dialogue **User Identification Request** s'ouvre.

Si la carte à puce contient plus d'un certificat, sélectionnez le certificat que vous souhaitez utiliser pour l'authentification dans la liste déroulante située sous **Choose a certificate to present as identification**.

4. Cliquez sur le bouton **OK**.

Vous êtes maintenant connecté avec succès à l'interface Web IdM.

The screenshot shows the Red Hat Identity Management (IdM) web interface. The top navigation bar includes 'Identity', 'Policy', 'Authentication', 'Network Services', and 'IPA Server'. Below this, there are sub-tabs for 'Users', 'Hosts', 'Services', 'Groups', 'ID Views', and 'Automember'. The 'Users' sub-tab is active, and the 'Active users' category is selected in the left sidebar. The main content area is titled 'Active users' and contains a search bar, a 'Refresh' button, and buttons for 'Delete', '+ Add', '- Disable', '✓ Enable', and 'Actions'. A table lists the active users:

	User login	First name	Last name	Status	UID	Email address	Telephone Number	Job Title
<input type="checkbox"/>	admin		Administrator	✓ Enabled	427200000			

Showing 1 to 1 of 1 entries.

2.10. SE CONNECTER À GDM EN UTILISANT L'AUTHENTIFICATION PAR CARTE À PUCE SUR UN CLIENT IDM

Le Gnome Desktop Manager (GDM) nécessite une authentification. Vous pouvez utiliser votre mot de passe, mais vous pouvez également utiliser une carte à puce pour l'authentification.

Cette section décrit l'authentification par carte à puce pour accéder à GDM.

Conditions préalables

- Le système a été configuré pour l'authentification par carte à puce. Pour plus de détails, voir [Configuration du client IdM pour l'authentification par carte à puce](#).
- La carte à puce contient votre certificat et votre clé privée.
- Le compte d'utilisateur est membre du domaine IdM.
- Le certificat de la carte à puce correspond à l'entrée de l'utilisateur :
 - Affecter le certificat à une entrée utilisateur particulière. Pour plus de détails, voir [Ajouter un certificat à une entrée utilisateur dans l'interface Web IdM](#) ou [Ajouter un certificat à une entrée utilisateur dans l'interface CLI IdM](#).
 - Les données de mappage de certificats appliquées au compte. Pour plus de détails, voir [Règles de mappage de certificats pour la configuration de l'authentification par carte à puce](#).

Procédure

1. Insérez la carte à puce dans le lecteur.
2. Saisissez le code PIN de la carte à puce.
3. Cliquez sur **Sign In**.

Vous êtes connecté avec succès au système RHEL et vous disposez d'un TGT fourni par le serveur IdM.

Verification steps

- Dans la fenêtre **Terminal**, entrez **klist** et vérifiez le résultat :

```
$ 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
```

2.11. UTILISATION DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC LA COMMANDE SU

Le passage à un autre utilisateur nécessite une authentification. Vous pouvez utiliser un mot de passe ou un certificat. Cette section décrit l'utilisation de votre carte à puce avec la commande **su**. Cela signifie qu'après avoir entré la commande **su**, vous êtes invité à saisir le code PIN de la carte à puce.

Conditions préalables

- Votre serveur et votre client IdM ont été configurés pour l'authentification par carte à puce.
 - Voir [Configuration du serveur IdM pour l'authentification par carte à puce](#)
 - Voir [Configuration du client IdM pour l'authentification par carte à puce](#)
- La carte à puce contient votre certificat et votre clé privée. Voir [Stocker un certificat sur une carte à puce](#)
- La carte est insérée dans le lecteur et connectée à l'ordinateur.

Procédure

- Dans une fenêtre de terminal, changez d'utilisateur à l'aide de la commande **su**:

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

Si la configuration est correcte, vous êtes invité à saisir le code PIN de la carte à puce.

CHAPITRE 3. CONFIGURATION DES CERTIFICATS ÉMIS PAR ADCS POUR L'AUTHENTIFICATION PAR CARTE À PUCE DANS IDM

Ce scénario décrit la situation suivante :

- Votre déploiement est basé sur une confiance inter-forêts entre Identity Management (IdM) et Active Directory (AD).
- Vous souhaitez autoriser l'authentification par carte à puce pour les utilisateurs dont les comptes sont stockés dans AD.
- Les certificats sont créés et stockés dans Active Directory Certificate Services (ADCS).

Pour une vue d'ensemble de l'authentification par carte à puce, voir [Comprendre l'authentification par carte à puce](#).

La configuration s'effectue selon les étapes suivantes :

- [Copie des certificats d'autorité de certification et d'utilisateur d'Active Directory vers le serveur et le client IdM](#)
- [Configuration du serveur IdM et des clients pour l'authentification par carte à puce à l'aide de certificats ADCS](#)
- [Conversion d'un fichier PFX \(PKCS#12\) pour pouvoir stocker le certificat et la clé privée dans la carte à puce](#)
- [Configuration des délais d'attente dans le fichier sssd.conf](#)
- [Création de règles de mappage de certificats pour l'authentification par carte à puce](#)

Conditions préalables

- La gestion des identités (IdM) et la confiance dans Active Directory (AD) sont installées. Pour plus de détails, voir [Installer la confiance entre IdM et AD](#).
- Active Directory Certificate Services (ADCS) est installé et les certificats pour les utilisateurs sont générés.

3.1. PARAMÈTRES DU SERVEUR WINDOWS REQUIS POUR LA CONFIGURATION DE LA CONFIANCE ET L'UTILISATION DU CERTIFICAT

Cette section résume ce qui doit être configuré sur Windows Server :

- Active Directory Certificate Services (ADCS) est installé
- L'autorité de certification est créée
- [Facultatif] Si vous utilisez l'inscription Web de l'autorité de certification, les services d'information Internet (IIS) doivent être configurés

Exporter le certificat :

- La clé doit avoir **2048** bits ou plus
- Inclure une clé privée
- Vous aurez besoin d'un certificat au format suivant : Échange d'informations personnelles -**PKCS #12(.PFX)**
 - Activer la confidentialité des certificats

3.2. COPIER DES CERTIFICATS À PARTIR D'ACTIVE DIRECTORY À L'AIDE DE SFTP

Pour pouvoir utiliser l'authentification par carte à puce, vous devez copier les fichiers de certificats suivants :

- Un certificat d'autorité de certification racine au format **CER: adcs-winservice-ca.cer** sur votre serveur IdM.
- Un certificat d'utilisateur avec une clé privée au format **PFX: aduser1.pfx** sur un client IdM.



NOTE

Cette procédure suppose que l'accès SSH est autorisé. Si SSH n'est pas disponible, l'utilisateur doit copier le fichier du serveur AD vers le serveur IdM et le client.

Procédure

1. Connectez-vous à partir de **the IdM server** et copiez le certificat racine de **adcs-winservice-ca.cer** sur le serveur IdM :

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

2. Connectez-vous à partir de **the IdM client** et copiez le certificat d'utilisateur de **aduser1.pfx** sur le client :

```
[root@client1 ~]# sftp Administrator@winservice.ad.example.com
Administrator@winservice.ad.example.com's password:
Connected to Administrator@winservice.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
```

Le certificat de l'autorité de certification est stocké dans le serveur IdM et les certificats des utilisateurs sont stockés sur la machine du client.

3.3. CONFIGURATION DU SERVEUR IDM ET DES CLIENTS POUR L'AUTHENTIFICATION PAR CARTE À PUCE À L'AIDE DE CERTIFICATS ADCS

Vous devez configurer le serveur IdM (Identity Management) et les clients pour pouvoir utiliser l'authentification par carte à puce dans l'environnement IdM. IdM inclut les scripts **ipa-advise** qui effectuent tous les changements nécessaires :

- installer les paquets nécessaires
- il configure le serveur et les clients IdM
- copier les certificats de l'autorité de certification dans les emplacements prévus

Vous pouvez exécuter **ipa-advise** sur votre serveur IdM.

Cette procédure décrit

- Sur un serveur IdM : Préparation du script **ipa-advise** pour configurer votre serveur IdM pour l'authentification par carte à puce.
- Sur un serveur IdM : Préparation du script **ipa-advise** pour configurer votre client IdM pour l'authentification par carte à puce.
- Sur un serveur IdM : Appliquer le script du serveur **ipa-advise** sur le serveur IdM en utilisant le certificat AD.
- Déplacement du script client vers la machine client IdM.
- Sur un client IdM : Appliquer le script du client **ipa-advise** sur le client IdM en utilisant le certificat AD.

Conditions préalables

- Le certificat a été copié sur le serveur IdM.
- Obtenir le ticket Kerberos.
- Connectez-vous en tant qu'utilisateur disposant de droits d'administration.

Procédure

1. Sur le serveur IdM, utilisez le script **ipa-advise** pour configurer un client :

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

2. Sur le serveur IdM, utilisez le script **ipa-advise** pour configurer un serveur :

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

3. Sur le serveur IdM, exécuter le script :

■

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

- Il configure le serveur HTTP Apache de l'IdM.
- Il active la cryptographie à clé publique pour l'authentification initiale dans Kerberos (PKINIT) sur le centre de distribution de clés (KDC).
- Il configure l'interface Web IdM pour qu'elle accepte les demandes d'autorisation de carte à puce.

4. Copiez le script **sc_client.sh** sur le système client :

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

5. Copiez le certificat Windows sur le système client :

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

6. Sur le système client, exécutez le script client :

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

Le certificat de l'autorité de certification est installé dans le bon format sur le serveur IdM et les systèmes clients, et l'étape suivante consiste à copier les certificats des utilisateurs sur la carte à puce elle-même.

