Red Hat Enterprise Linux 8

IdM での証明書の管理

証明書の発行、証明書ベースの認証の設定、および証明書の有効性の制御

Last Updated: 2024-06-26

Red Hat Enterprise Linux 8 IdM での証明書の管理

証明書の発行、証明書ベースの認証の設定、および証明書の有効性の制御

法律上の通知

概要

管理者は X.509 証明書を使用してユーザー、ホスト、サービスを認証し、デジタル署名と暗号化を有効にします。Red Hat Identity Management (IdM) では、統合認証局 (CA) または外部の CA を使用して証明書を管理できます。 certmonger サービス、certutil ツール、または Ansible Playbook を使用して、証明書を要求および更新できます。IdM サーバーの Web サーバー証明書と LDAP サーバー証明書を置き換えるには、手動アクションを実行する必要があります。 管理者は、軽量のサブ CA を作成して、VPN ゲートウェイのユーザー証明書など、特定の目的の証明書を発行できます。管理者は、この VPN ゲートウェイが不要になったときにサブ CA の証明書を取り消すことで、このサービスのすべての証明書を無効にすることができます。

目次

RED HAT ドキュメントへのフィードバック (英語のみ) 6

第1章 IDENTITY MANAGEMENT の公開鍵証明書 7
1.1. IDM の認証局 7
1.2. 証明書および KERBEROS の比較 8
1.3. IDM でユーザーを認証する証明書を使用する利点と問題点 8

第2章 統合 IDM CA を使用したユーザー、ホスト、およびサービスの証明書の管理 10
2.1. IDM WEB UI でのユーザー、ホスト、またはサービスの新規証明書の要求 11
2.2. CERTUTIL を使用した IDM CA からのユーザー、ホスト、またはサービスの新規証明書の要求 12
2.3. OPENSSL を使用した IDM CA からのユーザー、ホスト、またはサービスの新規証明書の要求 13
2.4. 関連情報 14

第3章 IDM ユーザー、ホスト、およびサービスの外部署名証明書の管理 15
3.1. IDM CLI を使用した外部 CA からの IDM ユーザー、ホスト、またはサービスへの証明書の追加 15
3.2. IDM WEB UI を使用した外部 CA からの IDM ユーザー、ホスト、またはサービスへの証明書の追加 16
3.3. IDM CLI を使用した IDM ユーザー、ホスト、またはサービスアカウントからの外部 CA 発行の証明書削除 16
3.4. IDM WEB UI を使用した IDM ユーザー、ホスト、またはサービスアカウントからの外部 CA 発行証明書の削除 17
3.5. 関連情報 18

第4章 IDM と機能する証明書形式への変換 19
4.1. IDM での証明書の形式およびエンコード 19
4.2. IDM ユーザーアカウントに読み込む外部証明書の変換 20
4.3. 証明書をブラウザーに読み込むための準備 23
4.4. IDM における証明書関連のコマンドおよび形式 23

第5章 IDENTITY MANAGEMENT での証明書プロファイルの作成および管理 25
5.1. 証明書プロファイルの概要 25
5.2. 証明書プロファイルの作成 26
5.3. CA アクセス制御リストの概要 27
5.4. 証明書プロファイルへのアクセスを制御する CA ACL の定義 27
5.5. 証明書プロファイルおよび CA ACL を使用した証明書発行 29
5.6. 証明書プロファイルの変更 31
5.7. 証明書プロファイルの設定パラメーター 32

第6章 IDM での証明書の有効性の管理 35
6.1. IDM CA が発行した既存の証明書の効力の管理 35
6.2. IDM CA が発行する将来の証明書の効力の管理 35
6.3. IDM WEBUI での証明書の有効期限の表示 35
6.4. CLI での証明書の有効期限の表示 36
6.5. 統合 IDM CA を使用した証明書の失効 36
6.6. 統合 IDM CA を使用した証明書の復元 38

第7章 スマートカード認証用の IDENTITY MANAGEMENT の設定 40
7.1. スマートカード認証用の IDM サーバーの設定 40
7.2. ANSIBLE を使用したスマートカード認証用の IDM サーバー設定 42
7.3. スマートカード認証用の IDM クライアントの設定 46
7.4. ANSIBLE を使用したスマートカード認証用の IDM クライアント設定 48
7.5. IDM WEB UI のユーザーエントリーへの証明書の追加 50
7.6. IDM CLI でユーザーエントリーへの証明書の追加 52
7.7. スマートカードを管理および使用するツールのインストール 53

7.8. スマートカードを準備し、証明書と鍵をスマートカードにアップロードする 54
7.9. スマートカードを使用して IDM へのログイン 55
7.10. IDM クライアントでスマートカード認証を使用して GDM にログインする 57
7.11. SU コマンドでのスマートカード認証の使用 57

第8章 IDM でスマートカード認証用に ADCS が発行した証明書の設定 59
8.1. 信頼の設定と証明書の使用に必要な WINDOWS SERVER 設定 59
8.2. SFTP を使用して ACTIVE DIRECTORY から証明書のコピー 60
8.3. ADCS 証明書を使用したスマートカード認証用の IDM サーバーおよびクライアントの設定 60
8.4. PFX ファイルの変換 62
8.5. スマートカードを管理および使用するツールのインストール 62
8.6. スマートカードを準備し、証明書と鍵をスマートカードにアップロードする 63
8.7. SSSD.CONF でタイムアウトの設定 65
8.8. スマートカード認証用の証明書マッピングルールの作成 66

第9章 IDENTITY MANAGEMENT での証明書マッピングルールの設定 67
9.1. 認証を設定するための証明書マッピングルール 67
9.2. IDM における ID マッピングルールのコンポーネント 68
9.3. マッチングルールで使用する証明書からのデータの取得 69
9.4. IDM に保存されたユーザーの証明書マッピングの設定 69
9.5. ACTIVE DIRECTORY ドメインとの信頼に対する証明書マッピングルール 75
9.6. AD ユーザーエントリーに証明書全体が含まれるユーザーに証明書マッピングを設定 76
9.7. ユーザー証明書をユーザーアカウントにマッピングするように AD が設定されている場合に、証明書マッピングの設定 78
9.8. AD ユーザーエントリーに証明書やマッピングデータが含まれていない場合に、証明書マッピングの設定 81
9.9. 複数のアイデンティティーマッピングルールを 1 つに結合 86
9.10. 関連情報 87

第10章 IDM クライアントのデスクトップに保存されている証明書を使用した認証の設定 88
10.1. WEB UI での証明書認証用の IDENTITY MANAGEMENT SERVER の設定 88
10.2. 新しいユーザー証明書を要求し、クライアントにエクスポート 89
10.3. 証明書とユーザーが互いにリンクしていることの確認 91
10.4. 証明書認証を有効にするためのブラウザーの設定 91
10.5. IDENTITY MANAGEMENT ユーザーとして証明書を使用した IDENTITY MANAGEMENT WEB UI の認証 94
10.6. 証明書を使用して CLI への認証を可能にするように IDM クライアントを設定 95

第11章 IDM CA 更新サーバーの使用 96
11.1. IDM CA 更新サーバーの説明 96
11.2. IDM CA 更新サーバーの変更およびリセット 97

第12章 外部署名 CA 証明書の管理 99
12.1. IDM での外部署名 CA から自己署名 CA への切り替え 99
12.2. IDM での自己署名 CA から外部署名 CA への切り替え 100
12.3. 外部 CA を使用した IDM CA 更新サーバー証明書の更新 100

第13章 IDM がオフライン時に期限切れのシステム証明書の更新 103
13.1. CA 更新サーバーでの期限切れのシステム証明書の更新 103
13.2. 更新後の IDM ドメイン内の他の IDM サーバーの検証 104

第14章 IDM レプリカでまだ有効期限が切れていない場合の WEB サーバーと LDAP サーバーの証明書の置き換え 106

第15章 IDM デプロイメント全体で期限切れになった WEB サーバーと LDAP サーバーの証明書を置き換える 108

第16章 IDM CA サーバーでの CRL の生成 112
16.1. IDM サーバーでの CRL 生成の停止 112

16.2. IDM レプリカサーバーでの CRL 生成の開始 112
16.3. CRL 更新間隔の変更 113

第17章 CA 更新サーバーと CRL パブリッシャーのロールを実行するサーバーの使用の停止 115

第18章 CERTMONGER を使用したサービスの IDM 証明書の取得 119
18.1. CERTMONGER の概要 119
18.2. CERTMONGER を使用したサービスの IDM 証明書の取得 120
18.3. サービス証明書を要求する CERTMONGER の通信フロー 121
18.4. CERTMONGER が追跡する証明書要求の詳細を表示 124
18.5. 証明書追跡の開始および停止 125
18.6. 証明書を手動で更新 126
18.7. CERTMONGER が CA レプリカでの IDM 証明書の追跡を再開 127
18.8. CERTMONGER での SCEP の使用 128

第19章 RHEL システムロールを使用して証明書を要求する 133
19.1. CERTIFICATE RHEL システムロール 133
19.2. CERTIFICATE RHEL システムロールを使用した新しい自己署名証明書の要求 133
19.3. CERTIFICATE RHEL システムロールを使用した IDM CA からの新しい証明書の要求 134
19.4. CERTIFICATE RHEL システムロールを使用して証明書発行前または発行後に実行するコマンドを指定する 135

第20章 証明書のサブセットだけに信頼するアプリケーションを制限する手順 137
20.1. 軽量サブ CA の管理 137
20.2. IDM WEB UI からのサブ CA 証明書のダウンロード 144
20.3. WEB サーバーおよびクライアント認証用の CA ACL の作成 144
20.4. CERTMONGER を使用したサービスの IDM 証明書の取得 148
20.5. サービス証明書を要求する CERTMONGER の通信フロー 150
20.6. シングルインスタンスの APACHE HTTP SERVER 設定 153
20.7. APACHE HTTP SERVER への TLS 暗号化の追加 154
20.8. APACHE HTTP サーバーでサポートされる TLS プロトコルバージョンの設定 156
20.9. APACHE HTTP サーバーで対応している暗号の設定 157
20.10. TLS クライアント証明書認証の設定 158
20.11. 新しいユーザー証明書を要求し、クライアントにエクスポート 159
20.12. 証明書認証を有効にするためのブラウザーの設定 161

第21章 関連する証明書の特定グループの迅速な無効化 164
21.1. IDM CLI での CA ACL の無効化 164
21.2. IDM サブ CA の無効化 165

第22章 ANSIBLE を使用した IDM 証明書の管理 167
22.1. ANSIBLE を使用した IDM ホスト、サービス、ユーザーの SSL 証明書の要求 167
22.2. ANSIBLE を使用して IDM ホスト、サービス、ユーザーの SSL 証明書を取り消す 168
22.3. ANSIBLE を使用して IDM ユーザー、ホスト、およびサービスの SSL 証明書を復元する 169
22.4. ANSIBLE を使用して IDM ユーザー、ホスト、およびサービスの SSL 証明書を取得する 170

第23章 IDM HEALTHCHECK を使用した証明書の検証 172
23.1. IDM 証明書の HEALTHCHECK テスト 172
23.2. HEALTHCHECK ツールを使用した証明書のスクリーニング 173

第24章 IDM HEALTHCHECK を使用したシステム証明書の検証 175
24.1. システム証明書の HEALTHCHECK テスト 175
24.2. HEALTHCHECK を使用したシステム証明書のスクリーニング 176

第25章 IDM が内部で使用する証明書について 177
25.1. IDM の内部証明書について 177

25.2. IDM の内部の証明書 178
25.3. IDM 内部証明書の更新プロセス 183
25.4. 関連情報 183

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに関するご意見やご感想をお寄せください。また、改善点があればお知らせください。

Jira からのフィードバック送信 (アカウントが必要)

1. Jira の Web サイトにログインします。
2. 上部のナビゲーションバーで **Create** をクリックします。
3. **Summary** フィールドにわかりやすいタイトルを入力します。
4. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも追加してください。
5. ダイアログの下部にある **Create** をクリックします。

第1章 IDENTITY MANAGEMENT の公開鍵証明書

X.509 公開鍵証明書は、Identity Management (IdM)のユーザー、ホスト、およびサービスを認証するために使用されます。X.509 証明書は、認証のほかに、デジタル署名と暗号化も可能にし、プライバシー、完全性、否認不可を実現します。

証明書には、以下の情報が含まれます。

- 証明書が認証する発行先
- 証明書を署名した CA の発行元
- 証明書の有効性の開始日と終了日
- 証明書の有効な用途
- 発行先の公開鍵

公開鍵により暗号化したメッセージは、対応した秘密鍵により複号できます。証明書および、勝目所に含まれる公開鍵は、公開できますが、ユーザー、ホスト、またはサービスは、対応の秘密鍵を秘密にしておく必要があります。

1.1. IDM の認証局

認証局は信頼の階層で機能します。内部認証局 (CA) がある IdM 環境では、CA が署名している正しい証明書を、すべての IdM ホスト、ユーザー、およびサービスが信頼します。このルート CA とは別に、IdM は、ルート CA から証明書に署名する権限を付与されたサブ CA にも対応します。多くの場合、このようなサブ CA が署名できる証明書は、VPN 証明書 など、特定の種類の証明書です。最後に、IdM は外部 CA の使用に対応します。以下 の表は、IdM の各種 CA の用途に関する詳細を紹介します。

表1.1 IdM での統合および外部 CA の用途の比較

CA の名前	説明	用途	関連リンク
ipa CA	Dogtag アップストリームプロジェクトをベースとする統合 CA	統合 CA は、ユーザー、ホスト、およびサービスの証明書の作成、取り消し、および発行が可能です。	ipa CA を使用した新規ユーザー証明書の要求およびクライアントへのエクスポート
IdM サブ CA	**ipa** CA の下位となる統合 CA	IdM サブ CA は、**ipa** CA が証明書の署名を許可した CA です。多くの場合に、これらの証明書は、VPN 証明書など、特定の種類の証明書です。	証明書のサブセットだけに信頼するアプリケーションを制限する手順
外部 CA	外部 CA は、統合 IdM CA またはそのサブ CA 以外の CA です。	IdM ツールを使用して、これらの CA が発行する証明書をユーザー、サービス、またはホストに対して追加し、削除します。	IdM ユーザー、ホスト、およびサービスの外部署名証明書の管理

証明書の観点からは、自己署名 IdM CA と、外部署名の間に違いがありません。

CA のロールの目的は以下のとおりです。

- デジタル証明書を発行する。
- 証明書を署名して、証明書に名前のある発行先が公開鍵を所有することを認定する。発行先は、ユーザー、ホスト、またはサービスのいずれかです。
- 証明書をキャンセルして、証明書失効リスト (CRL) および Online Certificate Status Protocol (OCSP) で失効ステータスを提供できる。

関連情報

- CA サービスの計画 を参照してください。

1.2. 証明書および KERBEROS の比較

証明書は、Kerberos チケットが実行したのと同様の機能を実行します。Kerberos は、セキュアでないネットワーク上で通信するノードが、安全な方法で互いに ID を証明できるように、チケットベースで機能するコンピューターネットワーク認証プロトコルです。以下の表では、Kerberos および X.509 の証明書を比較します。

表1.2 証明書および Kerberos の比較

特徴	Kerberos	X.509
認証	はい	はい
プライバシー	任意	はい
インテグリティー	任意	はい
関係する暗号の種類	対称	非対称
デフォルトの有効性	短い (1 日)	長い (2 年)

デフォルトでは、Identity Management の Kerberos は、通信相手の ID のみを保証します。

1.3. IDM でユーザーを認証する証明書を使用する利点と問題点

IdM でユーザーを認証する証明書を使用する利点は次のとおりです。

- 通常、スマートカードの秘密鍵を保護する PIN は、通常のパスワードよりも簡単で覚えやすいものになります。
- デバイスによっては、スマートカードに保存されている秘密鍵をエクスポートできません。これによりセキュリティーが強化されます。
- スマートカードはログアウトを自動化できます。IdM は、ユーザーがリーダーからスマートカードを取り外すと、ユーザーをログアウトするように設定できます。
- 秘密鍵を盗むには、スマートカードへの物理的なアクセスが必要で、スマートカードはハッキング攻撃に対して安全になります。

- スマートカード認証は 2 要素認証の一例です。つまり、所有しているもの (カード) と、知っているもの (PIN) の両方が必要です。
- スマートカードは、電子メールの暗号化など、他の目的に使用できる鍵を提供するため、パスワードよりも柔軟性があります。
- IdM クライアントである共有マシンでスマートカードを使用しても、通常、システム管理者に対して追加で発生する問題はありません。実際、スマートカード認証は共有マシンにとって理想的な選択肢です。

IdM のユーザーを認証する証明書を使用する問題点は次のとおりです。

- スマートカードまたは証明書を紛失したり忘れたりすることで、実質的にロックアウトされる可能性があります。
- PIN を複数回誤入力すると、カードがロックされる可能性があります。
- 一般に、セキュリティー担当者や承認者などによる要求と承認の間には中間ステップがあります。セキュリティー担当者または管理者が、IdM で **ipa cert-request** コマンドを実行する必要があります。
- スマートカードとリーダーは、ベンダーとドライバーに固有である傾向があります。多くのリーダーはさまざまなカードに使用できますが、一部のベンダーのスマートカードは、別のベンダーのリーダーや、その目的で設計されていないリーダーでは機能しない場合があります。
- 証明書とスマートカードの場合は、管理者が短期間で習得しなければならない内容が多くなります。

第2章 統合 IDM CA を使用したユーザー、ホスト、およびサービスの証明書の管理

統合 CA、**ipa** CA、およびそのサブ CA を使用して Identity Management (IdM)で証明書を管理する方法の詳細は、以下のセクションを参照してください。

- IdM Web UI を使用したユーザー、ホスト、またはサービスの新しい証明書の要求
- IdM CLI を使用した IdM CA からのユーザー、ホスト、またはサービスの新規証明書の要求
  - certutil を使用した IdM CA からのユーザー、ホスト、またはサービスの新規証明書の要求
    - **certutil** ユーティリティーを使用して IdM CA から新規ユーザー証明書を要求して、その証明書を IdM クライアントにエクスポートする具体的な例については、新しいユーザー証明書を要求し、クライアントにエクスポート を参照してください。
  - openssl を使用した IdM CA からのユーザー、ホスト、またはサービスの新規証明書の要求

certmonger ユーティリティーを使用して、IdM CA からサービスの新しい証明書を要求することもできます。詳細は certmonger を使用した IdM CA からのサービスの新規証明書の要求 を参照してください。

前提条件

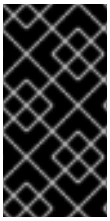
- IdM デプロイメントに統合 CA が含まれている。
  - IdM で CA サービスを計画する方法の詳細は、CA サービスの計画 を参照してください。
  - 統合 DNS と統合 CA をルート CA として使用する IdM サーバーをインストールする方法は、IdM サーバーのインストール: 統合 DNS と統合 CA をルート CA として使用する場合 を参照してください。
  - 統合 DNS と外部 CA を root CA として使用する IdM サーバーをインストールする方法は、IdM サーバーのインストール: 統合 DNS と外部 CA を root CA として使用する場合 を参照してください。
  - 統合 DNS なしで、統合 CA をルート CA として使用する IdM サーバーをインストールする方法は、IdM サーバーのインストール: 統合 DNS がなく統合 CA を root CA として使用する場合 を参照してください。
  - (オプション) IdM デプロイメントは、証明書で認証するユーザーに対応します。
    - IdM デプロイメントが IdM クライアントファイルシステムに保存されている証明書を使用したユーザー認証をサポートするように設定する方法は、IdM クライアントのデスクトップに保存されている証明書を使用した認証の設定 を参照してください。
    - IdM クライアントに挿入するスマートカードに保存されている証明書を使用してユーザー認証をサポートするように IdM デプロイメントを設定する方法は、スマートカード認証用の Identity Management の設定 を参照してください。
    - Active Directory 証明書システムが発行するスマートカードを使用してユーザー認証をサポートするように IdM デプロイメントを設定する方法は、IdM でスマートカード認証用に ADCS が発行した証明書の設定 を参照してください。

2.1. IDM WEB UI でのユーザー、ホスト、またはサービスの新規証明書の要求

Identity Management (IdM) の Web UI を使用して、統合 IdM 認証局 (CA) から IdM エンティティー (**ipa** CA またはそのサブ CA) の新しい証明書を要求するするには、次の手順に従います。

IdM エンティティーには、以下が含まれます。

- ユーザー
- ホスト
- サービス



重要

サービスは通常、秘密鍵の保存先となる専用のサービスノードで実行されます。サービスの秘密鍵を IdM サーバーにコピーすることは、安全ではないとみなされます。したがって、サービスの証明書を要求する場合には、サービスノードで証明書署名要求 (CSR) を作成します。

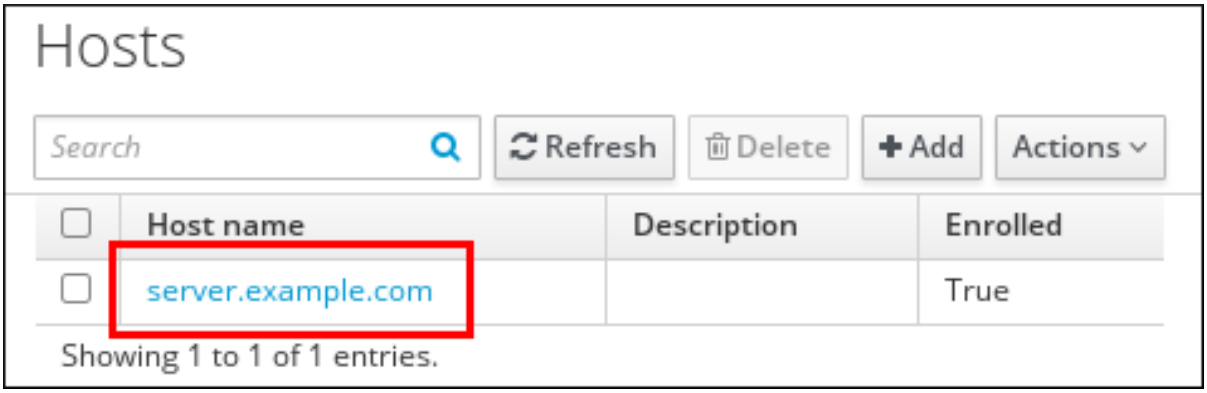
前提条件

- IdM デプロイメントに統合 CA が含まれている。
- IdM 管理者として IdM Web UI にログインしている。

手順

1. **Identity** タブで、 **Users**、**Hosts**、または **Services** のサブタブを選択します。
2. ユーザー、ホスト、またはサービス名をクリックして、設定ページを開きます。

   図2.1 ホストのリスト

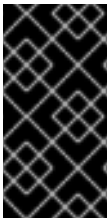


3. **Actions → New Certificate** をクリックします。
4. 必要に応じて、発行元の CA およびプロファイル ID を選択します。
5. 画面の **certutil** コマンドライン (CLI) ユーティリティーの使用手順に従います。
6. **Issue** をクリックします。

2.2. CERTUTIL を使用した IDM CA からのユーザー、ホスト、またはサービスの新規証明書の要求

標準の IdM 状況で Identity Management (IdM) ユーザー、ホスト、またはサービスの証明書を要求するには、**certutil** ユーティリティーを使用できます。ホストまたはサービスの Kerberos エイリアスが証明書を使用できるようにするには、代わりに openssl ユーティリティーを使用して証明書を要求 します。

以下の手順に従って、**certutil** を使用して、**ipa** (IdM 認証局(CA)) から IdM ユーザー、ホスト、またはサービスの証明書を要求します。



重要

サービスは通常、秘密鍵の保存先となる専用のサービスノードで実行されます。サービスの秘密鍵を IdM サーバーにコピーすることは、安全ではないとみなされます。したがって、サービスの証明書を要求する場合には、サービスノードで証明書署名要求 (CSR) を作成します。

前提条件

- IdM デプロイメントに統合 CA が含まれている。
- IdM 管理者として IdM コマンドラインインターフェイス (CLI) にログインしている。

手順

1. 証明書データベースの一時ディレクトリーを作成します。

```
# mkdir ~/certdb/
```

2. 以下のように、新しい一時証明書データベースを作成します。

```
# certutil -N -d ~/certdb/
```

3. CSR を作成し、出力をファイルにリダイレクトします。たとえば、4096 ビット証明書の CSR を作成し、発行先を **CN=server.example.com,O=EXAMPLE.COM** に設定するには、以下を実行します。

```
# certutil -R -d ~/certdb/ -a -g 4096 -s "CN=server.example.com,O=EXAMPLE.COM" -8 server.example.com > certificate_request.csr
```

4. IdM サーバーで実行している CA に証明書要求ファイルを送信します。新しく発行した証明書に関連付ける Kerberos プリンシパルを指定します。

```
# ipa cert-request certificate_request.csr --principal=host/server.example.com
```

IdM の **ipa cert-request** コマンドは、次のデフォルトを使用します。

- **caIPAserviceCert** 証明書プロファイル
  カスタムプロファイルを選択するには、**--profile-id** オプションを使用します。
- 統合 IdM のルート CA (**ipa**)
  サブ CA を選択するには、**--ca** オプションを使用します。

関連情報

関連情報

- **ipa cert-request --help** コマンドの出力を参照してください。
- Identity Management での証明書プロファイルの作成および管理 を参照してください。

2.3. OPENSSL を使用した IDM CA からのユーザー、ホスト、またはサービスの新規証明書の要求

ホストまたはサービスの Kerberos エイリアスが証明書を使用できるようにするには、**openssl** ユーティリティーを使用して、Identity Management (IdM) ホストまたはサービスの証明書を要求してください。標準の状況では、代わりに certutil ユーティリティーを使用して新しい証明書を要求することを検討してください。

以下の手順に従って、**openssl** を使用して、IdM ホスト、または **ipa** (IdM 認証局) からサービスの証明書を要求します。

重要

サービスは通常、秘密鍵の保存先となる専用のサービスノードで実行されます。サービスの秘密鍵を IdM サーバーにコピーすることは、安全ではないとみなされます。したがって、サービスの証明書を要求する場合には、サービスノードで証明書署名要求 (CSR) を作成します。

前提条件

- IdM デプロイメントに統合 CA が含まれている。
- IdM 管理者として IdM コマンドラインインターフェイス (CLI) にログインしている。

手順

1. Kerberos プリンシパル **test/server.example.com** のエイリアスを 1 つ以上作成します。例: **test1/server.example.com** および **test2/server.example.com**
2. CSR で dnsName (**server.example.com**) と otherName (**test2/server.example.com**) の subjectAltName を追加します。これには、UPN otherName および subjectAltName を指定する以下の行を追加するように、**openssl.conf** ファイルを 設定します。

```
otherName=1.3.6.1.4.1.311.20.2.3;UTF8:test2/server.example.com@EXAMPLE.COM
DNS.1 = server.example.com
```

3. **openssl** を使用して証明書要求を作成します。

```
openssl req -new -newkey rsa:2048 -keyout test2service.key -sha256 -nodes -out
certificate_request.csr -config openssl.conf
```

4. IdM サーバーで実行している CA に証明書要求ファイルを送信します。新しく発行した証明書に関連付ける Kerberos プリンシパルを指定します。

```
# ipa cert-request certificate_request.csr --principal=host/server.example.com
```

IdM の **ipa cert-request** コマンドは、次のデフォルトを使用します。

- **caIPAserviceCert** 証明書プロファイル
  カスタムプロファイルを選択するには、**--profile-id** オプションを使用します。

- 統合 IdM のルート CA (**ipa**)
  サブ CA を選択するには、**--ca** オプションを使用します。

関連情報

- **ipa cert-request --help** コマンドの出力を参照してください。
- Identity Management での証明書プロファイルの作成および管理 を参照してください。

2.4. 関連情報

- 統合 IdM CA を使用した証明書の失効 を参照してください。
- 統合 IdM CA を使用した証明書の復元 を参照してください。
- 証明書のサブセットのみを信頼するアプリケーションの制限 を参照してください。

第3章 IDM ユーザー、ホスト、およびサービスの外部署名証明書の管理

この章では、Identity Management (IdM) コマンドラインインターフェイス (CLI) および IdM Web UI を使用して、外部認証局 (CA) によって発行されたユーザー、ホスト、またはサービス証明書を追加または削除する方法について説明します。

3.1. IDM CLI を使用した外部 CA からの IDM ユーザー、ホスト、またはサービスへの証明書の追加

Identity Management (IdM) 管理者は、Identity Management (IdM) CLI を使用して、外部署名証明書を IdM ユーザー、ホスト、またはサービスのアカウントに追加できます。

前提条件

- 管理ユーザーの Ticket-Granting Ticket (TGT) を取得している。

手順

- IdM ユーザーに証明書を追加するには、次のコマンドを実行します。

```
$ ipa user-add-cert user --certificate=MIQTPrajQAwg...
```

このコマンドでは、以下の情報を指定する必要があります。

  - ユーザーの名前。
  - base64 でエンコードされた DER 証明書

注記

証明書の内容をコマンドラインにコピーして貼り付ける代わりに、証明書を DER 形式に変換し、これを base64 に再プロビジョニングできます。たとえば、**user_cert.pem**証明書を**ユーザー**に追加するには、次のコマンドを入力します。

```
$ ipa user-add-cert user --certificate="$(openssl x509 -outform der -in
user_cert.pem | base64 -w 0)"
```

ipa user-add-cert コマンドは、オプションを追加せずに対話的に実行できます。

IdM ホストに証明書を追加するには、次のコマンドを実行します。

- **ipa host-add-cert**

IdM サービスに証明書を追加するには、次のコマンドを実行します。

- **ipa service-add-cert**

関連情報

- 統合 IdM CA を使用したユーザー、ホスト、およびサービスの証明書の管理

3.2. IDM WEB UI を使用した外部 CA からの IDM ユーザー、ホスト、またはサービスへの証明書の追加

Identity Management (IdM) 管理者は、Identity Management (IdM) Web UI を使用して、外部署名証明書を IdM ユーザー、ホスト、またはサービスのアカウントに追加できます。

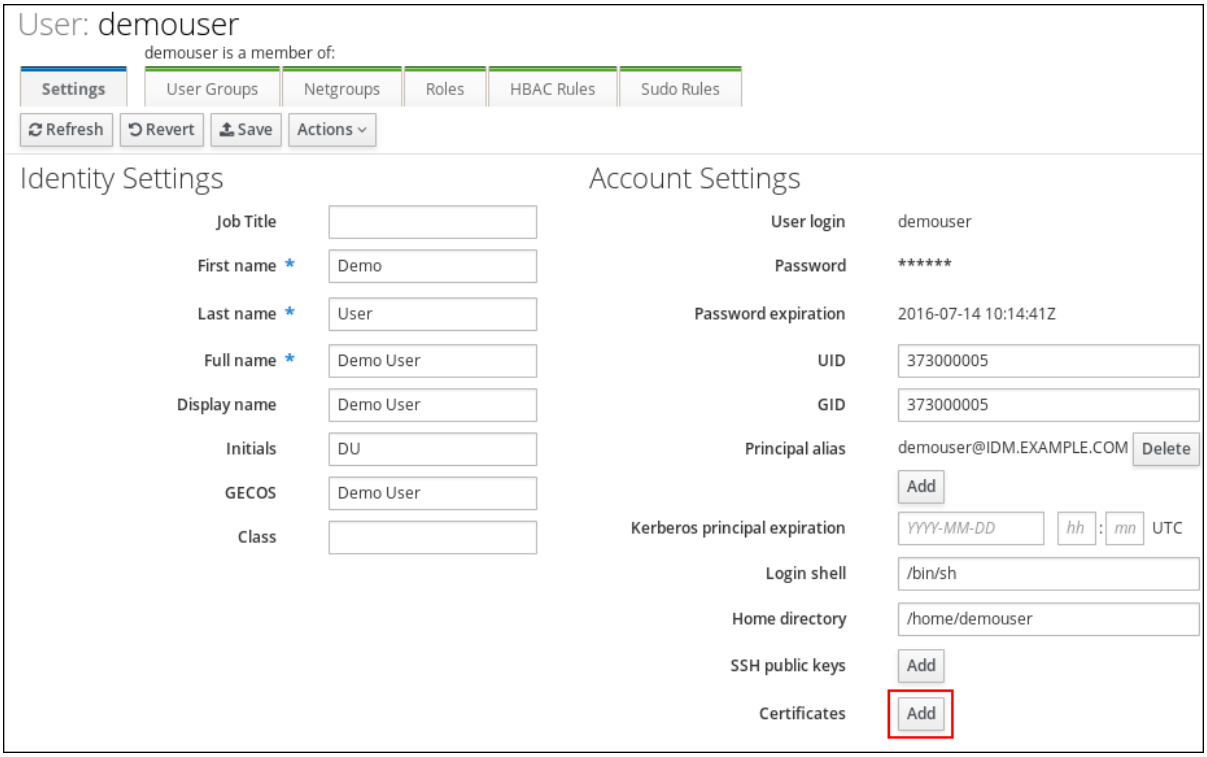
前提条件

- 管理者として Identity Management (IdM) Web UI にログインしている。

手順

1. **Identity** タブを開き、**Users**、**Hosts**、または **Services** サブタブを選択します。
2. ユーザー、ホスト、またはサービス名をクリックして、設定ページを開きます。
3. **Certificates** エントリーの横にある **Add** をクリックします。

図3.1 ユーザーアカウントへの証明書の追加



4. Base64 または PEM でエンコードされた形式で証明書をテキストフィールドに貼り付け、**Add** をクリックします。
5. **Save** をクリックして変更を保存します。

3.3. IDM CLI を使用した IDM ユーザー、ホスト、またはサービスアカウントからの外部 CA 発行の証明書削除

Identity Management (IdM) 管理者は、Identity Management (IdM) CLI を使用して、外部署名証明書を IdM ユーザー、ホスト、またはサービスのアカウントから削除できます。

前提条件

- 管理ユーザーの Ticket-Granting Ticket (TGT) を取得している。

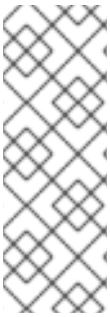
手順

- IdM ユーザーから証明書を削除するには、次のコマンドを実行します。

```
$ ipa user-remove-cert user --certificate=MIQTPrajQAwg...
```

このコマンドでは、以下の情報を指定する必要があります。

  - ユーザーの名前。
  - base64 でエンコードされた DER 証明書



注記

証明書の内容をコマンドラインにコピーして貼り付ける代わりに、証明書を DER 形式に変換し、これを base64 に再プロビジョニングできます。たとえば、**user_cert.pem** 証明書を **user** から削除するには、次のように入力します。

```
$ ipa user-remove-cert user --certificate="$(openssl x509 -outform der -in
user_cert.pem | base64 -w 0)"
```

ipa user-remove-cert コマンドは、オプションを追加せずに対話的に実行できます。

IdM ホストから証明書を削除するには、次のコマンドを実行します。

- **ipa host-remove-cert**

IdM サービスから証明書を削除するには、次のコマンドを実行します。

- **ipa service-remove-cert**

関連情報

- 統合 IdM CA を使用したユーザー、ホスト、およびサービスの証明書の管理

3.4. IDM WEB UI を使用した IDM ユーザー、ホスト、またはサービスアカウントからの外部 CA 発行証明書の削除

Identity Management (IdM) 管理者は、Identity Management (IdM) Web UI を使用して、外部署名証明書を IdM ユーザー、ホスト、またはサービスのアカウントから削除できます。

前提条件

- 管理者として Identity Management (IdM) Web UI にログインしている。

手順

1. **Identity** タブを開き、**Users**、**Hosts**、または **Services** サブタブを選択します。
2. ユーザー、ホスト、またはサービス名をクリックして、設定ページを開きます。

3. 削除する証明書の横にある **Actions** をクリックし、**Delete** を選択します。
4. **Save** をクリックして変更を保存します。

3.5. 関連情報

- Ansible Playbook を使用して IdM サービスエントリーに外部署名証明書を存在させる手順

第4章 IDM と機能する証明書形式への変換

このユーザーストーリーでは、IdM システム管理者が、特定の IdM コマンドで証明書の正しい形式を使用するようにする方法を説明します。これは、たとえば以下の状況で役に立ちます。

- ユーザープロファイルに外部証明書を読み込んでいる。詳細は、IdM ユーザーアカウントに読み込む外部証明書の変換 を参照してください。
- IdM サーバーをスマートカード認証用に設定する場合、または IdM クライアントをスマートカード認証用に設定する場合 に、外部 CA 証明書を使用して、ユーザーが外部認証局から発行された証明書が含まれるスマートカードで IdM を認証できるようにしている。
- NSS データベースから、証明書と秘密鍵の両方を含む pkcs #12 形式で、証明書をエクスポートしている。詳細は、NSS データベースから PKCS #12 ファイルへの証明書と秘密鍵のエクスポート を参照してください。

4.1. IDM での証明書の形式およびエンコード

IdM におけるスマートカード認証を含む証明書認証は、ユーザーが提示する証明書と、ユーザーの IdM プロファイルに保存されている証明書または証明書データを比較することによって進められます。