3.4. CONVERSION DU FICHIER PFX

Avant d'enregistrer le fichier PFX (PKCS#12) dans la carte à puce, vous devez

- convertir le fichier au format PEM
- extraire la clé privée et le certificat dans deux fichiers différents

Conditions préalables

- Le fichier PFX est copié sur la machine du client IdM.

Procédure

1. Sur le client IdM, dans le format PEM :

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

2. Extraire la clé dans un fichier séparé :

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

3. Extraire le certificat public dans un fichier séparé :

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

À ce stade, vous pouvez enregistrer les adresses **aduser1.key** et **aduser1.crt** dans la carte à puce.

3.5. INSTALLATION D'OUTILS DE GESTION ET D'UTILISATION DES CARTES À PUCE

Pour configurer votre carte à puce, vous avez besoin d'outils qui peuvent générer des certificats et les stocker sur une carte à puce.

Vous devez :

- Installez le paquet **gnutls-utils**, qui vous aide à gérer les certificats.
- Installez le paquetage **opensc**, qui fournit un ensemble de bibliothèques et d'utilitaires pour travailler avec des cartes à puce.
- Démarrez le service **pcscd**, qui communique avec le lecteur de cartes à puce.

Procédure

1. Installez les paquets **opensc** et **gnutls-utils**:

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

2. Démarrez le service **pcscd**.

```
# systemctl start pcscd
```

Vérifiez que le service **pcscd** est opérationnel.

3.6. PRÉPARATION DE VOTRE CARTE À PUCE ET TÉLÉCHARGEMENT DE VOS CERTIFICATS ET CLÉS SUR VOTRE CARTE À PUCE

Cette section décrit la configuration de la carte à puce avec l'outil **pkcs15-init**, qui vous aide à configurer :

- Effacer votre carte à puce
- Définition de nouveaux codes PIN et de clés de déblocage de code PIN (PUK) en option
- Création d'un nouvel emplacement sur la carte à puce
- Stockage du certificat, de la clé privée et de la clé publique dans la fente
- Si nécessaire, verrouiller les paramètres de la carte à puce, car certaines cartes à puce nécessitent ce type de finalisation



NOTE

L'outil **pkcs15-init** peut ne pas fonctionner avec toutes les cartes à puce. Vous devez utiliser les outils qui fonctionnent avec la carte à puce que vous utilisez.

Conditions préalables

- Le paquet **opensc**, qui comprend l'outil **pkcs15-init**, est installé.
Pour plus de détails, voir [Installation des outils de gestion et d'utilisation des cartes à puce](#).
- La carte est insérée dans le lecteur et connectée à l'ordinateur.
- Vous disposez de la clé privée, de la clé publique et du certificat à stocker sur la carte à puce. Dans cette procédure, **testuser.key**, **testuserpublic.key**, et **testuser.crt** sont les noms utilisés pour la clé privée, la clé publique et le certificat.
- Vous disposez du code PIN de l'utilisateur de votre carte à puce actuelle et du code PIN de l'agent de sécurité (SO-PIN).

Procédure

1. Effacez votre carte à puce et authentifiez-vous avec votre code PIN :

```
$ 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]:
```

La carte a été effacée.

2. Initialisez votre carte à puce, définissez votre code PIN et PUK d'utilisateur, ainsi que le code PIN et PUK de votre responsable de la sécurité :

```
$ 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
```

L'outil **pkcs15-init** crée un nouvel emplacement sur la carte à puce.

3. Définir l'étiquette et l'ID d'authentification pour l'emplacement :

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

L'étiquette est définie sur une valeur lisible par l'homme, dans ce cas, **testuser**. L'adresse **auth-id** doit être composée de deux valeurs hexadécimales ; dans ce cas, elle est fixée à **01**.

4. Stockez et étiquetez la clé privée dans le nouvel emplacement de la carte à puce :

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



NOTE

La valeur que vous indiquez pour **--id** doit être la même lorsque vous stockez votre clé privée et votre certificat à l'étape suivante. Il est recommandé de spécifier votre propre valeur pour **--id**, sinon l'outil calculera une valeur plus complexe.

5. Stockez et étiquetez le certificat dans le nouvel emplacement de la carte à puce :

```
$ 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. (Facultatif) Stockez et étiquetez la clé publique dans le nouvel emplacement de la carte à puce :

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



NOTE

Si la clé publique correspond à une clé privée ou à un certificat, indiquez le même ID que celui de la clé privée ou du certificat.

7. (Facultatif) Certaines cartes à puce exigent que vous finalisiez la carte en verrouillant les paramètres :

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

À ce stade, votre carte à puce comprend le certificat, la clé privée et la clé publique dans l'emplacement nouvellement créé. Vous avez également créé votre code PIN et PUK d'utilisateur ainsi que le code PIN et PUK de l'agent de sécurité.

3.7. CONFIGURATION DES DÉLAIS D'ATTENTE DANS SSSD.CONF

L'authentification à l'aide d'un certificat de carte à puce peut prendre plus de temps que les délais par défaut utilisés par SSSD. L'expiration du délai peut être causée par :

- lecteur lent
- un transfert d'un dispositif physique vers un environnement virtuel
- trop de certificats stockés sur la carte à puce
- réponse lente du répondeur OCSP (Online Certificate Status Protocol) si OCSP est utilisé pour vérifier les certificats

Dans ce cas, vous pouvez prolonger les délais suivants dans le fichier **sssd.conf**, par exemple jusqu'à 60 secondes :

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

Conditions préalables

- Vous devez être connecté en tant que root.

Procédure

1. Ouvrez le fichier **sssd.conf**:

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

2. Modifier la valeur de **p11_child_timeout**:

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

3. Modifier la valeur de **krb5_auth_timeout**:

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

4. Sauvegarder les paramètres.

Maintenant, l'interaction avec la carte à puce est autorisée pendant 1 minute (60 secondes) avant que l'authentification n'échoue avec un délai d'attente.

3.8. CRÉATION DE RÈGLES DE MAPPAGE DE CERTIFICATS POUR L'AUTHENTIFICATION PAR CARTE À PUCE

Si vous souhaitez utiliser un seul certificat pour un utilisateur qui possède des comptes dans AD (Active Directory) et dans IdM (Identity Management), vous pouvez créer une règle de mappage des certificats sur le serveur IdM.

Après avoir créé une telle règle, l'utilisateur peut s'authentifier avec sa carte à puce dans les deux domaines.

Pour plus d'informations sur les règles de mappage des certificats, voir [Règles de mappage des certificats pour la configuration de l'authentification sur les cartes à puce](#).

CHAPITRE 4. RÈGLES DE MAPPAGE DES CERTIFICATS POUR LA CONFIGURATION DE L'AUTHENTIFICATION SUR LES CARTES À PUCE

Les règles de mappage de certificats sont un moyen pratique de permettre aux utilisateurs de s'authentifier à l'aide de certificats dans des scénarios où l'administrateur de la gestion des identités (IdM) n'a pas accès aux certificats de certains utilisateurs. Ce manque d'accès est généralement dû au fait que les certificats ont été délivrés par une autorité de certification externe. Un cas d'utilisation particulier est représenté par les certificats émis par le système de certification d'un Active Directory (AD) avec lequel le domaine IdM entretient une relation de confiance.

Les règles de mappage des certificats sont également pratiques si l'environnement IdM est vaste et que de nombreux utilisateurs utilisent des cartes à puce. Dans ce cas, l'ajout de certificats complets peut s'avérer compliqué. Le sujet et l'émetteur sont prévisibles dans la plupart des scénarios et donc plus faciles à ajouter à l'avance que le certificat complet. En tant qu'administrateur système, vous pouvez créer une règle de mappage de certificats et ajouter des données de mappage de certificats à une entrée utilisateur avant même qu'un certificat ne soit délivré à un utilisateur particulier. Une fois le certificat émis, l'utilisateur peut se connecter à l'aide du certificat, même si le certificat complet n'a pas encore été téléchargé dans l'entrée utilisateur.

En outre, comme les certificats doivent être renouvelés à intervalles réguliers, les règles de mappage des certificats réduisent la charge administrative. Lorsque le certificat d'un utilisateur est renouvelé, l'administrateur ne doit pas mettre à jour l'entrée de l'utilisateur. Par exemple, si le mappage est basé sur les valeurs **Subject** et **Issuer**, et si le nouveau certificat a le même sujet et le même émetteur que l'ancien, le mappage s'applique toujours. En revanche, si le certificat complet est utilisé, l'administrateur doit télécharger le nouveau certificat dans l'entrée utilisateur pour remplacer l'ancien.

Pour configurer le mappage des certificats :

1. Un administrateur doit charger les données de mappage du certificat (généralement l'émetteur et le sujet) ou le certificat complet dans un compte utilisateur.
2. Un administrateur doit créer une règle de mappage de certificats pour permettre à un utilisateur de se connecter avec succès à l'IdM
 - a. dont le compte contient une entrée de données de mappage de certificats
 - b. dont la saisie des données de mappage du certificat correspond aux informations figurant sur le certificat

Pour plus d'informations sur les différents composants d'une règle de correspondance et sur la manière de les obtenir et de les utiliser, voir [Composants d'une règle de correspondance des identités dans IdM](#) et [Obtention de l'émetteur d'un certificat en vue de son utilisation dans une règle de correspondance](#).

Ensuite, lorsque l'utilisateur final présente le certificat, stocké soit dans le [système de fichiers](#), soit sur une [carte à puce](#), l'authentification est réussie.

4.1. RÈGLES DE MAPPAGE DES CERTIFICATS POUR LES TRUSTS AVEC LES DOMAINES ACTIVE DIRECTORY

Cette section décrit les différents cas d'utilisation du mappage de certificats qui sont possibles si un déploiement IdM est en relation de confiance avec un domaine Active Directory (AD).

Les règles de mappage des certificats sont un moyen pratique d'autoriser l'accès aux ressources IdM pour les utilisateurs qui possèdent des certificats de carte à puce émis par le système de certification AD de confiance. Selon la configuration d'AD, les scénarios suivants sont possibles :

- Si le certificat est émis par AD mais que l'utilisateur et le certificat sont stockés dans IdM, le mappage et l'ensemble du traitement de la demande d'authentification ont lieu du côté d'IdM. Pour plus de détails sur la configuration de ce scénario, voir [Configuration du mappage des certificats pour les utilisateurs stockés dans IdM](#)
- Si l'utilisateur est enregistré dans AD, le traitement de la demande d'authentification a lieu dans AD. Il existe trois sous-cas différents :
 - L'entrée utilisateur AD contient l'intégralité du certificat. Pour plus de détails sur la configuration de l'IdM dans ce scénario, voir [Configuration du mappage de certificats pour les utilisateurs dont l'entrée utilisateur AD contient le certificat entier](#).
 - AD est configuré pour associer des certificats d'utilisateur à des comptes d'utilisateur. Dans ce cas, l'entrée utilisateur AD ne contient pas l'intégralité du certificat, mais un attribut appelé **altSecurityIdentities**. Pour plus d'informations sur la configuration de l'IdM dans ce scénario, voir [Configuration du mappage de certificats si AD est configuré pour mapper des certificats d'utilisateur sur des comptes d'utilisateur](#).
 - L'entrée de l'utilisateur AD ne contient ni le certificat complet ni les données de mappage. Dans ce cas, la seule solution consiste à utiliser la commande **ipa idoverrideuser-add** pour ajouter le certificat complet à l'ID override de l'utilisateur AD dans IdM. Pour plus de détails, voir [Configuration du mappage de certificats si l'entrée de l'utilisateur AD ne contient pas de certificat ou de données de mappage](#).

4.2. COMPOSANTS D'UNE RÈGLE DE MAPPAGE D'IDENTITÉ DANS IDM

Cette section décrit les composants d'un site *identity mapping rule* dans IdM et la manière de les configurer. Chaque composant a une valeur par défaut que vous pouvez remplacer. Vous pouvez définir les composants dans l'interface Web ou dans l'interface de ligne de commande. Dans la CLI, la règle de mappage d'identité est créée à l'aide de la commande **ipa certmaprule-add**.

Règle de cartographie

Le composant règle de mappage associe (ou *maps*) un certificat à un ou plusieurs comptes d'utilisateurs. La règle définit un filtre de recherche LDAP qui associe un certificat au compte d'utilisateur voulu.

Les certificats délivrés par différentes autorités de certification (AC) peuvent avoir des propriétés différentes et être utilisés dans des domaines différents. C'est pourquoi l'IdM n'applique pas les règles de mappage de manière inconditionnelle, mais uniquement aux certificats appropriés. Les certificats appropriés sont définis à l'aide de *matching rules*.

Notez que si vous laissez l'option "mapping rule" vide, les certificats sont recherchés dans l'attribut **userCertificate** sous la forme d'un fichier binaire encodé en DER.

Définissez la règle de mappage dans le CLI en utilisant l'option **--maprule**.

Règle de correspondance

Le composant règle de correspondance sélectionne un certificat auquel vous souhaitez appliquer la règle de correspondance. La règle de correspondance par défaut fait correspondre les certificats avec l'utilisation **digitalSignature key** et **clientAuth extended key**.

Définissez la règle de correspondance dans le CLI en utilisant l'option **--matchrule**.

Liste des domaines

La liste des domaines spécifie les domaines d'identité dans lesquels vous souhaitez que l'IdM recherche les utilisateurs lors du traitement des règles de mappage d'identité. Si l'option n'est pas spécifiée, l'IdM recherche les utilisateurs uniquement dans le domaine local auquel appartient le client IdM.

Définissez le domaine dans le CLI en utilisant l'option **--domain**.

Priorité

Lorsque plusieurs règles s'appliquent à un certificat, la règle ayant la priorité la plus élevée est prioritaire. Toutes les autres règles sont ignorées.

- Plus la valeur numérique est faible, plus la priorité de la règle de mise en correspondance des identités est élevée. Par exemple, une règle de priorité 1 est plus prioritaire qu'une règle de priorité 2.
- Si une règle n'a pas de valeur de priorité définie, elle a la priorité la plus basse.

Définissez la priorité de la règle de mappage dans le CLI à l'aide de l'option **--priority**.

Exemple de règle de mappage de certificats

Définir, à l'aide de la CLI, une règle de mappage de certificats appelée **simple_rule** qui autorise l'authentification d'un certificat émis par **Smart Card CA** de l'organisation **EXAMPLE.ORG** tant que le **Subject** de ce certificat correspond à une entrée **certmapdata** dans un compte d'utilisateur de l'IdM :

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

4.3. OBTENIR L'ÉMETTEUR D'UN CERTIFICAT POUR L'UTILISER DANS UNE RÈGLE DE CORRESPONDANCE

Cette procédure décrit comment obtenir les informations relatives à l'émetteur d'un certificat afin de pouvoir les copier et les coller dans la règle de correspondance d'une règle de mappage de certificats. Pour obtenir le format de l'émetteur requis par une règle de correspondance, utilisez l'utilitaire **openssl x509**.

Conditions préalables

- Vous disposez du certificat d'utilisateur au format **.pem** ou **.crt**

Procédure

1. Obtenez les informations sur l'utilisateur à partir du certificat. Utilisez l'utilitaire d'affichage et de signature de certificats **openssl x509** avec :
 - l'option **-noout** pour empêcher la sortie d'une version encodée de la requête

- l'option **-issuer** pour éditer le nom de l'émetteur
- l'option **-in** pour spécifier le nom du fichier d'entrée à partir duquel le certificat doit être lu
- l'option **-nameopt** avec la valeur **RFC2253** pour afficher la sortie avec le nom distinctif relatif (RDN) le plus spécifique en premier
Si le fichier d'entrée contient un certificat de gestion d'identité, la sortie de la commande montre que l'émetteur est défini à l'aide des informations de **Organisation**:

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

Si le fichier d'entrée contient un certificat Active Directory, la sortie de la commande montre que l'émetteur est défini à l'aide des informations de **Domain Component**:

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

2. Optionnellement, pour créer une nouvelle règle de mappage dans le CLI basée sur une règle de correspondance qui spécifie que l'émetteur du certificat doit être le **AD-WIN2012R2-CA** extrait du domaine **ad.example.com** et que le sujet du certificat doit correspondre à l'entrée **certmapdata** dans un compte d'utilisateur dans l'IdM :

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

4.4. RESSOURCES SUPPLÉMENTAIRES

- Voir la page de manuel **sss-certmap(5)**.

CHAPITRE 5. CONFIGURATION DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC LA CONSOLE WEB POUR LES UTILISATEURS GÉRÉS DE MANIÈRE CENTRALISÉE

Configurer l'authentification par carte à puce dans la console web RHEL pour les utilisateurs qui sont gérés de manière centralisée par :

- Gestion de l'identité
- Active Directory, qui est relié à la gestion des identités dans le cadre de la confiance inter-forêts

Conditions préalables

- Le système pour lequel vous souhaitez utiliser l'authentification par carte à puce doit être membre d'un domaine Active Directory ou Identity Management.
- Le certificat utilisé pour l'authentification par carte à puce doit être associé à un utilisateur particulier dans Identity Management ou Active Directory.
Pour plus de détails sur l'association d'un certificat à l'utilisateur dans la gestion des identités, voir [Ajouter un certificat à une entrée utilisateur dans l'interface Web Id M](#) ou [Ajouter un certificat à une entrée utilisateur dans l'interface CLI IdM](#).

5.1. AUTHENTIFICATION PAR CARTE À PUCE POUR LES UTILISATEURS GÉRÉS DE MANIÈRE CENTRALISÉE

Une carte à puce est un dispositif physique qui peut fournir une authentification personnelle à l'aide de certificats stockés sur la carte. L'authentification personnelle signifie que vous pouvez utiliser les cartes à puce de la même manière que les mots de passe des utilisateurs.

Vous pouvez stocker les informations d'identification de l'utilisateur sur la carte à puce sous la forme d'une clé privée et d'un certificat. Un logiciel et un matériel spécifiques sont utilisés pour y accéder. Vous insérez la carte à puce dans un lecteur ou une prise USB et fournissez le code PIN de la carte à puce au lieu de votre mot de passe.

La gestion des identités (IdM) prend en charge l'authentification par carte à puce avec :

- Certificats d'utilisateur délivrés par l'autorité de certification de l'IdM.
- Certificats d'utilisateur émis par l'autorité de certification Active Directory Certificate Service (ADCS).



NOTE

Si vous souhaitez commencer à utiliser l'authentification par carte à puce, consultez la configuration matérielle requise : [Prise en charge des cartes à puce dans RHEL8](#).

5.2. INSTALLATION D'OUTILS DE GESTION ET D'UTILISATION DES CARTES À PUCE

Pour configurer votre carte à puce, vous avez besoin d'outils qui peuvent générer des certificats et les stocker sur une carte à puce.

Vous devez :

- Installez le paquet **gnutls-utils**, qui vous aide à gérer les certificats.
- Installez le paquetage **opensc**, qui fournit un ensemble de bibliothèques et d'utilitaires pour travailler avec des cartes à puce.
- Démarrez le service **pcscd**, qui communique avec le lecteur de cartes à puce.

Procédure

1. Installez les paquets **opensc** et **gnutls-utils**:

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

2. Démarrez le service **pcscd**.