システムの設定

IdM プロファイルに格納されるものは証明書のみで、対応する秘密鍵ではありません。また、認証中に、ユーザーが、対応する秘密鍵の所有していることを表示する必要があります。ユーザーは、証明書と秘密鍵の両方が含まれる PKCS #12 ファイル、またはこれら 2 つのファイル (証明書が含まれるファイルと、秘密鍵が含まれているファイル) のいずれかを提示して行います。

したがって、ユーザープロファイルに証明書を読み込むなどのプロセスでは、秘密鍵を含まない証明書ファイルのみが使用できます。

同様に、システム管理者が外部の CA 証明書を提供している場合は、パブリックデータ (秘密鍵がない証明書) のみを提供します。スマートカード認証用の IdM サーバーまたは IdM クライアントを設定する **ipa-advise** ユーティリティーでは、外部 CA の証明書が含まれる入力ファイルが必要ですが、秘密鍵は必要ありません。

証明書のエンコーディング

2 つの一般的な証明書エンコーディング **PEM** (Privacy-enhanced Electronic Mail) および **DER** (Distinguished Encoding Rules) があります。**base64** 形式は **PEM** 形式とほぼ同じですが、ヘッダーおよびフッター (**-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----**) は含まれません。

DER を使用してエンコードされた証明書は、バイナリーファイルの X509 デジタル証明書です。証明書はバイナリーファイルで、人間は判読できません。**DER** ファイルは、ファイル名の拡張子 **.der** を使用することもありますが、ファイル名の拡張子 **.crt** および **.cer** が **DER** 証明書が含まれることもあります。鍵を含む **DER** ファイルの名前は **.key** です。

PEM Base64 を使用してエンコードされた証明書は、人間が判読できるファイルです。このファイルには、-----BEGIN ...の行頭に付けられた ASCII (Base64) のデータが含まれます。**PEM** ファイルは、**.pem** ファイル拡張子を使用することもありますが、ファイル名の拡張子 **.crt** および **.cer** に **PEM** 証明書が含まれる場合もあります。鍵を含む **PEM** ファイルの名前は **.key** です。

ipa コマンドには、許可される証明書の種類に、さまざまな制限があります。たとえば、**ipa user-add-cert** コマンドでは、**base64** 形式でエンコードされた証明書のみが使用できますが、**ipa-server-certinstall** は、**PEM**、**DER**、**PKCS #7**、**PKCS #8** および **PKCS #12** の証明書が使用できます。

表4.1 証明書のエンコーディング

エンコーディング形式	人間が判別可能	一般的なファイル名の拡張子	エンコーディング形式を使用できる IdM コマンドの例
PEM/base64	はい	.pem、.crt、.cer	ipa user-add-cert、ipa-server-certinstall など
DER	いいえ	.der、.crt、.cer	ipa-server-certinstall など

IdM の証明書関連のコマンドおよび形式 は、コマンドが受け入れる証明書形式を含むその他の **ipa** コマンドをリスト表示します。

ユーザー認証

ブラウザーのデータベースに秘密鍵と証明書の両方を保存することで、Web UI を使用して IdM にアクセスすると、証明書に対応する秘密鍵をユーザーが所有していることを証明します。

CLI を使用して IdM にアクセスすると、以下のいずれかの方法で、証明書に対応する秘密鍵をユーザーが所有していることを証明します。

- ユーザーは、**kinit -X** コマンドの **X509_user_identity** パラメーターの値として、証明書と鍵の両方が含まれるスマートカードに接続するスマートカードモジュールへのパスを追加します。

```
$ kinit -X X509_user_identity='PKCS11:opensc-pkcs11.so' idm_user
```

- ユーザーは、**kinit -X** コマンドの **X509_user_identity** パラメーターの値として、証明書が含まれるファイルと、秘密鍵が含まれるファイルの 2 つを追加します。

```
$ kinit -X X509_user_identity='FILE:`/path/to/cert.pem,/path/to/cert.key`' idm_user
```

便利な証明書コマンド

証明書データ (発行先や発行者など) を表示するには、次のコマンドを実行します。

```
$ openssl x509 -noout -text -in ca.pem
```

2 つの証明書で、異なる行を比較するには、次のコマンドを実行します。

```
$ diff cert1.crt cert2.crt
```

2 つの証明書の出力を 2 列で表示して、2 つの証明書で異なる行を比較するには、次のコマンドを実行します。

```
$ diff cert1.crt cert2.crt -y
```

4.2. IDM ユーザーアカウントに読み込む外部証明書の変換

このセクションでは、外部証明書がユーザーエントリーに追加される前に、適切にエンコードされおよび形式が正しいことを確認する方法を説明します。

4.2.1. 前提条件

- 証明書が Active Directory 認証局によって発行され、**PEM** エンコーディングを使用する場合は、**PEM** ファイルが **UNIX** 形式に変換されていることを確認します。ファイルを変換するには、dos2unix パッケージが提供する **dos2unix** ユーティリティーを使用します。

4.2.2. IdM CLI での外部証明書の変換および IdM ユーザーアカウントへの読み込み

IdM CLI では、最初の行および最後の行 (-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----) が削除された **PEM** 証明書のみが使用できます。

以下の手順に従って、外部証明書を **PEM** 形式に変換し、IdM CLI を使用して IdM ユーザーアカウントに追加します。

手順

1. 証明書を **PEM** 形式に変換します。
   - 証明書が **DER** 形式の場合は、次のようになります。

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

   - ファイルが **PKCS #12** 形式で、共通のファイル名の拡張子が **.pfx** および **.p12** で、証明書、秘密鍵、およびその他のデータが含まれている場合は、**openssl pkcs12** ユーティリティーを使用して証明書をデプロイメントします。プロンプトが表示されたら、そのファイルに保存されている秘密鍵をパスワードで保護します。

     ```
     $ openssl pkcs12 -in cert_and_key.p12 -clcerts -nokeys -out cert.pem
     Enter Import Password:
     ```

2. 管理者の認証情報を取得します。

   ```
   $ kinit admin
   ```

3. 以下のいずれかの方法で、**IdM CLI** を使用して証明書をユーザーアカウントに追加します。
   - 文字列を **ipa user-add-cert** コマンドに追加する前に、**sed** ユーティリティーを使用して **PEM** ファイルの最初の行および最後の行 (-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----) を削除します。

     ```
     $ ipa user-add-cert some_user --certificate="$(sed -e '/BEGIN
     CERTIFICATE/d;/END CERTIFICATE/d' cert.pem)"
     ```

   - 最初の行と最後の行 (-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----) がない証明書ファイルのコンテンツを、**ipa user-add-cert** コマンドにコピーして貼り付けます。

     ```
     $ ipa user-add-cert some_user --
     certificate=MIIDlzCCAn+gAwIBAgIBATANBgkqhki...
     ```

注記

最初の行および最後の行 (-----BEGIN CERTIFICATE----- および -----END CERTIFICATE-----) を削除せずに、直接証明書を含む **PEM** ファイルを、**ipa user-add-cert** コマンドへの入力として直接渡すことはできません。

```
$ ipa user-add-cert some_user --cert=some_user_cert.pem
```

このコマンドの結果、ipa: ERROR: Base64 decoding failed: Incorrect padding エラーメッセージが表示されます。

4. 必要に応じて、システムで証明書が許可されているかどうかを確認するには、次のコマンドを実行します。

```
[idm_user@r8server]$ ipa user-show some_user
```

4.2.3. IdM ユーザーアカウントに読み込むための IdM Web UI での外部証明書の変換

以下の手順に従って、外部証明書を **PEM** 形式に変換し、これを IdM Web UI の IdM ユーザーアカウントに追加します。

手順

1. **CLI** を使用して、証明書を **PEM** 形式に変換します。
   - 証明書が **DER** 形式の場合は、次のようになります。

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

   - ファイルが **PKCS #12** 形式で、共通のファイル名の拡張子が **.pfx** および **.p12** で、証明書、秘密鍵、およびその他のデータが含まれている場合は、**openssl pkcs12** ユーティリティーを使用して証明書をデプロイメントします。プロンプトが表示されたら、そのファイルに保存されている秘密鍵をパスワードで保護します。

```
$ openssl pkcs12 -in cert_and_key.p12 -clcerts -nokeys -out cert.pem
Enter Import Password:
```

2. エディターで証明書を開き、コンテンツをコピーします。IdM Web UI には **PEM** 形式および **base64** 形式が使用できるため、ヘッダー行-----BEGIN CERTIFICATE-----およびフッター行-----END CERTIFICATE-----を含めることができます。
3. IdM Web UI で、セキュリティー担当者としてログインします。
4. **Identity → Users → some_user** の順に選択します。
5. **証明書** の横にある **追加** をクリックします。
6. 表示されるウインドウに、PEM 形式のコンテンツを貼り付けます。
7. **Add** をクリックします。

証明書がシステムに受け入れられる場合は、ユーザープロファイルの **証明書** 間でリストに表示されるのを確認できます。

4.3. 証明書をブラウザーに読み込むための準備

ユーザー証明書をブラウザーにインポートする前に、証明書と対応する秘密鍵が **PKCS #12** 形式にあることを確認してください。その他の準備作業が必要な一般的な状況は、次の 2 つです。

- 証明書が NSS データベースにある。この状況での続行方法の詳細は、Exporting a certificate and private key from an NSS database into a PKCS #12 file を参照してください。
- 証明書と秘密鍵が別々の **PEM** ファイルにある。この状況での続行方法の詳細は、Combining certificate and private key PEM files into a PKCS #12 file を参照してください。

その後、**PEM** 形式の CA 証明書と、**PKCS #12** 形式のユーザー証明書をブラウザーにインポートするには、証明書認証を有効にするためのブラウザーの設定 および Identity Management ユーザーとして証明書を使用した Identity Management Web UI の認証 の手順に従います。

4.3.1. NSS データベースから PKCS #12 ファイルへの証明書と秘密鍵のエクスポート

手順

1. 証明書を、NSS データベースから **PKCS12** 形式にエクスポートするには、**pk12util** コマンドを使用します。たとえば、**~/certdb** ディレクトリーに保存されている NSS データベースから、**~/some_user.p12** ファイルに、**some_user** ニックネームを持つ証明書をエクスポートする場合は、次のコマンドを実行します。

```
$ pk12util -d ~/certdb -o ~/some_user.p12 -n some_user
Enter Password or Pin for "NSS Certificate DB":
Enter password for PKCS12 file:
Re-enter password:
pk12util: PKCS12 EXPORT SUCCESSFUL
```

2. **.p12** ファイルに適切なパーミッションを設定します。

```
# chmod 600 ~/some_user.p12
```

PKCS #12 ファイルには秘密鍵も含まれるため、その他のユーザーがファイルを使用できないように保護する必要があります。それ以外の場合は、ユーザー権限になりすますことができます。

4.3.2. 証明書と秘密鍵の PEM ファイルを PKCS #12 ファイルに統合

以下の手順に従って、証明書と、別の **PEM** ファイルに保存されている対応する鍵を **PKCS #12** ファイルに組み合わせます。

手順

- **certfile.cer** に保存されている証明書と、**certfile.key** に保存されている鍵を、証明書および鍵の両方が含まれる **certfile.p12** ファイルに追加します。

```
$ openssl pkcs12 -export -in certfile.cer -inkey certfile.key -out certfile.p12
```

4.4. IDM における証明書関連のコマンドおよび形式

次の表に、IdM における証明書関連のコマンドを、使用できる形式と共に示します。

表4.2 IdM 証明書コマンドおよび形式

コマンド	使用できる形式	備考
ipa user-add-cert some_user --certificate	base64 PEM 証明書	
ipa-server-certinstall	PEM および DER 証明書、PKCS#7 証明書チェーン、PKCS#8 および生の秘密鍵、PKCS#12 証明書および秘密鍵	
ipa-cacert-manage install	DER、PEM、PKCS#7	
ipa-cacert-manage renew --external-cert-file	PEM および DER 証明書、PKCS#7 証明書チェーン	
ipa-ca-install --external-cert-file	PEM および DER 証明書、PKCS#7 証明書チェーン	
ipa cert-show <cert serial> --certificate-out /path/to/file.pem	該当なし	**<cert_serial>** シリアル番号がある証明書で、PEM でエンコードされた **file.pem** ファイルを作成します。
ipa cert-show <cert serial> --certificate-out /path/to/file.pem	該当なし	**<cert_serial>** シリアル番号がある証明書で、PEM でエンコードされた **file.pem** ファイルを作成します。**--chain** オプションを使用すると、PEM ファイルに、証明書チェーンを含む証明書が含まれます。
ipa cert-request --certificate-out=FILE /path/to/req.csr	該当なし	新しい証明書で、PEM 形式の **req.csr** ファイルを作成します。
ipa cert-request --certificate-out=FILE /path/to/req.csr	該当なし	新しい証明書で、PEM 形式の **req.csr** ファイルを作成します。**--chain** オプションを使用する場合には、PEM ファイルに、証明書チェーンを含む証明書が含まれます。

第5章 IDENTITY MANAGEMENT での証明書プロファイルの作成および管理

証明書プロファイルは、証明書署名要求 (CSR) が受け入れ可能であるかどうかを判断するために、証明書の署名時に認証局 (CA) により使用されます。受け入れ可能な場合には、証明書にどのような機能および拡張機能が含まれるかを判断します。特定のタイプの証明書を発行するのに、証明書プロファイルを関連付けます。証明書プロファイルと CA アクセス制御リスト (ACL) を組み合わせることで、カスタム証明書プロファイルへのアクセスを定義し、制御できます。

証明書プロファイルの作成方法の説明では、例として S/MIME 証明書を使用します。一部のメールプログラムは、Secure Multipurpose Internet Mail Extension (S/MIME) プロトコルを使用して、デジタル署名および暗号化されたメールをサポートします。S/MIME を使用して電子メールメッセージの署名または暗号化を行うには、メッセージの送信者に S/MIME 証明書が必要です。

- 証明書プロファイルの概要
- 証明書プロファイルの作成
- CA アクセス制御リストの概要
- 証明書プロファイルへのアクセスを制御する CA ACL の定義
- 証明書プロファイルおよび CA ACL を使用した証明書発行
- 証明書プロファイルの変更
- 証明書プロファイルの設定パラメーター

5.1. 証明書プロファイルの概要

証明書プロファイルを使用すると、証明書の内容や、以下のような証明書の発行時の制約を決定できます。

- 証明書署名要求をエンコードするために使用する署名アルゴリズム。
- 証明書のデフォルトの有効性。
- 証明書の取り消しに使用できる失効理由。
- プリンシパルのコモンネーム (cn) が subject alternative name フィールドにコピーされる場合。
- 証明書に存在する必要がある機能およびエクステンション。

特定のタイプの証明書を発行するのに、証明書プロファイルを 1 つ関連付けます。IdM のユーザー、サービス、およびホストには、さまざまな証明書プロファイルを定義できます。IdM には、デフォルトで次の証明書プロファイルが含まれています。

- **caIPAserviceCert**
- **IECUserRoles**
- **kdcs_PKINIT_Certs** (内部で使用)

さらに、特定の目的で証明書を発行できるカスタムプロファイルを作成およびインポートできます。たとえば、特定のプロファイルの使用を 1 つのユーザーまたはグループに制限し、他のユーザーやグルー

プがそのプロファイルを使用して証明書を認証用に発行できないようにさせます。カスタム証明書プロファイルを作成するには、**ipa certprofile** コマンドを使用します。

関連情報

- **ipa help certprofile** コマンドを参照してください。

5.2. 証明書プロファイルの作成

S/MIME 証明書を要求するプロファイル設定ファイルを作成して、コマンドラインから証明書プロファイルを作成するには、次の手順に従います。

手順

1. 既存のデフォルトプロファイルをコピーしてカスタムプロファイルを作成します。

```
$ ipa certprofile-show --out smime.cfg caIPAserviceCert
-----------------------------------------------
Profile configuration stored in file 'smime.cfg'
-----------------------------------------------
  Profile ID: caIPAserviceCert
  Profile description: Standard profile for network services
  Store issued certificates: TRUE
```

2. テキストエディターで新たに作成されたプロファイル設定ファイルを開きます。

```
$ vi  smime.cfg
```

3. **プロファイル ID** は、**smime** など、プロファイルの用途を反映する名前に変更します。

注記

新規作成されたプロファイルをインポートする場合は、**profileId** フィールドがある場合には、profiledId はコマンドラインで指定した ID と一致する必要があります。

4. Extended Key Usage 設定を更新します。Extended Key Usage のデフォルト拡張設定は、TLS サーバーとクライアント認証向けです。たとえば、S/MIME の場合に、電子メール保護のために Extended Key Usage を設定する必要があります。

```
policyset.serverCertSet.7.default.params.exKeyUsageOIDs=1.3.6.1.5.5.7.3.4
```

5. 新しいプロファイルをインポートします。

```
$ ipa certprofile-import smime --file smime.cfg \
  --desc "S/MIME certificates" --store TRUE

-----------------------
Imported profile "smime"
-----------------------
  Profile ID: smime
  Profile description: S/MIME certificates
  Store issued certificates: TRUE
```

検証手順

- 新しい証明書プロファイルがインポートされたことを確認します。

```
$ ipa certprofile-find

------------------
4 profiles matched
------------------
  Profile ID: caIPAserviceCert
  Profile description: Standard profile for network services
  Store issued certificates: TRUE

  Profile ID: IECUserRoles
  Profile description: User profile that includes IECUserRoles extension from request
  Store issued certificates: TRUE

  Profile ID: KDCs_PKINIT_Certs
  Profile description: Profile for PKINIT support by KDCs
  Store issued certificates: TRUE

  Profile ID: smime
  Profile description: S/MIME certificates
  Store issued certificates: TRUE
----------------------------
Number of entries returned 4
----------------------------
```

関連情報

- **ipa help certprofile** を参照してください。
- RFC 5280, section 4.2.1.12 を参照してください。

5.3. CA アクセス制御リストの概要

認証局のアクセス制御リスト (CA ACL) ルールは、どのプリンシパルにどのプロファイルを使用して証明書を発行するかを定義します。これには、CA ACL を使用できます。以下に例を示します。

- 特定のプロファイルで証明書を発行できるユーザー、ホスト、またはサービスを決定します。
- 証明書を発行できる IdM 認証局またはサブ CA の特定

たとえば、CA ACL を使用して、ロンドンオフィス関連の IdM グループに所属するユーザーだけが、ロンドンのオフィスから作業する社員向けのプロファイルを使用するように限定できます。

CA ACL ルールを管理する **ipa caacl** ユーティリティーを使用すると、特権ユーザーは、指定した CA ACL を追加、表示、変更、または削除できます。

関連情報

- **ipa help caacl** を参照してください。

5.4. 証明書プロファイルへのアクセスを制御する CA ACL の定義

caacl ユーティリティーを使用して CA アクセス制御リスト (ACL) ルールを定義し、グループ内のユーザーがカスタム証明書プロファイルにアクセスできるようにするには、次の手順に従います。この場合は、S/MIME ユーザーのグループと CA ACL を作成し、そのグループのユーザーが **smime** 証明書プロファイルにアクセスできるようにする方法を説明します。

前提条件

- IdM 管理者の認証情報を取得していることを確認している。

手順

1. 証明書プロファイルのユーザーに新しいグループを作成します。

```
$ ipa group-add smime_users_group
--------------------------------
Added group "smime users group"
--------------------------------
  Group name: smime_users_group
  GID: 75400001
```

2. **smime_user_group** グループに追加する新規ユーザーを作成します。

```
$ ipa user-add smime_user
First name: smime
Last name: user
----------------------
Added user "smime_user"
----------------------
  User login: smime_user
  First name: smime
  Last name: user
  Full name: smime user
  Display name: smime user
  Initials: TU
  Home directory: /home/smime_user
  GECOS: smime user
  Login shell: /bin/sh
  Principal name: smime_user@IDM.EXAMPLE.COM
  Principal alias: smime_user@IDM.EXAMPLE.COM
  Email address: smime_user@idm.example.com
  UID: 1505000004
  GID: 1505000004
  Password: False
  Member of groups: ipausers
  Kerberos keys available: False
```

3. **smime_user** は **smime_users_group** に追加します。

```
$ ipa group-add-member smime_users_group --users=smime_user
  Group name: smime_users_group
  GID: 1505000003
  Member users: smime_user
-------------------------
Number of members added 1
-------------------------
```

4. CA ACL を作成し、グループのユーザーが証明書プロファイルにアクセスできるようにします。

```
$ ipa caacl-add smime_acl
-------------------------
Added CA ACL "smime_acl"
-------------------------
  ACL name: smime_acl
  Enabled: TRUE
```

5. CA ACL にユーザーグループを追加します。

```
$ ipa caacl-add-user smime_acl --group smime_users_group
  ACL name: smime_acl
  Enabled: TRUE
  User Groups: smime_users_group
-------------------------
Number of members added 1
-------------------------
```

6. CA ACL に証明書プロファイルを追加します。

```
$ ipa caacl-add-profile smime_acl --certprofile smime
  ACL name: smime_acl
  Enabled: TRUE
  Profiles: smime
  User Groups: smime_users_group
-------------------------
Number of members added 1
-------------------------
```

検証手順

- 作成した CA ACL の詳細を表示します。

```
$ ipa caacl-show smime_acl
  ACL name: smime_acl
  Enabled: TRUE
  Profiles: smime
  User Groups: smime_users_group
...
```

関連情報

- **ipa** の man ページを参照してください。
- **ipa help caacl** を参照してください。

5.5. 証明書プロファイルおよび CA ACL を使用した証明書発行

認証局のアクセス制御リスト (CA ACL) が許可する場合は、証明書プロファイルを使用して証明書を要求できます。CA ACL 経由でアクセスが付与されたカスタム証明書プロファイルを使用して、ユーザーの S/MIME 証明書を要求するするには、次の手順に従います。

前提条件

- 証明書プロファイルが作成されている。
- ユーザーが必要な証明書プロファイルを使用して証明書を要求可能な CA ACL が作成されている。

注記

cert-request コマンドを実行するユーザーが以下の条件に該当する場合には、CA ACL チェックを回避できます。

- **admin** ユーザーである。
- **Request Certificate ignoring CA ACLs** パーミッションがある。

手順

1. ユーザーの証明書要求を生成します。例: OpenSSL の使用:

```
$ openssl req -new -newkey rsa:2048 -days 365 -nodes -keyout private.key -out cert.csr -subj '/CN=smime_user'
```

2. IdM CA からユーザーの新しい証明書を要求します。

```
$ ipa cert-request cert.csr --principal=smime_user --profile-id=smime
```

必要に応じて、--ca **sub-CA_name** オプションをコマンドに指定して、ルート CA ではなくサブ CA から証明書を要求します。

検証手順

- 新たに発行した証明書がユーザーに割り当てられていることを確認します。

```
$ ipa user-show user
  User login: user
  ...
  Certificate: MIICfzCCAWcCAQA...
  ...
```

関連情報

- **ipa(a)** の man ページを参照してください。
- **ipa help user-show** コマンドを参照してください。
- **ipa help cert-request** コマンドを参照してください。
- **openssl(lssl)** の man ページを参照してください。

5.6. 証明書プロファイルの変更

ipa certprofile-mod コマンドを使用して、コマンドラインで証明書プロファイルを直接変更するには、次の手順に従います。

手順

1. 変更する証明書プロファイルの証明書プロファイル ID を確認します。IdM に現在保存されている証明書プロファイルをすべて表示するには、次のコマンドを実行します。

```
# ipa certprofile-find

------------------
4 profiles matched
------------------
  Profile ID: caIPAserviceCert
  Profile description: Standard profile for network services
  Store issued certificates: TRUE

  Profile ID: IECUserRoles
  ...

  Profile ID: smime
  Profile description: S/MIME certificates
  Store issued certificates: TRUE
----------------------------
Number of entries returned
----------------------------
```

2. 証明書プロファイルの説明を変更します。たとえば、既存のプロファイルを使用して S/MIME 証明書のカスタム証明書プロファイルを作成した場合は、説明を新しい使用方法に合わせて変更します。

```
# ipa certprofile-mod smime --desc "New certificate profile description"
-----------------------------------
Modified Certificate Profile "smime"
-----------------------------------
   Profile ID: smime
   Profile description: New certificate profile description
   Store issued certificates: TRUE
```

3. テキストエディターでカスタマー証明書プロファイルファイルを開き、要件に合わせてを変更します。

```
# vi smime.cfg
```

証明書プロファイルの設定ファイルで指定できるオプションの詳細は、証明書プロファイル設定パラメーター を参照してください。

4. 既存の証明書プロファイルの設定ファイルを更新します。

```
# ipa certprofile-mod _profile_ID_ --file=smime.cfg
```

検証手順

- 証明書プロファイルが更新されたことを確認します。

```
$ ipa certprofile-show smime
  Profile ID: smime
  Profile description: New certificate profile description
  Store issued certificates: TRUE
```

関連情報

- **ipa(a)** の man ページを参照してください。
- **ipa help certprofile-mod** を参照してください。

5.7. 証明書プロファイルの設定パラメーター

証明書プロファイル設定パラメーターは、CA プロファイルディレクトリー **/var/lib/pki/pki-tomcat/ca/profiles/ca** の **profile_name** ファイルに保存されます。プロファイルのパラメーター (defaults、inputs、outputs および constraints) はすべて、ポリシーセット 1 つで設定されます。証明書プロファイルのポリシーセットには **policyset.policyName.policyNumber** という名前があります。たとえば、ポリシーセット **serverCertSet** の場合、以下を実行します。

```
policyset.list=serverCertSet
policyset.serverCertSet.list=1,2,3,4,5,6,7,8
policyset.serverCertSet.1.constraint.class_id=subjectNameConstraintImpl
policyset.serverCertSet.1.constraint.name=Subject Name Constraint
policyset.serverCertSet.1.constraint.params.pattern=CN=[^,]+,.+
policyset.serverCertSet.1.constraint.params.accept=true
policyset.serverCertSet.1.default.class_id=subjectNameDefaultImpl
policyset.serverCertSet.1.default.name=Subject Name Default
policyset.serverCertSet.1.default.params.name=CN=$request.req_subject_name.cn$, OU=pki-ipa,
O=IPA
policyset.serverCertSet.2.constraint.class_id=validityConstraintImpl
policyset.serverCertSet.2.constraint.name=Validity Constraint
policyset.serverCertSet.2.constraint.params.range=740
policyset.serverCertSet.2.constraint.params.notBeforeCheck=false
policyset.serverCertSet.2.constraint.params.notAfterCheck=false
policyset.serverCertSet.2.default.class_id=validityDefaultImpl
policyset.serverCertSet.2.default.name=Validity Default
policyset.serverCertSet.2.default.params.range=731
policyset.serverCertSet.2.default.params.startTime=0
```

各ポリシーセットには、評価順で証明書プロファイルに設定されたポリシーのリスト (ポリシー ID 番号) が含まれます。サーバーは、受信するリクエストごとに各ポリシーセットを評価します。証明書要求を 1 つ受信すると、セット 1 つが評価され、その他のプロファイルセットは無視されます。デュアルキーペアが発行されると、最初の証明書要求に対して最初のポリシーセットが評価され、2 番目の証明書要求に対して 2 番目のセットが評価されます。デュアルキーペアを発行するときに、単一の証明書を発行する場合や 2 つ以上のセットを発行する場合は、複数のポリシーセットは必要ありません。

表5.1 証明書プロファイル設定ファイルパラメーター

パラメーター	説明

パラメーター	説明
desc	end-entities ページに表示される証明書プロファイルの説明 (フリーテキスト)。例: **desc=This certificate profile is for enrolling server certificates with agent authentication**。
enable	プロファイルを有効にし、end-entities ページからアクセスできるようにします。例: **enable=true**
auth.instance_id	証明書要求の認証に使用する認証マネージャープラグインを設定します。自動登録では、認証に成功すると、CA は証明書をすぐに発行します。認証に失敗した場合や、認証プラグインが指定されていない場合には、エージェントにより手動で承認されるように、要求はキューに配置されます。例: **auth.instance_id=AgentCertAuth**
authz.acl	承認の制約を指定します。これは主に、グループ評価のアクセス制御リスト (ACL) を設定するために使用されます。たとえば、**caCMCUserCert** パラメーターでは、CMC リクエストの署名者が Certificate Manager Agents グループに所属する必要があります。 **authz.acl=group="Certificate Manager Agents** ディレクトリーベースのユーザー証明書の更新では、このオプションを使用して、元の要求元と現在認証されているユーザーが同じであることを確認できます。エンティティーは、承認の評価前に認証を行う (バインドまたは基本的にはシステムにログイン) 必要があります。
name	証明書プロファイルの名前。たとえば、**name=Agent-Authenticated Server Certificate Enrollment** など。この名前は、エンドユーザーの登録または更新ページに表示されます。
input.list	証明書プロファイルが対応する入力を名前別に表示します。たとえば、**input.list=i1,i2** のようになります。
input.input_id.class_id	入力 ID (input.list にリスト表示される入力の名前) 別に入力の java クラス名を示します。例: **input.i1.class_id=certReqInputImpl**。

パラメーター	説明
output.list	証明書プロファイルの出力形式を名前別に表示します。例: **output.list=o1**。
output.output_id.class_id	output.list に名前が付けられた出力形式の Java クラス名を指定します。例: **output.o1.class_id=certOutputImpl**。
policyset.list	設定した証明書プロファイルルールをリスト表示します。デュアル証明書の場合には、1 セットのルールが署名鍵に、もう 1 つは暗号鍵に適用されます。単一の証明書は、証明書プロファイルルールのセットを 1 つだけ使用します。例: **policyset.list=serverCertSet**
policyset.policyset_id.list	評価順で証明書プロファイルに設定されたポリシーのリスト (ポリシー ID 番号) を表示します。例: **policyset.serverCertSet.list=1,2,3,4,5,6,7,8**
policyset.policyset_id.policy_number.constraint.class_id	プロファイルルールにデフォルト設定された制約プラグインの java クラス名を指定します。例: policyset.serverCertSet.1.constraint.class_id=subjectNameConstraintImpl。
policyset.policyset_id.policy_number.constraint.name	制約のユーザー定義の名前を指定します。例: policyset.serverCertSet.1.constraint.name=Subject Name Constraint
policyset.policyset_id.policy_number.constraint.params.attribute	制約で使用可能な属性値を指定します。設定可能な属性は、制約のタイプによって異なります。例: policyset.serverCertSet.1.constraint.params.pattern=CN=.*
policyset.policyset_id.policy_number.default.class_id	プロファイルルールに、デフォルトセットの java クラス名を指定します。例: policyset.serverCertSet.1.default.class_id=userSubjectNameDefaultImpl
policyset.policyset_id.policy_number.default.name	デフォルトのユーザー定義名を指定します。例: policyset.serverCertSet.1.default.name=Subject Name Default
policyset.policyset_id.policy_number.default.params.attribute	デフォルトに使用可能な属性の値を指定します。設定可能な属性は、デフォルトのタイプによって異なります。例: policyset.serverCertSet.1.default.params.name=CN=(Name)$request.requestor_name$

第6章 IDM での証明書の有効性の管理

Identity Management (IdM) では、既存の証明書と将来発行する証明書の両方の有効性を管理できますが、その方法は異なります。

6.1. IDM CA が発行した既存の証明書の効力の管理

IdM では、証明書の有効期限を表示する次の方法を使用できます。

- IdM WebUI での有効期限の表示
- CLI での有効期限の表示

IdM CA により発行した既存の証明書の効力を次の方法で管理できます。

- 元の証明書署名要求 (CSR) または秘密鍵から生成された新しい CSR を使用して新しい証明書を要求することにより、証明書を更新します。次のユーティリティーを使用して、新しい証明書を要求できます。

  certmonger

  certmonger を使用して、サービス証明書を要求できます。証明書の有効期限が切れる前に、**certmonger** は証明書を自動的に更新するため、サービス証明書の継続的な効力が保証されます。詳細は、certmonger でサービスの IdM 証明書の取得 を参照してください。

  certutil

  certutil を使用して、ユーザー、ホスト、およびサービスの証明書を更新できます。ユーザー証明書を要求する方法は、新しいユーザー証明書を要求し、クライアントにエクスポート を参照してください。

  openssl

  openssl を使用して、ユーザー、ホスト、およびサービスの証明書を更新できます。

- 証明書を取り消します。詳細は、次を参照してください。
  - IdM WebUI で統合 IdM CA での証明書の失効
  - IdM CLI で統合 IdM CA での証明書の失効
- 証明書が一時的に取り消された場合は、証明書を復元します。詳細は、次を参照してください。
  - IdM WebUI で統合 IdM CA で証明書の復元
  - IdM CLI で統合 IdM CA で証明書の復元

6.2. IDM CA が発行する将来の証明書の効力の管理

IdM CA が発行する将来の証明書の効力を管理するには、証明書プロファイルを変更、インポート、または作成します。詳細は Identity Management での証明書プロファイルの作成および管理 を参照してください。

6.3. IDM WEBUI での証明書の有効期限の表示

IdM WebUI を使用して、IdM CA が発行したすべての証明書の有効期限を表示できます。

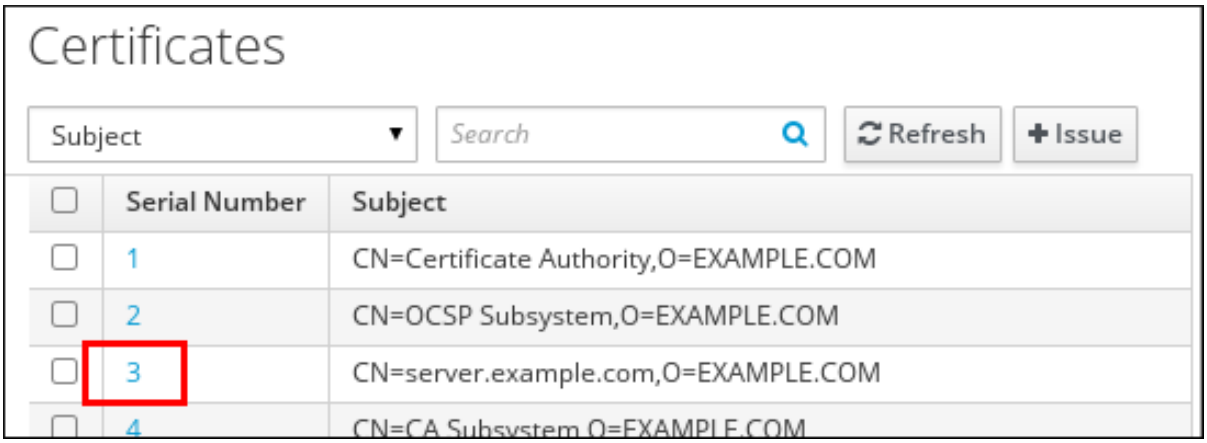
前提条件

- 管理者の認証情報を取得していることを確認している。

手順

1. **Authentication** メニューで、**Certificates** > **Certificates** をクリックします。
2. 証明書のシリアル番号をクリックして、証明書情報ページを開きます。

図6.1 証明書のリスト



3. 証明書の情報ページで、**失効日** 情報を見つけます。

6.4. CLI での証明書の有効期限の表示

コマンドラインインターフェイス (CLI) を使用して、証明書の有効期限を表示できます。

手順

- **openssl** ユーティリティーを使用して、人間が読める形式でファイルを開きます。

```
$ openssl x509 -noout -text -in ca.pem
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: O = IDM.EXAMPLE.COM, CN = Certificate Authority
        Validity
            Not Before: Oct 30 19:39:14 2017 GMT
            Not After : Oct 30 19:39:14 2037 GMT
```

6.5. 統合 IDM CA を使用した証明書の失効

6.5.1. 証明書失効の理由

失効した証明書は無効であり、認証に使用できません。理由 6 の **証明書の保留** を除き、すべての失効は永続的です。

デフォルトの失効理由は 0 (**未指定**) です。

表6.1 証明書の失効理由

ID	理由	説明
0	指定なし	
1	鍵が侵害された	証明書を発行した鍵が信頼されなくなった。 考えられる原因 - トークンの消失、ファイルへの不適切なアクセス。
2	CA が侵害された	証明書を発行した CA は信頼されなくなった。
3	所属が変更した	考えられる原因: * 退職したか、別の部門に移動した。 * ホストまたはサービスが廃止された。
4	置き換え	現在の証明書から新しい証明書に置き換えられた。
5	運用停止	ホストまたはサービスの使用を停止している。
6	証明書が保留になっている	証明書は一時的に取り消されている。証明書は後で復元できます。
8	CRL から削除された	証明書は、証明書失効リスト (CRL) に含まれていない。
9	特権が撤回された	ユーザー、ホスト、またはサービスは、証明書の使用を許可されなくなった。
10	侵害された属性機関 (Attribute Authority)	属性機関証明書は信頼されなくなった。