```
# systemctl start pcscd
```

Vérifiez que le service **pcscd** est opérationnel.

5.3. PRÉPARATION DE VOTRE CARTE À PUCE ET TÉLÉCHARGEMENT DE VOS CERTIFICATS ET CLÉS SUR VOTRE CARTE À PUCE

Cette section décrit la configuration de la carte à puce avec l'outil **pkcs15-init**, qui vous aide à configurer :

- Effacer votre carte à puce
- Définition de nouveaux codes PIN et de clés de déblocage de code PIN (PUK) en option
- Création d'un nouvel emplacement sur la carte à puce
- Stockage du certificat, de la clé privée et de la clé publique dans la fente
- Si nécessaire, verrouiller les paramètres de la carte à puce, car certaines cartes à puce nécessitent ce type de finalisation



NOTE

L'outil **pkcs15-init** peut ne pas fonctionner avec toutes les cartes à puce. Vous devez utiliser les outils qui fonctionnent avec la carte à puce que vous utilisez.

Conditions préalables

- Le paquet **opensc**, qui comprend l'outil **pkcs15-init**, est installé.
Pour plus de détails, voir [Installation des outils de gestion et d'utilisation des cartes à puce](#) .
- La carte est insérée dans le lecteur et connectée à l'ordinateur.
- Vous disposez de la clé privée, de la clé publique et du certificat à stocker sur la carte à puce.
Dans cette procédure, **testuser.key**, **testuserpublic.key**, et **testuser.crt** sont les noms utilisés pour la clé privée, la clé publique et le certificat.
- Vous disposez du code PIN de l'utilisateur de votre carte à puce actuelle et du code PIN de l'agent de sécurité (SO-PIN).

Procédure

1. Effacez votre carte à puce et authentifiez-vous avec votre code PIN :

```
$ 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]:
```

La carte a été effacée.

2. Initialisez votre carte à puce, définissez votre code PIN et PUK d'utilisateur, ainsi que le code PIN et PUK de votre responsable de la sécurité :

```
$ 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
```

L'outil **pkcs15-init** crée un nouvel emplacement sur la carte à puce.

3. Définir l'étiquette et l'ID d'authentification pour l'emplacement :

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

L'étiquette est définie sur une valeur lisible par l'homme, dans ce cas, **testuser**. L'adresse **auth-id** doit être composée de deux valeurs hexadécimales ; dans ce cas, elle est fixée à **01**.

4. Stockez et étiquetez la clé privée dans le nouvel emplacement de la carte à puce :

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



NOTE

La valeur que vous indiquez pour **--id** doit être la même lorsque vous stockez votre clé privée et votre certificat à l'étape suivante. Il est recommandé de spécifier votre propre valeur pour **--id**, sinon l'outil calculera une valeur plus complexe.

5. Stockez et étiquetez le certificat dans le nouvel emplacement de la carte à puce :

```
$ 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. (Facultatif) Stockez et étiquetez la clé publique dans le nouvel emplacement de la carte à puce :

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

**NOTE**

Si la clé publique correspond à une clé privée ou à un certificat, indiquez le même ID que celui de la clé privée ou du certificat.

7. (Facultatif) Certaines cartes à puce exigent que vous finalisiez la carte en verrouillant les paramètres :

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

À ce stade, votre carte à puce comprend le certificat, la clé privée et la clé publique dans l'emplacement nouvellement créé. Vous avez également créé votre code PIN et PUK d'utilisateur ainsi que le code PIN et PUK de l'agent de sécurité.

5.4. ACTIVATION DE L'AUTHENTIFICATION PAR CARTE À PUCE POUR LA CONSOLE WEB

Pour pouvoir utiliser l'authentification par carte à puce dans la console web, activez l'authentification par carte à puce dans le fichier **cockpit.conf**.

En outre, vous pouvez désactiver l'authentification par mot de passe dans le même fichier.

Conditions préalables

- La console web RHEL a été installée.

Procédure

1. Connectez-vous à la console web RHEL avec des privilèges d'administrateur.
2. Cliquez sur **Terminal**.
3. Dans le site **/etc/cockpit/cockpit.conf**, le site **ClientCertAuthentication** est remplacé par le site **yes**:

```
[WebService]
ClientCertAuthentication = yes
```

4. Il est possible de désactiver l'authentification par mot de passe dans **cockpit.conf** avec :

```
[Basic]
action = none
```

Cette configuration désactive l'authentification par mot de passe et vous devez toujours utiliser la carte à puce.

5. Redémarrez la console web pour vous assurer que le site **cockpit.service** accepte la modification :

```
# systemctl restart cockpit
```

5.5. SE CONNECTER À LA CONSOLE WEB AVEC DES CARTES À PUCE

Vous pouvez utiliser des cartes à puce pour vous connecter à la console web.

Conditions préalables

- Un certificat valide stocké dans votre carte à puce et associé à un compte d'utilisateur créé dans un domaine Active Directory ou Identity Management.
- PIN pour déverrouiller la carte à puce.
- La carte à puce a été introduite dans le lecteur.

Procédure

1. Ouvrez votre navigateur web et ajoutez l'adresse de la console web dans la barre d'adresse. Le navigateur vous demande d'ajouter le code PIN protégeant le certificat stocké sur la carte à puce.
2. Dans la boîte de dialogue **Password Required**, saisissez le code PIN et cliquez sur **OK**.
3. Dans la boîte de dialogue **User Identification Request**, sélectionnez le certificat stocké dans la carte à puce.
4. Sélectionnez **Remember this decision**.
Le système n'ouvre pas cette fenêtre la prochaine fois.



NOTE

Cette étape ne s'applique pas aux utilisateurs de Google Chrome.

5. Cliquez sur **OK**.

Vous êtes maintenant connecté et la console web affiche son contenu.

5.6. ACTIVATION DE SUDO SANS MOT DE PASSE POUR LES UTILISATEURS DE CARTES À PUCE

Une fois que vous vous êtes connecté à la console web avec un certificat, il se peut que vous deviez passer en mode administratif (privilèges de racine via **sudo**). Si votre compte utilisateur possède un mot de passe, vous pouvez l'utiliser pour vous authentifier sur **sudo**.

Comme alternative, si vous utilisez Red Hat Identity Management, vous pouvez déclarer l'authentification initiale du certificat de la console web comme étant de confiance pour l'authentification à **sudo**, SSH, ou d'autres services. À cette fin, la console Web crée automatiquement un ticket Kerberos S4U2Proxy dans la session de l'utilisateur.

Conditions préalables

- Gestion de l'identité
- Active Directory connecté à la confiance entre les forêts grâce à la gestion des identités
- Carte à puce configurée pour se connecter à la console web. Pour plus d'informations, voir [Configuration de l'authentification par carte à puce avec la console web pour les utilisateurs gérés de manière centralisée](#).

Procédure

1. Définir des règles de délégation des contraintes pour dresser la liste des hôtes auxquels le ticket peut accéder.

Exemple 5.1. Mise en place de règles de délégation de contraintes

La session de la console web s'exécute sur l'hôte **host.example.com** et doit être autorisée à accéder à son propre hôte avec **sudo**. De plus, nous ajoutons un deuxième hôte de confiance - **remote.example.com**.

- Créer la délégation suivante :
 - Exécutez les commandes suivantes pour ajouter une liste de machines cibles auxquelles une règle particulière peut accéder :

```
# ipa servicedelegationtarget-add cockpit-target
# ipa servicedelegationtarget-add-member cockpit-target \
  --principals=host/host.example.com@EXAMPLE.COM \
  --principals=host/remote.example.com@EXAMPLE.COM
```

- Pour autoriser les sessions de la console web (HTTP/principal) à accéder à cette liste d'hôtes, exécutez les commandes suivantes :

```
# ipa servicedelegationrule-add cockpit-delegation
# ipa servicedelegationrule-add-member cockpit-delegation \
  --principals=HTTP/host.example.com@EXAMPLE.COM
# ipa servicedelegationrule-add-target cockpit-delegation \
  --servicedelegationtargets=cockpit-target
```

2. Activer l'authentification GSS dans les services correspondants :

- a. Pour sudo, activez le module **pam_sss_gss** dans le fichier **/etc/sss/sss.conf**:

- i. En tant que root, ajoutez une entrée pour votre domaine dans le fichier de configuration **/etc/sss/sss.conf**.

```
[domain/example.com]
pam_gssapi_services = sudo, sudo-i
```

- ii. Activez le module dans le fichier **/etc/pam.d/sudo** sur la première ligne.

```
auth sufficient pam_sss_gss.so
```

- b. Pour SSH, mettez à jour l'option **GSSAPIAuthentication** du fichier **/etc/ssh/sshd_config** en **yes**.



AVERTISSEMENT

Le ticket S4U délégué n'est pas transmis aux hôtes SSH distants lorsque l'on s'y connecte depuis la console web. L'authentification sudo sur un hôte distant avec votre ticket ne fonctionnera pas.

Vérification

1. Connectez-vous à la console web à l'aide d'une carte à puce.
2. Cliquez sur le bouton **Limited access**.
3. Authentifiez-vous à l'aide de votre carte à puce.

OU

1. Essayez de vous connecter à un autre hôte avec SSH.

5.7. LIMITATION DES SESSIONS D'UTILISATEURS ET DE LA MÉMOIRE POUR ÉVITER UNE ATTAQUE DOS

L'authentification par certificat est protégée en séparant et en isolant les instances du serveur web **cockpit-ws** contre les attaquants qui veulent se faire passer pour un autre utilisateur. Cependant, cela introduit un risque d'attaque par déni de service (DoS) : Un attaquant distant pourrait créer un grand nombre de certificats et envoyer un grand nombre de requêtes HTTPS à **cockpit-ws**, chacune utilisant un certificat différent.

Pour éviter ce déni de service, les ressources collectives de ces instances de serveur web sont limitées. Par défaut, les limites du nombre de connexions et de l'utilisation de la mémoire sont fixées à 200 threads et à une limite de mémoire de 75 % (soft) / 90 % (hard).

La procédure suivante décrit la protection des ressources en limitant le nombre de connexions et la mémoire.