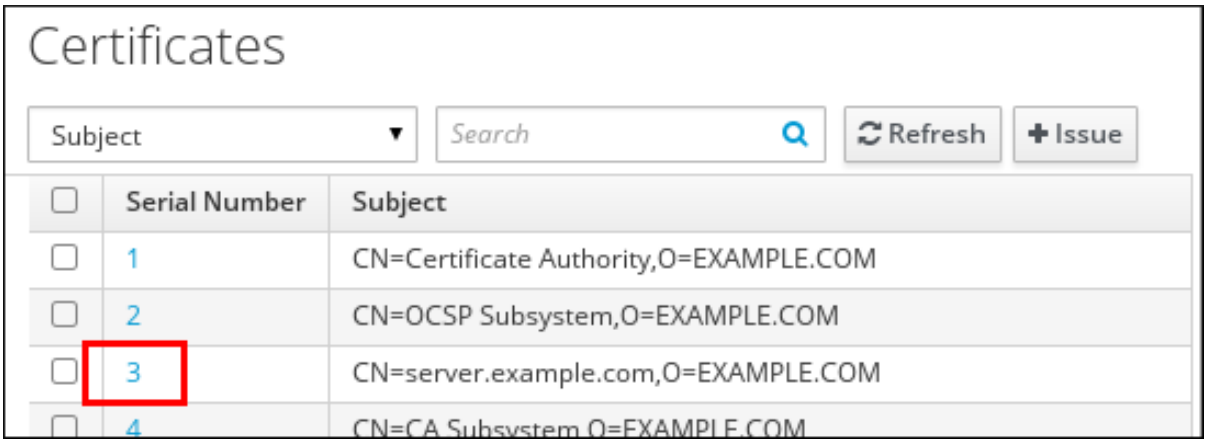
6.5.2. IdM WebUI を使用して統合 IdM CA で証明書の失効

証明書の秘密鍵を紛失した場合は、証明書を無効にして不正使用を防ぐ必要があります。IdM CA が発行した証明書を IdM WebUI を使用して取り消すには、この手順を完了します。

手順

1. **Authentication** > **Certificates** > **Certificates** をクリックします。
2. 証明書のシリアル番号をクリックして、証明書情報ページを開きます。

図6.2 証明書のリスト



3. 証明書情報ページで、**Actions → Revoke Certificate** をクリックします。
4. 取り消しの理由を選択し、**Revoke** をクリックします。詳細は 証明書失効の理由 を参照してください。

6.5.3. IdM CLI を使用した統合 IdM CA での証明書の失効

証明書の秘密鍵を紛失した場合は、証明書を無効にして不正使用を防ぐ必要があります。IdM CLI で、IdM CA が発行した証明書を取り消すには、以下の手順を行います。

手順

- **ipa cert-revoke** コマンドを使用して、次を指定します。
  - 証明書のシリアル番号
  - 失効理由の ID 番号。詳細は 証明書失効の理由 を参照してください。

たとえば、理由 1 (**侵害された鍵**) のためにシリアル番号 **1032** の証明書を失効させるには、次のコマンドを実行します。

```
$ ipa cert-revoke 1032 --revocation-reason=1
```

新しい証明書の要求の詳細は、次のドキュメントを参照してください。

- 新しいユーザー証明書を要求し、クライアントにエクスポート
- certmonger でサービスの IdM 証明書の取得

6.6. 統合 IDM CA を使用した証明書の復元

理由 6 (**証明書の保留**) のために証明書が失効し、証明書の秘密鍵が侵害されていない場合は、証明書を再度復元できます。証明書を復元するには、次のいずれかの手順を使用します。

- IdM WebUI で統合 IdM CA を使用した証明書の復元
- IdM CLI で統合 IdM CA を使用した証明書の復元

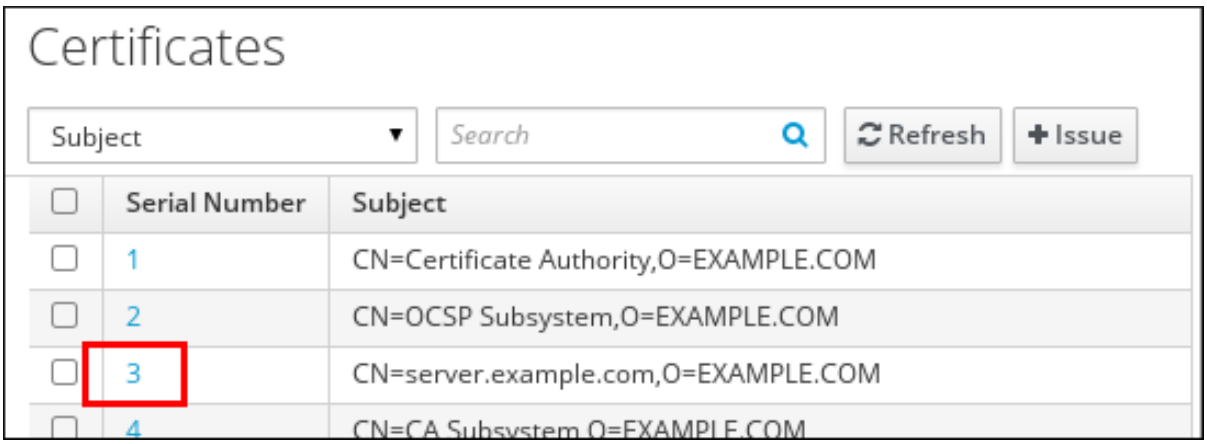
6.6.1. IdM WebUI を使用して統合 IdM CA で証明書の復元

IdM WebUI を使用して、理由 6 (**証明書の保留**) のために取り消された IdM 証明書を復元するには、この手順を完了します。

手順

1. **Authentication** メニューで、**Certificates** > **Certificates** をクリックします。
2. 証明書のシリアル番号をクリックして、証明書情報ページを開きます。

図6.3 証明書のリスト



3. 証明書情報ページで、**Actions → Restore Certificate** をクリックします。

6.6.2. IdM CA を使用して、統合 IdM CA で証明書の失効

IdM CLI を使用して、理由 6 (**証明書の保留**) のために取り消された IdM 証明書を復元するには、この手順を完了します。

手順

- **ipa cert-remove-hold** コマンドを使用して、証明書のシリアル番号を指定します。以下に例を示します。

```
$ ipa cert-remove-hold 1032
```

第7章 スマートカード認証用の IDENTITY MANAGEMENT の設定

Identity Management (IdM) では、以下によるスマートカード認証に対応しています。

- IdM 認証局が発行するユーザー証明書
- 外部認証局が発行するユーザー証明書

両方のタイプの証明書に対して、IdM でスマートカード認証を設定できます。このシナリオでは、**rootca.pem** CA 証明書は、信頼された外部の認証局の証明書を含むファイルです。

IdM でのスマートカード認証については、スマートカード認証について を参照してください。

スマートカード認証の設定に関する詳細は、以下を参照してください。

- スマートカード認証用の IdM サーバーの設定
- スマートカード認証用の IdM クライアントの設定
- IdM Web UI のユーザーエントリーへの証明書の追加
- IdM CLI でユーザーエントリーへの証明書の追加
- スマートカードを管理および使用するツールのインストール
- スマートカードでの証明書の保存
- スマートカードを使用して IdM へのログイン
- スマートカード認証で GDM アクセスの設定
- スマートカード認証で su アクセスの設定

7.1. スマートカード認証用の IDM サーバーの設定

Identity Management (IdM) CA が信頼する <EXAMPLE.ORG> ドメインの認証局 (CA) によって証明書が発行されたユーザーに対してスマートカード認証を有効にする場合は、次の証明書を取得し、IdM サーバーを設定する **ipa-advise** スクリプトを実行するときにそれらを追加できるようにする必要があります。

- <EXAMPLE.ORG> CA の証明書を直接、または 1 つ以上のサブ CA を通じて発行した root CA の証明書。証明書チェーンは、その CA が証明書を発行した Web ページからダウンロードできます。詳細は、証明書認証を有効にするためのブラウザーの設定 の手順 1 - 4a を参照してください。
- IdM CA 証明書。IdM CA インスタンスが実行されている IdM サーバーの **/etc/ipa/ca.crt** ファイルから CA 証明書を取得できます。
- すべての中間 CA、つまり <EXAMPLE.ORG> CA と IdM CA の中間 CA の証明書。

スマートカード認証用に IdM サーバーを設定するには、以下を行います。

1. PEM 形式の CA 証明書を含むファイルを取得します。
2. ビルトイン **ipa-advise** スクリプトを実行します。
3. システム設定をリロードします。

前提条件

- IdM サーバーへの root アクセス権限がある。
- ルート CA 証明書とすべての中間 CA 証明書があります。

手順

1. 設定を行うディレクトリーを作成します。

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

2. そのディレクトリーに移動します。

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

3. PEM 形式のファイルに保存されている関連する CA 証明書を取得します。CA 証明書が DER などの異なる形式のファイルに保存されている場合は、これを PEM 形式に変換します。IdM 認証局の証明書は PEM 形式で、**/etc/ipa/ca.crt** ファイルにあります。
DER ファイルを PEM ファイルに変換します。

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

4. 便宜上、設定を行うディレクトリーに証明書をコピーします。

```
[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. 必要に応じて、外部の認証局の証明書を使用する場合は、**openssl x509** ユーティリティーを使用して **PEM** 形式のファイルの内容を表示し、**Issuer** および **Subject** の値が正しいことを確認します。

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

6. 管理者の権限を使用して、組み込みの **ipa-advise** ユーティリティーで設定スクリプトを生成します。

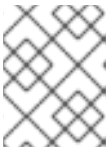
```
[root@server SmartCard]# kinit admin
[root@server SmartCard]# ipa-advise config-server-for-smart-card-auth > config-server-for-smart-card-auth.sh
```

config-server-for-smart-card-auth.sh スクリプトは、以下の操作を実行します。

- IdM Apache HTTP サーバーを設定します。
- キー配布センター (KDC) の Kerberos (PKINIT) で、初回認証用の公開鍵暗号化機能を有効にします。
- スマートカード認可要求を受け入れるように IdM Web UI を設定します。

7. スクリプトを実行し、root CA と サブ CA 証明書が含まれる PEM ファイルを引数として追加します。

```
[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
```



注記

ルート CA 証明書を、サブ CA 証明書の前に引数として追加します。また、CA またはサブ CA 証明書の有効期限が切れていないことを確認します。

8. 必要に応じて、ユーザー証明書を発行した認証局が Online Certificate Status Protocol (OCSP) レスポンダーを提供しない場合は、IdM Web UI への認証に対する OCSP チェックを無効にすることが必要になる場合があります。

   a. **/etc/httpd/conf.d/ssl.conf** ファイルで **SSLOCSPEnable** パラメーターを **off** に設定します。

   ```
   SSLOCSPEnable off
   ```

   b. 変更をすぐに有効にするには、Apache デーモン (httpd) を再起動します。

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

警告

IdM CA が発行したユーザー証明書のみを使用する場合は、OCSP チェックを無効にしないでください。OCSP レスポンダーは IdM に含まれます。

ユーザー証明書を発行した CA が、OCSP サービスリクエストをリッスンする場所に関する情報がユーザー証明書に含まれていない場合に、OCSP チェックを有効にしたまま、ユーザー証明書が IdM サーバーにより拒否されないようにする方法は、Apache mod_ssl 設定オプションの **SSLOCSPDefaultResponder** ディレクティブを参照してください。

これで、スマートカード認証にサーバーが設定されました。

注記

トポロジー全体でスマートカード認証を有効にするには、各 IdM サーバーで手順を実行します。

7.2. ANSIBLE を使用したスマートカード認証用の IDM サーバー設定

Ansible を使用して、Identity Management (IdM) CA が信頼する <EXAMPLE.ORG> ドメインの認証局 (CA) によって証明書が発行されたユーザーのスマートカード認証を有効にできます。そのために

は、**ipasmartcard_server ansible-freeipa** ロールスクリプトを使用して Ansible Playbook を実行するときに使用できるように、次の証明書を取得する必要があります。

- <EXAMPLE.ORG> CA の証明書を直接、または 1 つ以上のサブ CA を通じて発行した root CA の証明書。証明書チェーンは、その CA が証明書を発行した Web ページからダウンロードできます。詳細は、証明書認証を有効にするためのブラウザーの設定 の手順 4 を参照してください。
- IdM CA 証明書。CA 証明書は、任意の IdM CA サーバーの **/etc/ipa/ca.crt** ファイルから取得できます。
- <EXAMPLE.ORG> CA と IdM CA の中間にあるすべての CA の証明書。

前提条件

- IdM サーバーへの **root** アクセス権限がある。
- IdM **admin** のパスワードを把握している。
- ルート CA 証明書、IdM CA 証明書、すべての中間 CA の証明書がある。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. CA 証明書が **DER** などをはじめとする別の形式のファイルに保存されている場合、それらを **PEM** 形式に変換します。

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

   IdM 認証局の証明書は **PEM** 形式で、**/etc/ipa/ca.crt** ファイルにあります。

2. 必要に応じて、**openssl x509** ユーティリティーを使用して **PEM** 形式のファイルの内容を表示し、**Issuer** および **Subject** の値が正しいことを確認します。

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

3. **~/MyPlaybooks/** ディレクトリーに移動します。

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

4. CA 証明書専用のサブディレクトリーを作成します。

```
$ mkdir SmartCard/
```

5. 便宜上、必要なすべての証明書を **~/MyPlaybooks/SmartCard/** ディレクトリーにコピーします。

```
# 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. Ansible インベントリーファイルで、以下を指定します。
   - スマートカード認証用に設定する IdM サーバー。
   - IdM 管理者パスワード。
   - CA の証明書へのパス (次の順序に従う)。
     - ルート CA 証明書ファイル
     - 中間 CA 証明書ファイル
     - IdM CA 証明書ファイル

   ファイルは次のようになります。

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

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

[ipacluster:children]
ipaserver
ipareplicas

[ipacluster:vars]
ipaadmin_password= "{{ ipaadmin_password }}"
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. 次の内容で **install-smartcard-server.yml** playbook を作成します。

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

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

8. ファイルを保存します。

9. Ansible Playbook を実行します。Playbook ファイル、**secret.yml** ファイルを保護するパスワードを格納するファイル、およびインベントリーファイルを指定します。

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

ipasmartcard_server Ansible ロールは、次のアクションを実行します。

- IdM Apache HTTP サーバーを設定します。
- キー配布センター (KDC) の Kerberos (PKINIT) で、初回認証用の公開鍵暗号化機能を有効にします。
- スマートカード認可要求を受け入れるように IdM Web UI を設定します。

10. 必要に応じて、ユーザー証明書を発行した認証局が Online Certificate Status Protocol (OCSP) レスポンダーを提供しない場合は、IdM Web UI への認証に対する OCSP チェックを無効にすることが必要になる場合があります。

a. **root** として IdM サーバーに接続します。

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

b. **/etc/httpd/conf.d/ssl.conf** ファイルで **SSLOCSPEnable** パラメーターを **off** に設定します。

```
SSLOCSPEnable off
```

c. 変更をすぐに有効にするには、Apache デーモン (httpd) を再起動します。

```
# systemctl restart httpd
```

警告

IdM CA が発行したユーザー証明書のみを使用する場合は、OCSP チェックを無効にしないでください。OCSP レスポンダーは IdM に含まれます。

ユーザー証明書を発行した CA が、OCSP サービスリクエストをリッスンする場所に関する情報がユーザー証明書に含まれていない場合に、OCSP チェックを有効にしたまま、ユーザー証明書が IdM サーバーにより拒否されないようにする方法は、Apache mod_ssl 設定オプションの **SSLOCSPDefaultResponder** ディレクティブを参照してください。

これで、インベントリーファイルにリストされているサーバーがスマートカード認証用に設定されました。

注記

トポロジー全体でスマートカード認証を有効にするには、Ansible Playbook の **hosts** 変数を **ipacluster** に設定します。

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

関連情報

- **/usr/share/doc/ansible-freeipa/playbooks/** ディレクトリーで **ipasmartcard_server** ロールを使用するサンプル Playbook

7.3. スマートカード認証用の IDM クライアントの設定

以下の手順に従って、スマートカード認証用に IdM クライアントを設定します。この手順は、認証にスマートカードを使用しているときに接続する各 IdM システム、クライアント、またはサーバーで実行する必要があります。たとえば、ホスト A からホスト B への **ssh** 接続を有効にするには、スクリプトをホスト B で実行する必要があります。

管理者として、以下を使用して、この手順でスマートカード認証を有効にします。

- **ssh** プロトコル
  詳細は、スマートカード認証で SSH アクセスの設定 を参照してください。
- コンソールのログイン
- GNOME Display Manager (GDM)
- **su** コマンド

この手順は、IdM Web UI に対する認証には必要ありません。IdM Web UI の認証には 2 つのホストが関係しますが、どちらも IdM クライアントである必要はありません。

- ブラウザーが実行されているマシン。マシンは IdM ドメインの外にある場合があります。
- **httpd** が実行している IdM サーバー

以下の手順は、IdM サーバーではなく、IdM クライアントでスマートカード認証を設定していることを前提としています。このため、2 台のコンピューターが必要です。設定スクリプトを生成する IdM サーバーと、スクリプトを実行する IdM クライアントが必要になります。

前提条件

- Configuring the IdM server for smart card authentication に従って、IdM サーバーがスマートカード認証用に設定されている。
- IdM サーバーと IdM クライアントに root アクセス権限がある。
- ルート CA 証明書とすべての中間 CA 証明書があります。

- **--mkhomedir** オプションを使用して IdM クライアントをインストールし、リモートユーザーが正常にログインできるようにしている。ホームディレクトリーを作成しない場合、デフォルトのログイン場所はディレクトリー構造のルート / になります。

手順

1. IdM サーバーで、管理者権限を使用して、**ipa-advise** で設定スクリプトを生成します。

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

config-client-for-smart-card-auth.sh スクリプトは、以下の操作を実行します。

- スマートカードデーモンを設定する。
- システム全体のトラストストアを設定します。
- System Security Services Daemon (SSSD) を設定して、ユーザーがユーザー名とパスワード、またはスマートカードで認証できるようにします。スマートカード認証用の SSSD プロファイルオプションの詳細は、RHEL のスマートカード認証オプション を参照してください。

2. IdM サーバーから、IdM クライアントマシンの任意のディレクトリーに、スクリプトをコピーします。

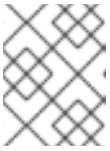
```
[root@server SmartCard]# scp config-client-for-smart-card-auth.sh root@client.idm.example.com:/root/SmartCard/
Password:
config-client-for-smart-card-auth.sh        100%   2419       3.5MB/s   00:00
```

3. 便宜上、IdM サーバーから、上の手順で使用した IdM クライアントマシンのディレクトリーに、PEM 形式の CA 証明書ファイルをコピーします。

```
[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. クライアントマシンで、スクリプトを実行し、CA 証明書を含む PEM ファイルを引数として追加します。

```
[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
```



注記

ルート CA 証明書を、サブ CA 証明書の前に引数として追加します。また、CA またはサブ CA 証明書の有効期限が切れていないことを確認します。

これで、クライアントがスマートカード認証に対して設定されました。

7.4. ANSIBLE を使用したスマートカード認証用の IDM クライアント設定

ansible-freeipa ipasmartcard_client モジュールを使用して特定の Identity Management (IdM) クライアントを設定し、IdM ユーザーがスマートカードで認証できるようにするには、次の手順に従います。この手順を実行し、以下のいずれかを使用して IdM にアクセスする IdM ユーザーのスマートカード認証を有効にします。

- **ssh** プロトコル
  詳細は、スマートカード認証で SSH アクセスの設定 を参照してください。
- コンソールのログイン
- GNOME Display Manager (GDM)
- **su** コマンド

注記

この手順は、IdM Web UI に対する認証には必要ありません。IdM Web UI の認証には 2 つのホストが関係しますが、どちらも IdM クライアントである必要はありません。

- ブラウザーが実行されているマシン。マシンは IdM ドメインの外にある場合があります。
- **httpd** が実行している IdM サーバー

前提条件

- Ansible を使用したスマートカード認証用の IdM サーバー設定 に説明されているとおり、IdM サーバーがスマートカード認証用に設定されている。
- IdM サーバーと IdM クライアントに root アクセス権限がある。
- ルート CA 証明書、IdM CA 証明書、すべての中間 CA の証明書がある。
- 次の要件を満たすように Ansible コントロールノードを設定している。
  - Ansible バージョン 2.14 以降を使用している。
  - Ansible コントローラーに **ansible-freeipa** パッケージがインストールされている。
  - この例では、**~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成したことを前提としている。
  - この例では、**secret.yml** Ansible vault に **ipaadmin_password** が保存されていることを前提としています。
- ターゲットノード (**ansible-freeipa** モジュールが実行されるノード) が、IdM クライアント、サーバー、またはレプリカとして IdM ドメインに含まれている。

手順

1. CA 証明書が **DER** などをはじめとする別の形式のファイルに保存されている場合、それらを **PEM** 形式に変換します。

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

IdM CA 証明書は **PEM** 形式で、**/etc/ipa/ca.crt** ファイルにあります。

2. 必要に応じて、**openssl x509** ユーティリティーを使用して **PEM** 形式のファイルの内容を表示し、**Issuer** および **Subject** の値が正しいことを確認します。

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

3. Ansible コントロールノードで、**~/MyPlaybooks/** ディレクトリーに移動します。

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

4. CA 証明書専用のサブディレクトリーを作成します。

```
$ mkdir SmartCard/
```

5. 便宜上、必要なすべての証明書を **~/MyPlaybooks/SmartCard/** ディレクトリーにコピーします。以下はその例です。

```
# 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. Ansible インベントリーファイルで、以下を指定します。

- スマートカード認証用に設定する IdM クライアント。
- IdM 管理者パスワード。
- CA の証明書へのパス (次の順序に従う)。
  - ルート CA 証明書ファイル
  - 中間 CA 証明書ファイル
  - IdM CA 証明書ファイル

ファイルは次のようになります。

```
[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
```

7. 次の内容で **install-smartcard-clients.yml** playbook を作成します。

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

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

8. ファイルを保存します。

9. Ansible Playbook を実行します。Playbook とインベントリーファイルを指定します。

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

ipasmartcard_client Ansible ロールは、次のアクションを実行します。

- スマートカードデーモンを設定する。
- システム全体のトラストストアを設定します。
- System Security Services Daemon (SSSD) を設定して、ユーザーがユーザー名とパスワード、またはスマートカードで認証できるようにします。スマートカード認証用の SSSD プロファイルオプションの詳細は、RHEL のスマートカード認証オプション を参照してください。

これで、インベントリーファイルの **ipaclients** セクションにリストされているクライアントがスマートカード認証用に設定されました。

注記

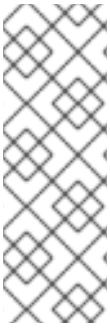
--mkhomedir オプションを使用して IdM クライアントをインストールしている場合、リモートユーザーはホームディレクトリーにログインできます。それ以外の場合、デフォルトのログイン場所はディレクトリー構造のルート / です。

関連情報

- **/usr/share/doc/ansible-freeipa/playbooks/** ディレクトリーで **ipasmartcard_server** ロールを使用するサンプル Playbook

7.5. IDM WEB UI のユーザーエントリーへの証明書の追加

IdM Web UI のユーザーエントリーに外部証明書を追加するには、以下の手順に従います。



注記

証明書全体をアップロードする代わりに、IdM のユーザーエントリーに証明書マッピングデータをアップロードすることもできます。システム管理者向けのスマートカード認証の設定を容易にするために、完全な証明書または証明書マッピングデータのいずれかが含まれるユーザーエントリーを、対応する証明書マッピングルールと併用できます。詳細は以下を参照してください。

認証を設定するための証明書マッピングルール

注記

ユーザーの証明書が IdM 認証局によって発行された場合、証明書はユーザーエントリーにすでに保存されているため、この手順に従う必要はありません。

前提条件

- ユーザーエントリーに追加できる証明書がある。

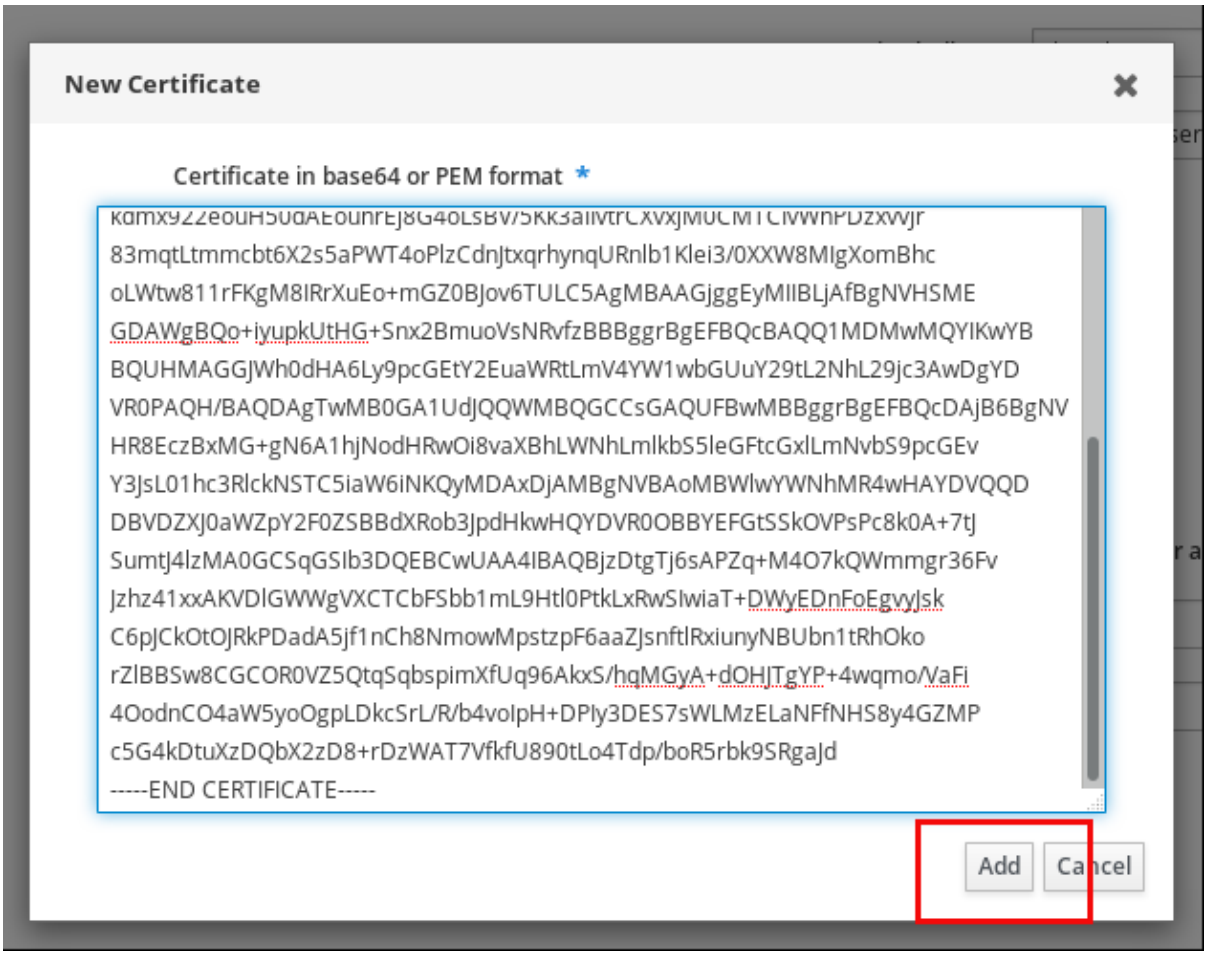
手順

1. 別のユーザーに証明書を追加する場合は、管理者として IdM Web UI にログインします。独自のプロファイルに証明書を追加する場合は、管理者の認証情報が必要ありません。
2. **Users → Active users → sc_user** の順に移動します。
3. **Certificate** オプションを探して、**Add** をクリックします。
4. コマンドラインインターフェイスで、**cat** ユーティリティーまたはテキストエディターを使用して、**PEM** 形式の証明書を表示します。

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

5. CLI で、証明書をコピーし、Web UI で開いたウィンドウにこれを貼り付けます。
6. **Add** をクリックします。

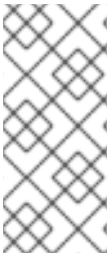
図7.1 IdM Web UI で新しい証明書の追加



sc_user エントリーに外部証明書が含まれるようになりました。

7.6. IDM CLI でユーザーエントリーへの証明書の追加

IdM CLI のユーザーエントリーに外部証明書を追加するには、以下の手順に従います。



注記

証明書全体をアップロードする代わりに、IdM のユーザーエントリーに証明書マッピングデータをアップロードすることもできます。システム管理者向けのスマートカード認証の設定を容易にするために、完全な証明書または証明書マッピングデータのいずれかが含まれるユーザーエントリーを、対応する証明書マッピングルールと併用できます。詳細は、認証を設定するための証明書マッピングルール を参照してください。

注記

ユーザーの証明書が IdM 認証局によって発行された場合、証明書はユーザーエントリーにすでに保存されているため、この手順に従う必要はありません。

前提条件

- ユーザーエントリーに追加できる証明書がある。

手順

1. 別のユーザーに証明書を追加する場合は、管理者として IdM Web CLI にログインします。

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

独自のプロファイルに証明書を追加する場合は、管理者の認証情報は必要ありません。

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

2. ヘッダーとフッターのある証明書を含む環境変数を作成し、1 行に連結します。これは、**ipa user-add-cert** コマンドで必要な形式です。

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

testuser.crt ファイルの証明書は、**PEM** 形式である必要があることに注意してください。

3. **ipa user-add-cert** コマンドを使用して、sc_user のプロファイルに証明書を追加します。

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

sc_user エントリーに外部証明書が含まれるようになりました。

7.7. スマートカードを管理および使用するツールのインストール

前提条件

- **gnutls-utils** パッケージがインストールされている。
- **opensc** パッケージがインストールされている。
- **pcscd** サービスを実行している。

スマートカードを設定する前に、対応するツール (証明書を生成して **pscd** サービスを起動できるもの) をインストールする必要があります。

手順

1. **opensc** パッケージおよび **gnutls-utils** パッケージをインストールします。

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

2. **pcscd** サービスを開始します。

```
# systemctl start pcscd
```

検証手順

- **pcscd** サービスが稼働していることを確認します。

```
# systemctl status pcscd
```

7.8. スマートカードを準備し、証明書と鍵をスマートカードにアップロードする

pkcs15-init ツールを使用してスマートカードを設定するには、この手順に従います。このツールは、以下を設定するのに役立ちます。

- スマートカードの消去
- 新しい PIN および任意の PIN ブロック解除キー (PUK) の設定
- スマートカードでの新規スロットの作成
- スロットへの証明書、秘密鍵、および公開鍵の保存
- 必要に応じて、特定のスマートカードではこのタイプのファイナライズが必要なため、スマートカードの設定をロックします。

注記

pkcs15-init ツールは、すべてのスマートカードで機能するとは限りません。使用しているスマートカードで動作するツールを使用する必要があります。

前提条件

- **pkcs15-init** ツールを含む **opensc** パッケージがインストールされている。
  詳細は スマートカードを管理および使用するツールのインストール を参照してください。
- カードがリーダーに挿入され、コンピューターに接続されている。
- スマートカードに保存する秘密鍵、公開鍵、および証明書がある。この手順の **testuser.key**、**testuserpublic.key**、および **testuser.crt** は、秘密鍵、公開鍵、および証明書に使用される名前です。
- 現在のスマートカードユーザー PIN およびセキュリティーオフィス PIN (SO-PIN)

手順

1. スマートカードを消去して PIN で自身を認証します。

```
$ 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]:
```

   カードが削除されました。

2. スマートカードを初期化し、ユーザーの PIN と PUK を設定します。また、セキュリティー担当者の PIN と PUK を設定します。

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

   pcks15-init ツールは、スマートカードに新しいスロットを作成します。

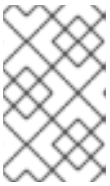
3. スロットのラベルと認証 ID を設定します。

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

ラベルは人間が判読できる値に設定されます (この場合は **testuser**)。**auth-id** は 16 進数の値である必要があります。この場合、**01** に設定されます。

4. スマートカードの新しいスロットに秘密鍵を保存し、ラベルを付けます。

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



注記

--id に指定する値は、秘密鍵を保存するときと、次の手順で証明書を保存するときに同じである必要があります。**--id** に独自の値を指定することを推奨します。そうしないと、より複雑な値がツールによって計算されます。

5. スマートカードの新しいスロットに証明書を保存し、ラベル付けします。

```
$ 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. オプション: スマートカードの新しいスロットに公開鍵を保存し、ラベルを付けます。

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

注記

公開鍵が秘密鍵または証明書に対応する場合は、秘密鍵または証明書の ID と同じ ID を指定します。

7. オプション: スマートカードの中には、設定をロックしてカードをファイナライズする必要があるものもあります。

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

この段階では、スマートカードには、新たに作成されたスロットに証明書、秘密鍵、および公開鍵が含まれます。ユーザーの PIN と PUK、およびセキュリティー担当者の PIN と PUK も作成しました。

7.9. スマートカードを使用して IDM へのログイン

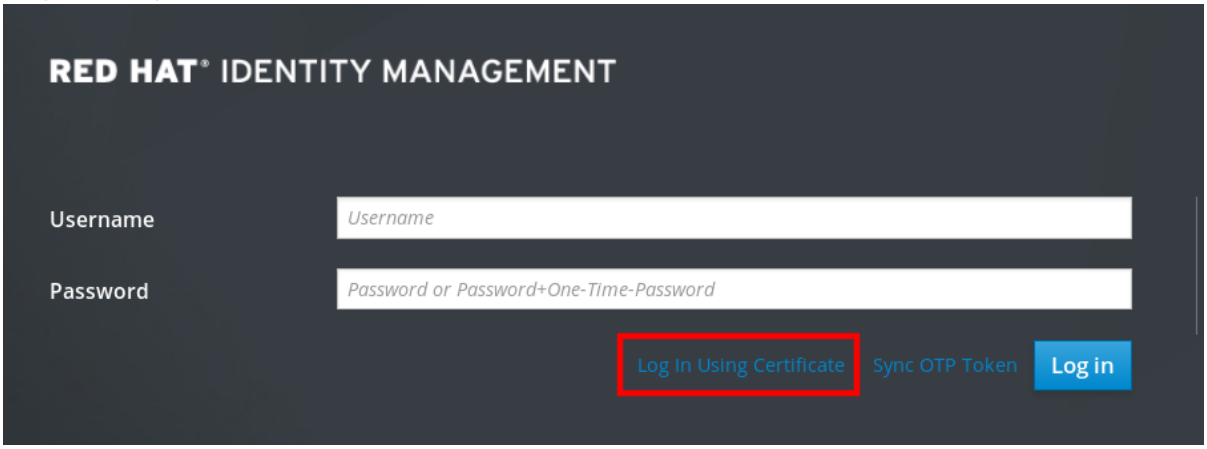
IdM Web UI へのログインにスマートカードを使用するには、以下の手順に従います。

前提条件

- Web ブラウザーが、スマートカード認証を使用できるように設定されている。
- IdM サーバーはスマートカード認証用に設定されています。
- スマートカードにインストールされた証明書は、IdM サーバーによって発行されるか、IdM のユーザーエントリーに追加されています。
- スマートカードのロックを解除するために必要な PIN を知っています。
- スマートカードがリーダーに挿入されました。

手順

1. ブラウザーで IdM Web UI を開きます。
2. **Log In Using Certificate** をクリックします。

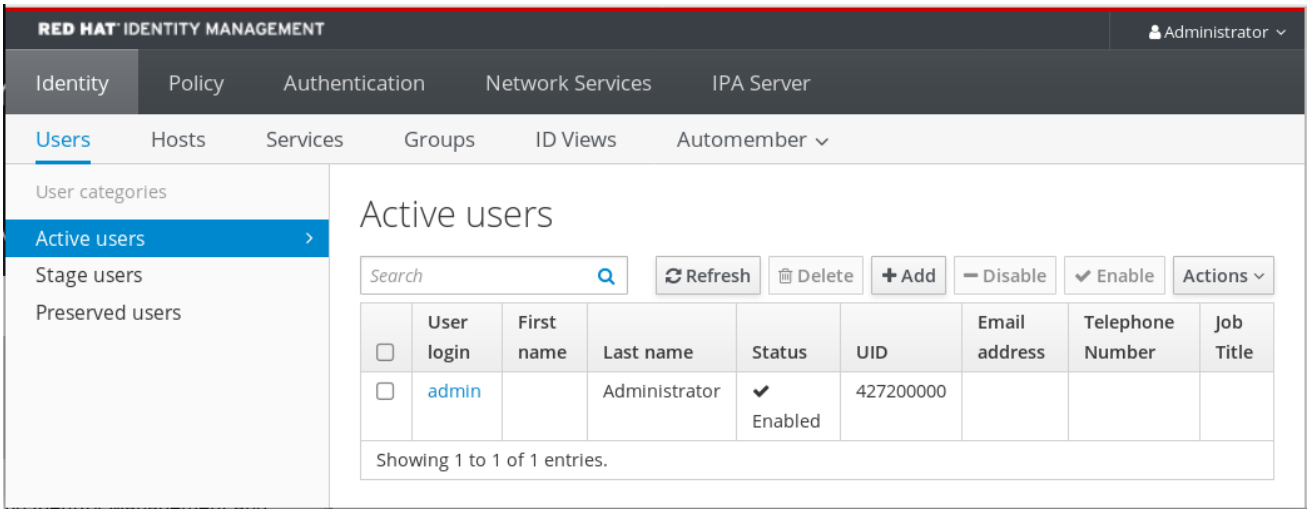


3. **Password Required** ダイアログボックスが開いたら、スマートカードのロックを解除する PIN を追加して、**OK** ボタンをクリックします。
User Identification Request ダイアログボックスが開きます。

   スマートカードに複数の証明書が含まれている場合は、**Choose a certificate to present as identification** の下にあるドロップダウンリストで、認証に使用する証明書を選択します。

4. **OK** ボタンをクリックします。

これで、IdM Web UI に正常にログインできるようになりました。



7.10. IDM クライアントでスマートカード認証を使用して GDM にログインする

GNOME Desktop Manager (GDM) には認証が必要です。パスワードは使用できますが、認証にスマートカードを使用することもできます。

以下の手順に従って、スマートカード認証を使用して GDM にアクセスします。

前提条件

- システムはスマートカード認証用に設定されています。詳細は、スマートカード認証用の IdM クライアントの設定 を参照してください。
- スマートカードに、証明書と秘密鍵が含まれている。
- ユーザーアカウントは、IdM ドメインのメンバーです。
- スマートカードの証明書は、以下を使用してユーザーエントリーにマッピングします。
  - 特定のユーザーエントリーへの証明書の割り当て。詳細は Adding a certificate to a user entry in the IdM Web UI または Adding a certificate to a user entry in the IdM CLI を参照してください。
  - アカウントに適用される証明書マッピングデータ。詳細は、スマートカードにおける認証を設定するための証明書マッピングルール を参照してください。

手順

1. スマートカードをリーダーに挿入します。
2. スマートカード PIN を入力します。
3. **Sign In** をクリックします。

RHEL システムにログインし、IdM サーバーが提供する TGT がある。

検証手順

- **端末** ウィンドウで **klist** を入力し、結果を確認します。