Procédure

1. Dans le terminal, ouvrez le fichier de configuration **system-cockpithttps.slice**:

```
# systemctl edit system-cockpithttps.slice
```

2. Limitez les **TasksMax** à 100 et les **CPUQuota** à 30%:

```
[Slice]
# change existing value
TasksMax=100
# add new restriction
CPUQuota=30%
```

3. Pour appliquer les modifications, redémarrez le système :

```
# systemctl daemon-reload  
# systemctl stop cockpit
```

Désormais, les nouvelles limites de mémoire et de session utilisateur protègent le serveur web **cockpit-
ws** contre les attaques DoS.

CHAPITRE 6. CONFIGURATION DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC DES CERTIFICATS LOCAUX

Ce chapitre décrit un scénario dans lequel

- L'hôte n'est pas connecté à un domaine.
- Vous souhaitez vous authentifier avec une carte à puce sur cet hôte.
- Vous souhaitez configurer l'accès SSH en utilisant l'authentification par carte à puce.
- Vous voulez configurer la carte à puce avec **authselect**.

Utilisez la configuration suivante pour réaliser ce scénario :

- Obtenez un certificat d'utilisateur pour l'utilisateur qui souhaite s'authentifier à l'aide d'une carte à puce. Le certificat doit être généré par une autorité de certification fiable utilisée dans le domaine.
Si vous ne pouvez pas obtenir le certificat, vous pouvez générer un certificat utilisateur signé par une autorité de certification locale à des fins de test,
- Stockez le certificat et la clé privée dans une carte à puce.
- Configurer l'authentification par carte à puce pour l'accès SSH.



IMPORTANT

Si un hôte peut faire partie du domaine, ajoutez-le au domaine et utilisez les certificats générés par Active Directory ou l'autorité de certification de la gestion des identités.

Pour plus d'informations sur la création de certificats IdM pour une carte à puce, voir [Configuration de la gestion des identités pour l'authentification par carte à puce](#) .

Conditions préalables

- Authselect installé
L'outil authselect configure l'authentification des utilisateurs sur les hôtes Linux et vous pouvez l'utiliser pour configurer les paramètres d'authentification par carte à puce. Pour plus d'informations sur authselect, voir [Explication de authselect](#).
- Carte à puce ou périphériques USB pris en charge par RHEL 9
Pour plus de détails, voir la [prise en charge des cartes à puce dans RHEL9](#) .

6.1. CRÉATION DE CERTIFICATS LOCAUX

Cette section décrit comment effectuer ces tâches :

- Générer l'autorité de certification OpenSSL
- Créer une demande de signature de certificat



AVERTISSEMENT

Les étapes suivantes sont destinées à des fins de test uniquement. Les certificats générés par une autorité de certification locale auto-signée ne sont pas aussi sûrs qu'une autorité de certification AD, IdM ou RHCS. Vous devez utiliser un certificat généré par l'autorité de certification de votre entreprise, même si l'hôte ne fait pas partie du domaine.

Procédure

1. Créez un répertoire où vous pourrez générer le certificat, par exemple :

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

2. Mettre en place le certificat (copier ce texte sur votre ligne de commande dans le répertoire **ca**):

```
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. Créez les répertoires suivants :

```
# mkdir certs crl newcerts
```

4. Créez les fichiers suivants :

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

5. Inscrivez le numéro 01 dans le fichier série :

```
# echo 01 > serial
```

Cette commande écrit un numéro 01 dans le fichier de série. Il s'agit du numéro de série du certificat. Ce numéro augmente d'une unité à chaque nouveau certificat délivré par l'autorité de certification.

6. Créer une clé d'autorité de certification racine OpenSSL :

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

7. Créer un certificat d'autorité de certification racine auto-signé :

```
# 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. Créez la clé de votre nom d'utilisateur :

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

Cette clé est générée dans le système local, qui n'est pas sécurisé. Il faut donc retirer la clé du système lorsqu'elle est stockée dans la carte.

Vous pouvez également créer une clé directement dans la carte à puce. Pour ce faire, suivez les instructions du fabricant de votre carte à puce.

9. Créez le fichier de configuration de la demande de signature de certificat (copiez ce texte sur votre ligne de commande dans le répertoire ca) :

```
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. Créez une demande de signature de certificat pour votre certificat example.user :

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

11. Configurez le nouveau certificat. La période d'expiration est fixée à 1 an :

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

À ce stade, l'autorité de certification et les certificats sont générés avec succès et préparés pour l'importation dans une carte à puce.

6.2. COPIE DES CERTIFICATS DANS LE RÉPERTOIRE SSSD

Gnome Desktop Manager (GDM) nécessite SSSD. Si vous utilisez GDM, vous devez copier le certificat PEM dans le répertoire `/etc/sss/pki`.

Conditions préalables

- L'autorité locale de l'AC et les certificats ont été générés

Procédure

1. Assurez-vous que SSSD est installé sur le système.

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

2. Créez un répertoire `/etc/sss/pki`:

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

3. Copiez le fichier `rootCA.crt` en tant que fichier PEM dans le répertoire `/etc/sss/pki`:

```
# cp /tmp/ca/rootCA.crt /etc/sss/pki/sss_auth_ca_db.pem
```

Vous avez maintenant généré avec succès l'autorité de certification et les certificats, et vous les avez enregistrés dans le répertoire `/etc/sss/pki`.



NOTE

Si vous souhaitez partager les certificats de l'autorité de certification avec une autre application, vous pouvez modifier l'emplacement dans `sss.conf` :

- SSSD PAM responder : `pam_cert_db_path` dans la section `[pam]`
- SSSD ssh responder : `ca_db` dans la section `[ssh]`

Pour plus de détails, voir la page de manuel de `sss.conf`.

Red Hat recommande de conserver le chemin d'accès par défaut et d'utiliser un fichier de certificat d'autorité de certification dédié pour SSSD afin de s'assurer que seules les autorités de certification approuvées pour l'authentification sont répertoriées ici.

6.3. INSTALLATION D'OUTILS DE GESTION ET D'UTILISATION DES CARTES À PUCE

Pour configurer votre carte à puce, vous avez besoin d'outils qui peuvent générer des certificats et les stocker sur une carte à puce.

Vous devez :

- Installez le paquet `gnutls-utils`, qui vous aide à gérer les certificats.
- Installez le paquetage `opensc`, qui fournit un ensemble de bibliothèques et d'utilitaires pour travailler avec des cartes à puce.
- Démarrez le service `pcscd`, qui communique avec le lecteur de cartes à puce.

Procédure

1. Installez les paquets `opensc` et `gnutls-utils`:

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

2. Démarrez le service `pcscd`.

```
# systemctl start pcscd
```

Vérifiez que le service `pcscd` est opérationnel.

6.4. PRÉPARATION DE VOTRE CARTE À PUCE ET TÉLÉCHARGEMENT DE VOS CERTIFICATS ET CLÉS SUR VOTRE CARTE À PUCE

Cette section décrit la configuration de la carte à puce avec l'outil **pkcs15-init**, qui vous aide à configurer :

- Effacer votre carte à puce
- Définition de nouveaux codes PIN et de clés de déblocage de code PIN (PUK) en option
- Création d'un nouvel emplacement sur la carte à puce
- Stockage du certificat, de la clé privée et de la clé publique dans la fente
- Si nécessaire, verrouiller les paramètres de la carte à puce, car certaines cartes à puce nécessitent ce type de finalisation



NOTE

L'outil **pkcs15-init** peut ne pas fonctionner avec toutes les cartes à puce. Vous devez utiliser les outils qui fonctionnent avec la carte à puce que vous utilisez.

Conditions préalables

- Le paquet **opensc**, qui comprend l'outil **pkcs15-init**, est installé.
Pour plus de détails, voir [Installation des outils de gestion et d'utilisation des cartes à puce](#).
- La carte est insérée dans le lecteur et connectée à l'ordinateur.
- Vous disposez de la clé privée, de la clé publique et du certificat à stocker sur la carte à puce. Dans cette procédure, **testuser.key**, **testuserpublic.key**, et **testuser.crt** sont les noms utilisés pour la clé privée, la clé publique et le certificat.
- Vous disposez du code PIN de l'utilisateur de votre carte à puce actuelle et du code PIN de l'agent de sécurité (SO-PIN).

Procédure

1. Effacez votre carte à puce et authentifiez-vous avec votre code PIN :

```
$ 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]:
```

La carte a été effacée.

2. Initialisez votre carte à puce, définissez votre code PIN et PUK d'utilisateur, ainsi que le code PIN et PUK de votre responsable de la sécurité :

```
$ 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
```

L'outil **pkcs15-init** crée un nouvel emplacement sur la carte à puce.

3. Définir l'étiquette et l'ID d'authentification pour l'emplacement :

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

L'étiquette est définie sur une valeur lisible par l'homme, dans ce cas, **testuser**. L'adresse **auth-id** doit être composée de deux valeurs hexadécimales ; dans ce cas, elle est fixée à **01**.

4. Stockez et étiquetez la clé privée dans le nouvel emplacement de la carte à puce :

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



NOTE

La valeur que vous indiquez pour **--id** doit être la même lorsque vous stockez votre clé privée et votre certificat à l'étape suivante. Il est recommandé de spécifier votre propre valeur pour **--id**, sinon l'outil calculera une valeur plus complexe.

5. Stockez et étiquetez le certificat dans le nouvel emplacement de la carte à puce :

```
$ 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. (Facultatif) Stockez et étiquetez la clé publique dans le nouvel emplacement de la carte à puce :

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



NOTE

Si la clé publique correspond à une clé privée ou à un certificat, indiquez le même ID que celui de la clé privée ou du certificat.