```
$ 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
```

7.11. SU コマンドでのスマートカード認証の使用

別のユーザーへの変更には認証が必要です。パスワードまたは証明書を使用できます。以下の手順に従って、**su** コマンドでスマートカードを使用します。これは、**su** コマンドを入力すると、スマートカード PIN の入力が求められます。

前提条件

- IdM サーバーとクライアントがスマートカード認証用に設定されました。
  - スマートカード認証用の IdM サーバーの設定 を参照してください。
  - スマートカード認証用の IdM クライアントの設定 を参照してください。
- スマートカードに、証明書と秘密鍵が含まれている。スマートカードでの証明書の保存 を参照してください。
- カードがリーダーに挿入され、コンピューターに接続されている。

手順

- 端末で、**su** コマンドで別のユーザーに移動します。

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

  設定が正しい場合は、スマートカードの PIN を入力するよう求められます。

第8章 IDM でスマートカード認証用に ADCS が発行した証明書の設定

Active Directory (AD)証明書サービスが証明書を発行するユーザーに IdM でスマートカード認証を設定するには、次のコマンドを実行します。

- デプロイメントは、Identity Management (IdM) と Active Directory (AD) と間のフォレスト間の信頼に基づいている場合
- AD にアカウントが保存されているユーザーに対してスマートカード認証を許可したい場合
- 証明書が作成され、Active Directory Certificate Services (ADCS) に保存される場合

スマートカード認証の概要については、スマートカード認証について を参照してください。

設定は次の手順で実行できます。

- Active Directory から IdM サーバーおよびクライアントへの CA 証明書およびユーザー証明書のコピー
- ADCS 証明書を使用したスマートカード認証用の IdM サーバーおよびクライアントの設定
- 証明書および秘密鍵をスマートカードに保存できるように PFX (PKCS#12) ファイルの変換
- sssd.conf ファイルでのタイムアウトの設定
- スマートカード認証用の証明書マッピングルールの作成

前提条件

- Identity Management (IdM) および Active Directory (AD) 信頼がインストールされている。詳細は、IdM と AD との間の信頼のインストール を参照してください。
- Active Directory 証明書サービス (ADCS) がインストールされ、ユーザーの証明書が生成されている。

8.1. 信頼の設定と証明書の使用に必要な WINDOWS SERVER 設定

Windows Server で以下を設定する必要があります。

- Active Directory 証明書サービス (ADCS) がインストールされる
- 認証局が作成される
- 必要に応じて、認証機関の Web 登録を使用している場合は、IIS (Internet Information Services) を設定する必要がある。

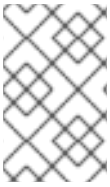
証明書をエクスポートします。

- 鍵には **2048** ビット以上が必要
- 秘密鍵を含める
- Personal Information Exchange (**PKCS #12(.PFX)**) の形式の証明書が必要
    - 証明書のプライバシーを有効にする

8.2. SFTP を使用して ACTIVE DIRECTORY から証明書のコピー

スマートカード認証を使用できるようにするには、次の証明書ファイルをコピーする必要があります。

- IdM サーバーにある **CER** 形式のルート CA 証明書 (**adcs-winserver-ca.cer**)
- IdM クライアントの **PFX** 形式の秘密鍵を持つユーザー証明書 (**aduser1.pfx**)



注記

この手順では、SSH アクセスが許可されていることを想定しています。SSH が使用できない場合、ユーザーは AD サーバーから IdM サーバーおよびクライアントにファイルをコピーする必要があります。

手順

1. **IdM サーバー** から接続し、**adcs-winserver-ca.cer** ルート証明書を IdM サーバーにコピーします。

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

2. **IdM クライアント** から接続し、**aduser1.pfx** ユーザー証明書をクライアントにコピーします。

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

これで、CA 証明書は IdM サーバーに保存され、ユーザー証明書はクライアントマシンに保存されます。

8.3. ADCS 証明書を使用したスマートカード認証用の IDM サーバーおよびクライアントの設定

IdM 環境でスマートカード認証を使用できるように、IdM (Identity Management) サーバーおよびクライアントを設定する必要があります。IdM には、以下に示す必要な変更をすべて行う **ipa-advise** スクリプトが含まれています。

- 必要なパッケージをインストールする

- IdM サーバーおよびクライアントを設定する
- CA 証明書を所定の場所にコピーする

IdM サーバーで **ipa-advise** を実行できるようになります。

以下の手順に従って、スマートカード認証用にサーバーとクライアントを設定します。

- IdM サーバー - **ipa-advise** スクリプトを準備して、スマートカード認証用に IdM サーバーを設定します。
- IdM サーバー - **ipa-advise** スクリプトを準備して、スマートカード認証用に IdM クライアントを設定します。
- IdM サーバー - AD 証明書を使用して IdM サーバーに **ipa-advise** サーバースクリプトを適用します。
- クライアントスクリプトを IdM クライアントマシンに移動します。
- IdM サーバー - AD 証明書を使用して IdM クライアントに **ipa-advise** クライアントスクリプトを適用します。

前提条件

- 証明書が IdM サーバーにコピーされている。
- Kerberos チケットを取得している。
- 管理者権限を持つユーザーとしてログインしている。

手順

1. IdM サーバーで、クライアントを設定する **ipa-advise** スクリプトを使用します。

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

2. IdM サーバーで、サーバーを設定する **ipa-advise** スクリプトを使用します。

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

3. IdM サーバーで、スクリプトを実行します。

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

   - IdM Apache HTTP サーバーを設定します。
   - キー配布センター (KDC) の Kerberos (PKINIT) で、初回認証用の公開鍵暗号化機能を有効にします。
   - スマートカード認可要求を受け入れるように IdM Web UI を設定します。

4. **sc_client.sh** スクリプトをクライアントシステムにコピーします。

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

5. Windows 証明書をクライアントシステムにコピーします。

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

6. クライアントシステムで、クライアントスクリプトを実行します。

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

CA 証明書が IdM サーバーとクライアントシステムに正しい形式でインストールされました。次の手順は、ユーザー証明書をスマートカード自体にコピーすることです。

8.4. PFX ファイルの変換

PFX (PKCS#12) ファイルをスマートカードに保存する前に、以下を行う必要があります。

- ファイルを PEM 形式に変換する
- 秘密鍵と証明書を 2 つの異なるファイルに抽出する

前提条件

- PFX ファイルが IdM クライアントマシンにコピーされます。

手順

1. IdM クライアントで、PEM 形式に変換します。

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

2. 鍵を別のファイルにデプロイメントします。

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

3. パブリック証明書を別のファイルにデプロイメントします。

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

この時点で、**aduser1.key** および **aduser1.crt** をスマートカードに保存できます。

8.5. スマートカードを管理および使用するツールのインストール

前提条件

- **gnutls-utils** パッケージがインストールされている。
- **opensc** パッケージがインストールされている。
- **pcscd** サービスを実行している。

スマートカードを設定する前に、対応するツール (証明書を生成して **pscd** サービスを起動できるもの) をインストールする必要があります。

手順

1. **opensc** パッケージおよび **gnutls-utils** パッケージをインストールします。

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

2. **pcscd** サービスを開始します。

```
# systemctl start pcscd
```

検証手順

- **pcscd** サービスが稼働していることを確認します。

```
# systemctl status pcscd
```

8.6. スマートカードを準備し、証明書と鍵をスマートカードにアップロードする

pkcs15-init ツールを使用してスマートカードを設定するには、この手順に従います。このツールは、以下を設定するのに役立ちます。

- スマートカードの消去
- 新しい PIN および任意の PIN ブロック解除キー (PUK) の設定
- スマートカードでの新規スロットの作成
- スロットへの証明書、秘密鍵、および公開鍵の保存
- 必要に応じて、特定のスマートカードではこのタイプのファイナライズが必要なため、スマートカードの設定をロックします。

注記

pkcs15-init ツールは、すべてのスマートカードで機能するとは限りません。使用しているスマートカードで動作するツールを使用する必要があります。

前提条件

- **pkcs15-init** ツールを含む **opensc** パッケージがインストールされている。
  詳細は スマートカードを管理および使用するツールのインストール を参照してください。

- カードがリーダーに挿入され、コンピューターに接続されている。
- スマートカードに保存する秘密鍵、公開鍵、および証明書がある。この手順の **testuser.key**、**testuserpublic.key**、および **testuser.crt** は、秘密鍵、公開鍵、および証明書に使用される名前です。
- 現在のスマートカードユーザー PIN およびセキュリティーオフィス PIN (SO-PIN)

手順

1. スマートカードを消去して PIN で自身を認証します。

```
$ 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]:
```

   カードが削除されました。

2. スマートカードを初期化し、ユーザーの PIN と PUK を設定します。また、セキュリティー担当者の PIN と PUK を設定します。

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

   pcks15-init ツールは、スマートカードに新しいスロットを作成します。

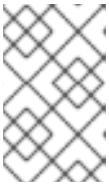
3. スロットのラベルと認証 ID を設定します。

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

   ラベルは人間が判読できる値に設定されます (この場合は **testuser**)。**auth-id** は 16 進数の値である必要があります。この場合、**01** に設定されます。

4. スマートカードの新しいスロットに秘密鍵を保存し、ラベルを付けます。

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



注記

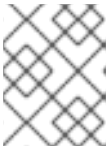
--id に指定する値は、秘密鍵を保存するときと、次の手順で証明書を保存するときに同じである必要があります。**--id** に独自の値を指定することを推奨します。そうしないと、より複雑な値がツールによって計算されます。

5. スマートカードの新しいスロットに証明書を保存し、ラベル付けします。

```
$ 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. オプション: スマートカードの新しいスロットに公開鍵を保存し、ラベルを付けます。

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



注記

公開鍵が秘密鍵または証明書に対応する場合は、秘密鍵または証明書の ID と同じ ID を指定します。

7. オプション: スマートカードの中には、設定をロックしてカードをファイナライズする必要があるものもあります。

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

この段階では、スマートカードには、新たに作成されたスロットに証明書、秘密鍵、および公開鍵が含まれます。ユーザーの PIN と PUK、およびセキュリティー担当者の PIN と PUK も作成しました。

8.7. SSSD.CONF でタイムアウトの設定

スマートカード証明書による認証は、SSSD で使用されるデフォルトのタイムアウトよりも時間がかかる場合があります。タイムアウトの期限切れは次の原因で発生する可能性があります。

- 読み込みが遅い
- 物理デバイスから仮想環境への転送
- スマートカードに保存されている証明書が多すぎる
- OCSP レスポンダーからの応答が遅い (証明書の検証に OCSP (Online Certificate Status Protocol) を使用している場合)

この場合は、**sssd.conf** ファイルにある次のタイムアウトを、たとえば 60 秒まで延長できます。

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

前提条件

- root としてログインしている。

手順

1. **sssd.conf** ファイルを開きます。

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

2. **p11_child_timeout** の値を変更します。

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

3. **krb5_auth_timeout** の値を変更します。

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

4. 設定を保存します。

現在、スマートカードとの相互作用は 1 分間 (60 秒) 実行でき、その後、認証はタイムアウトで失敗します。

8.8. スマートカード認証用の証明書マッピングルールの作成

AD (Active Directory) および IdM (Identity Management) にアカウントを持つユーザーに対して証明書を 1 つ使用する場合は、IdM サーバーで証明書マッピングルールを作成できます。

このようなルールを作成すると、ユーザーは両方のドメインのスマートカードで認証できます。

証明書マッピングルールの詳細は、認証を設定するための証明書マッピングルール を参照してください。

第9章 IDENTITY MANAGEMENT での証明書マッピングルールの設定

証明書マッピングルールは、Identity Management (IdM) 管理者が特定のユーザーの証明書にアクセスしない場合に、シナリオで証明書を使用して認証できるため便利な方法です。これは通常、証明書が外部の認証局によって発行されたことが原因です。

9.1. 認証を設定するための証明書マッピングルール

次のシナリオでは、証明書マッピングルールの設定が必要になる場合があります。

- 証明書は、IdM ドメインが信頼関係にある Active Directory (AD) の証明書システムによって発行されています。
- 証明書は外部の認証局によって発行されています。
- IdM 環境は大規模で、多くのユーザーがスマートカードを使用しています。この場合、完全な証明書の追加は複雑になる可能性があります。件名と発行者はほとんどのシナリオで予測可能なため、完全な証明書よりも事前に追加する方が簡単です。

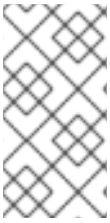
システム管理者は、証明書マッピングルールを作成し、特定のユーザーに証明書を発行する前に、ユーザーエントリーに証明書マッピングデータを追加できます。証明書を発行すると、完全な証明書がまだユーザーエントリーにアップロードされていなくても、ユーザーは証明書を使用してログインできます。

さらに、証明書は定期的に更新されるため、証明書マッピングルールにより管理オーバーヘッドが削減されます。ユーザーの証明書が更新される場合、管理者はユーザーエントリーを更新する必要はありません。たとえば、マッピングが **Subject** と **Issuer** の値に基づいている場合、および新しい証明書の Subject と Issuer が以前と同じ場合は、マッピングは引き続き適用されます。一方で、完全な証明書を使用した場合、管理者は古い証明書に置き換わる新しい証明書をユーザーエントリーにアップロードする必要があります。

証明書マッピングを設定するには、以下を実行します。

1. 管理者は、証明書マッピングデータまたは完全な証明書をユーザーアカウントにロードする必要があります。
2. 管理者は、証明書の情報と一致する証明書マッピングデータエントリーがアカウントに含まれているユーザーが IdM に正常にログインできるように、証明書マッピングルールを作成する必要があります。

証明書マッピングルールが作成されると、エンドユーザーが ファイルシステム または スマートカード に保存されている証明書を提示すると、認証は成功します。



注記

キー配布センター (KDC) には、証明書マッピングルールのキャッシュがあります。キャッシュは最初の **certauth** 要求の際に入力され、タイムアウトは 300 秒にハードコーディングされています。KDC は、再起動するかキャッシュが期限切れにならない限り、証明書マッピングルールへの変更を認識しません。

マッピングルールを設定する個々のコンポーネントの詳細と、そのコンポーネントの取得方法および使用方法は、IdM における ID マッピングルールのコンポーネント および マッチングルールで使用する証明書からの発行者の取得 を参照してください。

注記

証明書マッピングルールは、証明書を使用するユースケースに応じて異なります。たとえば、証明書を使用して SSH を使用している場合、証明書から公開鍵を抽出するには完全な証明書が必要です。

9.2. IDM における ID マッピングルールのコンポーネント

IdM で **ID マッピングルール** を作成する際に、異なるコンポーネントを設定します。各コンポーネントには、オーバーライドできるデフォルト値があります。コンポーネントは、Web UI または CLI のいずれかで定義できます。CLI では、**ipa certmaprule-add** コマンドを使用して、ID マッピングルールが作成されます。

マッピングルール

マッピングルールコンポーネントでは、証明書を 1 人または複数のユーザーアカウントに関連付けます (または **マップ** します)。ルールは、証明書を目的のユーザーアカウントに関連付ける LDAP 検索フィルターを定義します。
さまざまな認証局 (CA) が発行する証明書にはさまざまなプロパティーがあり、さまざまなドメインで使用される可能性があります。そのため、IdM はマッピングルールを無条件に適用せず、適切な証明書にのみ適用されます。適切な証明書は、**マッチングルール** を使用して定義されます。

マッピングルールのオプションを空のままにすると、証明書は、DER でエンコードされたバイナリーファイルとして、**userCertificate** 属性で検索されることに注意してください。

--maprule オプションを使用して、CLI でマッピングルールを定義します。

マッチングルール

マッチングルールコンポーネントは、マッピングルールを適用する証明書を選択します。デフォルトのマッチングルールは、**digitalSignature 鍵** の使用と、**clientAuth 拡張鍵** の使用で、証明書と一致します。
--matchrule オプションを使用して、CLI にマッチングルールを定義します。

ドメインリスト

ドメインリストは、ID マッピングルールの処理時に IdM がユーザーを検索する ID ドメインを指定します。このオプションを指定しないと、IdM は、IdM クライアントが所属しているローカルドメイン内でのみユーザーを検索します。
--domain オプションを使用して CLI にドメインを定義します。

優先度

複数のルールが証明書に適用される場合は、最も優先度が高いルールが優先されます。その他のルールはすべて無視されます。

- 数値が低いほど、ID マッピングルールの優先度が高くなります。たとえば、優先度 1 のルールは、優先度 2 のルールよりも高く設定されています。
- ルールに優先度の値が定義されていないと、優先度が最も低くなります。

--priority オプションを使用して、CLI にマッピングルールの優先度を定義します。

証明書マッピングルールの例 1

CLI を使用して、証明書の **Subject** が IdM のユーザーアカウントの **certmapdata** エントリーと一致する場合に、**EXAMPLE.ORG** 組織の **Smart Card CA** によって発行された証明書の認証を許可する **simple_rule** という証明書マッピングルールを定義するには、以下を実行します。

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

9.3. マッチングルールで使用する証明書からのデータの取得

この手順では、証明書からデータを取得して、そのデータをコピーして証明書マッピングルールの一致ルールに貼り付ける方法について説明します。一致ルールに必要なデータを取得するには、**sssctl cert-show** または **sssctl cert-eval-rule** コマンドを使用します。

前提条件

- PEM 形式のユーザー証明書があります。

手順

1. 必要なデータを取得できるように、正しくエンコードされていることを確認する証明書を指す変数を作成します。

```
# CERT=$(openssl x509 -in /path/to/certificate -outform der|base64 -w0)
```

2. **sssctl cert-eval-rule** を使用して、一致するデータを確認します。次の例では、証明書のシリアル番号が使用されています。

```
# sssctl cert-eval-rule $CERT --match='<ISSUER>CN=adcs19-WIN1-
CA,DC=AD,DC=EXAMPLE,DC=COM' --map='LDAPU1:(altSecurityIdentities=X509:<I>
{issuer_dn!ad_x500}<SR>{serial_number!hex_ur})'
Certificate matches rule.
Mapping filter:

    (altSecurityIdentities=X509:<I>DC=com,DC=example,DC=ad,CN=adcs19-WIN1-
CA<SR>0F0000000000DB8852DD7B246C9C0F0000003B)
```

この場合、**altSecurityIdentities=** 以降のすべてをユーザーの AD の **altSecurityIdentities** 属性に追加します。SKI マッピングを使用する場合は、**--map='LDAPU1:(altSecurityIdentities=X509:<SKI>{subject_key_id!hex_u})'** を使用します。

3. オプションで、証明書の発行者が **ad.example.com** ドメインの **adcs19-WIN1-CA** と一致し、証明書のシリアル番号がユーザーアカウントの **altSecurityIdentities** エントリーと一致する必要があることを指定する一致ルールに基づいて、CLI で新しいマッピングルールを作成するには、以下を実行します。

```
# ipa certmaprule-add simple_rule --matchrule '<ISSUER>CN=adcs19-WIN1-
CA,DC=AD,DC=EXAMPLE,DC=COM' --maprule 'LDAPU1:(altSecurityIdentities=X509:<I>
{issuer_dn!ad_x500}<SR>{serial_number!hex_ur})'
```

9.4. IDM に保存されたユーザーの証明書マッピングの設定

ユーザーに設定されている証明書認証が IdM に保存されている場合に IdM で証明書マッピングを有効にするには、システム管理者は次のタスクを完了する必要があります。

- 証明書マッピングルールを設定して、マッピングルールおよび証明書マッピングデータエントリーで指定された条件に一致する証明書を持つ IdM ユーザーが IdM に対して認証できるようにします。
- IdM ユーザーエントリーに証明書マッピングデータを入力すると、証明書マッピングデータエントリーで指定された値がすべて含まれている場合に、ユーザーが複数の証明書を使用して認証できるようになります。

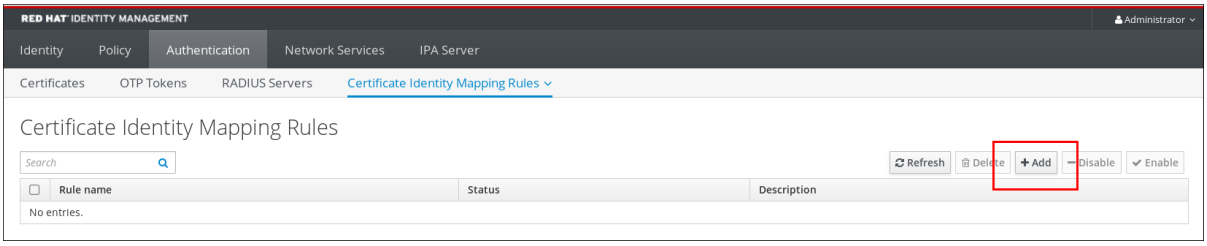
前提条件

- IdM にユーザーがアカウントがある。
- 管理者が、ユーザーエントリーに追加する証明書全体または証明書マッピングデータのいずれかを所有している。

9.4.1. IdM Web UI で証明書マッピングルールの追加

1. 管理者として IdM Web UI にログインします。
2. **Authentication → Certificate Identity Mapping Rules → Certificate Identity Mapping Rules** の順に移動します。
3. **Add** をクリックします。

図9.1 IdM Web UI で新しい証明書マッピングルールの追加



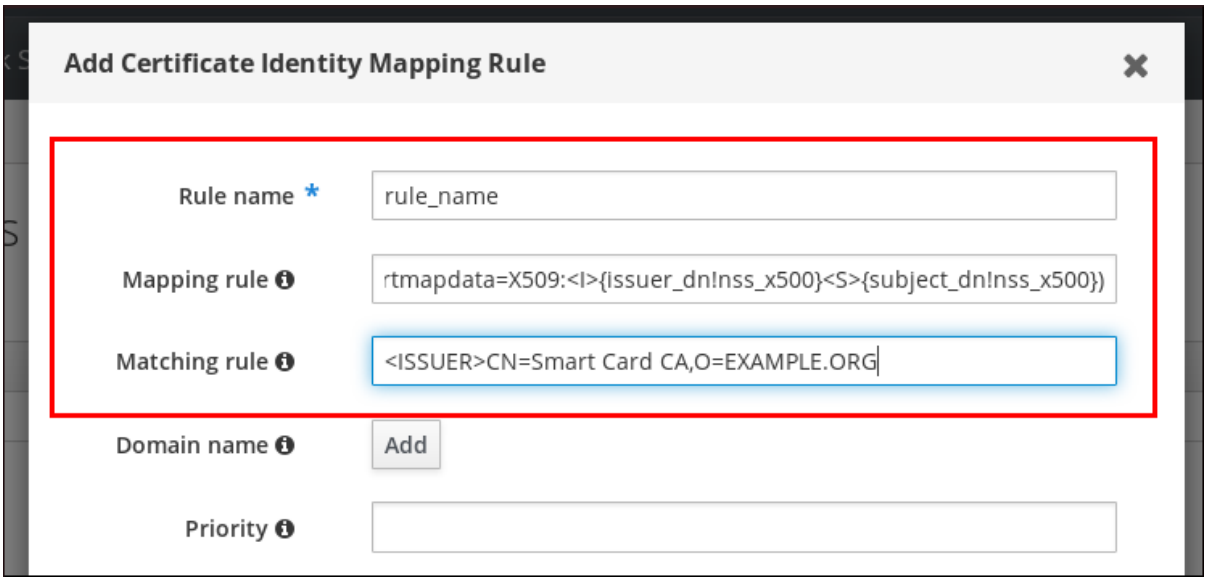
4. ルール名を入力します。
5. マッピングルールを入力します。たとえば、IdM に提示された証明書の **Issuer** および **Subject** エントリーを Idm で検索し、提示された証明書に含まれるこの 2 つのエントリーで見つかった情報に基づいて認証するかどうかを決定するには、次のコマンドを実行します。

```
(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})
```

6. マッチングルールを入力します。たとえば、**EXAMPLE.ORG** 組織の **スマートカード CA** が発行する証明書のみが IdM に対して認証できるようにするには、次のコマンドを実行します。

```
<ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG
```

図9.2 IdM Web UI への証明書マッピングルールの詳細の入力



7. ダイアログボックスの下部にある **Add** をクリックして、ルールを追加し、ダイアログボックスを閉じます。

8. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

```
# systemctl restart sssd
```

これで、証明書マッピングルールセットが設定され、スマートカードの証明書で検出されたマッピングルールで指定されたデータの種類と、IdM ユーザーエントリーの証明書マッピングデータを比較します。一致するファイルが見つかると、一致するユーザーが認証されます。

9.4.2. IdM CLI での証明書マッピングルールの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. マッピングルールを入力し、マッピングルールの基となっているマッチングルールを入力します。たとえば、提示する証明書内の **Issuer** および **Subject** のエントリーを IdM で検索し、提示された証明書に含まれるこの 2 つのエントリーで見つかった情報に基づいて認証するかどうかを決定し、**EXAMPLE.ORG** 組織の **Smart Card CA** が発行する証明書のみを認識するには、次のコマンドを実行します。

```
# ipa certmaprule-add rule_name --matchrule '<ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG' --maprule '(ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})'
----------------------------------------------------
Added Certificate Identity Mapping Rule "rule_name"
----------------------------------------------------
  Rule name: rule_name
  Mapping rule: (ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})
  Matching rule: <ISSUER>CN=Smart Card CA,O=EXAMPLE.ORG
  Enabled: TRUE
```

3. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

```
# systemctl restart sssd
```

これで、証明書マッピングルールセットが設定され、スマートカードの証明書で検出されたマッピングルールで指定されたデータの種類と、IdM ユーザーエントリーの証明書マッピングデータを比較します。一致するファイルが見つかると、一致するユーザーが認証されます。

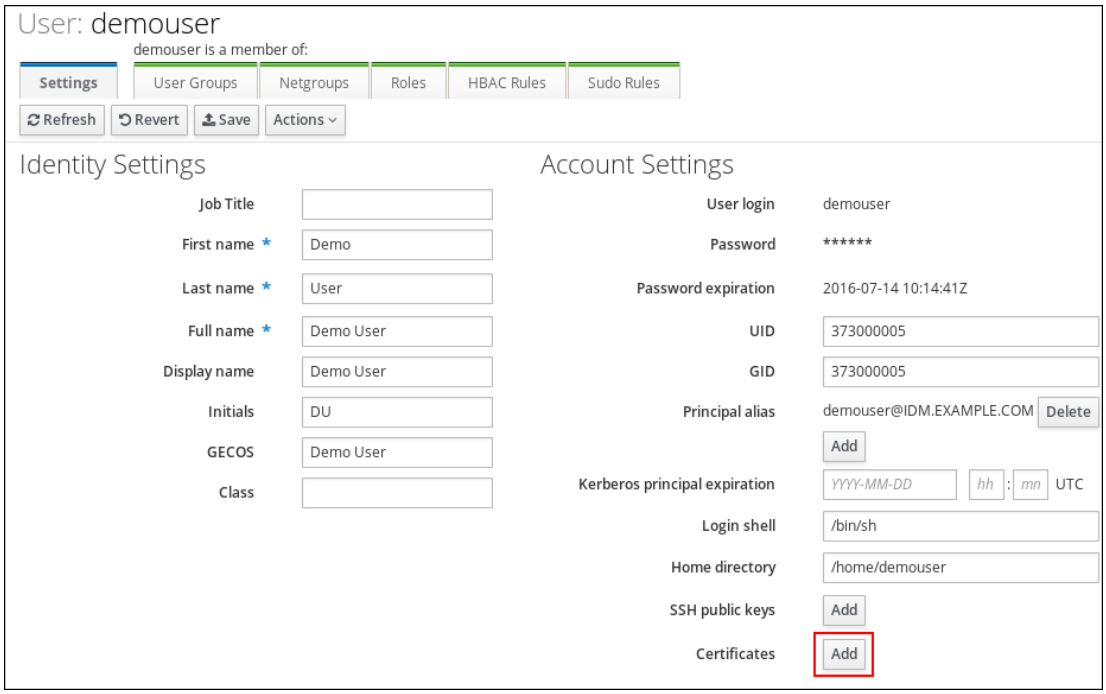
9.4.3. IdM Web UI のユーザーエントリーへの証明書マッピングデータの追加

1. 管理者として IdM Web UI にログインします。
2. **Users → Active users → idm_user** に移動します。
3. **Certificate mapping data** オプションを見つけ、**Add** をクリックします。
4. 以下のいずれかのオプションを選択します。
   - **idm_user** の証明書がある場合:
     a. コマンドラインインターフェイスで、**cat** ユーティリティーまたはテキストエディターを使用して証明書を表示します。

```
[root@server ~]# cat idm_user_certificate.pem
-----BEGIN CERTIFICATE-----
MIIFFTCCA/2gAwIBAgIBEjANBgkqhkiG9w0BAQsFADA6MRgwFgYDVQQKDA9JRE0u
RVhBTVBMRS5DT00xHjAcBgNVBAMMFUNlcnRpZmljYXRlIEF1dGhvcml0eTAeFw0x

ODA5MDIxODE1MzlaFw0yMDA5MDIxODE1MzlaMCwxGDAWBgNVBAoMD0lETS5FWEFN
[...output truncated...]
```

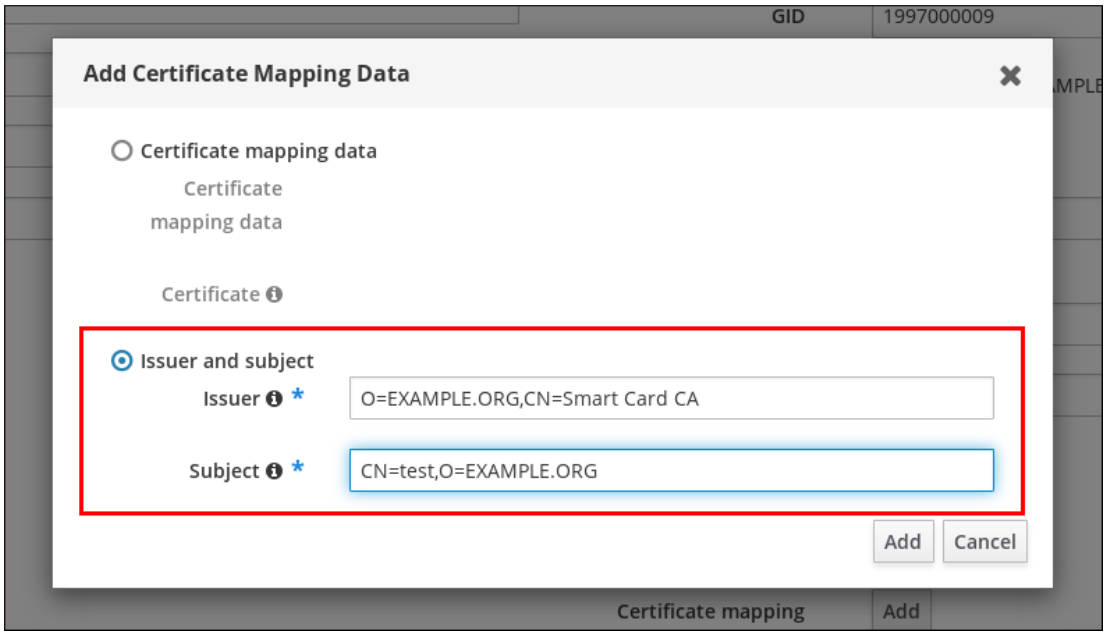
     b. 証明書をコピーします。
     c. IdM Web UI で、**Certificate** の横にある **Add** をクリックして、開いたウィンドウに証明書を貼り付けます。

図9.3 ユーザーの証明書マッピングデータの追加 - 証明書



- **idm_user** の証明書を自由に使用できないけれど、証明書の **Issuer** および **Subject** がわかっている場合は、**Issuer and subject** のラジオボタンをオンにして、2 つのそれぞれのボックスに値を入力します。

図9.4 ユーザーの証明書マッピングデータの追加 - 発行者および発行先



5. **Add** をクリックします。

検証手順

必要に応じて、**.pem** 形式の証明書全体へのアクセスがある場合は、ユーザーと証明書がリンクされていることを確認します。

1. **sss_cache** ユーティリティーを使用して、SSSD キャッシュで **idm_user** の記録を無効にし、**idm_user** 情報を再読み込みします。

```
# sss_cache -u idm_user
```

2. **ipa certmap-match** コマンドに、IdM ユーザーの証明書が含まれるファイルの名前を付けて実行します。

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

この出力では、証明書マッピングデータが **idm_user** に追加され、対応するマッピングルールが存在することを確認します。これは、定義した証明書マッピングデータに一致する証明書を使用して、**idm_user** として認証できることを意味します。

9.4.4. IdM CLI のユーザーエントリーへの証明書マッピングデータの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. 以下のいずれかのオプションを選択します。

- **idm_user** の証明書をお持ちの場合は、**ipa user-add-cert** コマンドを使用して証明書をユーザーアカウントに追加します。

```
# CERT=$(openssl x509 -in idm_user_cert.pem -outform der|base64 -w0)
# ipa user-add-certmapdata idm_user --certificate $CERT
```

- **idm_user** の証明書を持っていないが、ユーザーの証明書の **Issuer** および **Subject** がわかっている場合は、以下を実行します。

```
# ipa user-add-certmapdata idm_user --subject "O=EXAMPLE.ORG,CN=test" --
issuer "CN=Smart Card CA,O=EXAMPLE.ORG"
-----------------------------------------
Added certificate mappings to user "idm_user"
-----------------------------------------
  User login: idm_user
  Certificate mapping data: X509:<I>O=EXAMPLE.ORG,CN=Smart Card
CA<S>CN=test,O=EXAMPLE.ORG
```

検証手順

必要に応じて、**.pem** 形式の証明書全体へのアクセスがある場合は、ユーザーと証明書がリンクされていることを確認します。

1. **sss_cache** ユーティリティーを使用して、SSSD キャッシュで **idm_user** の記録を無効にし、**idm_user** 情報を再読み込みします。

```
# sss_cache -u idm_user
```

2. **ipa certmap-match** コマンドに、IdM ユーザーの証明書が含まれるファイルの名前を付けて実行します。

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

この出力では、証明書マッピングデータが **idm_user** に追加され、対応するマッピングルールが存在することを確認します。これは、定義した証明書マッピングデータに一致する証明書を使用して、**idm_user** として認証できることを意味します。

9.5. ACTIVE DIRECTORY ドメインとの信頼に対する証明書マッピングルール

IdM デプロイメントが Active Directory (AD) ドメインと信頼関係にある場合、さまざまな証明書マッピングの使用例が可能です。

AD 設定によっては、以下の状況が考えられます。

- 証明書が AD Certificate System によって発行され、ユーザーと証明書が IdM に保存されている場合、認証リクエストのマッピングと処理全体が IdM 側で行われます。このシナリオの設定に関する詳細は IdM に保存されたユーザーの証明書マッピングの設定 を参照してください。
- ユーザーが AD に保存されている場合は、認証要求の処理が AD で実行されます。サブケースは 3 つあります。
  - AD ユーザーエントリーに、証明書全体が含まれる場合。このシナリオで IdM を設定する方法は、AD ユーザーエントリーに証明書全体が含まれるユーザー用の証明書マッピングの設定 を参照してください。
  - AD が、ユーザー証明書をユーザーアカウントにマップするように設定されている場合。この場合、AD ユーザーエントリーには証明書全体が含まれず、代わりに **altSecurityIdentities** と呼ばれる属性が含まれます。このシナリオで IdM を設定する方法は、AD がユーザー証明書をユーザーアカウントにマッピングするように設定している場合は、証明書マッピングの設定 を参照してください。
  - AD ユーザーエントリーに、証明書全体またはマッピングデータが含まれない場合。この場合、次の 2 つのオプションがあります。
    - ユーザー証明書が AD Certificate System によって発行された場合、証明書にはサブジェクト代替名 (SAN) としてユーザープリンシパル名が含まれるか、証明書の SID 拡張子のユーザーの SID (最新の更新が AD に適用されている場合) が含まれます。これらは両方とも、証明書をユーザーにマッピングするために使用できます。
    - ユーザー証明書がスマートカード上にある場合、スマートカードで SSH を有効にするには、SSSD は証明書から公開 SSH キーを取得する必要があるため、完全な証明書が必要です。唯一の解決策は、**ipa idoverrideuser-add** コマンドを使用して、証明書全体を IdM の AD ユーザーの ID オーバーライドに追加することです。詳細は AD ユーザーエントリーに証明書やマッピングデータが含まれていない場合の証明書マッピングの設定 を参照してください。

AD ドメイン管理者は、**altSecurityIdentities** 属性を使用して証明書を AD 内のユーザーに手動でマッピングできます。この属性には 6 つの値がサポートされていますが、3 つのマッピングは安全ではないと考えられています。2022 年 5 月 10 日のセキュリティー更新 の一環として、インストールされると、すべてのデバイスは互換モードになり、証明書がユーザーに弱くマッピングされている場合、認証は期待どおりに行われます。ただし、完全強制モードと互換性のない証明書を特定する警告メッセージがログに記録されます。2023 年 11 月 14 日以降、すべてのデバイスは完全強制モードに更新され、証明書が強力なマッピング基準を満たさない場合、認証は拒否されます。

たとえば、AD ユーザーが証明書 (PKINIT) を含む IdM Kerberos チケットをリクエストすると、AD は証明書を内部でユーザーにマップする必要があり、これに新しいマッピングルールを使用します。ただし、IdM では、IdM クライアント上のユーザーに証明書をマップするために IdM が使用されている場合、以前のルールが引き続き機能します。

IdM は新しいマッピングテンプレートをサポートしているため、AD 管理者は新しいルールを使用し、両方を維持する必要がなくなります。IdM は、Active Directory に追加された以下を含む新しいマッピングテンプレートをサポートするようになりました。

- シリアル番号: LDAPU1:(altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<SR>{serial_number!hex_ur})
- サブジェクトキー ID: LDAPU1:(altSecurityIdentities=X509:<SKI>{subject_key_id!hex_u})
- User SID: LDAPU1:(objectsid={sid})

新しい SID 拡張子を使用して証明書を再発行したくない場合は、AD のユーザーの **altSecurityIdentities** 属性に適切なマッピング文字列を追加して、手動マッピングを作成できます。

9.6. AD ユーザーエントリーに証明書全体が含まれるユーザーに証明書マッピングを設定

このユーザーストーリーでは、IdM デプロイメントが Active Directory (AD) を信頼し、そのユーザーが AD に保存され、AD のユーザーエントリーに証明書全体が含まれる場合に、IdM で証明書マッピングを有効にするのに必要な手順を説明します。

前提条件

- IdM にユーザーアカウントがない。
- ユーザーに、証明書を含む AD のアカウントがある。
- IdM 管理者が、IdM 証明書マッピングルールが基になっているデータにアクセスできる。

注記

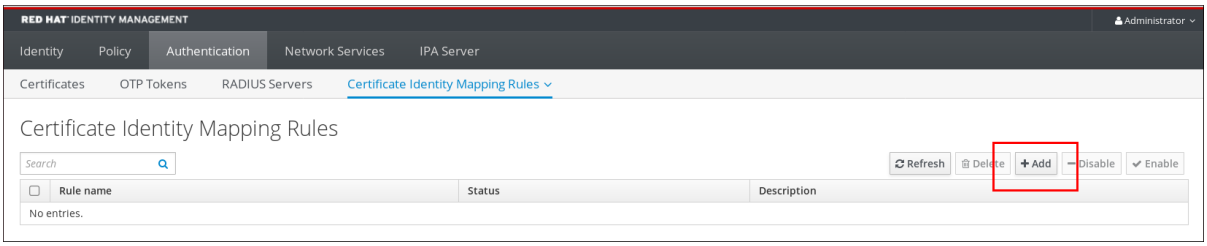
PKINIT がユーザーに対して確実に機能するには、次の条件のいずれかが適用される必要があります。

- ユーザーエントリーの証明書には、ユーザープリンシパル名またはユーザーの SID 拡張子が含まれます。
- AD のユーザーエントリーには、**altSecurityIdentities** 属性に適切なエントリーがあります。

9.6.1. IdM Web UI で証明書マッピングルールの追加

1. 管理者として IdM Web UI にログインします。
2. **Authentication → Certificate Identity Mapping Rules → Certificate Identity Mapping Rules** の順に移動します。
3. **Add** をクリックします。

図9.5 IdM Web UI で新しい証明書マッピングルールの追加



4. ルール名を入力します。
5. マッピングルールを入力します。認証のために IdM に提示された証明書全体を、AD で利用可能な証明書全体と比較するには、次のコマンドを実行します。

```
(userCertificate;binary={cert!bin})
```

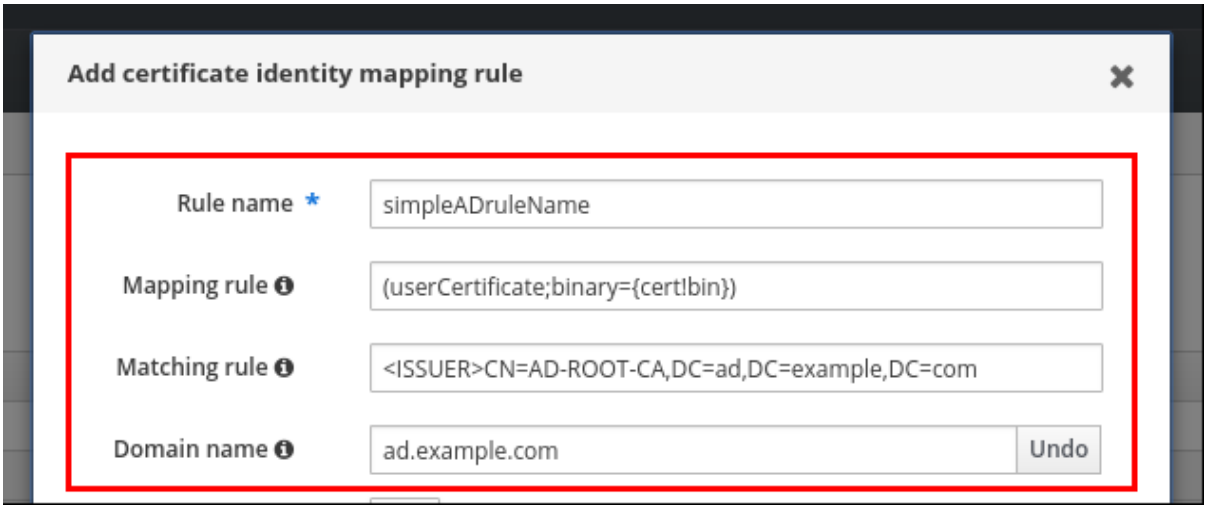
注記

完全な証明書を使用してマッピングする場合、また、証明書を更新する場合は、新しい証明書を AD ユーザーオブジェクトに必ず追加する必要があります。

6. マッチングルールを入力します。たとえば、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** が発行する証明書のみを認証できるようにするには、次のコマンドを実行します。

```
<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
```

図9.6 AD に保存されている証明書があるユーザーの証明書マッピングルール



7. **Add** をクリックします。
8. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにするには、CLI で SSSD を再起動します。

```
# systemctl restart sssd
```

9.6.2. IdM CLI での証明書マッピングルールの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. マッピングルールを入力し、マッピングルールの基となっているマッチングルールを入力します。AD で利用可能な証明書と比較する、認証用に提示される証明書全体を取得して、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** により発行された証明書のみの認証を許可するには、次のコマンドを実行します。

```
# ipa certmaprule-add simpleADrule --matchrule '<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com' --maprule '(userCertificate;binary={cert!bin})' --domain ad.example.com
----------------------------------------------------
Added Certificate Identity Mapping Rule "simpleADrule"
----------------------------------------------------
  Rule name: simpleADrule
  Mapping rule: (userCertificate;binary={cert!bin})
  Matching rule: <ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
  Domain name: ad.example.com
  Enabled: TRUE
```

注記

完全な証明書を使用してマッピングする場合、また、証明書を更新する場合は、新しい証明書を AD ユーザーオブジェクトに必ず追加する必要があります。

3. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

```
# systemctl restart sssd
```

9.7. ユーザー証明書をユーザーアカウントにマッピングするように AD が設定されている場合に、証明書マッピングの設定

このユーザーストーリーでは、IdM デプロイメントが Active Directory (AD) を信頼し、そのユーザーが AD に保存され、AD のユーザーエントリーに証明書マッピングデータが含まれる場合に、IdM で証明書マッピングを有効にするのに必要な手順を説明します。

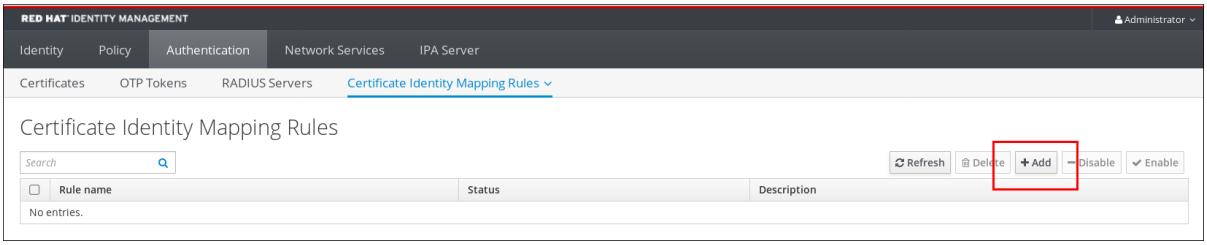
前提条件

- IdM にユーザーアカウントがない。
- このユーザーに、**altSecurityIdentities** 属性を含む AD にアカウントがある。AD は、IdM の **certmapdata** 属性に相当します。
- IdM 管理者が、IdM 証明書マッピングルールが基になっているデータにアクセスできる。

9.7.1. IdM Web UI で証明書マッピングルールの追加

1. 管理者として IdM Web UI にログインします。
2. **Authentication → Certificate Identity Mapping Rules → Certificate Identity Mapping Rules** の順に移動します。
3. **Add** をクリックします。

図9.7 IdM Web UI で新しい証明書マッピングルールの追加



4. ルール名を入力します。
5. マッピングルールを入力します。たとえば、提示された証明書で **Issuer** エントリーおよび **Subject** エントリーを AD DC で検索し、提示された証明書に含まれるこの 2 つのエントリーで見つかった情報に基づいて認証するかどうかを決定するには、次のコマンドを実行します。

```
(altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500})
```

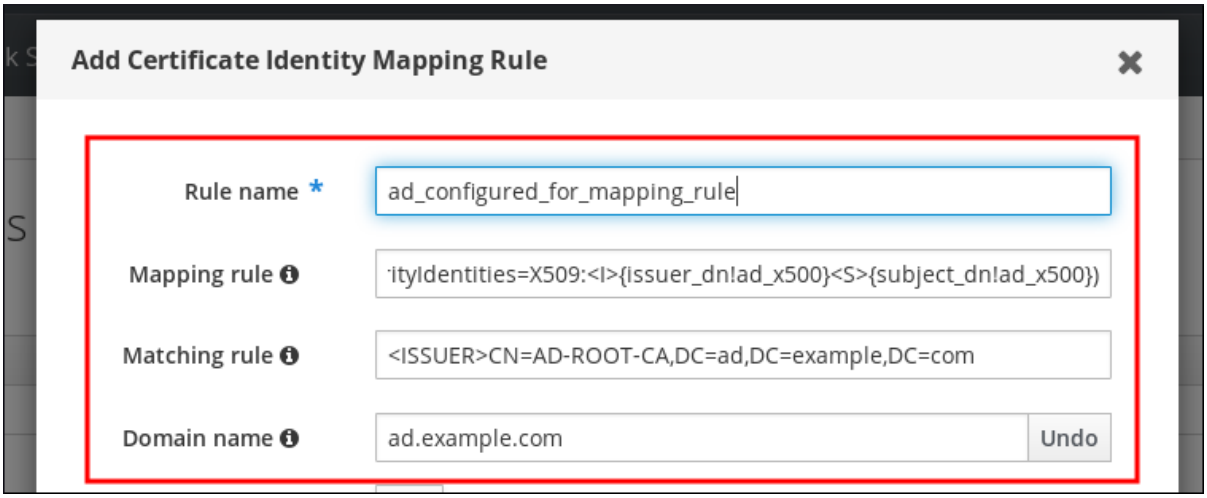
6. マッチングルールを入力します。たとえば、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** が発行する証明書のみを許可し、IdM に対してユーザーを認証するには、次のコマンドを実行します。

```
<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
```

7. ドメインを入力します。

```
ad.example.com
```

図9.8 AD がマッピング用に設定されている場合の証明書マッピングルール



8. **Add** をクリックします。

9. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにするには、CLI で SSSD を再起動します。

```
# systemctl restart sssd
```

9.7.2. IdM CLI での証明書マッピングルールの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. マッピングルールを入力し、マッピングルールの基となっているマッチングルールを入力します。たとえば、提示する証明書の **Issuer** エントリーおよび **Subject** エントリーを AD で検索し、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** により発行された証明書のみを許可するには、次のコマンドを実行します。

```
# ipa certmaprule-add ad_configured_for_mapping_rule --matchrule
'<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com' --maprule
'(altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500})' --
domain=ad.example.com
----------------------------------------------------
Added Certificate Identity Mapping Rule "ad_configured_for_mapping_rule"
----------------------------------------------------
  Rule name: ad_configured_for_mapping_rule
  Mapping rule: (altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>
{subject_dn!ad_x500})
  Matching rule: <ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
  Domain name: ad.example.com
  Enabled: TRUE
```

3. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

```
# systemctl restart sssd
```

9.7.3. AD で証明書マッピングデータの確認

altSecurityIdentities 属性は、IdM の **certmapdata** ユーザー属性と同等の Active Directory (AD) です。信頼されている AD ドメインが、ユーザーアカウントにユーザー証明書をマッピングするように設定されている時に IdM で証明書マッピングを設定する場合は、IdM システム管理者が、AD のユーザーエントリーに **altSecurityIdentities** 属性が正しく設定されていることを確認する必要があります。

前提条件

- ユーザーアカウントにはユーザー管理アクセス権が必要です。

手順

- AD に保存されているユーザーの適切な情報が AD に含まれていることを確認するには、**ldapsearch** コマンドを使用します。たとえば、次のコマンドを入力して、以下の条件が適用される **adserver.ad.example.com** サーバーで、チェックします。

- **altSecurityIdentities** 属性は、**ad_user** のユーザーエントリーに設定されます。
- matchrule では、以下の条件が適用されるように指定します。
  - **ad_user** が AD への認証に使用する証明書が **ad.example.com** ドメインの **AD-ROOT-CA** により発行されている。
  - 発行者が **<S>DC=com,DC=example,DC=ad,CN=Users,CN=ad_user** である。

```
$ ldapsearch -o ldif-wrap=no -LLL -h adserver.ad.example.com \
-p 389 -D cn=Administrator,cn=users,dc=ad,dc=example,dc=com \
-W -b cn=users,dc=ad,dc=example,dc=com "(cn=ad_user)" \
altSecurityIdentities
Enter LDAP Password:
dn: CN=ad_user,CN=Users,DC=ad,DC=example,DC=com
altSecurityIdentities: X509:<I>DC=com,DC=example,DC=ad,CN=AD-ROOT-
CA<S>DC=com,DC=example,DC=ad,CN=Users,CN=ad_user
```

9.8. AD ユーザーエントリーに証明書やマッピングデータが含まれていない場合に、証明書マッピングの設定

このユーザーストーリーでは、IdM デプロイメントが Active Directory (AD) を信頼し、そのユーザーが AD に保存され、AD のユーザーエントリーに証明書全体または証明書マッピングデータが含まれる場合に、IdM で証明書マッピングを有効にするのに必要な手順を説明します。

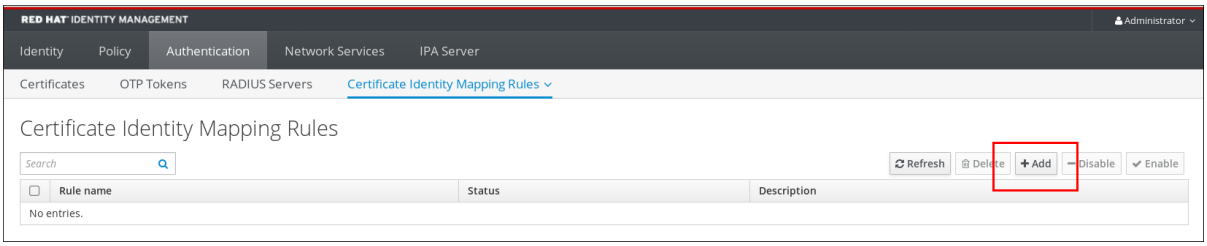
前提条件

- IdM にユーザーアカウントがない。
- ユーザーのアカウントがある AD に、証明書全体、または **altSecurityIdentities** 属性、IdM **certmapdata** 属性で AD に相当するものがない。
- IdM 管理者は次のいずれかを実行しました。
  - AD ユーザー証明書全体を IdM の AD ユーザーの **user ID override** に追加しました。
  - サブジェクト代替名やユーザーの SID など、証明書内の代替フィールドにマップする証明書マッピングルールを作成しました。

9.8.1. IdM Web UI で証明書マッピングルールの追加

1. 管理者として IdM Web UI にログインします。
2. **Authentication → Certificate Identity Mapping Rules → Certificate Identity Mapping Rules** の順に移動します。
3. **Add** をクリックします。

図9.9 IdM Web UI で新しい証明書マッピングルールの追加



4. ルール名を入力します。

5. マッピングルールを入力します。認証するために IdM に提示された証明書全体を、IdM の AD ユーザーエントリーのユーザー ID オーバーライドエントリーに保存されている証明書と比較できるようにするには、次のコマンドを実行します。

```
(userCertificate;binary={cert!bin})
```

注記

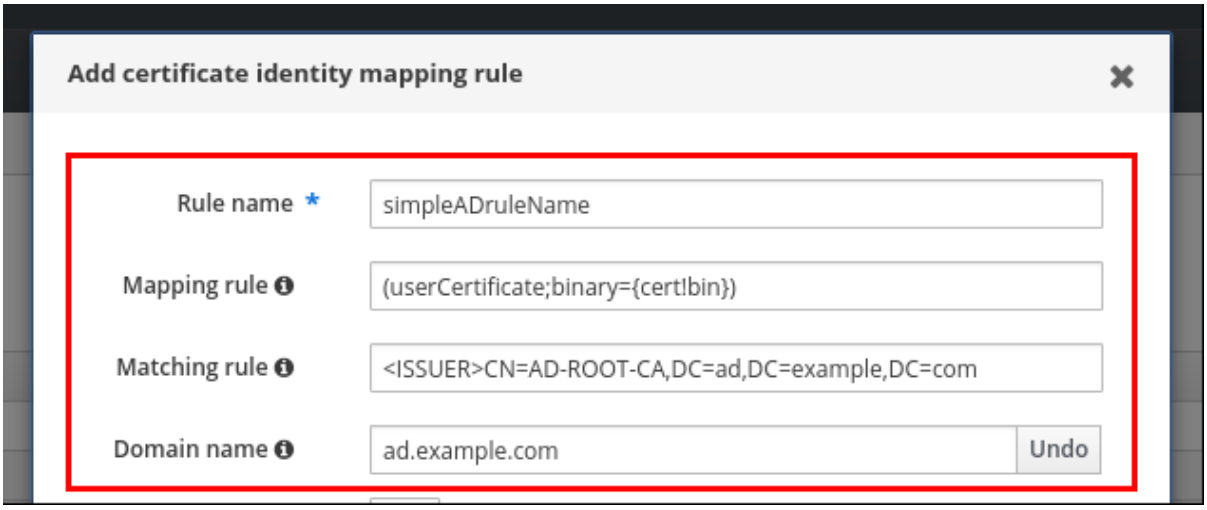
証明書には SAN としてのユーザープリンシパル名も含まれており、最新の更新では証明書の SID 拡張子にユーザーの SID も含まれているため、これらのフィールドを使用して証明書をユーザーにマップすることもできます。たとえば、ユーザーの SID を使用する場合は、このマッピングルールを **LDAPU1: (objectsid={sid})** に置き換えます。証明書マッピングの詳細は、**sss-certmap** の man ページを参照してください。

6. マッチングルールを入力します。たとえば、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** が発行する証明書のみを認証できるようにするには、次のコマンドを実行します。

```
<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
```

7. ドメイン名を入力します。たとえば、**ad.example.com** ドメインでユーザーを検索するには、以下を実行します。

図9.10 AD に証明書やマッピングデータが保存されていないユーザーに対する証明書マッピングルール



8. **Add** をクリックします。

9. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにするには、CLI で SSSD を再起動します。

```
# systemctl restart sssd
```

9.8.2. IdM CLI での証明書マッピングルールの追加

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. マッピングルールを入力し、マッピングルールの基となっているマッチングルールを入力します。IdM の AD ユーザーエントリーのユーザー ID オーバーライドエントリーに保存されている証明書と比較する、認証用に提示される証明書全体を取得して、**AD.EXAMPLE.COM** ドメインの **AD-ROOT-CA** により発行された証明書のみを認証できるようにするには、以下のコマンドを実行します。

```
# ipa certmaprule-add simpleADrule --matchrule '<ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com' --maprule '(userCertificate;binary={cert!bin})' --domain ad.example.com
----------------------------------------------------
Added Certificate Identity Mapping Rule "simpleADrule"
----------------------------------------------------
  Rule name: simpleADrule
  Mapping rule: (userCertificate;binary={cert!bin})
  Matching rule: <ISSUER>CN=AD-ROOT-CA,DC=ad,DC=example,DC=com
  Domain name: ad.example.com
  Enabled: TRUE
```

注記

証明書には SAN としてのユーザープリンシパル名も含まれており、最新の更新では証明書の SID 拡張子にユーザーの SID も含まれているため、これらのフィールドを使用して証明書をユーザーにマップすることもできます。たとえば、ユーザーの SID を使用する場合は、このマッピングルールを **LDAPU1:(objectsid={sid})** に置き換えます。証明書マッピングの詳細は、**sss-certmap** の man ページを参照してください。

3. System Security Services Daemon (SSSD) は、証明書マッピングルールを定期的に再読み込みします。新たに作成したルールがすぐに読み込まれるようにする場合は、次のコマンドを実行して SSSD を再起動します。

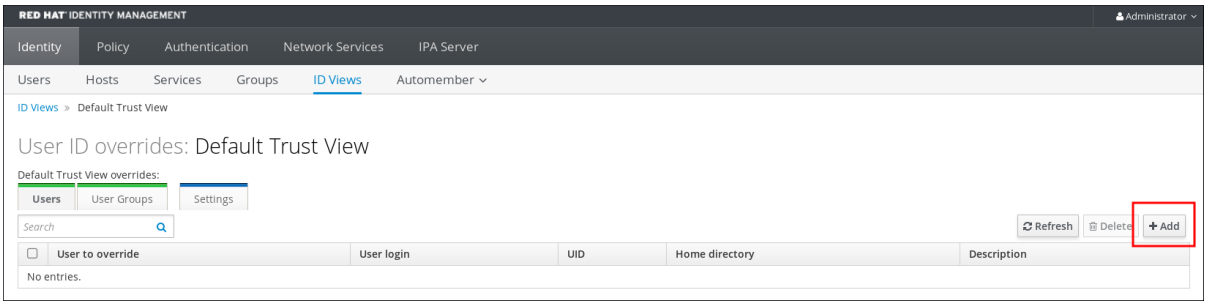
```
# systemctl restart sssd
```

9.8.3. IdM Web UI で、AD ユーザーの ID オーバーライドに証明書を追加

1. **Identity → ID Views → Default Trust View** の順に選択します。

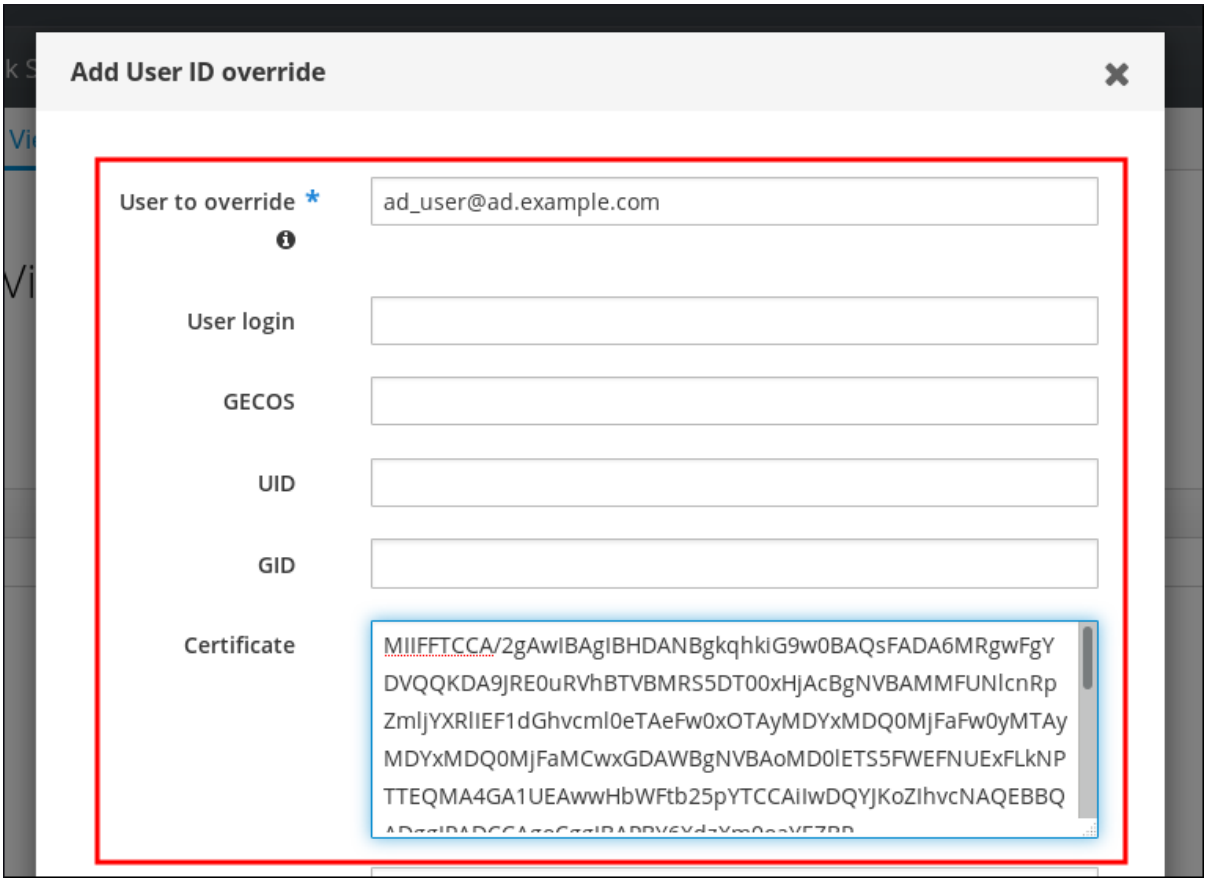
2. **Add** をクリックします。

図9.11 IdM Web UI で新規ユーザー ID オーバーライドの追加



3. **User to override** フィールドに、**ad_user@ad.example.com** と入力します。
4. **ad_user** の証明書を、**Certificate** フィールドにコピーアンドペーストします。

図9.12 AD ユーザーにユーザー ID オーバーライドの設定



5. **Add** をクリックします。

検証手順

ユーザーと証明書がリンクしていることを確認します。

1. **sss_cache** ユーティリティーを使用して、SSSD キャッシュで **ad_user@ad.example.com** のレコードを無効にし、**ad_user@ad.example.com** 情報の再読み込みを強制します。

```
# sss_cache -u ad_user@ad.example.com
```

2. AD ユーザーの証明書が含まれるファイルの名前で、**ipa certmap-match** コマンドを実行します。

```
# ipa certmap-match ad_user_cert.pem
--------------
1 user matched
--------------
 Domain: AD.EXAMPLE.COM
 User logins: ad_user@ad.example.com
----------------------------
Number of entries returned 1
----------------------------
```

この出力では、**ad_user@ad.example.com** に追加した証明書マッピングデータがあり、Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data で定義した対応するマッピングルールが存在することを確認します。これは、定義した証明書マッピングデータに一致する証明書を使用して、**ad_user@ad.example.com** として認証できることを意味します。

関連情報

Active Directory ユーザーの ID ビューの使用

9.8.4. IdM CLI で、AD ユーザーの ID オーバーライドに証明書を追加する

1. 管理者の認証情報を取得します。

```
# kinit admin
```

2. 証明書 blob を **CERT** という新しい変数に格納します。

```
# CERT=$(openssl x509 -in /path/to/certificate -outform der|base64 -w0)
```

3. **ipa idoverrideuser-add-cert** コマンドを使用して、**ad_user@ad.example.com** の証明書をユーザーアカウントに追加します。

```
# ipa idoverrideuser-add-cert ad_user@ad.example.com --certificate $CERT
```

検証手順

ユーザーと証明書がリンクしていることを確認します。

1. **sss_cache** ユーティリティーを使用して、SSSD キャッシュで **ad_user@ad.example.com** のレコードを無効にし、**ad_user@ad.example.com** 情報の再読み込みを強制します。

```
# sss_cache -u ad_user@ad.example.com
```

2. AD ユーザーの証明書が含まれるファイルの名前で、**ipa certmap-match** コマンドを実行します。

```
# ipa certmap-match ad_user_cert.pem
--------------
1 user matched
--------------
 Domain: AD.EXAMPLE.COM
 User logins: ad_user@ad.example.com
```

```
----------------------------
Number of entries returned 1
----------------------------
```

この出力では、**ad_user@ad.example.com** に追加した証明書マッピングデータがあり、Adding a certificate mapping rule if the AD user entry contains no certificate or mapping data で定義した対応するマッピングルールが存在することを確認します。これは、定義した証明書マッピングデータに一致する証明書を使用して、**ad_user@ad.example.com** として認証できることを意味します。

関連情報

Active Directory ユーザーの ID ビューの使用

9.9. 複数のアイデンティティーマッピングルールを 1 つに結合

複数の ID マッピングルールを 1 つのルールに結合するには、個々のマッピングルールの前に | (or) 文字を追加し、括弧 **()** で区切ります。以下に例を示します。

証明書マッピングフィルターの例 1

```
$ ipa certmaprule-add ad_cert_for_ipa_and_ad_users \ --maprule='(|(ipacertmapdata=X509:<I>
{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})(altSecurityIdentities=X509:<I>
{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}))' \ --matchrule='<ISSUER>CN=AD-ROOT-
CA,DC=ad,DC=example,DC=com' \ --domain=ad.example.com
```

上記の例では、**--maprule** オプションのフィルター定義には、以下の基準が含まれます。

- **ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500}** は、IdM での証明書マッピングルールの追加 の説明のとおりに、IdM ユーザーアカウントの **ipacertmapdata** 属性の値に、スマートカードの発行先および発行者をリンクさせるフィルターです。
- **altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}** は、スマートカード証明書から発行先および発行者を、AD ユーザーアカウントの **altSecurityIdentities** の値にリンクするフィルターです。これは、信頼された AD ドメインがユーザー証明書をマッピングするように設定されている場合には、証明書マッピングルールの追加 で説明されています。
- **--domain=ad.example.com** オプションを追加すると、指定した証明書にマッピングされたユーザーが、ローカルの **idm.example.com** ドメインだけでなく、ad.example.com**ad.example.com** ドメイン内でも検索されます。

--maprule オプションのフィルターの定義では、論理演算子 | (or) が使用できるため、複数の基準を指定できます。この場合、ルールは、1 つ以上の基準を満たすユーザーアカウントをすべてマップします。

証明書マッピングフィルターの例 2

```
$ ipa certmaprule-add ipa_cert_for_ad_users \
  --maprule='(|(userCertificate;binary={cert!bin})(ipacertmapdata=X509:<I>
{issuer_dn!nss_x500}<S>{subject_dn!nss_x500})(altSecurityIdentities=X509:<I>
{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}))' \
  --matchrule='<ISSUER>CN=Certificate Authority,O=REALM.EXAMPLE.COM' \
  --domain=idm.example.com --domain=ad.example.com
```

上記の例では、**--maprule** オプションのフィルター定義には、以下の基準が含まれます。

- **userCertificate;binary={cert!bin}** は、証明書全体を含むユーザーエントリーを返すフィルターです。AD ユーザーが、この種のフィルターを作成する場合は、AD ユーザーエントリーに証明書またはマッピングデータがない場合の証明書マッピングルールの追加 を参照してください。

- **ipacertmapdata=X509:<I>{issuer_dn!nss_x500}<S>{subject_dn!nss_x500}** は、IdM での証明書マッピングルールの追加 の説明のとおりに、IdM ユーザーアカウントの **ipacertmapdata** 属性の値に、スマートカードの発行先および発行者をリンクさせるフィルターです。

- **altSecurityIdentities=X509:<I>{issuer_dn!ad_x500}<S>{subject_dn!ad_x500}** は、スマートカード証明書から発行先および発行者を、AD ユーザーアカウントの **altSecurityIdentities** の値にリンクするフィルターです。これは、信頼された AD ドメインがユーザー証明書をマッピングするように設定されている場合には、証明書マッピングルールの追加 で説明されています。

--maprule オプションのフィルターの定義では、論理演算子 | (or) が使用できるため、複数の基準を指定できます。この場合、ルールは、1 つ以上の基準を満たすユーザーアカウントをすべてマップします。

9.10. 関連情報

- **sss-certmap(5)** の man ページを参照してください。

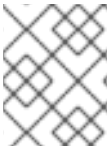
第10章 IDM クライアントのデスクトップに保存されている証明書を使用した認証の設定

Identity Management (IdM) を設定すると、IdM システム管理者は、認証局 (CA) がユーザーに発行した証明書を使用して、IdM Web UI とコマンドラインインターフェイス (CLI) に対してユーザーを認証できます。証明書は IdM クライアントのデスクトップに保存されます。

Web ブラウザーは、IdM ドメイン外のシステムで実行できます。

証明書を使用した認証設定中に、以下の点に注意してください。

- 証明書を使用して認証するユーザーに証明書がすでにある場合は、新しいユーザー証明書を要求し、クライアントにエクスポート を省略できます。
- ユーザーの証明書が IdM CA により発行された場合は、証明書とユーザーが互いにリンクしていることの確認 を省略できます。



注記

Identity Management ユーザーのみが、証明書を使用して Web UI にログインできます。Active Directory ユーザーは、ユーザー名とパスワードを使用してログインできます。

10.1. WEB UI での証明書認証用の IDENTITY MANAGEMENT SERVER の設定

Identity Management (IdM) 管理者は、ユーザーが、証明書を使用して IdM 環境で認証できるように設定できます。

手順

Identity Management 管理者が、以下を行います。

1. Identity Management サーバーで管理者権限を取得し、サーバーを設定するシェルスクリプトを作成します。
   a. **ipa-advise config-server-for-smart-card-auth** コマンドを実行し、その出力をファイル (例: **server_certificate_script.sh**) に保存します。

```
# kinit admin
# ipa-advise config-server-for-smart-card-auth > server_certificate_script.sh
```

   b. **chmod** ユーティリティーを使用して、実行パーミッションをファイルに追加します。

```
# chmod +x server_certificate_script.sh
```

2. Identity Management ドメインの全サーバーで、**server_certificate_script.sh** スクリプトを実行します。
   a. 証明書認証を有効にするユーザーの証明書を発行した唯一の認証局が IdM CA である場合は、IdM Certificate Authority 証明書へのパス (**/etc/ipa/ca.crt**) を使用します。

```
# ./server_certificate_script.sh /etc/ipa/ca.crt
```

b. 証明書認証を有効にするユーザーの証明書を外部の複数の CA が署名した場合は、関連する CA 証明書へのパスを使用します。

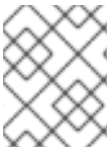
```
# ./server_certificate_script.sh /tmp/ca1.pem /tmp/ca2.pem
```