7. (Facultatif) Certaines cartes à puce exigent que vous finalisiez la carte en verrouillant les paramètres :

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

À ce stade, votre carte à puce comprend le certificat, la clé privée et la clé publique dans l'emplacement nouvellement créé. Vous avez également créé votre code PIN et PUK d'utilisateur ainsi que le code PIN et PUK de l'agent de sécurité.

6.5. CONFIGURATION DE L'ACCÈS SSH À L'AIDE DE L'AUTHENTIFICATION PAR CARTE À PUCE

Les connexions SSH nécessitent une authentification. Vous pouvez utiliser un mot de passe ou un certificat. Cette section décrit la configuration nécessaire pour activer l'authentification à l'aide d'un certificat stocké sur une carte à puce

Pour plus de détails sur la configuration des cartes à puce avec **authselect**, voir [Configuration des cartes à puce avec authselect](#).

Conditions préalables

- La carte à puce contient votre certificat et votre clé privée.
- La carte est insérée dans le lecteur et connectée à l'ordinateur.
- SSSD est installé et configuré.
- Votre nom d'utilisateur correspond au nom commun (CN) ou à l'ID utilisateur (UID) figurant dans le SUJET du certificat.
- Le service **pcscd** est en cours d'exécution sur votre machine locale.
Pour plus de détails, voir [Installation des outils de gestion et d'utilisation des cartes à puce](#).

Procédure

1. Créez un nouveau répertoire pour les clés SSH dans le répertoire personnel de l'utilisateur qui utilise l'authentification par carte à puce :

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

2. Exécutez la commande **ssh-keygen -D** avec la bibliothèque **opensc** pour récupérer la clé publique existante associée à la clé privée de la carte à puce, et ajoutez-la à la liste **authorized_keys** du répertoire des clés SSH de l'utilisateur pour activer l'accès SSH avec l'authentification par carte à puce.

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

3. SSH nécessite la configuration des droits d'accès au répertoire **/.ssh** et au fichier **authorized_keys**. Pour définir ou modifier les droits d'accès, entrez :

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

4. En option, afficher les touches :

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

Le terminal affiche les touches.

5. Vérifiez que l'authentification par carte à puce est activée dans le fichier **/etc/sss/sss.conf**:
Dans la section **[pam]**, activez le module d'authentification par certificat pam : **pam_cert_auth = True**

Si le fichier **sss.conf** n'a pas encore été créé, vous pouvez créer la configuration fonctionnelle minimale en copiant le script suivant sur la ligne de commande :

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

[nss]

[pam]
pam_cert_auth = True

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

6. Pour utiliser les clés SSH, configurez l'authentification avec la commande **authselect**:

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

Vous pouvez maintenant vérifier l'accès SSH à l'aide de la commande suivante :

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

Si la configuration est réussie, vous êtes invité à saisir le code PIN de la carte à puce.

La configuration fonctionne désormais localement. Vous pouvez maintenant copier la clé publique et la distribuer dans les fichiers **authorized_keys** situés sur tous les serveurs sur lesquels vous souhaitez utiliser SSH.

CHAPITRE 7. CONFIGURATION DE L'AUTHENTIFICATION PAR CARTE À PUCE À L'AIDE DE AUTHSELECT

Cette section décrit comment configurer votre carte à puce pour atteindre l'un des objectifs suivants :

- Activer l'authentification par mot de passe et par carte à puce
- Désactiver le mot de passe et activer l'authentification par carte à puce
- Activer le verrouillage lors du retrait

Conditions préalables

- Authselect installé
L'outil authselect configure l'authentification des utilisateurs sur les hôtes Linux et vous pouvez l'utiliser pour configurer les paramètres d'authentification par carte à puce. Pour plus d'informations sur authselect, voir [Configuration de l'authentification utilisateur à l'aide d'authselect](#).
- Carte à puce ou périphériques USB pris en charge par RHEL 9
Pour plus de détails, voir la [prise en charge des cartes à puce dans RHEL9](#).

7.1. CERTIFICATS ÉLIGIBLES AUX CARTES À PUCE

Avant de pouvoir configurer une carte à puce avec **authselect**, vous devez importer un certificat dans votre carte. Vous pouvez utiliser les outils suivants pour générer le certificat :

- Active Directory (AD)
- Gestion de l'identité (IdM)
Pour plus d'informations sur la création de certificats IdM, voir [Demander un nouveau certificat d'utilisateur et l'exporter vers le client](#).
- Système de certification Red Hat (RHCS)
Pour plus d'informations, voir [Gestion des cartes à puce avec Enterprise Security Client](#).
- Autorité de certification (AC) tierce
- Autorité de certification locale. Vous pouvez utiliser un certificat généré par l'autorité de certification locale si l'utilisateur ne fait pas partie d'un domaine ou à des fins de test.
Pour plus d'informations sur la création et l'importation de certificats locaux dans une carte à puce, voir [Configuration et importation de certificats locaux dans une carte à puce](#).

7.2. CONFIGUREZ VOTRE SYSTÈME POUR ACTIVER L'AUTHENTIFICATION PAR CARTE À PUCE ET PAR MOT DE PASSE

Cette section décrit comment activer l'authentification par carte à puce et par mot de passe sur votre système.

Conditions préalables

- La carte à puce contient votre certificat et votre clé privée.
- La carte est insérée dans le lecteur et connectée à l'ordinateur.

- L'outil **authselect** est installé sur votre système.

Procédure

- Entrez la commande suivante pour autoriser l'authentification par carte à puce et par mot de passe :

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

À ce stade, l'authentification par carte à puce est activée, mais l'authentification par mot de passe fonctionnera si vous oubliez votre carte à puce à la maison.

7.3. CONFIGURATION DE VOTRE SYSTÈME POUR APPLIQUER L'AUTHENTIFICATION PAR CARTE À PUCE

L'outil **authselect** vous permet de configurer l'authentification par carte à puce sur votre système et de désactiver l'authentification par mot de passe par défaut. La commande **authselect** comprend les options suivantes :

- **with-smartcard** - permet l'authentification par carte à puce en plus de l'authentification par mot de passe
- **with-smartcard-required** - active l'authentification par carte à puce et désactive l'authentification par mot de passe



NOTE

L'option **with-smartcard-required** n'impose l'authentification exclusive par carte à puce que pour les services de connexion, tels que **login**, **gdm**, **xdm**, **kdm**, **xscreensaver**, **gnome-screensaver** et **kscreensaver**. D'autres services, tels que **su** ou **sudo** pour les utilisateurs de commutation, n'utilisent pas l'authentification par carte à puce par défaut et continueront à vous demander un mot de passe.

Conditions préalables

- La carte à puce contient votre certificat et votre clé privée.
- La carte est insérée dans le lecteur et connectée à l'ordinateur.
- L'outil **authselect** est installé sur votre système local.

Procédure

- Entrez la commande suivante pour appliquer l'authentification par carte à puce :

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



NOTE

Une fois cette commande exécutée, l'authentification par mot de passe ne fonctionnera plus et vous ne pourrez vous connecter qu'à l'aide d'une carte à puce. Assurez-vous que l'authentification par carte à puce fonctionne avant d'exécuter cette commande, sinon vous risquez d'être bloqué sur votre système.

7.4. CONFIGURATION DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC VERROUILLAGE EN CAS DE RETRAIT

Le service **authselect** vous permet de configurer l'authentification de votre carte à puce pour verrouiller votre écran instantanément après avoir retiré la carte à puce du lecteur. La commande **authselect** doit inclure les variables suivantes :

- **with-smartcard**- activation de l'authentification par carte à puce
- **with-smartcard-required**- l'activation de l'authentification exclusive par carte à puce (l'authentification par mot de passe est désactivée)
- **with-smartcard-lock-on-removal**- imposer la déconnexion après le retrait de la carte à puce



NOTE

L'option **with-smartcard-lock-on-removal** ne fonctionne que sur les systèmes dotés de l'environnement de bureau GNOME. Si vous utilisez un système basé sur **tty** ou une console et que vous retirez votre carte à puce de son lecteur, vous n'êtes pas automatiquement verrouillé.

Conditions préalables

- La carte à puce contient votre certificat et votre clé privée.
- La carte est insérée dans le lecteur et connectée à l'ordinateur.
- L'outil **authselect** est installé sur votre système local.

Procédure

- Entrez la commande suivante pour activer l'authentification par carte à puce, désactiver l'authentification par mot de passe et appliquer le verrouillage lors de la suppression :

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

Désormais, lorsque vous retirez la carte, l'écran se verrouille. Vous devez réinsérer votre carte à puce pour le déverrouiller.

CHAPITRE 8. S'AUTHENTIFIER À SUDO À DISTANCE À L'AIDE DE CARTES À PUCE

Cette section décrit comment s'authentifier à sudo à distance à l'aide de cartes à puce. Une fois que le service **ssh-agent** fonctionne localement et qu'il peut transmettre le socket **ssh-agent** à une machine distante, vous pouvez utiliser le protocole d'authentification SSH dans le module sudo PAM pour authentifier les utilisateurs à distance.

Après vous être connecté localement à l'aide d'une carte à puce, vous pouvez vous connecter via SSH à la machine distante et exécuter la commande **sudo** sans être invité à saisir un mot de passe en utilisant la transmission SSH de l'authentification par carte à puce.

Dans cet exemple, un client se connecte au serveur IPA via SSH et exécute la commande sudo sur le serveur IPA avec des informations d'identification stockées sur une carte à puce.

- [Création de règles sudo dans IdM](#)
- [Mise en place du module PAM pour sudo](#)
- [Se connecter à sudo à distance à l'aide d'une carte à puce](#)

8.1. CRÉATION DE RÈGLES SUDO DANS IDM

Cette procédure décrit comment créer des règles sudo dans IdM pour donner à **ipausers1** la permission d'exécuter sudo sur l'hôte distant.