注記

トポロジー全体でユーザーの証明書認証を有効にする場合は、今後新たにシステムに追加する各レプリカに対して、必ずスクリプトを実行してください。

10.2. 新しいユーザー証明書を要求し、クライアントにエクスポート

Identity Management (IdM) 管理者は、IdM 環境でユーザーの証明書を作成し、作成した証明書を、ユーザーの証明書認証を有効にする IdM クライアントにエクスポートできます。



注記

証明書を使用して認証するユーザーがすでに証明書を持っている場合は、この手順を実行する必要はありません。

手順

1. 必要に応じて、新しいディレクトリー (例: ~/**certdb**/) を作成し、証明書の一時データベースを作成します。要求されたら、NSS 証明書の DB パスワードを作成し、後続の手順で生成される証明書への鍵を暗号化します。

```
# mkdir ~/certdb/
# certutil -N -d ~/certdb/
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
```

2. 証明書署名要求 (CSR) を作成し、その出力をファイルにリダイレクトします。たとえば、**IDM.EXAMPLE.COM** レルムの **idm_user** ユーザーの **4096** ビット証明書に対して、**certificate_request.csr** という名前の CSR を作成する場合は、判別を簡単にするために、証明書の秘密鍵のニックネームを **idm_user** に設定し、発行先を **CN=idm_user,O=IDM.EXAMPLE.COM** に設定します。

```
# certutil -R -d ~/certdb/ -a -g 4096 -n idm_user -s "CN=idm_user,O=IDM.EXAMPLE.COM"
> certificate_request.csr
```

3. プロンプトが表示されたら、**certutil** を使用して一時データベースを作成したときに入力したパスワードを入力します。その後、止めるように言われるまで、ランダムにタイピングし続けます。

```
Enter Password or Pin for "NSS Certificate DB":

A random seed must be generated that will be used in the
creation of your key.  One of the easiest ways to create a
random seed is to use the timing of keystrokes on a keyboard.
```

```
To begin, type keys on the keyboard until this progress meter
is full.  DO NOT USE THE AUTOREPEAT FUNCTION ON YOUR KEYBOARD!


Continue typing until the progress meter is full:
```

4. 証明書要求ファイルをサーバーに送信します。新しく発行した証明書に関連付ける Kerberos プリンシパルと、証明書を保存する出力ファイルを指定し、必要に応じて証明書のプロファイルを指定します。たとえば、**IECUserRoles** プロファイル (**idm_user@IDM.EXAMPLE.COM** プリンシパルに追加したユーザーロール拡張を持つプロファイル) の証明書を取得して、それを **~/idm_user.pem** ファイルに保存する場合は、次のコマンドを実行します。

```
# ipa cert-request certificate_request.csr --principal=idm_user@IDM.EXAMPLE.COM --
profile-id=IECUserRoles --certificate-out=~/idm_user.pem
```

5. 証明書を NSS データベースに追加します。証明書が NSS データベースの秘密鍵に一致するように、CSR を作成する際に使用したニックネームを設定するには、**-n** オプションを使用します。**-t** オプションは信頼レベルを設定します。詳細は、certutil(1) man ページを参照してください。**-i** オプションは、入力証明書ファイルを指定します。たとえば、**idm_user** ニックネームを持つ証明書を NSS データベースに追加するには、次のコマンドを実行します。証明書は、**~/certdb/** データベースの **~/idm_user.pem** ファイルに保存されます。

```
# certutil -A -d ~/certdb/ -n idm_user -t "P,," -i ~/idm_user.pem
```

6. NSS データベースの鍵で、ニックネームが **(orphan)** と表示されていないことを確認します。たとえば、**~/certdb/** データベースに保存されている証明書で、対応する鍵が存在することを確認するには、以下のコマンドを実行します。

```
# certutil -K -d ~/certdb/
< 0> rsa      5ad14d41463b87a095b1896cf0068ccc467df395   NSS Certificate
DB:idm_user
```

7. 証明書を、NSS データベースから PKCS12 形式にエクスポートするには、**pk12util** コマンドを使用します。たとえば、NSS データベース **/root/certdb** から **~/idm_user.p12** ファイルへ、**idm_user** ニックネームを持つ証明書をエクスポートする場合は、次のコマンドを実行します。

```
# pk12util -d ~/certdb -o ~/idm_user.p12 -n idm_user
Enter Password or Pin for "NSS Certificate DB":
Enter password for PKCS12 file:
Re-enter password:
pk12util: PKCS12 EXPORT SUCCESSFUL
```

8. **idm_user** の証明書認証を有効にするホストに、証明書を転送します。

```
# scp ~/idm_user.p12 idm_user@client.idm.example.com:/home/idm_user/
```

9. セキュリティー上の理由から、証明書が転送されたホストの、.pkcs12 ファイルが格納されているディレクトリーに、other グループがアクセスできないようにします。

```
# chmod o-rwx /home/idm_user/
```

10. セキュリティー上の理由から、一時 NSS データベースおよび .pkcs12 ファイルを、サーバーから削除します。

```
# rm ~/certdb/
# rm ~/idm_user.p12
```

10.3. 証明書とユーザーが互いにリンクしていることの確認

注記

ユーザーの証明書が IdM CA によって発行された場合は、この手順を実行する必要はありません。

証明書が機能するには、証明書が、それを使用して Identity Management (IdM) に認証を受けるユーザーにリンクされていることを確認する必要があります。

- 証明書が、Identity Management 環境外の認証局から提供されている場合は、ユーザーアカウントの証明書へのリンク に記載されている手順に従って、ユーザーと証明書をリンクします。
- 証明書が Identity Management CA により提供されている場合は、その証明書がユーザーエントリーに自動的に追加されているため、証明書をユーザーアカウントにリンクする必要はありません。IdM で新しい証明書を作成する方法の詳細は、新しいユーザー証明書を要求し、クライアントにエクスポート を参照してください。

10.4. 証明書認証を有効にするためのブラウザーの設定

Web UI を使用した Identity Management (IdM) へのログイン時に証明書で認証できるようにするには、ユーザーおよび関連の認証局 (CA) 証明書を Mozilla Firefox または Google Chrome ブラウザーにインポートする必要があります。ブラウザーが実行しているホスト自体は、IdM ドメインの一部である必要はありません。

IdM が Web UI への接続をサポートしているブラウザーは以下のとおりです。

- Mozilla Firefox 38 以降
- Google Chrome 46 以降

次の手順は、Mozilla Firefox 57.0.1 ブラウザーを設定する方法を説明します。

前提条件

- PKCS #12 形式で自由にブラウザーにインポートできる ユーザー証明書 がある。

手順

1. Firefox を開き、**設定 → プライバシーとセキュリティー** に移動します。

図10.1 設定のプライバシーおよびセキュリティーセクション

2. **証明書を表示** をクリックします。

図10.2 プライバシーおよびセキュリティーで証明書を表示

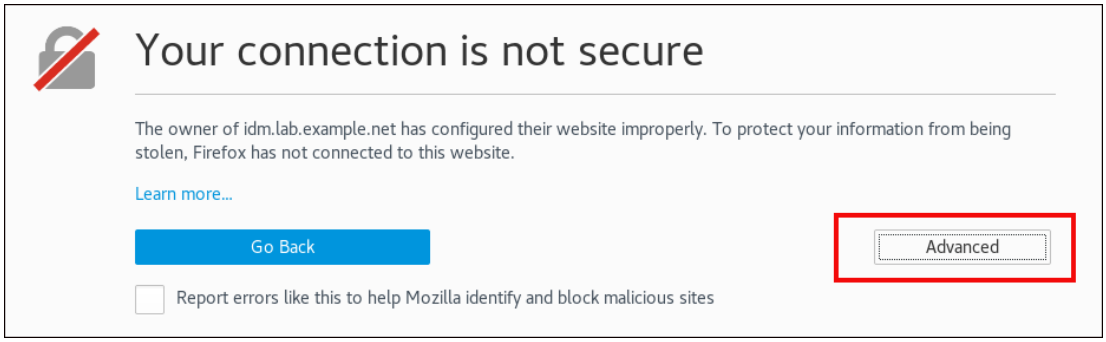
3. **あなたの証明書** タブで、**インポート** をクリックします。PKCS12 形式のユーザー証明書を見つけて開きます。**OK** をクリックし、**OK** をクリックします。

4. Identity Management 認証局が、Firefox で信頼できる認証局として認識されていることを確認します。

   a. IdM CA 証明書をローカルに保存します。

   - Firefox アドレスバーに IdM サーバーの名前を入力し、IdM の Web UI に移動します。接続が安全ではないことを警告するページで、**詳細** をクリックします。

図10.3 安全ではない接続

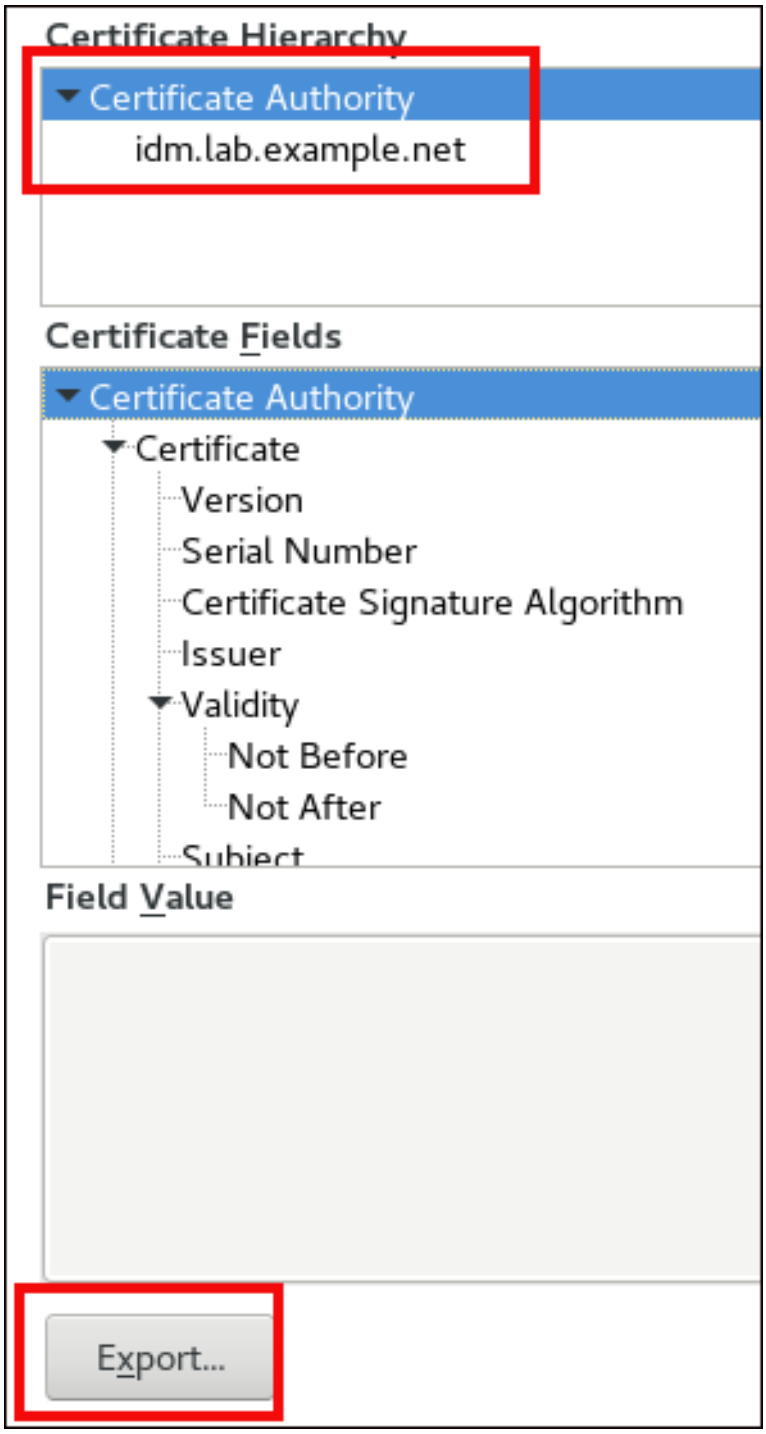


   - **例外を追加** します。**表示** をクリックします。

図10.4 証明書の詳細の表示

- **詳細** タブで、**認証局** フィールドを強調表示します。

図10.5 CA 証明書のエクスポート



- **Export** をクリックします。CA 証明書を、**CertificateAuthority.crt** ファイルとして保存し、**閉じる** をクリックして、**キャンセル** をクリックします。

b. IdM CA 証明書を、信頼できる認証局の証明書として Firefox にインポートします。

- Firefox を起動し、設定に移動して、**プライバシーおよびセキュリティー**に移動します。

図10.6 設定のプライバシーおよびセキュリティーセクション

- **証明書を表示** をクリックします。

図10.7 プライバシーおよびセキュリティーで証明書を表示

- **認証機関** タブで、**インポート** をクリックします。**CertificateAuthority.crt** ファイルで、上の手順で保存した CA 証明書を見つけて開きます。証明書を信頼し、Web サイトを識別したら、**OK** をクリックし、**OK** をクリックします。

5. Identity Management ユーザーとして証明書を使用した Identity Management Web UI の認証に進みます。

10.5. IDENTITY MANAGEMENT ユーザーとして証明書を使用した IDENTITY MANAGEMENT WEB UI の認証

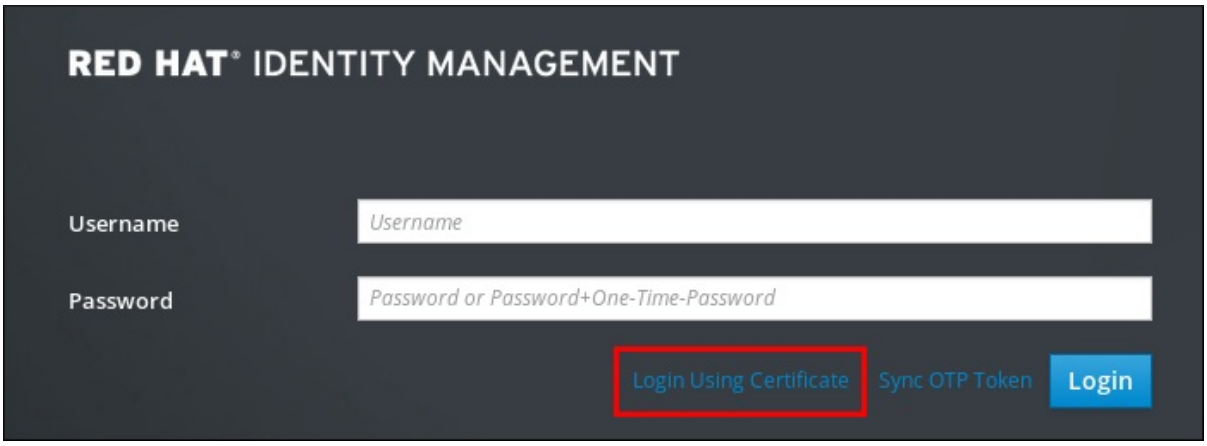
Identity Management クライアントのデスクトップに保存されている証明書を使用して、ユーザーが Identity Management (IdM) の Web UI を認証するには、次の手順に従います。

手順

1. ブラウザーで、Identity Management の Web UI (例: **https://server.idm.example.com/ipa/ui**) に移動します。

2. **Login Using Certificate** をクリックします。

図10.8 Identity Management の Web UI でLogin Using Certificate



3. ユーザーの証明書がすでに選択されているはずです。**Remember this decision** の選択を解除して、**OK** をクリックします。

これで、証明書に対応するユーザーとして認証されました。

関連情報

- Configuring Identity Management for smart card authentication を参照してください。

10.6. 証明書を使用して CLI への認証を可能にするように IDM クライアントを設定

IdM クライアントのコマンドラインインターフェイス (CLI) で、IdM ユーザーに対して証明書認証を有効にするには、IdM ユーザーの証明書および秘密鍵を IdM クライアントにインポートします。ユーザー証明書の作成と転送方法の詳細は、新しいユーザー証明書を要求し、クライアントにエクスポート を参照してください。

手順

- IdM クライアントにログインし、ユーザーの証明書と秘密鍵を含む .p12 ファイルを用意します。Kerberos TGT (Ticket Granting Ticket) ファイルを取得してキャッシュを取得するには、**-X** オプションに **X509_username:/path/to/file.p12** 属性を使用し、ユーザーのプリンシパルで **kinit** コマンドを実行して、ユーザーの X509 識別情報の場所を指定します。たとえば、**~/idm_user.p12** ファイルに保存されている識別情報を使用して **idm_user** の TGT を取得するには、次のコマンドを実行します。

```
$ kinit -X X509_idm_user='PKCS12:~/idm_user.p12' idm_user
```

注記

このコマンドは、.pem ファイル形式 (**kinit -X X509_username='FILE:/path/to/cert.pem,/path/to/key' user_principal**) にも対応しています。

第11章 IDM CA 更新サーバーの使用

11.1. IDM CA 更新サーバーの説明

組み込みの認証局 (CA) を使用する Identity Management (IdM) デプロイメントでは、CA 更新サーバーが IdM システム証明書を維持および更新します。IdM デプロイメントを確実に堅牢化します。

IdM システム証明書には、以下が含まれます。

- **IdM CA** 認証
- **OCSP** 署名証明書
- **IdM CA サブシステム** 証明書
- **IdM CA 監査署名** 証明書
- **IdM 更新エージェント** (RA) 証明書
- **KRA** トランスポート証明書およびストレージ証明書

システム証明書の特徴は、鍵がすべての CA レプリカで共有されることです。一方、IdM サービス証明書 (**LDAP** 証明書、**HTTP** 証明書、**PKINIT** 証明書など) には、さまざまな IdM CA サーバーに、さまざまな鍵ペアと発行先名があります。

IdM トポロジーでは、デフォルトで、最初の IdM CA サーバーが CA 更新サーバーになります。

注記

IdM CA は、アップストリームのドキュメントでは **Dogtag** と呼ばれています。

CA 更新サーバーのロール

IdM CA 証明書、**IdM CA サブシステム** 証明書、および **IdM RA** 証明書は、IdM デプロイメントに重要です。各証明書は、**/etc/pki/pki-tomcat/** ディレクトリーの NSS データベースおよび LDAP データベースのエントリーに保存されます。LDAP に保存されている証明書は、NSS データベースに保存されている証明書に一致する必要があります。一致しない場合は、IdM フレームワークと IdM CA、および IdM CA と LDAP との間に認証エラーが発生します。

すべての IdM CA レプリカは、すべてのシステム証明書への追跡リクエストがあります。統合 CA を備えた IdM デプロイメントに CA 更新サーバーが含まれていない場合には、各 IdM CA サーバーはシステム証明書の更新を個別に要求します。これにより、異なる CA レプリカにさまざまなシステム証明書が含まれるようになり、認証に失敗します。

CA レプリカの 1 つを更新サーバーとすることで、システム証明書が必要に応じて一度だけ更新されるため、認証が失敗しないようになります。

CA レプリカ上の certmonger サービスのロール

すべての IdM CA レプリカで実行している **certmonger** サービスは、**dogtag-ipa-ca-renew-agent** 更新ヘルパーを使用して、IdM システム証明書を追跡します。更新ヘルパープログラムは、CA 更新マスター設定を読み取ります。CA 更新サーバー以外の各 CA レプリカで、更新ヘルパーが **ca_renewal** LDAP エントリーから最新のシステム証明書を取得します。**certmonger** の更新の試みが正確に行われる際に、非決定論により、CA 更新サーバーが実際に証明書を更新する前に、**dogtag-ipa-ca-renew-agent** ヘルパーがシステム証明書の更新を試みることがあります。この状況が発生すると、CA レプリ

カの **certmonger** サービスに、すぐに有効期限が切れる古い証明書が提供されます。**certmonger** は、これデータベースにすでに保存されているのと同じ証明書であることを認識し、CA 更新サーバーから更新された証明書を取得できるまで、個々の試行の間に少し遅れて証明書の更新を試行し続けます。

IdM CA 更新サーバーの適切な機能

埋め込み CA のある IdM デプロイメントは、IdM CA でインストールされた、または後で IdM CA サーバーがインストールされた IdM デプロイメントです。組み込み CA を使用した IdM デプロイメントでは、常に更新サーバーとして設定された CA レプリカが 1 つだけ必要になります。更新サーバーはオンラインで完全に機能し、その他のサーバーで正しく複製する必要があります。

ipa server-del コマンド、**ipa-replica-manage del** コマンド、**ipa-csreplica-manage del** コマンド、または **ipa-server-install --uninstall** コマンドを使用して、現在の CA 更新サーバーを削除すると、別の CA レプリカが CA 更新サーバーとして自動的に割り当てられます。このポリシーは、更新されたサーバー設定を有効に保つようにします。

このポリシーは、以下の状況を対象としません。

- **オフライン更新サーバー**
  更新サーバーが長期間オフラインであると、更新ウィンドウが表示されない場合があります。この場合、更新されていないすべてのサーバーでは、証明書が期限切れになるまで現在のシステム証明書を再インストールし続けます。これが発生すると、証明書が失効した場合でも、その他の証明書の更新が失敗する可能性があるため、IdM デプロイメントが中断されます。

- **レプリケーションの問題**
  更新サーバーと他の CA レプリカとの間でレプリケーションの問題が存在する場合は、更新が成功する可能性もありますが、その他の CA レプリカが、期限切れになる前に更新された証明書を取得できなくなる可能性があります。

  この問題を回避するには、レプリカ合意が正しく機能することを確認してください。詳細は、RHEL 7 の **Linux ドメイン ID、認証、およびポリシーガイド** の 一般的 または 特定 のレプリケーションのトラブルシューティングガイドラインを参照してください。

11.2. IDM CA 更新サーバーの変更およびリセット

CA 更新サーバーの使用を止めると、Identity Management (IdM) により、IdM CA サーバーのリストから新しい CA 更新サーバーが自動的に選択されます。システム管理者は、選択に影響を与えることはできません。

新しい IdM CA 更新サーバーを選択できるようにするには、システム管理者が交換を手動で実行する必要があります。CA 更新サーバーを選択してから、現在の更新サーバーの使用を停止するプロセスを開始します。

現在の CA 更新サーバー設定が無効な場合は、IdM CA 更新サーバーをリセットします。

この手順では、CA 更新サーバーを変更またはリセットします。

前提条件

- IdM 管理者認証情報がある。

手順

1. IdM 管理者認証情報を取得します。

```
~]$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

2. 任意で、デプロイメント内のどの IdM サーバーが新しい CA 更新サーバーになるのに必要な CA のロールを持っているかを確認するには、次のコマンドを実行します。

```
~]$ ipa server-role-find --role 'CA server'
----------------------
2 server roles matched
----------------------
  Server name: server.idm.example.com
  Role name: CA server
  Role status: enabled

  Server name: replica.idm.example.com
  Role name: CA server
  Role status: enabled
----------------------------
Number of entries returned 2
----------------------------
```

デプロイメントには、2 つの CA サーバーがあります。

3. 必要に応じて、どの CA サーバーが現在の CA 更新サーバーであるかを確認するには、次のコマンドを実行します。

```
~]$ ipa config-show | grep 'CA renewal'
  IPA CA renewal master: server.idm.example.com
```

現在の更新サーバーは **server.idm.example.com** です。

4. 更新サーバー設定を変更するには、**--ca-renewal-master-server** オプションを指定して **ipa config-mod** ユーティリティーを使用します。

```
~]$ ipa config-mod --ca-renewal-master-server replica.idm.example.com | grep 'CA renewal'
  IPA CA renewal master: replica.idm.example.com
```

重要

以下を使用して、新しい CA 更新サーバーに切り替えることもできます。

- **ipa-cacert-manage --renew** コマンド。このコマンドは、CA 証明書を更新し、**かつ**コマンドを実行する CA サーバーを新しい CA 更新サーバーにします。
- **ipa-cert-fix** コマンド。このコマンドは、期限切れの証明書が失敗の原因になっている場合にデプロイメントを回復します。また、コマンドを実行する CA サーバーを新しい CA 更新サーバーにします。
  詳細は IdM がオフライン時に期限切れのシステム証明書の更新 を参照してください。

第12章 外部署名 CA 証明書の管理

Identity Management (IdM) は、さまざまな種類の認証局 (CA) 設定を提供します。IdM は、統合 CA または外部 CA を使用してインストールできます。インストール時に使用している CA の種類を指定する必要があります。ただし、一度インストールすれば、外部署名 CA から自己署名 CA (またはその逆) に移行することができます。また、自己署名 CA 証明書は自動的に更新されますが、外部署名 CA 証明書は必ず手動で更新する必要があります。外部署名 CA 証明書を管理するには、必要に応じて関連するセクションを参照してください。

- 外部署名 CA を使用した IdM のインストール:
  - 統合 DNS を使用し、外部 CA をルート CA として使用して IdM サーバーをインストールする
  - 統合 DNS を使用せず、外部 CA をルート CA として使用して IdM サーバーをインストールする
- 外部署名 CA から自己署名 CA への切り替え
- 自己署名 CA から外部署名 CA への切り替え
- 外部署名 CA 証明書の更新

12.1. IDM での外部署名 CA から自己署名 CA への切り替え

この手順は、外部署名から、Identity Management (IdM) 認証局 (CA) の自己署名証明書に切り替えます。自己署名の CA の場合、CA 証明書の更新は自動的に管理されます。システム管理者は、外部認証局に証明書署名リクエスト (CSR) を提出する必要はありません。

外部署名から自己署名の CA へ切り替える場合は、CA 証明書を置き換えます。以前の CA が署名する証明書は有効のままで、今でも使用されています。たとえば、**LDAP** 証明書の証明書チェーンは、自己署名の CA に移動した後も変更されません。

> **external_CA** certificate > **IdM CA** certificate > **LDAP** certificate

前提条件

- IdM CA 更新サーバーおよびすべての IdM クライアントとサーバーへの **root** アクセス権がある。

手順

1. IdM CA 更新サーバーで、CA 証明書を自己署名として更新します。

```
# ipa-cacert-manage renew --self-signed
Renewing CA certificate, please wait
CA certificate successfully renewed
The ipa-cacert-manage command was successful
```

2. **root** として、残りのすべての IdM サーバーとクライアントに **SSH** で接続します。以下に例を示します。

```
# ssh root@idmclient01.idm.example.com
```

3. IdM クライアントで、サーバーからの証明書を使用して、ローカルの IdM 証明書データベースを更新します。

```
[idmclient01 ~]# ipa-certupdate
Systemwide CA database updated.
Systemwide CA database updated.
The ipa-certupdate command was successful
```

4. 必要に応じて、更新が成功したかどうかを確認して、新しい CA 証明書を **/etc/ipa/ca.crt** ファイルに追加します。

```
[idmclient01 ~]$ openssl crl2pkcs7 -nocrl -certfile /etc/ipa/ca.crt | openssl pkcs7 -
print_certs -text -noout
[...]
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 39 (0x27)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: O=IDM.EXAMPLE.COM, CN=Certificate Authority
        Validity
            Not Before: Jul  1 16:32:45 2019 GMT
            Not After : Jul  1 16:32:45 2039 GMT
        Subject: O=IDM.EXAMPLE.COM, CN=Certificate Authority
[...]
```

この出力は、新規 CA 証明書が古い CA 証明書と共にリストされているため、更新が正常に行われたことを示しています。

12.2. IDM での自己署名 CA から外部署名 CA への切り替え

IdM で、自己署名 CA から外部署名 CA に切り替えることができます。IdM で外部署名 CA に切り替えると、IdM の CA サーバーが外部 CA の subCA になります。また、CA 証明書の更新が自動的に管理されません。システム管理者は、証明書署名要求 (CSR) を外部認証局に送信する必要があります。

外部署名 CA に切り替えるには、CSR が外部 CA によって署名されている必要があります。IdM で自己署名 CA に切り替えるには、外部 CA を使用した IdM CA 更新サーバー証明書の更新 の手順に従います。

12.3. 外部 CA を使用した IDM CA 更新サーバー証明書の更新

以下の手順に従って、外部 CA を使用して Identity Management (IdM) 認証局 (CA) 証明書を更新し、証明書署名要求 (CSR) に署名します。この設定では、IdM CA サーバーは、外部 CA のサブ CA です。外部の CA は、Active Directory Certificate Server (AD CS) を使用することができますが、必須ではありません。

外部認証局が AD CS の場合は、CSR に、IdM CA 証明書に必要なテンプレートを指定できます。証明書テンプレートは、証明書要求の受信時に CA が使用するポリシーおよびルールを定義します。AD の証明書テンプレートは、IdM の証明書プロファイルに対応します。

オブジェクト識別子 (OID) で特定の AD CS テンプレートを定義できます。OID は、分散アプリケーションのデータ要素、構文などを一意に識別するために、さまざまな発行機関が発行する一意の数値です。

または、特定の AD CS テンプレートを名前で定義することもできます。たとえば、IdM CA から AD CS に送信された CSR で使用されるデフォルトプロファイルの名前は **subCA** です。

CSR で OID または名前を指定してプロファイルを定義するには、**external-ca-profile** オプションを使用します。詳細は、**ipa-cacert-manage** man ページを参照してください。

既製の証明書テンプレートを使用する以外に、AD CS でカスタム証明書テンプレートを作成し、CSR で使用できます。

前提条件

- IdM CA 更新サーバーへの root のアクセス権がある。

手順

この手順では、現在の CA 証明書が自己署名の証明書であるか、外部署名であるかに関係なく、外部署名を使用して IdM CA の証明書を更新します。

1. 外部 CA に送信される CSR を作成します。

   - 外部 CA が AD CS の場合は、**--external-ca-type=ms-cs** オプションを使用します。デフォルトの **subCA** テンプレート以外のテンプレートが必要な場合は、**--external-ca-profile** を使用してこれを指定します。

     ```
     ~]# ipa-cacert-manage renew --external-ca --external-ca-type=ms-cs [--external-ca-profile=PROFILE]
     Exporting CA certificate signing request, please wait
     The next step is to get /var/lib/ipa/ca.csr signed by your CA and re-run ipa-cacert-manage as:
     ipa-cacert-manage renew --external-cert-file=/path/to/signed_certificate --external-cert-file=/path/to/external_ca_certificate
     The ipa-cacert-manage command was successful
     ```

   - 外部 CA が AD CS ではない場合は、以下のようになります。

     ```
     ~]# ipa-cacert-manage renew --external-ca
     Exporting CA certificate signing request, please wait
     The next step is to get /var/lib/ipa/ca.csr signed by your CA and re-run ipa-cacert-manage as:
     ipa-cacert-manage renew --external-cert-file=/path/to/signed_certificate --external-cert-file=/path/to/external_ca_certificate
     The ipa-cacert-manage command was successful
     ```

     この出力は、CSR が作成され、**/var/lib/ipa/ca.csr** ファイルに保存されていることを示しています。

2. **/var/lib/ipa/ca.csr** にある CSR を外部 CA に送信します。このプロセスは、外部 CA として使用するサービスにより異なります。

3. 発行された証明書と、発行元 CA の CA 証明書チェーンを Base 64 でエンコードされた Blob 形式で取得します。これは次のどちらかの形式です。

   - PEM ファイル (外部 CA が AD CS でない場合)

   - Base_64 証明書 (外部 CA が AD CS である場合)

プロセスは、各証明書サービスによって異なります。通常は Web ページか通知メールにダウンロードリンクがあり、管理者が、必要なすべての証明書をダウンロードできるようになっています。

外部 CA が AD CS で、Microsoft Windows 認証局管理ウィンドウから既知のテンプレートで CSR を送信した場合、AD CS は証明書を直ちに発行し、その証明書を保存する証明書の保存ダイアログが AD CS Web インターフェイスに表示されます。

4. **ipa-cacert-manage renew** コマンドを再度実行し、完全な証明書チェーンを提供するのに必要な CA 証明書ファイルをすべて追加します。**--external-cert-file** オプションを複数回使用して、必要な数だけファイルを指定します。

```
~]# ipa-cacert-manage renew --external-cert-file=/path/to/signed_certificate --external-cert-file=/path/to/external_ca_certificate_1 --external-cert-file=/path/to/external_ca_certificate_2
```

5. すべての IdM サーバーおよびクライアントで、サーバーの証明書でローカルの IdM 証明書データベースを更新します。

```
[client ~]$ ipa-certupdate
Systemwide CA database updated.
Systemwide CA database updated.
The ipa-certupdate command was successful
```

6. 必要に応じて、更新が成功したかどうかを確認して、新しい CA 証明書を **/etc/ipa/ca.crt** ファイルに追加します。

```
[client ~]$ openssl crl2pkcs7 -nocrl -certfile /etc/ipa/ca.crt | openssl pkcs7 -print_certs -text -noout
[...]
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 39 (0x27)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: O=IDM.EXAMPLE.COM, CN=Certificate Authority
        Validity
            Not Before: Jul  1 16:32:45 2019 GMT
            Not After : Jul  1 16:32:45 2039 GMT
        Subject: O=IDM.EXAMPLE.COM, CN=Certificate Authority
[...]
```

この出力は、新規 CA 証明書が古い CA 証明書と共にリストされているため、更新が正常に行われたことを示しています。

第13章 IDM がオフライン時に期限切れのシステム証明書の更新

システム証明書の期限が切れると、Identity Management (IdM) が起動できません。IdM は、**ipa-cert-fix** ツールを使用して、このような状況であってもシステム証明書の更新に対応します。

前提条件

- IdM が、Red Hat Enterprise Linux 8.1 以降にのみインストールされている。
- ホストで **ipactl start --ignore-service-failures** コマンドを入力して、LDAP サービスが実行中であることを確認します。

13.1. CA 更新サーバーでの期限切れのシステム証明書の更新

以下の手順に従って、期限切れの IdM 証明書に **ipa-cert-fix** ツールを適用します。

重要

CA 更新サーバーではない CA (認証局) ホストで **ipa-cert-fix** ツールを実行し、ユーティリティーが共有証明書を更新すると、そのホストは自動的にドメイン内の新しい CA 更新サーバーになります。不整合を避けるために、ドメインには常に CA 更新サーバー 1 つだけを設定する必要があります。

前提条件

- 管理者権限でサーバーにログインしている。

手順

1. (オプション) システムをバックアップします。これは、**ipa-cert-fix** が **nssdbs** に対して元に戻せない変更を行うため、強く推奨されます。**ipa-cert-fix** は LDAP に対しても変更を行うため、クラスター全体をバックアップすることも推奨されます。
2. **ipa-cert-fix** ツールを起動して、システムを分析し、更新を必要とする期限切れの証明書のリストを表示します。

```
# ipa-cert-fix
...
The following certificates will be renewed:

Dogtag sslserver certificate:
  Subject: CN=ca1.example.com,O=EXAMPLE.COM 201905222205
  Serial:  13
  Expires: 2019-05-12 05:55:47
...
Enter "yes" to proceed:
```

3. 更新プロセスを開始するには、**yes** を入力します。

```
Enter "yes" to proceed: true
Proceeding.
Renewed Dogtag sslserver certificate:
  Subject: CN=ca1.example.com,O=EXAMPLE.COM 201905222205
  Serial:  268369925
```

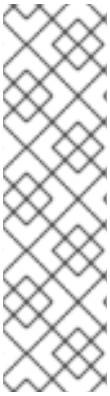
```
  Expires: 2021-08-14 02:19:33
...