Pour les besoins de cet exemple, les commandes **less** et **whoami** sont ajoutées en tant que commandes sudo pour tester la procédure.

Conditions préalables

- L'utilisateur IdM a été créé. Pour les besoins de cet exemple, l'utilisateur est **ipausers1**.
- Vous avez le nom d'hôte du système sur lequel vous exécutez sudo à distance. Dans le cadre de cet exemple, l'hôte est **server.ipa.test**.

Procédure

1. Créez une règle **sudo** nommée **adminrule** pour permettre à un utilisateur d'exécuter des commandes.

```
ipa sudorule-add adminrule
```

2. Ajouter **less** et **whoami** en tant que commandes **sudo**:

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

3. Ajoutez les commandes **less** et **whoami** à la commande **adminrule**:

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

4. Ajouter l'utilisateur **ipausers1** à l'utilisateur **adminrule**:

```
ipa sudorule-add-user adminrule --users ipausers1
```

5. Ajoutez l'hôte sur lequel vous exécutez **sudo** à la liste **adminrule**:

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

Ressources supplémentaires

- Voir **ipa sudorule-add --help**.
- Voir **ipa sudocmd-add --help**.

8.2. MISE EN PLACE DU MODULE PAM POUR SUDO

Cette procédure décrit comment installer et configurer le module PAM de **pam_ssh_agent_auth.so** pour l'authentification sudo avec une carte à puce sur n'importe quel hôte où vous exécutez sudo.

Procédure

1. Installer l'agent PAM SSH :

```
dnf -y install pam_ssh_agent_auth
```

2. Ajoutez l'adresse **authorized_keys_command** pour **pam_ssh_agent_auth.so** au fichier **/etc/pam.d/sudo** avant toute autre entrée **auth**:

```
##%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. Pour permettre à la redirection de l'agent SSH de fonctionner lorsque vous exécutez des commandes sudo, ajoutez ce qui suit au fichier **/etc/sudoers**:

```
Valeurs par défaut env_keep = "SSH_AUTH_SOCK"
```

Cela permet aux utilisateurs dont les clés publiques des cartes à puce sont stockées dans IPA/SSSD de s'authentifier auprès de sudo sans saisir de mot de passe.

4. Redémarrez le service **sssd**:

```
systemctl restart sssd
```

Ressources supplémentaires

- Voir la page de manuel **pam**.

8.3. SE CONNECTER À SUDO À DISTANCE À L'AIDE D'UNE CARTE À PUCE

Cette procédure décrit comment configurer l'agent et le client SSH pour se connecter à sudo à distance à l'aide d'une carte à puce.

Conditions préalables

- Vous avez créé des règles sudo dans IdM.
- Vous avez installé et configuré le module PAM de **pam_ssh_agent_auth** pour l'authentification sudo sur le système distant où vous allez exécuter sudo.

Procédure

1. Démarrez l'agent SSH (s'il n'est pas déjà en cours d'exécution).

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

2. Ajoutez votre carte à puce à l'agent SSH. Saisissez votre code PIN lorsque vous y êtes invité :

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

3. Connectez-vous au système sur lequel vous devez exécuter **sudo** à distance en utilisant SSH avec la fonction ssh-agent forwarding activée. Utilisez l'option **-A**:

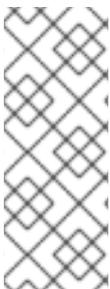
```
ssh -A ipauser1@server.ipa.test
```

Verification steps

- Exécutez la commande **whoami** avec **sudo**:

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

Aucun code PIN ou mot de passe n'est demandé lorsque la carte à puce est insérée.



NOTE

Si l'agent SSH est configuré pour utiliser d'autres sources, telles que le trousseau GNOME, et que vous exécutez la commande **sudo** après avoir retiré la carte à puce, il se peut que vous ne soyez pas invité à saisir un code PIN ou un mot de passe, car l'une des autres sources peut fournir un accès à une clé privée valide. Pour vérifier les clés publiques de toutes les identités connues par l'agent SSH, exécutez la commande **ssh-add -L**.

Ressources supplémentaires

- [Utiliser des communications sécurisées entre deux systèmes avec OpenSSH](#)
- [Se connecter à des machines distantes avec des clés SSH en utilisant ssh-agent](#)

CHAPITRE 9. DÉPANNAGE DE L'AUTHENTIFICATION PAR CARTE À PUCE

Les sections suivantes décrivent comment résoudre certains des problèmes que vous pouvez rencontrer lors de la mise en place de l'authentification par carte à puce.

- [Test de l'authentification par carte à puce](#)
- [Dépannage de l'authentification par carte à puce avec SSSD](#)
- [Vérification que le KDC Kerberos d'IdM peut utiliser PKINIT et que les certificats de l'autorité de certification sont correctement localisés](#)
- [Augmentation des délais d'attente SSSD](#)
- [Dépannage des règles de mappage et de correspondance des certificats](#)

9.1. TEST DE L'ACCÈS PAR CARTE À PUCE SUR LE SYSTÈME

Cette procédure décrit comment tester si vous pouvez accéder à votre carte à puce.

Conditions préalables

- Vous avez installé et configuré votre serveur IdM et votre client pour une utilisation avec des cartes à puce.
- Vous avez installé l'outil **certutil** à partir du paquetage **nss-tools**.
- Vous avez le code PIN ou le mot de passe de votre carte à puce.

Procédure

1. À l'aide de la commande **lsusb**, vérifiez que le lecteur de cartes à puce est visible par le système d'exploitation :

```
$ 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
```

Pour plus d'informations sur les cartes à puce et les lecteurs testés et pris en charge par RHEL, voir [Prise en charge des cartes à puce dans RHEL 9](#) .

2. Assurez-vous que le service et le socket **pcscd** sont activés et en cours d'exécution :

```
$ 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. À l'aide de la commande **p11-kit list-modules**, affichez des informations sur la carte à puce configurée et les jetons présents sur la carte à puce :

```

$ 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. Vérifiez que vous pouvez accéder au contenu de votre carte à puce :

```

$ 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. Affichez le contenu du certificat de votre carte à puce à l'aide de la commande **certutil**:
- a. Exécutez la commande suivante pour déterminer le nom correct de votre certificat :

```
$ 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. Affichez le contenu du certificat sur votre carte à puce :



NOTE

Veillez à ce que le nom du certificat corresponde exactement à la sortie affichée à l'étape précédente, dans cet exemple **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

```

Ressources supplémentaires

- Voir la page de manuel **certutil(1)**.

9.2. DÉPANNAGE DE L'AUTHENTIFICATION PAR CARTE À PUCE AVEC SSSD

Cette procédure décrit comment dépanner l'authentification avec SSSD à l'aide de cartes à puce.

Conditions préalables

- Vous avez installé et configuré votre serveur IdM et votre client pour une utilisation avec des cartes à puce.
- Vous avez installé le paquetage **sssd-tools**.

- Vous pouvez détecter votre lecteur de carte à puce et afficher le contenu de votre carte à puce. Voir [Test de l'accès par carte à puce sur le système](#) .

Procédure

1. Vérifiez que vous pouvez vous authentifier avec votre carte à puce en utilisant **su**:

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

Si le code PIN de la carte à puce ne vous est pas demandé et qu'une invite de mot de passe ou une erreur d'autorisation est renvoyée, vérifiez les journaux SSSD. Reportez-vous à la section [Dépannage de l'authentification avec SSSD dans IdM](#) pour obtenir des informations sur la journalisation dans SSSD. Voici un exemple d'échec d'authentification :

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

Si les journaux SSSD indiquent un problème à l'adresse **krb5_child**, similaire à ce qui suit, il se peut que vous ayez un problème avec vos certificats d'autorité de certification. Pour résoudre les problèmes liés aux certificats, voir [Vérifier que le KDC Kerberos IdM peut utiliser Pkinit et que les certificats d'autorité de certification sont correctement localisés](#).

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

Si les journaux SSSD indiquent un dépassement de délai à partir de **p11_child** ou **krb5_child**, vous devrez peut-être augmenter les délais d'attente SSSD et réessayer de vous authentifier à l'aide de votre carte à puce. Voir [Augmentation des délais d'attente SSSD](#) pour plus de détails sur la manière d'augmenter les délais d'attente.

2. Vérifiez que la configuration de l'authentification par carte à puce GDM est correcte. Un message de succès pour l'authentification PAM devrait être renvoyé comme indiqué ci-dessous :

```
# 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:
```

Si une erreur d'authentification, semblable à la suivante, est renvoyée, vérifiez les journaux SSSD pour essayer de déterminer la cause du problème. Voir [Dépannage de l'authentification avec SSSD dans IdM](#) pour plus d'informations sur la journalisation dans SSSD.

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

```
PAM Environment:
- no env -
```

Si l'authentification PAM continue d'échouer, videz votre cache et exécutez à nouveau la commande.

```
# 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
```

9.3. VÉRIFICATION QUE LE KDC KERBEROS D'IDM PEUT UTILISER PKINIT ET QUE LES CERTIFICATS DE L'AUTORITÉ DE CERTIFICATION SONT CORRECTEMENT LOCALISÉS

Cette procédure décrit comment vérifier que le KDC Kerberos d'IdM peut utiliser PKINIT et comment vérifier que les certificats de l'autorité de certification sont correctement localisés.

Conditions préalables

- Vous avez installé et configuré votre serveur IdM et votre client pour une utilisation avec des cartes à puce.
- Vous pouvez détecter votre lecteur de carte à puce et afficher le contenu de votre carte à puce. Voir [Test de l'accès par carte à puce sur le système](#) .

Procédure

1. Exécutez l'utilitaire **kinit** pour vous authentifier en tant que **idmuser1** avec le certificat stocké sur votre carte à puce :

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

2. Saisissez le code PIN de votre carte à puce. Si vous n'êtes pas invité à saisir votre code PIN, vérifiez que vous pouvez détecter votre lecteur de carte à puce et afficher le contenu de votre carte à puce. Voir [Test de l'authentification de la carte à puce](#).
3. Si votre code PIN est accepté et que vous êtes invité à saisir votre mot de passe, il se peut que le certificat de signature de l'autorité de certification soit manquant.
 - a. Vérifiez que la chaîne de l'autorité de certification est répertoriée dans le fichier du paquet de certificats par défaut à l'aide des commandes **openssl**:

```
$ 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. Vérifiez la validité de vos certificats :
 - i. Recherchez l'ID du certificat d'authentification de l'utilisateur pour **idmuser1**:

```
$ pkcs11-tool --list-objects --login
[...]
Certificate Object; type = X.509 cert
label: Certificate
subject: DN: O=IDM.EXAMPLE.COM, CN=idmuser1
ID: 01
```

- ii. Lire les informations relatives au certificat de l'utilisateur sur la carte à puce au format DER :

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

- iii. Convertir le certificat DER au format PEM :

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

- iv. Vérifiez que le certificat comporte des signatures d'émetteur valides jusqu'à l'autorité de certification :

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

4. Si votre carte à puce contient plusieurs certificats, il se peut que **kinit** ne parvienne pas à choisir le bon certificat pour l'authentification. Dans ce cas, vous devez spécifier l'ID du certificat comme argument de la commande **kinit** en utilisant l'option **certid=<ID>**.

- a. Vérifiez combien de certificats sont stockés sur la carte à puce et obtenez l'identifiant du certificat que vous utilisez :

```
$ 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. Exécutez **kinit** avec le certificat ID 01 :

```

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

```

5. Exécutez **klist** pour afficher le contenu du cache des informations d'identification Kerberos :

```

$ 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. Détruisez vos tickets Kerberos actifs une fois que vous avez terminé :

```

$ kdestroy -A

```

Ressources supplémentaires

- Voir la page de manuel **kinit**.
- Voir la page de manuel **kdestroy**.

9.4. AUGMENTATION DES DÉLAIS D'ATTENTE SSSD

Si vous rencontrez des problèmes d'authentification avec une carte à puce, vérifiez dans le fichier **krb5_child.log** et **p11_child.log** la présence d'entrées de délai d'attente similaires à celles qui suivent :

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

Si le fichier journal contient une entrée de délai d'attente, essayez d'augmenter les délais d'attente de SSSD comme indiqué dans cette procédure.

Conditions préalables

- Vous avez configuré votre serveur IdM et votre client pour l'authentification par carte à puce.

Procédure

1. Ouvrez le fichier **sssd.conf** sur le client IdM :

```

# vim /etc/sss/sss.conf

```

2. Dans la section de votre domaine, par exemple **[domain/idm.example.com]**, ajoutez l'option suivante :

```
krb5_auth_timeout = 60
```

3. Dans la section **[pam]**, ajouter ce qui suit :

```
p11_child_timeout = 60
```

4. Effacer le cache SSSD :

```
# 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
```

Une fois que vous avez augmenté les délais d'attente, essayez à nouveau de vous authentifier à l'aide de votre carte à puce. Voir [Tester l'authentification par carte à puce](#) pour plus de détails.

9.5. DÉPANNAGE DES RÈGLES DE MAPPAGE ET DE CORRESPONDANCE DES CERTIFICATS

Si vous rencontrez des problèmes d'authentification avec une carte à puce, vérifiez que vous avez correctement associé votre certificat de carte à puce à un utilisateur. Par défaut, un certificat est associé à un utilisateur lorsque l'entrée de l'utilisateur contient le certificat complet dans le cadre de l'attribut **usercertificate**. Toutefois, si vous avez défini des règles de mappage de certificats, vous avez peut-être modifié la manière dont les certificats sont associés aux utilisateurs. Pour résoudre les problèmes liés aux règles de mappage et de correspondance des certificats, reportez-vous aux sections suivantes :

- [Vérification de la correspondance entre les certificats et les utilisateurs](#)
- [Vérification de l'utilisateur associé à un certificat de carte à puce](#)



NOTE

Si vous utilisez votre carte à puce pour vous authentifier à l'aide de SSH, vous devez ajouter le certificat complet à l'entrée de l'utilisateur dans Identity Management (IdM). Si vous n'utilisez pas votre carte à puce pour vous authentifier à l'aide de SSH, vous pouvez ajouter des données de mappage de certificat à l'aide de la commande **ipa user-add-certmapdata**.

9.5.1. Vérification de la correspondance entre les certificats et les utilisateurs

Par défaut, un certificat est associé à un utilisateur lorsque l'entrée de l'utilisateur contient le certificat complet dans le cadre de l'attribut **usercertificate**. Toutefois, si vous avez défini des règles de mappage de certificats, il se peut que vous ayez modifié la manière dont les certificats sont associés aux utilisateurs. Cette procédure décrit comment vérifier vos règles de mappage de certificats.

Conditions préalables

- Vous avez installé et configuré votre serveur et votre client de gestion des identités (IdM) pour une utilisation avec des cartes à puce.
- Vous pouvez détecter votre lecteur de carte à puce et afficher le contenu de votre carte à puce. Voir [Test de l'accès par carte à puce sur le système](#) .
- Vous avez associé votre certificat de carte à puce à un utilisateur IdM. Voir [Règles de mappage des certificats pour la configuration de l'authentification sur les cartes à puce](#).

Procédure

1. Vérifier les règles de mappage des certificats actuellement configurées pour IdM :

```
# ipa certmaprule-find
-----
1 Certificate Identity Mapping Rule matched
-----
Rule name: smartcardrule
Mapping rule: (ipacertmapdata=X509:<|>{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
-----
```

Vous pouvez vous attendre à ce que l'une des règles de mappage suivantes soit définie :

- **ipacertmapdata** indique que l'attribut **certmapdata** de l'entrée utilisateur IdM est utilisé.
- **altSecurityIdentities** spécifie que l'attribut de mappage du nom d'entrée de l'utilisateur d'Active Directory est utilisé.
- **userCertificate;binary=** indique que le certificat entier de IdM ou AD est utilisé.

Vous pouvez définir de nombreuses options de correspondance, mais certaines des options généralement configurées sont les suivantes :

- **<ISSUER>CN=[...]** spécifie que l'attribut de l'émetteur du certificat utilisé est vérifié pour s'assurer qu'il correspond à celui-ci.
- **<SUBJECT>.*,DC=MY,DC=DOMAIN** indique que l'objet du certificat est vérifié.

2. Activez la journalisation du System Security Services Daemon (SSSD) en ajoutant **debug_level = 9** au fichier **/etc/sss/sss.conf** sur le serveur IdM :

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

3. Restart SSSD:

```
# systemctl restart sssd
```

4. Vous devriez voir l'entrée suivante dans le fichier **/var/log/sss/sss_idm.example.com.log** si le mappage a été lu correctement :

■

```
[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:<l>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})].
```

- Si votre règle de mappage contient une syntaxe invalide, une entrée similaire à la suivante peut être observée dans le fichier journal :

```
[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:<l>{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
```

- Vérifiez la syntaxe de votre règle de mappage.

```
# ipa certmaprule-show smartcardrule
Rule name: smartcardrule
Mapping rule: |(userCertificate;binary={cert!bin})(ipacertmapdata=X509:<l>{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
```

- Si nécessaire, modifiez votre règle de mappage des certificats :

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

Ressources supplémentaires

- Voir la page de manuel **sss-certmap**.

9.5.2. Vérification de l'utilisateur associé à un certificat de carte à puce

Si vous rencontrez des problèmes d'authentification avec une carte à puce, vérifiez que le bon utilisateur est associé à votre certificat de carte à puce.

Conditions préalables

- Vous avez installé et configuré votre serveur et votre client de gestion des identités (IdM) pour une utilisation avec des cartes à puce.
- Vous pouvez détecter votre lecteur de carte à puce et afficher le contenu de votre carte à puce. Voir [Test de l'accès par carte à puce sur le système](#) .

- Vous avez associé votre certificat de carte à puce à un utilisateur IdM. Voir [Règles de mappage des certificats pour la configuration de l'authentification sur les cartes à puce](#).
- Vous disposez d'une copie du certificat de votre carte à puce au format PEM, par exemple **cert.pem**.

Procédure

1. Vérifiez que l'utilisateur est associé à votre certificat de carte à puce :

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

Si l'utilisateur ou le domaine ne sont pas corrects, vérifiez comment vos certificats sont associés aux utilisateurs. Voir [Vérification de la correspondance entre les certificats et les utilisateurs](#) .

2. Vérifier si l'entrée utilisateur contient le certificat :

```
# ipa user-show idmuser1
User login: idmuser1
[...]
Certificate:MIIEejCCAUkGAWIBAgIBCzANBgkqhkiG9w0BAQsFADAzMREwDwYDVQQKDAhJ
UEEuVEVTVDEeMBwGA1UEAwwVQ2VydGlmaWNhdGUgQXV0aG9yaXR5MB4XD
```

3. Si votre entrée utilisateur ne contient pas le certificat, ajoutez votre certificat codé en base-64 à l'entrée utilisateur :
 - a. Créez une variable d'environnement contenant le certificat dont l'en-tête et le pied de page ont été supprimés et concaténés en une seule ligne, ce qui correspond au format attendu par la commande **ipa user-add-cert**:

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

Notez que le certificat contenu dans le fichier **idmuser1.crt** doit être au format PEM.

- b. Ajoutez le certificat au profil de **idmuser1** à l'aide de la commande **ipa user-add-cert**:

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

- c. Effacer le cache du System Security Services Daemon (SSSD).

```
# 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. Exécutez à nouveau **ipa certmap-match** pour confirmer que l'utilisateur est associé à votre certificat de carte à puce.