Becoming renewal master.
The ipa-cert-fix command was successful
```

ipa-cert-fix が期限切れの証明書をすべて更新する前に、最大 1 分かかる場合があります。

4. 必要に応じて、すべてのサービスが現在実行していることを確認します。

```
# ipactl status
Directory Service: RUNNING
krb5kdc Service: RUNNING
kadmin Service: RUNNING
httpd Service: RUNNING
ipa-custodia Service: RUNNING
pki-tomcatd Service: RUNNING
ipa-otpd Service: RUNNING
ipa: INFO: The ipactl command was successful
```

この時点で、証明書が更新され、サービスが実行しています。次の手順は、IdM ドメイン内のその他のサーバーを確認します。



注記

複数の CA サーバーで証明書を修復する必要がある場合は、次のコマンドを実行します。

1. トポロジー全体で LDAP レプリケーションが機能していることを確認したら、上記の手順に従って、最初に 1 つの CA サーバーで **ipa-cert-fix** を実行します。
2. 別の CA サーバーで **ipa-cert-fix** を実行する前に、(別の CA サーバーの) **getcert-resubmit** を介して共有証明書の Certmonger 更新をトリガーして、共有証明書の不必要な更新を回避します。

13.2. 更新後の IDM ドメイン内の他の IDM サーバーの検証

ipa-cert-fix ツールで、CA 更新サーバーを更新したら、以下を行う必要があります。

- ドメインのその他の Identity Management (IdM) サーバーをすべて再起動する。
- certmonger が証明書を更新したかどうかを確認する。
- 期限切れのシステム証明書でその他の認証局 (CA) レプリカがある場合は、証明書も **ipa-cert-fix** ツールで更新する。

前提条件

- 管理者権限でサーバーにログインしている。

手順

1. **--force** パラメーターで IdM を再起動します。

```
# ipactl restart --force
```

--force パラメーターを使用すると、**ipactl** ユーティリティーは個々のサービスの起動失敗を無視します。たとえば、サーバーに期限切れの証明書を持つ CA もあると、**pki-tomcat** サービスが起動に失敗します。**--force** パラメーターを使用しているため、これが予想され、無視されます。

2. 再起動後に、**certmonger** サービスが証明書を更新することを確認します (証明書の状態は MONITORING になります)。

```
# getcert list | egrep '^Request|status:|subject:'
Request ID '20190522120745':
        status: MONITORING
        subject: CN=IPA RA,O=EXAMPLE.COM 201905222205
Request ID '20190522120834':
        status: MONITORING
        subject: CN=Certificate Authority,O=EXAMPLE.COM 201905222205
...
```

certmonger がレプリカ上で共有証明書を更新する前に時間がかかる場合があります。

3. サーバーも CA の場合、上記のコマンドは、**pki-tomcat** サービスが使用する証明書の **CA_UNREACHABLE** を報告します。

```
Request ID '20190522120835':
        status: CA_UNREACHABLE
        subject: CN=ca2.example.com,O=EXAMPLE.COM 201905222205
...
```

4. この証明書を更新するには、**ipa-cert-fix** ユーティリティーを使用します。

```
# ipa-cert-fix
Dogtag sslserver certificate:
  Subject: CN=ca2.example.com,O=EXAMPLE.COM
  Serial:  3
  Expires: 2019-05-11 12:07:11

Enter "yes" to proceed: true
Proceeding.
Renewed Dogtag sslserver certificate:
  Subject: CN=ca2.example.com,O=EXAMPLE.COM 201905222205
  Serial:  15
  Expires: 2019-08-14 04:25:05

The ipa-cert-fix command was successful
```

これで、すべての IdM 証明書が更新され、正常に機能するようになりました。

第14章 IDM レプリカでまだ有効期限が切れていない場合の WEB サーバーと LDAP サーバーの証明書の置き換え

Identity Management (IdM) システム管理者は、IdM サーバーで実行している Web (または **httpd**) サービスおよび LDAP (または **Directory**) サービスの証明書を手動で置き換えることができます。たとえば、証明書の有効期限が近づいていて、**certmonger** ユーティリティーが証明書を自動的に更新するように設定されていない場合、または証明書が外部の認証局 (CA) によって署名されている場合に、これが必要になることがあります。

この例では、**server.idm.example.com** IdM サーバーで実行しているサービスの証明書をインストールします。外部 CA から証明書を取得する。

注記

HTTP サービス証明書と LDAP サービス証明書には異なる IdM サーバーに異なるキーペアとサブジェクト名があるため、各 IdM サーバーで証明書を個別に更新する必要があります。

前提条件

- IdM サーバーが複製合意を持つトポロジー内の少なくとも 1 つの他の IdM レプリカで、Web および LDAP 証明書は引き続き有効です。これは **ipa-server-certinstall** コマンドの前提条件です。このコマンドは、他の IdM レプリカと通信するために **TLS** 接続を必要とします。ただし、証明書が無効な場合、そのような接続は確立できず、**ipa-server-certinstall** コマンドは失敗します。その場合は、Web サーバーと LDAP サーバーの証明書が IdM デプロイメント全体で期限切れになった場合の置き換え を参照してください。
- IdM サーバーへの **root** アクセス権限がある。
- **Directory Manager** パスワードを把握している。
- 外部 CA の CA 証明書チェーンを保存しているファイル (**ca_certificate_chain_file.crt**) にアクセスできる。

手順

1. **ca_certificate_chain_file.crt** に含まれる証明書を、追加の CA 証明書として IdM にインストールします。

```
# ipa-cacert-manage install
```

2. **ca_certicate_chain_file.crt** からの証明書を使用して、ローカルの IdM 証明書データベースを更新します。

```
# ipa-certupdate
```

3. **OpenSSL** ユーティリティーを使用して、秘密鍵と証明書署名要求 (CSR) を生成します。

```
$ openssl req -new -newkey rsa:4096 -days 365 -nodes -keyout new.key -out new.csr -addext "subjectAltName = DNS:server.idm.example.com" -subj '/CN=server.idm.example.com,O=IDM.EXAMPLE.COM'
```

CSR を外部 CA に送信します。このプロセスは、外部 CA として使用するサービスにより異なります。CA が証明書に署名したら、証明書を IdM サーバーにインポートします。

4. IdM サーバーで、Apache Web サーバーの古い秘密鍵および証明書を、新しい鍵と、新しく署名した証明書に置き換えます。

```
# ipa-server-certinstall -w --pin=password new.key new.crt
```

上記のコマンドでは、以下のようになります。

- **-w** オプションは、Web サーバーに証明書をインストールすることを指定します。
- **--pin** オプションは、秘密鍵を保護するパスワードを指定します。

5. プロンプトが表示されたら、**Directory Manager** パスワードを入力します。

6. LDAP サーバーの古い秘密鍵および証明書を、新しい鍵と、新しく署名した証明書に置き換えます。

```
# ipa-server-certinstall -d --pin=password new.key new.cert
```

上記のコマンドでは、以下のようになります。

- **-d** オプションは、LDAP サーバーに証明書をインストールすることを指定します。
- **--pin** オプションは、秘密鍵を保護するパスワードを指定します。

7. プロンプトが表示されたら、**Directory Manager** パスワードを入力します。

8. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

9. **Directory** サービスを再起動します。

```
# systemctl restart dirsrv@IDM.EXAMPLE.COM.service
```

10. サーバー上でサブ CA が削除または置き換えられている場合は、クライアントを更新します。

```
# ipa-certupdate
```

関連情報

- IdM と機能する証明書形式への変換
- **ipa-server-certinstall(1)** の man ページ

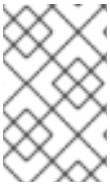
第15章 IDM デプロイメント全体で期限切れになった WEB サーバーと LDAP サーバーの証明書を置き換える

Identity Management (IdM) は、以下のサービス証明書を使用します。

- LDAP (または **Directory**) サーバー証明書
- Web (または **httpd**) サーバー証明書
- PKINIT 証明書

CA を使用しない IdM デプロイメントでは、**certmonger** はデフォルトで IdM サービス証明書を追跡したり、その有効期限を通知したりしません。IdM システム管理者がこれらの証明書の通知を手動で設定しない場合、または証明書を追跡するように **certmonger** を設定しない場合、証明書は予告なしに期限切れになります。

以下の手順に従って、**server.idm.example.com** IdM サーバーで実行している **httpd** および LDAP サービスの期限切れの証明書を手動で置き換えます。



注記

HTTP および LDAP サービス証明書は、異なる IdM サーバーで異なるキーペアとサブジェクト名を持ちます。したがって、各 IdM サーバーで個別に証明書を更新する必要があります。

前提条件

- トポロジー内の **すべての** IdM レプリカで、HTTP および LDAP 証明書の有効期限が切れています。そうでない場合は、Web サーバーと LDAP サーバーの証明書が IdM レプリカで期限切れになっていない場合の置き換え を参照してください。
- IdM サーバーとレプリカへの **root** アクセス権がある。
- **Directory Manager** パスワードを把握している。
- 次のディレクトリーとファイルのバックアップを作成しました:
  - **/etc/dirsrv/slapd-IDM-EXAMPLE-COM/**
  - **/etc/httpd/alias**
  - **/var/lib/certmonger**
  - **/var/lib/ipa/certs/**

手順

1. (オプション) **/var/lib/ipa/private** と **/var/lib/ipa/passwds** のバックアップを実行します。
2. 新しい証明書の署名に同じ CA を使用していない場合、またはすでにインストールされている CA 証明書が無効になっている場合は、ローカルデータベース内の外部 CA に関する情報を、外部 CA の有効な CA 証明書チェーンを含むファイルで更新します。.このファイルは、PEM および DER 証明書、PKCS#7 証明書チェーン、PKCS#8、生の秘密鍵、および PKCS#12 形式で受け入れられます。
   a. **ca_certificate_chain_file.crt** で利用可能な証明書を追加の CA 証明書として IdM にインス

トールします。

```
# ipa-cacert-manage install ca_certificate_chain_file.crt
```

b. **ca_certicate_chain_file.crt** からの証明書を使用して、ローカルの IdM 証明書データベースを更新します。

```
# ipa-certupdate
```

3. **httpd** および LDAP の証明書を要求します。

a. **OpenSSL** ユーティリティーを使用して、IdM インスタンスで実行している Apache Web サーバーの証明書署名要求 (CSR) をサードパーティー CA に作成します。

- 新しい秘密鍵の作成は任意です。元の秘密鍵がまだある場合は、**openssl req** コマンドで **-in** オプションを使用して、要求を読み取る入力ファイル名を指定できます。

```
$ openssl req -new -nodes -in /var/lib/ipa/private/httpd.key -out /tmp/http.csr -
addext 'subjectAltName = DNS:_server.idm.example.com_,
otherName:1.3.6.1.4.1.311.20.2.3;UTF8:HTTP/server.idm.example.com@IDM.EX
AMPLE.COM' -subj '/O=IDM.EXAMPLE.COM/CN=server.idm.example.com'
```

- 新しいキーを作成する場合:

```
$ openssl req -new -newkey rsa:2048 -nodes -keyout
/var/lib/ipa/private/httpd.key -out /tmp/http.csr -addext 'subjectAltName =
DNS:server.idm.example.com,
otherName:1.3.6.1.4.1.311.20.2.3;UTF8:HTTP/server.idm.example.com@IDM.EX
AMPLE.COM' -subj '/O=IDM.EXAMPLE.COM/CN=server.idm.example.com'
```

b. **OpenSSL** ユーティリティーを使用して、IdM インスタンスで実行している LDAP サーバーの証明書署名要求 (CSR) をサードパーティー CA に作成します。

```
$ openssl req -new -newkey rsa:2048 -nodes -keyout ~/ldap.key -out /tmp/ldap.csr -
addext 'subjectAltName = DNS:server.idm.example.com,
otherName:1.3.6.1.4.1.311.20.2.3;UTF8:ldap/server.idm.example.com@IDM.EXAMP
LE.COM' -subj '/O=IDM.EXAMPLE.COM/CN=server.idm.example.com'
```

c. CSR、**/tmp/http.csr** および **tmp/ldap.csr** を外部 CA に送信し、**httpd** の証明書と LDAP の証明書を取得します。このプロセスは、外部 CA として使用するサービスにより異なります。

4. **httpd** の証明書をインストールします。

```
# cp /path/to/httpd.crt /var/lib/ipa/certs/
```

5. LDAP 証明書を NSS データベースにインストールします。

a. [オプション] 利用可能な証明書を一覧表示します。

```
# certutil -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM/ -L
Certificate Nickname                                         Trust Attributes
                                                             SSL,S/MIME,JAR/XPI
```

```
Server-Cert                                    u,u,u
```

デフォルトの証明書のニックネームは **Server-Cert** ですが、別の名前が適用された可能性があります。

b. 前の手順の証明書のニックネームを使用して、古い無効な証明書を NSS データベース (**NSSDB**) から削除します。

```
# certutil -D -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM/ -n 'Server-Cert' -f
/etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt
```

c. **NSSDB** へのインポートプロセスを容易にするために、PKCS12 ファイルを作成します。

```
# openssl pkcs12 -export -in ldap.crt -inkey ldap.key -out ldap.p12 -name Server-
Cert
```

d. 作成した PKCS#12 ファイルを **NSSDB** にインストールします。

```
# pk12util -i ldap.p12 -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM/ -k
/etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt
```

e. 新しい証明書が正常にインポートされたことを確認します。

```
# certutil -L -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM/
```

6. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

7. **Directory** サービスを再起動します。

```
# systemctl restart dirsrv@IDM-EXAMPLE-COM.service
```

8. すべての IdM レプリカで前のすべての手順を実行します。これは、レプリカ間の **TLS** 接続を確立するための前提条件です。

9. 新しい証明書を LDAP ストレージに登録します。

a. Apache サーバーの古い秘密鍵および証明書を、新しい鍵と、新しく署名した証明書に置き換えます。

```
# ipa-server-certinstall -w --pin=password /var/lib/ipa/private/httpd.key
/var/lib/ipa/certs/httpd.crt
```

上記のコマンドでは、以下のようになります。

- **-w** オプションは、Web サーバーに証明書をインストールすることを指定します。
- **--pin** オプションは、秘密鍵を保護するパスワードを指定します。

b. プロンプトが表示されたら、**Directory Manager** パスワードを入力します。

c. LDAP サーバーの古い秘密鍵および証明書を、新しい鍵と、新しく署名した証明書に置き換えます。

```
# ipa-server-certinstall -d --pin=password /etc/dirsrv/slapd-IDM-EXAMPLE-COM/ldap.key /path/to/ldap.crt
```

上記のコマンドでは、以下のようになります。

- **-d** オプションは、LDAP サーバーに証明書をインストールすることを指定します。
- **--pin** オプションは、秘密鍵を保護するパスワードを指定します。

d. プロンプトが表示されたら、**Directory Manager** パスワードを入力します。

e. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

f. **Directory** サービスを再起動します。

```
# systemctl restart dirsrv@IDM-EXAMPLE-COM.service
```

10. 影響を受ける他のすべてのレプリカで、前の手順のコマンドを実行します。

関連情報

IdM で動作するように証明書形式を変換する * man **ipa-server-certinstall (1)** * How do I manually renew Identity Management (IPA) certificates on RHEL 8 after they have expired?(CA のない IPA)

第16章 IDM CA サーバーでの CRL の生成

IdM デプロイメントで埋め込み認証局 (CA) を使用する場合は、証明書失効リスト (CRL) の生成を 1 つの Identity Management (IdM) サーバーから別のサーバーに移動する必要になることがあります。たとえば、サーバーを別のシステムに移行する場合に必要になる場合があります。

CRL を生成するサーバーは 1 台だけ設定します。CRL パブリッシャーロールを実行する IdM サーバーは通常、CA 更新サーバーロールを実行するサーバーと同じですが、この設定は必須ではありません。CRL パブリッシャーサーバーの使用を停止する前に、別のサーバーを選択して CRL パブリッシャーサーバーロールを実行するように設定します。

16.1. IDM サーバーでの CRL 生成の停止

IdM CRL パブリッシャーサーバーで証明書失効リスト (CRL) の生成を停止するには、**ipa-crlgen-manage** コマンドを使用します。生成を無効にする前に、サーバーで実際に CRL が生成されることを確認してください。その後、無効にすることができます。

前提条件

- Identity Management (IdM) サーバーが、RHEL 8.1 システム以降にインストールされている。
- root としてログインしている。

手順

1. サーバーが CRL を生成しているかどうかを確認します。

```
[root@server ~]# ipa-crlgen-manage status
CRL generation: enabled
Last CRL update: 2019-10-31 12:00:00
Last CRL Number: 6
The ipa-crlgen-manage command was successful
```

2. サーバーで CRL の生成を停止します。

```
[root@server ~]# ipa-crlgen-manage disable
Stopping pki-tomcatd
Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
Starting pki-tomcatd
Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
Restarting httpd
CRL generation disabled on the local host. Please make sure to configure CRL generation on another master with ipa-crlgen-manage enable.
The ipa-crlgen-manage command was successful
```

3. サーバーが CRL の生成を停止したかどうかを確認します。

```
[root@server ~]# ipa-crlgen-manage status
```

サーバーで CRL の生成が停止されました。次の手順は、IdM レプリカで CRL 生成を有効にすることです。

16.2. IDM レプリカサーバーでの CRL 生成の開始

ipa-crlgen-manage コマンドを使用して、IdM CA サーバーで証明書失効リスト (CRL) の生成を開始できます。

前提条件

- Identity Management (IdM) サーバーが、RHEL 8.1 システム以降にインストールされている。
- RHEL システムは、IdM 認証局サーバーが必要である。
- root としてログインしている。

手順

1. CRL の生成を開始します。

```
[root@replica1 ~]# ipa-crlgen-manage enable
Stopping pki-tomcatd
Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
Starting pki-tomcatd
Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
Restarting httpd
Forcing CRL update
CRL generation enabled on the local host. Please make sure to have only a single CRL generation master.
The ipa-crlgen-manage command was successful
```

2. CRL が生成されているかどうかを確認します。

```
[root@replica1 ~]# ipa-crlgen-manage status
CRL generation: enabled
Last CRL update: 2019-10-31 12:10:00
Last CRL Number: 7
The ipa-crlgen-manage command was successful
```

16.3. CRL 更新間隔の変更

証明書失効リスト (CRL) ファイルは、デフォルトでは、Identity Management Certificate Authority (Idm CA) によって 4 時間ごとに自動的に生成されます。この間隔は以下の手順で変更できます。

手順

1. CRL 生成サーバーを停止します。

```
# systemctl stop pki-tomcatd@pki-tomcat.service
```

2. **/var/lib/pki/pki-tomcat/conf/ca/CS.cfg** ファイルを開き、**ca.crl.MasterCRL.autoUpdateInterval** の値を新しい間隔設定に変更します。たとえば、60 分ごとに CRL を生成するには、次のコマンドを実行します。

```
ca.crl.MasterCRL.autoUpdateInterval=60
```

注記

ca.crl.MasterCRL.autoUpdateInterval パラメーターを更新すると、すでにスケジュールされている CRL の次回更新後に変更が有効になります。

3. CRL 生成サーバーを起動します。

```
# systemctl start pki-tomcatd@pki-tomcat.service
```

関連情報

- IdM レプリカサーバーでの CRL 生成の詳細は、IdM レプリカサーバーでの CRL 生成の開始 を参照してください。

第17章 CA 更新サーバーと CRL パブリッシャーのロールを実行するサーバーの使用の停止

認証局 (CA) 更新サーバーロールおよび 証明書失効リスト (CRL) パブリッシャーロールの両方を実行しているサーバーが 1 台あるとします。このサーバーをオフラインにするか、使用を停止する必要がある場合は、別の CA サーバーを選択して、このロールを実行するように設定します。

この例では、CA 更新サーバーと CRL パブリッシャーのロールに対応しているホスト **server.idm.example.com** の使用を停止する必要があります。この手順では、CA 更新サーバーロールおよび CRL パブリッシャーロールをホスト **replica.idm.example.com** に転送し、IdM 環境から **server.idm.example.com** を削除します。

注記

CA 更新サーバーと CRL パブリッシャーの両方のロールを実行するために同じサーバーを設定する必要はありません。

前提条件

- IdM 管理者認証情報がある。
- 使用を停止しているサーバーの root パスワードがある。
- IdM 環境に、少なくとも 2 つの CA レプリカがある。

手順

1. IdM 管理者認証情報を取得します。

```
[user@server ~]$ kinit admin
Password for admin@IDM.EXAMPLE.COM:
```

2. **(オプション)** どのサーバーが CA 更新サーバーおよび CRL パブリッシャーのロールを実行しているかわからない場合は、以下を実行します。

   a. 現在の CA 更新サーバーを表示します。次のコマンドは、任意の IdM サーバーから実行できます。

```
[user@server ~]$ ipa config-show | grep 'CA renewal'
  IPA CA renewal master: server.idm.example.com
```

   b. ホストが現在の CRL パブリッシャーであるかどうかをテストします。

```
[user@server ~]$ ipa-crlgen-manage status
CRL generation: enabled
Last CRL update: 2019-10-31 12:00:00
Last CRL Number: 6
The ipa-crlgen-manage command was successful
```

   CRL を生成しない CA サーバーは、**CRL generation: disabled** を表示します。

```
[user@replica ~]$ ipa-crlgen-manage status
CRL generation: disabled
The ipa-crlgen-manage command was successful
```

CRL パブリッシャーサーバーが見つかるまで、CA サーバーでこのコマンドを入力し続けます。

c. このロールに対応するために昇格できるその他の CA サーバーをすべて表示します。この環境には 2 台の CA サーバーがあります。

```
[user@server ~]$ ipa server-role-find --role 'CA server'
----------------------
2 server roles matched
----------------------
  Server name: server.idm.example.com
  Role name: CA server
  Role status: enabled
  Server name: replica.idm.example.com
  Role name: CA server
  Role status: enabled
----------------------------
Number of entries returned 2
----------------------------
```

3. **replica.idm.example.com** を CA 更新サーバーとして設定します。

```
[user@server ~]$ ipa config-mod --ca-renewal-master-server replica.idm.example.com
```

4. **server.idm.example.com** で、以下を実行します。

a. 証明書更新タスクを無効にします。

```
[root@server ~]# pki-server ca-config-set ca.certStatusUpdateInterval 0
```

b. IdM サービスを再起動します。

```
[root@server ~]# ipactl restart
```

5. **replica.idm.example.com** で、以下を実行します。

a. 証明書更新タスクを有効にします。

```
[root@replica ~]# pki-server ca-config-unset ca.certStatusUpdateInterval
```

b. IdM サービスを再起動します。

```
[root@replica ~]# ipactl restart
```

6. **server.idm.example.com** で、CRL の生成を中止します。

```
[user@server ~]$ ipa-crlgen-manage disable
Stopping pki-tomcatd
Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
```

```
Starting pki-tomcatd
Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
Restarting httpd
CRL generation disabled on the local host. Please make sure to configure CRL generation on
another master with ipa-crlgen-manage enable.
The ipa-crlgen-manage command was successful
```

7. **replica.idm.example.com** で、CRL の生成を開始します。

```
[user@replica ~]$ ipa-crlgen-manage enable
Stopping pki-tomcatd
Editing /var/lib/pki/pki-tomcat/conf/ca/CS.cfg
Starting pki-tomcatd
Editing /etc/httpd/conf.d/ipa-pki-proxy.conf
Restarting httpd
Forcing CRL update
CRL generation enabled on the local host. Please make sure to have only a single CRL
generation master.
The ipa-crlgen-manage command was successful
```

8. **server.idm.example.com** で IdM サービスを停止します。

```
[root@server ~]# ipactl stop
```

9. **replica.idm.example.com** で、IdM 環境から **server.idm.example.com** を削除します。

```
[user@replica ~]$ ipa server-del server.idm.example.com
```

10. **server.idm.example.com** で、**ipa-server-install --uninstall** コマンドを root アカウントとして使用します。

```
[root@server ~]# ipa-server-install --uninstall
...
Are you sure you want to continue with the uninstall procedure? [no]: yes
```

検証手順

- 現在の CA 更新サーバーを表示します。

```
[user@replica ~]$ ipa config-show | grep 'CA renewal'
  IPA CA renewal master: replica.idm.example.com
```

- **replica.idm.example.com** ホストが CRL を生成していることを確認します。

```
[user@replica ~]$ ipa-crlgen-manage status
CRL generation: enabled
Last CRL update: 2019-10-31 12:10:00
Last CRL Number: 7
The ipa-crlgen-manage command was successful
```

関連情報

- IdM CA 更新サーバーの変更およびリセット

- IdM CA サーバーでの CRL の生成
- IdM レプリカのアンインストール

第18章 CERTMONGER を使用したサービスの IDM 証明書の取得

18.1. CERTMONGER の概要

Identity Management (IdM) が統合 IdM 認証局 (CA) とともにインストールされていると、**certmonger** サービスを使用してシステムおよびサービス証明書を追跡し、更新します。証明書の期限が切れると、**certmonger** は次の方法で更新プロセスを管理します。

- 元の要求で指定されているオプションを使用して、証明書署名要求 (CSR) を再生成します。
- IdM API の **cert-request** コマンドを使用して、CSR を IdM CA に送信します。
- IdM CA から証明書を受け取ります。
- pre-save コマンドを実行します (元の要求で指定されている場合)。
- 更新要求で指定されている場所 (**NSS** データベースまたはファイル) に新しい証明書をインストールします。
- post-save コマンドを実行します (元の要求で指定されている場合)。たとえば、post-save コマンドは、関連するサービスを再起動するように **certmonger** に指示し、サービスが新しい証明書が取得できるようにします。

certmonger が追跡する証明書の種類

証明書は、システム証明書とサービス証明書に分けることができます。

さまざまなサーバーのさまざまなキーペアと発行名を持つサービス証明書 (**HTTP**、**LDAP**、**PKINIT** など) とは異なり、IdM システム証明書とその鍵はすべての CA レプリカで共有されます。IdM システム証明書には、以下が含まれます。

- **IdM CA** 認証
- **OCSP** 署名証明書
- **IdM CA サブシステム** 証明書
- **IdM CA 監査署名** 証明書
- **IdM 更新エージェント** (RA) 証明書
- **KRA** トランスポート証明書およびストレージ証明書

certmonger サービスは、統合 CA を使用した IdM 環境のインストール時に要求された IdM システム証明書およびサービス証明書を追跡します。**certmonger** は、IdM ホストで実行しているその他のサービスに対して、システム管理者が手動で要求した証明書を追跡します。**Certmonger** では、外部 CA 証明書またはユーザー証明書は追跡されません。

Certmonger のコンポーネント

certmonger サービスは、2 つの主要コンポーネントで設定されています。

- **certmonger デーモン** - 証明書のリストを追跡し、更新コマンドを実行するエンジンです。
- **コマンドラインインターフェイス** (CLI) の **getcert** ユーティリティー。これにより、システム管理者は **certmonger** デーモンにアクティブにコマンドを送信できます。

具体的には、システム管理者は、**getcert** ユーティリティーを使用して以下のことができます。

- 新しい証明書を要求する
- **certmonger** が追跡する証明書のリストを表示する
- 証明書の追跡を開始または停止する
- 証明書を更新する

18.2. CERTMONGER を使用したサービスの IDM 証明書の取得

ブラウザーと、Identity Management (IdM) クライアントで実行している Web サービスとの間の通信が安全で暗号化されていることを確認するには、TLS 証明書を使用します。IdM 認証局 (CA) から Web サービスの TLS 証明書を取得します。

以下の手順に従って、**certmonger** を使用して、IdM クライアントで実行しているサービス (**HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM**) の IdM 証明書を取得します。

certmonger 使用して証明書を自動的に要求するということは、**certmonger** 更新の期限が切れたときに証明書を管理および更新することを意味します。

certmonger がサービス証明書を要求したときの動作の視覚的な表現については、サービス証明書を要求する certmonger の通信フロー を参照してください。

前提条件

- Web サーバーが、IdM クライアントとして登録されている。
- 手順を実行している IdM クライアントへのルートアクセス権限がある。
- 証明書を要求しているサービスは、前もって IdM に用意する必要はない。

手順

1. **HTTP** サービスが稼働している IdM クライアント **my_company.idm.example.com** で、以下を指定する **HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM** プリンシパルに対応するサービスの証明書を要求します。
   - 証明書は、ローカルの **/etc/pki/tls/certs/httpd.pem** ファイルに保存されます。
   - 秘密鍵は、ローカルの **/etc/pki/tls/private/httpd.key** ファイルに保存されます。
   - **SubjectAltName** の extensionRequest が、 **my_company.idm.example.com** の DNS 名の署名要求に追加されます。

     ```
     # ipa-getcert request -K HTTP/my_company.idm.example.com -k
     /etc/pki/tls/private/httpd.key -f /etc/pki/tls/certs/httpd.pem -g 2048 -D
     my_company.idm.example.com -C "systemctl restart httpd"
     New signing request "20190604065735" added.
     ```

     上記のコマンドでは、以下のようになります。

     - **ipa-getcert request** コマンドは、証明書が IdM CA から取得することを示しています。**ipa-getcert request** コマンドは、**getcert request -c IPA** のショートカットです。

- **-g** オプションは、生成先のキーのサイズ (設定されていない場合) を指定します。
- **-D** オプションは、要求に追加する DNS 値 **SubjectAltName** を指定します。
- **-C** オプションは、証明書の取得後に **httpd** サービスを再起動するように **certmonger** に指示します。
- 特定のプロファイルで証明書を発行するように指定する場合は、**-T** オプションを使用します。
- 指定した CA から名前付き発行者を使用して証明書を要求するには、**-X ISSUER** オプションを使用します。

注記

RHEL 8 は、RHEL 7 で使用されるものとは異なる SSL モジュール (Apache) を使用します。SSL モジュールは、NSS ではなく OpenSSL に依存しています。このため、RHEL 8 では、NSS データベースを使用して **HTTPS** 証明書と秘密鍵を保存することができません。

2. 必要に応じて、リクエストの状況を確認するには、次のコマンドを実行します。

```
# ipa-getcert list -f /etc/pki/tls/certs/httpd.pem
Number of certificates and requests being tracked: 3.
Request ID '20190604065735':
    status: MONITORING
    stuck: no
    key pair storage: type=FILE,location='/etc/pki/tls/private/httpd.key'
    certificate: type=FILE,location='/etc/pki/tls/certs/httpd.crt'
    CA: IPA
[...]
```

この出力は、要求が **MONITORING** 状況であることを表しています。これは、証明書が取得されていることを示しています。キーペアと証明書の場所は、要求された場所です。

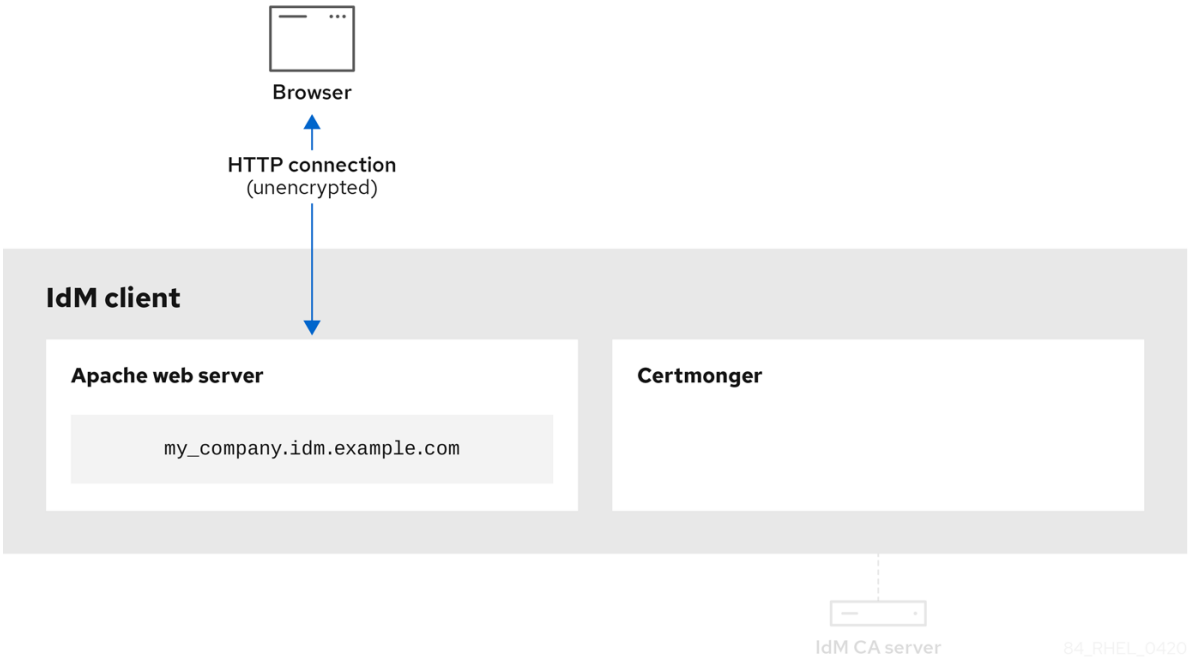
18.3. サービス証明書を要求する CERTMONGER の通信フロー

これらの図では、**certmonger** が Identity Management (IdM) 認証局 (CA) サーバーからサービス証明書を要求したときに何が起こるかを段階をおって紹介しています。シーケンスは次の図で設定されています。

- 暗号化されていない通信
- サービス証明書を要求する certmonger
- サービス証明書を発行する IdM CA
- Certmonger によるサービス証明書の適用
- 古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger

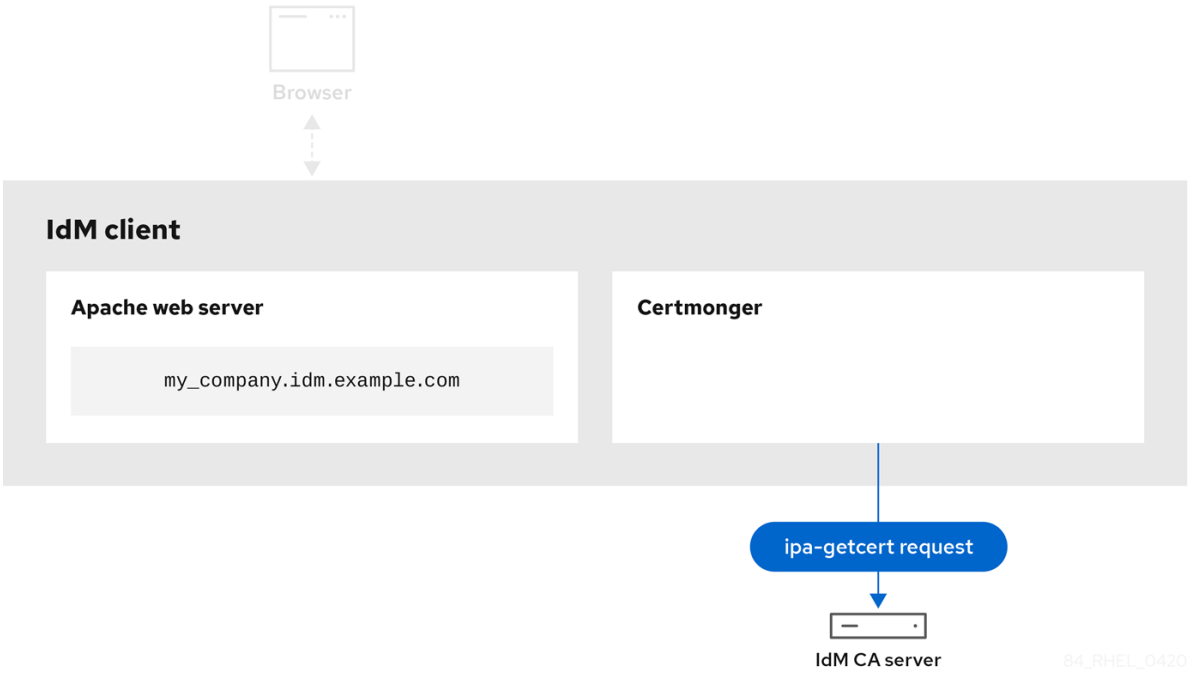
暗号化されていない通信 は、初期状態を示しています。HTTPS 証明書がないと、Web サーバーとブラウザー間の通信は暗号化されません。

図18.1 暗号化されていない通信



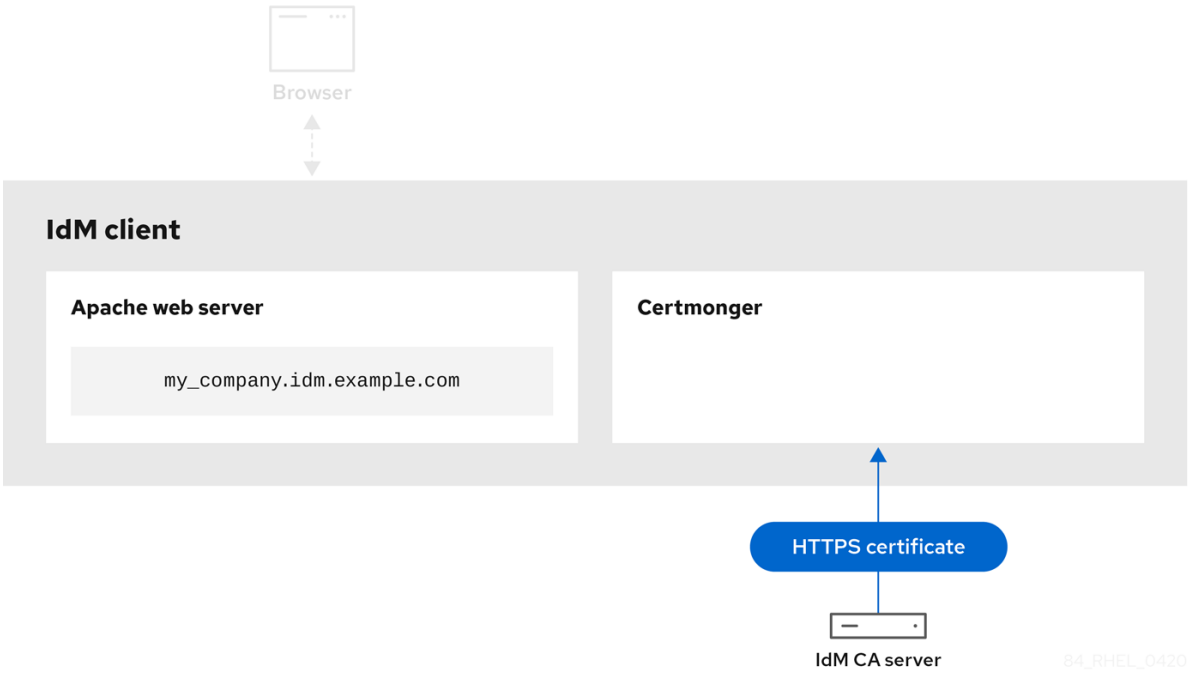
サービス証明書を要求する certmonger は、システム管理者が **certmonger** を使用して Apache Web サーバーの HTTPS 証明書を手動で要求していることを示しています。Web サーバー証明書を要求する場合、certmonger は CA と直接対話しないことに注意してください。IdM 経由でプロキシーが設定されます。

図18.2 サービス証明書を要求する certmonger



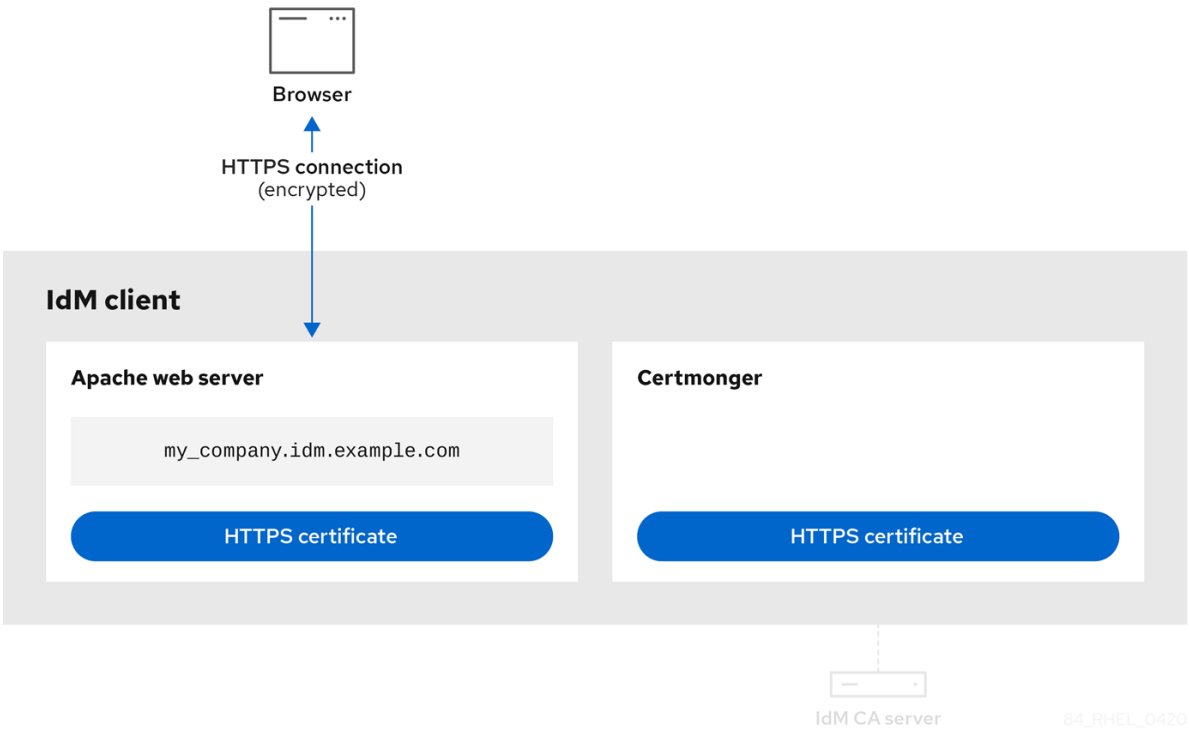
サービス証明書を発行する IdM CA は、Web サーバーで HTTPS 証明書を発行する IdM CA を示しています。

図18.3 サービス証明書を発行する IdM CA



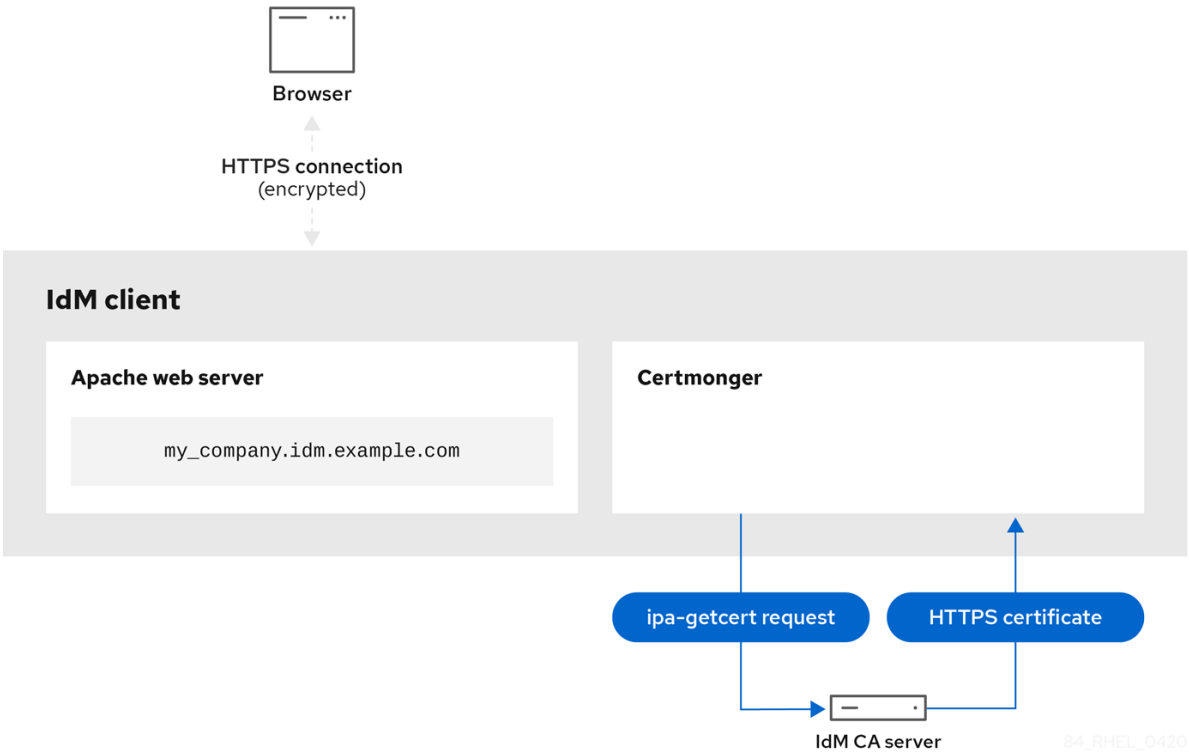
Certmonger によるサービス証明書の適用 は、**certmonger** が HTTPS 証明書を Id M クライアントの適切な場所に配置し、指示された場合は **httpd** サービスを再起動することを示します。その後、Apache サーバーは HTTPS 証明書を使用して、Apache サーバーとブラウザー間のトラフィックを暗号化します。

図18.4 Certmonger によるサービス証明書の適用



古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger は、証明書の有効期限が切れる前に、**certmonger** が Id MCA からのサービス証明書の更新を自動的に要求していることを示しています。IdM CA は、新しい証明書を発行します。

図18.5 古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger



18.4. CERTMONGER が追跡する証明書要求の詳細を表示

certmonger サービスは、証明書要求を監視します。証明書要求が正常に署名されると、証明書が作成されます。**certmonger** は、生成される証明書を含む証明書要求を管理します。以下の手順に従って、**certmonger** が管理する特定の証明書要求の詳細を表示します。

手順

- 証明書要求の指定方法が分からない場合は、特定の証明書要求のみの詳細をリスト表示します。たとえば、次を指定できます。
  - リクエスト ID
  - 証明書の場所
  - 証明書のニックネーム
    たとえば、要求 ID が 20190408143846 である証明書の詳細を表示するには、**-v** オプションを使用して、証明書のリクエストが失敗した場合にエラーの詳細をすべて表示します。

```
# getcert list -i 20190408143846 -v
Number of certificates and requests being tracked: 16.
Request ID '20190408143846':
 status: MONITORING
 stuck: no
 key pair storage: type=NSSDB,location='/etc/dirsrv/slapd-IDM-EXAMPLE-
```

```
COM',nickname='Server-Cert',token='NSS Certificate DB',pinfile='/etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt'
 certificate: type=NSSDB,location='/etc/dirsrv/slapd-IDM-EXAMPLE-COM',nickname='Server-Cert',token='NSS Certificate DB'
 CA: IPA
 issuer: CN=Certificate Authority,O=IDM.EXAMPLE.COM
 subject: CN=r8server.idm.example.com,O=IDM.EXAMPLE.COM
 expires: 2021-04-08 16:38:47 CEST
 dns: r8server.idm.example.com
 principal name: ldap/server.idm.example.com@IDM.EXAMPLE.COM
 key usage: digitalSignature,nonRepudiation,keyEncipherment,dataEncipherment
 eku: id-kp-serverAuth,id-kp-clientAuth
 pre-save command:
 post-save command: /usr/libexec/ipa/certmonger/restart_dirsrv IDM-EXAMPLE-COM
 track: true
 auto-renew: true
```

この出力では、証明書に関する情報の一部が表示されます。以下に例を示します。

- 証明書の場所 - 上記の例では、**/etc/dirsrv/slapd-IDM-EXAMPLE-COM** ディレクトリーの NSS データベースです。
- 証明書のニックネーム - 上記の例では **Server-Cert** になります。
- ピンを保存しているファイル - 上記の例では **/etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt** になります。
- 証明書の更新に使用される認証局 (CA) - 上記の例では **IPA** CA です。
- 有効期限 - 上記の例では **2021-04-08 16:38:47 CEST** になります。
- 証明書のステータス - 上記の例の **MONITORING** 状態は、証明書が有効であり、追跡されていることを意味します。
- post-save コマンド - 上記の例では、**LDAP** サービスの再起動です。

- 証明書要求の指定方法が分からない場合は、**certmonger** が監視または取得しようとしているすべての証明書の詳細をリスト表示します。

```
# getcert list
```

関連情報

- **getcert list** man ページを参照してください。

18.5. 証明書追跡の開始および停止

以下の手順に従って、**getcert stop-tracking** コマンドおよび **getcert start-tracking** コマンドを使用して証明書を監視します。この 2 つのコマンドは、**certmonger** サービスで利用できます。証明書追跡を有効にすると、Identity Management (IdM) 認証局 (CA) が発行する証明書を、別の IdM クライアントのマシンにインポートすると特に便利です。証明書の追跡を有効にすることは、次のプロビジョニングシナリオの最終ステップでもあります。

1. IdM サーバーでは、存在していないシステムの証明書を作成する。

2. 新しいシステムを作成する。
3. 新しいシステムを IdM クライアントとして登録する。
4. IdM クライアントの IdM サーバーから、証明書および鍵をインポートする。
5. **certmonger** を使用して証明書の追跡を開始し、有効期限が切れる時に証明書を更新するようにする。

手順

- 要求 ID が 20190408143846 の証明書の監視を無効にするには、次のコマンドを実行します。

```
# getcert stop-tracking -i 20190408143846
```

その他のオプションは、**getcert stop-tracking** man ページを参照してください。

- **/tmp/some_cert.crt** ファイルに保存されている証明書の監視を有効にするには、次のコマンドを実行します。秘密鍵が **/tmp/some_key.key** ファイルに保存されます。

```
# getcert start-tracking -c IPA -f /tmp/some_cert.crt -k /tmp/some_key.key
```

certmonger は、証明書を発行した CA タイプを自動的に特定できません。そのため、IdM CA が証明書を発行した場合は、**getcert start-tracking** コマンドに **IPA** 値を付けて **-c** オプションを追加します。**-c** オプションを追加しないと、**certmonger** が NEED_CA 状態になります。

その他のオプションは、**getcert start-tracking** man ページを参照してください。

注記

この 2 つのコマンドは証明書を操作しません。たとえば、**getcert stop-tracking** は、証明書を削除しなかったり、NSS データベースまたはファイルシステムから削除したりせず、監視する証明書のリストから証明書を削除します。同様に、**getcert start-tracking** は、監視された証明書のリストに証明書のみを追加します。

18.6. 証明書を手動で更新

証明書が失効日近くになると、**certmonger** デーモンは、認証局 (CA) ヘルパーを使用して更新コマンドを自動的に発行し、更新された証明書を取得し、以前の証明書を新しい証明書に置き換えます。

getcert resubmit コマンドを使用して、事前に証明書を手動で更新することもできます。これにより、SAN (Subject Alternative Name) を追加したりすると、証明書に含まれる情報を更新できます。

次の手順に従って、証明書を手動で更新します。

手順

- リクエスト ID が 20190408143846 の証明書を更新するには、次のコマンドを実行します。

```
# getcert resubmit -i 20190408143846
```

特定の証明書の要求 ID を取得するには、**getcert list** コマンドを使用します。詳細は、**getcert list** man ページを参照してください。

18.7. CERTMONGER が CA レプリカでの IDM 証明書の追跡を再開

この手順は、証明書の追跡が中断された後、**certmonger** が統合認証局で Identity Management (IdM) デプロイメントに不可欠な IdM システム証明書の追跡を再開する方法を説明します。中断は、システム証明書の更新中に IdM ホストが IdM から登録解除されたか、レプリケーショントポロジーが正しく機能しないことが原因である可能性があります。この手順では、**certmonger** が IdM サービス証明書 (**HTTP**、**LDAP**、**PKINIT**) の追跡を再開する方法も説明します。

前提条件

- システム証明書の追跡を再開するホストは、IdM 認証局 (CA) でもある IdM サーバーですが、IdM CA 更新サーバーではありません。

手順

1. サブシステムの CA 証明書の PIN を取得します。

```
# grep 'internal=' /var/lib/pki/pki-tomcat/conf/password.conf
```

2. サブシステムの CA 証明書に追跡を追加します。次のコマンドの **[internal PIN]** を、直前の手順で取得した PIN に置き換えます。

```
# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "caSigningCert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"caSigningCert cert-pki-ca"' -T caCACert

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "auditSigningCert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"auditSigningCert cert-pki-ca"' -T caSignedLogCert

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "ocspSigningCert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"ocspSigningCert cert-pki-ca"' -T caOCSPCert

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "subsystemCert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"subsystemCert cert-pki-ca"' -T caSubsystemCert

# getcert start-tracking -d /etc/pki/pki-tomcat/alias -n "Server-Cert cert-pki-ca" -c
'dogtag-ipa-ca-renew-agent' -P [internal PIN] -B
/usr/libexec/ipa/certmonger/stop_pkicad -C '/usr/libexec/ipa/certmonger/renew_ca_cert
"Server-Cert cert-pki-ca"' -T caServerCert
```

3. 残りの IdM 証明書、**HTTP** 証明書、**LDAP** 証明書、**IPA 更新エージェント** 証明書、および **PKINIT** 証明書の追跡を追加します。

```
# getcert start-tracking -f /var/lib/ipa/certs/httpd.crt -k /var/lib/ipa/private/httpd.key -p
/var/lib/ipa/passwds/idm.example.com-443-RSA -c IPA -C
/usr/libexec/ipa/certmonger/restart_httpd -T calPAserviceCert

# getcert start-tracking -d /etc/dirsrv/slapd-IDM-EXAMPLE-COM -n "Server-Cert" -c IPA
```

```
-p /etc/dirsrv/slapd-IDM-EXAMPLE-COM/pwdfile.txt -C
'/usr/libexec/ipa/certmonger/restart_dirsrv "IDM-EXAMPLE-COM"' -T calPAserviceCert

# getcert start-tracking -f /var/lib/ipa/ra-agent.pem -k /var/lib/ipa/ra-agent.key -c
dogtag-ipa-ca-renew-agent -B /usr/libexec/ipa/certmonger/renew_ra_cert_pre -C
/usr/libexec/ipa/certmonger/renew_ra_cert -T caSubsystemCert

# getcert start-tracking -f /var/kerberos/krb5kdc/kdc.crt -k
/var/kerberos/krb5kdc/kdc.key -c dogtag-ipa-ca-renew-agent -B
/usr/libexec/ipa/certmonger/renew_ra_cert_pre -C
/usr/libexec/ipa/certmonger/renew_kdc_cert -T KDCs_PKINIT_Certs
```

4. **certmonger** を再起動します。

```
# systemctl restart certmonger
```

5. **certmonger** の起動後 1 分待ってから、新しい証明書の状況を確認します。

```
# getcert list
```

関連情報

- IdM システム証明書の有効期限が切れている場合は、KCS (Knowledge Centered Support) ソリューション を参照して、CA 更新サーバー、および CRL パブリッシャーサーバーでもある IdM CA サーバーで IdM システム証明書を手動で更新します。次に、How do I manually renew Identity Management (IPA) certificates on RHEL7 after they have expired? (Replica IPA Server) で説明されている手順に従って、トポロジーにあるその他のすべての CA サーバーで IdM システム証明書を手動で更新します。

18.8. CERTMONGER での SCEP の使用

Simple Certificate Enrollment Protocol (SCEP) は、さまざまなデバイスやオペレーティングシステムで使用できる証明書管理プロトコルです。環境内で SCEP サーバーを外部認証局 (CA) として使用している場合、**certmonger** を使用して Identity Management (IdM) クライアントの証明書を取得できます。

18.8.1. SCEP の概要

Simple Certificate Enrollment Protocol (SCEP) は、さまざまなデバイスやオペレーティングシステムで使用できる証明書管理プロトコルです。SCEP サーバーを外部認証局 (CA) として使用できます。

Identity Management (IdM) クライアントを設定して、CA SCEP サービスから直接 HTTP 経由で証明書を要求および取得できます。このプロセスは、通常、限られた時間でのみ有効な共有シークレットで保護されます。

クライアント側で、SCEP は以下のコンポーネントを提供する必要があります。

- SCEP URL: CA SCEP インターフェイスの URL。
- SCEP 共有シークレット: CA と SCEP クライアントの間で共有される、証明書を取得するために使用される **challengePassword** PIN。

その後、クライアントは SCEP 経由で CA 証明書チェーンを取得し、CA に証明書署名要求を送信します。

certmonger で SCEP を設定する場合は、発行した証明書パラメーターを指定する新しい CA 設定プロファイルを作成します。

18.8.2. SCEP 経由での IdM CA 署名証明書の要求

以下の例では、**SCEP_example** SCEP CA 設定を **certmonger** に追加し、IdM クライアント **client.idm.example.com** で新しい証明書を要求します。**certmonger** は、NSS 証明書データベース形式と、OpenSSL などのファイルベース (PEM) 形式の両方をサポートしています。

前提条件

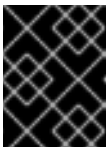
- SCEP URL を知っている。
- **challengePassword** PIN 共有シークレットがある。

手順

1. CA 設定を **certmonger** に追加します。

```
[root@client.idm.example.com ~]# getcert add-scep-ca -c SCEP_example -u SCEP_URL
```

   - **-c**: CA 設定に必要なニックネーム。後で同じ値を、他の **getcert** コマンドと合わせて使用できます。
   - **-u**: サーバーの SCEP インターフェイスへの URL。



重要

HTTPS URL を使用する場合は、**-R** オプションを使用して SCEP サーバー CA 証明書の PEM 形式のコピーの場所も指定する必要があります。

2. CA 設定が正常に追加されたことを確認します。

```
[root@client.idm.example.com ~]# getcert list-cas -c SCEP_example
CA 'SCEP_example':
        is-default: no
        ca-type: EXTERNAL
        helper-location: /usr/libexec/certmonger/scep-submit -u
http://SCEP_server_enrollment_interface_URL
        SCEP CA certificate thumbprint (MD5): A67C2D4B 771AC186 FCCA654A 5E55AAF7
        SCEP CA certificate thumbprint (SHA1): FBFF096C 6455E8E9 BD55F4A5 5787C43F
1F512279
```

   設定が正常に追加された場合、certmonger はリモート CA から CA チェーンを取得します。CA チェーンは、コマンド出力でサムプリントとして表示されます。暗号化されていない HTTP でサーバーにアクセスすると、中間者攻撃を防ぐため、サムプリントを SCEP サーバーに表示されるものと手動で比較します。

3. CA から証明書を要求します。

   - NSS を使用している場合:

```
[root@client.idm.example.com ~]# getcert request -I Example_Task -c SCEP_example
-d /etc/pki/nssdb -n ExampleCert -N cn="client.idm.example.com" -L one-time_PIN -D
client.idm.example.com
```

オプションを使用して、証明書要求の以下のパラメーターを指定できます。

- **-I** : **(オプション)** タスクの名前: リクエストの追跡 ID。後で **getcert list** コマンドで同じ値を使用できます。
- **-c**: 要求を送信する CA 設定。
- **-d**: 証明書およびキーを保存する NSS データベースを備えたディレクトリー。
- **-n**: NSS データベースで使用される証明書のニックネーム。
- **-N**: CSR のサブジェクト名。
- **-L** : CA が発行する期限付きの 1 回限りの **challengePassword** PIN。
- **-d**: 証明書のサブジェクト代替名。通常はホスト名と同じです。

- OpenSSL を使用している場合は、以下を行います。

```
[root@client.idm.example.com ~]# getcert request -I Example_Task -c SCEP_example
-f /etc/pki/tls/certs/server.crt -k /etc/pki/tls/private/private.key -N
cn="client.idm.example.com" -L one-time_PIN -D client.idm.example.com
```

オプションを使用して、証明書要求の以下のパラメーターを指定できます。

- **-I** : **(オプション)** タスクの名前: リクエストの追跡 ID。後で **getcert list** コマンドで同じ値を使用できます。
- **-c**: 要求を送信する CA 設定。
- **-f**: 証明書へのストレージパス。
- **-k**: キーへのストレージパス。
- **-N**: CSR のサブジェクト名。
- **-L** : CA が発行する期限付きの 1 回限りの **challengePassword** PIN。
- **-d**: 証明書のサブジェクト代替名。通常はホスト名と同じです。

検証

1. 証明書が発行され、ローカルデータベースに正しく保存されていることを確認します。

- NSS を使用している場合は、以下を入力します。

```
[root@client.idm.example.com ~]# getcert list -I Example_Task
 Request ID 'Example_Task':
        status: MONITORING
        stuck: no
        key pair storage:
type=NSSDB,location='/etc/pki/nssdb',nickname='ExampleCert',token='NSS Certificate
```

```
DB'
        certificate:
type=NSSDB,location='/etc/pki/nssdb',nickname='ExampleCert',token='NSS Certificate
DB'
        signing request thumbprint (MD5): 503A8EDD DE2BE17E 5BAA3A57 D68C9C1B
        signing request thumbprint (SHA1): B411ECE4 D45B883A 75A6F14D 7E3037F1
D53625F4
        CA: IPA
        issuer: CN=Certificate Authority,O=EXAMPLE.COM
        subject: CN=client.idm.example.com,O=EXAMPLE.COM
        expires: 2018-05-06 10:28:06 UTC
        key usage: digitalSignature,keyEncipherment
        eku: iso.org.dod.internet.security.mechanisms.8.2.2
        certificate template/profile: IPSECIntermediateOffline
        pre-save command:
        post-save command:
        track: true
 auto-renew: true
```

- OpenSSL を使用している場合は、以下を入力します。

```
[root@client.idm.example.com ~]# getcert list -I Example_Task
Request ID 'Example_Task':
        status: MONITORING
        stuck: no
        key pair storage: type=FILE,location='/etc/pki/tls/private/private.key'
        certificate: type=FILE,location='/etc/pki/tls/certs/server.crt'
        CA: IPA
        issuer: CN=Certificate Authority,O=EXAMPLE.COM
        subject: CN=client.idm.example.com,O=EXAMPLE.COM
        expires: 2018-05-06 10:28:06 UTC
        eku: id-kp-serverAuth,id-kp-clientAuth
        pre-save command:
        post-save command:
        track: true
        auto-renew: true
```

ステータス **MONITORING** は、発行した証明書の取得に成功したことを表します。**getcert-list(1)** の man ページには、その他の状態とその意味が記載されています。

関連情報

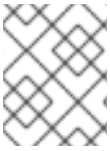
- 証明書を要求する場合の他のオプションは、**getcert-request(1)** の man ページを参照してください。

18.8.3. certmonger による AD SCEP 証明書の自動更新

certmonger が SCEP 証明書の更新要求を送信すると、この要求は既存の証明書の秘密鍵を使用して署名されます。ただし、**certmonger** によってデフォルトで送信される更新要求には、最初に証明書を取得するために使用された **challengePassword** PIN も含まれています。

SCEP サーバーとして機能する Active Directory (AD) Network Device Enrollment Service (NDES) サーバーは、元の **challengePassword** PIN を含む更新要求を自動的に拒否します。そのため、更新に失敗します。

AD での更新を機能させるには、**challengePassword** PIN なしで署名済みの更新要求を送信するように **certmonger** を設定する必要があります。また、更新時にサブジェクト名を比較しないように AD サーバーを設定する必要があります。



注記

challengePassword が含まれるリクエストも拒否する AD 以外の SCEP サーバーが存在する場合があります。この場合は、**certmonger** 設定を変更する必要もあります。

前提条件

- RHEL サーバーは RHEL 8.6 以降を実行している必要がある。

手順

1. AD サーバーで **regedit** を開きます。
2. **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography\MSCEP** サブキーに、新しい 32 ビットの REG_DWORD エントリー **DisableRenewalSubjectNameMatch** を追加し、その値を **1** に設定します。
3. **certmonger** が実行されているサーバーで、**/etc/certmonger/certmonger.conf** ファイルを開き、次のセクションを追加します。

```
[scep]
challenge_password_otp = yes
```

4. certmonger を再起動します。

```
# systemctl restart certmonger
```

第19章 RHEL システムロールを使用して証明書を要求する

certificate システムロールを使用すると、証明書を発行および管理できます。

19.1. CERTIFICATE RHEL システムロール

certificate システムロールを使用すると、Ansible Core を使用して TLS および SSL 証明書の発行と更新を管理できます。

ロールは **certmonger** を証明書プロバイダーとして使用し、自己署名証明書の発行と更新、および IdM 統合認証局 (CA) の使用を現時点でサポートしています。

certificate システムロールを含む Ansible Playbook では、次の変数を使用できます。

certificate_wait
タスクが証明書を発行するまで待機するかどうかを指定します。

certificate_requests
発行する各証明書とそのパラメーターを表すには、次のコマンドを実行します。

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.certificate/README.md** ファイル
- **/usr/share/doc/rhel-system-roles/certificate/** ディレクトリー

19.2. CERTIFICATE RHEL システムロールを使用した新しい自己署名証明書の要求

certificate システムロールを使用すると、Ansible Core を使用して自己署名証明書を発行できます。

このプロセスは、**certmonger** プロバイダーを使用し、**getcert** コマンドで証明書を要求します。

前提条件

- 制御ノードと管理ノードを準備している
- 管理対象ノードで Playbook を実行できるユーザーとしてコントロールノードにログインしている。
- 管理対象ノードへの接続に使用するアカウントに、そのノードに対する **sudo** 権限がある。

手順

1. 次の内容を含む Playbook ファイル (例: ~/**playbook.yml**) を作成します。

```
---
- hosts: managed-node-01.example.com
  roles:
    - rhel-system-roles.certificate
  vars:
    certificate_requests:
```

```
    - name: mycert
      dns: "*.example.com"
      ca: self-sign
```

- **name** パラメーターを希望する証明書の名前 (**mycert** など) に設定します。
- **dns** パラメーターを ***.example.com** などの証明書に含むドメインに設定します。
- **ca** パラメーターを **self-sign** に設定します。

デフォルトでは、**certmonger** は有効期限が切れる前に証明書の更新を自動的に試行します。これは、Ansible Playbook の **auto_renew** パラメーターを **no** に設定すると無効にできます。

2. Playbook の構文を検証します。

```
$ ansible-playbook --syntax-check ~/playbook.yml
```

このコマンドは構文を検証するだけであり、有効だが不適切な設定から保護するものではないことに注意してください。

3. Playbook を実行します。

```
$ ansible-playbook ~/playbook.yml
```

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.certificate/README.md** ファイル
- **/usr/share/doc/rhel-system-roles/certificate/** ディレクトリー

19.3. CERTIFICATE RHEL システムロールを使用した IDM CA からの新しい証明書の要求

certificate システムロールを使用すると、統合認証局 (CA) を持つ IdM サーバーを使用しながら、**ansible-core** を使用して証明書を発行できます。したがって、IdM を CA として使用する場合に、複数のシステムの証明書トラストチェーンを効率的かつ一貫して管理できます。

このプロセスは、**certmonger** プロバイダーを使用し、**getcert** コマンドで証明書を要求します。

前提条件

- 制御ノードと管理ノードを準備している
- 管理対象ノードで Playbook を実行できるユーザーとしてコントロールノードにログインしている。
- 管理対象ノードへの接続に使用するアカウントに、そのノードに対する **sudo** 権限がある。

手順

1. 次の内容を含む Playbook ファイル (例: **~/playbook.yml**) を作成します。

```
---
- hosts: managed-node-01.example.com
```

```
roles:
  - rhel-system-roles.certificate
vars:
  certificate_requests:
    - name: mycert
      dns: www.example.com
      principal: HTTP/www.example.com@EXAMPLE.COM
      ca: ipa
```

- **name** パラメーターを希望する証明書の名前 (**mycert** など) に設定します。
- **dns** パラメーターをドメインに設定し、証明書に追加します (例: **www.example.com**)。
- **principal** パラメーターを設定し、Kerberos プリンシパルを指定します (例: **HTTP/www.example.com@EXAMPLE.COM**)。
- **ca** パラメーターを **ipa** に設定します。

デフォルトでは、**certmonger** は有効期限が切れる前に証明書の更新を自動的に試行します。これは、Ansible Playbook の **auto_renew** パラメーターを **no** に設定すると無効にできます。

2. Playbook の構文を検証します。

```
$ ansible-playbook --syntax-check ~/playbook.yml
```

このコマンドは構文を検証するだけであり、有効だが不適切な設定から保護するものではないことに注意してください。

3. Playbook を実行します。

```
$ ansible-playbook ~/playbook.yml
```

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.certificate/README.md** ファイル
- **/usr/share/doc/rhel-system-roles/certificate/** ディレクトリー

19.4. CERTIFICATE RHEL システムロールを使用して証明書発行前または発行後に実行するコマンドを指定する

certificate ロールでは、Ansible Core を使用して、証明書の発行または更新の前後にコマンドを実行できます。

以下の例では、管理者が **www.example.com** の自己署名証明書を発行または更新する前に **httpd** サービスを停止し、後で再起動します。

前提条件

- 制御ノードと管理ノードを準備している
- 管理対象ノードで Playbook を実行できるユーザーとしてコントロールノードにログインしている。

- 管理対象ノードへの接続に使用するアカウントに、そのノードに対する **sudo** 権限がある。

手順

1. 次の内容を含む Playbook ファイル (例: **~/playbook.yml**) を作成します。

```
---
- hosts: managed-node-01.example.com
  roles:
    - rhel-system-roles.certificate
  vars:
    certificate_requests:
      - name: mycert
        dns: www.example.com
        ca: self-sign
        run_before: systemctl stop httpd.service
        run_after: systemctl start httpd.service
```

   - **name** パラメーターを希望する証明書の名前 (**mycert** など) に設定します。
   - **dns** パラメーターをドメインに設定し、証明書に追加します (例: **www.example.com**)。
   - **ca** パラメーターを証明書を発行する際に使用する CA に設定します (例: **self-sign**)。
   - この証明書を発行または更新する前に、**run_before** パラメーターを実行するコマンドに設定します (例: **systemctl stop httpd.service**)。
   - この証明書を発行または更新した後に、**run_after** パラメーターを実行するコマンドに設定します (例: **systemctl start httpd.service**)。

   デフォルトでは、**certmonger** は有効期限が切れる前に証明書の更新を自動的に試行します。これは、Ansible Playbook の **auto_renew** パラメーターを **no** に設定すると無効にできます。

2. Playbook の構文を検証します。

```
$ ansible-playbook --syntax-check ~/playbook.yml
```

   このコマンドは構文を検証するだけであり、有効だが不適切な設定から保護するものではないことに注意してください。

3. Playbook を実行します。

```
$ ansible-playbook ~/playbook.yml
```

関連情報

- **/usr/share/ansible/roles/rhel-system-roles.certificate/README.md** ファイル
- **/usr/share/doc/rhel-system-roles/certificate/** ディレクトリー

第20章 証明書のサブセットだけに信頼するアプリケーションを制限する手順

Identity Management (IdM) インストールが統合証明書システム (CS) 認証局 (CA) で設定されている場合は、軽量のサブ CA を作成できます。作成するすべてのサブ CA は、証明書システムのプライマリー CA である **ipa** CA に従属します。

このコンテキストでの **軽量サブ CA** は、**証明書を特定の目的で発行するサブ CA** になります。たとえば、軽量のサブ CA では、仮想プライベートネットワーク (VPN) ゲートウェイや Web ブラウザーなどのサービスを設定して、 **sub-CA A** が発行する証明書のみを受け入れることができます。**sub-CA B** が発行する証明書だけを受け入れるように他のサービスを設定すると、**sub-CA A**、プライマリー CA、**ipa** CA、およびこの 2 つの間にある中間サブ CA が発行する証明書を受け付けないようにすることができます。

サブ CA の中間証明書を取り消した場合、このサブ CA で発行されたすべての証明書 は、適切に設定されたクライアントによって自動的に無効とみなされます。ルート CA (**ipa**)、または別のサブ CA が直接発行するその他の証明書はすべて有効のままになります。

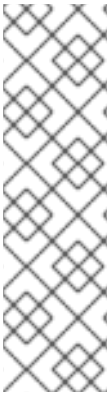
本セクションでは、Apache Web サーバーの例を使用して、アプリケーションが証明書のサブセットだけを信頼するように制限する方法を説明します。本セクションでは、IdM クライアントで実行している Web サーバーを、IdM サブ CA **webserver-ca** が発行する証明書を使用するように制限して、ユーザーに IdM サブ CA **webclient-ca** が発行するユーザー証明書を使用して、Web サーバーへの認証を行うようにするには、以下のセクションを実行します。

実行する必要がある手順は以下の通りです。

1. IdM サブ CA の作成
2. IdM WebUI からのサブ CA 証明書のダウンロード
3. 使用するユーザー、サービス、CA、および証明書プロファイルの正しい組み合わせを指定する CA ACL の作成
4. IdM サブ CA からの IdM クライアントで実行している Web サービスの証明書要求
5. シングルインスタンスの Apache HTTP サーバーの設定
6. Apache HTTP Server への TLS 暗号化の追加
7. Apache HTTP サーバーで対応している TLS プロトコルバージョンの設定
8. Apache HTTP サーバーで対応している暗号の設定
9. Web サーバーで TLS クライアント証明書認証の設定
10. IdM のサブ CA からのユーザー証明書の要求およびクライアントへのエクスポート
11. ブラウザーへのユーザー証明書のインポートおよびサブ CA 証明書を信頼するようなブラウザーの設定

20.1. 軽量サブ CA の管理

本セクションでは、軽量の従属認証局 (サブ CA) を管理する方法を説明します。作成するすべてのサブ CA は、証明書システムのプライマリー CA である **ipa** CA に従属します。サブ CA を無効にしたり、削除したりすることもできます。



注記

- サブ CA を削除すると、そのサブ CA の失効確認は機能しなくなります。**notAfter** の有効期限が近づいていて、そのサブ CA が発行した証明書がなくなった場合に限り、サブ CA を削除してください。
- サブ CA が発行した期限切れではない証明書がまだ存在する場合に限り、サブ CA を無効にしてください。サブ CA により発行された証明書がすべて期限切れになると、そのサブ CA を削除できます。
- IdM CA を無効にしたり、削除したりすることはできません。

サブ CA の管理の詳細は、次を参照してください。

- IdM WebUI からのサブ CA の作成
- IdM WebUI からのサブ CA の削除
- IdM CLI からのサブ CA の作成
- IdM CLI からのサブ CA の無効化
- IdM CLI からのサブ CA の削除

20.1.1. IdM WebUI からのサブ CA の作成

以下の手順に従って、IdM WebUI を使用して **webserver-ca** および **webclient-ca** という名前の新しいサブ CA を作成します。

前提条件

- 管理者の認証情報を取得していることを確認している。

手順

1. **Authentication** メニューで、**Certificates** をクリックします。
2. **Certificate Authorities** を選択し、**Add** をクリックします。
3. **webserver-ca** サブ CA の名前を入力します。サブジェクト DN フィールドにサブジェクト DN (例: **CN=WEBSERVER,O=IDM.EXAMPLE.COM**) を入力します。サブジェクト DN は、IdM CA インフラストラクチャー内で一意である必要があります。
4. **webclient-ca** サブ CA の名前を入力します。サブジェクト DN **CN=WEBCLIENT,O=IDM.EXAMPLE.COM** をサブジェクト DN フィールドに入力します。
5. コマンドラインインターフェイスで、**ipa-certupdate** コマンドを実行して、**webserver-ca** サブ CA 証明書および **webclient-ca** サブ CA 証明書の **certmonger** 追跡リクエストを作成します。

```
[root@ipaserver ~]# ipa-certupdate
```

重要

サブ CA の作成後に **ipa-certupdate** コマンドを実行しておかないと、サブ CA 証明書の有効期限が切れたときに、エンドエンティティー証明書の有効期限が切れていなくても、サブ CA が発行したエンドエンティティー証明書は無効と見なされます。

検証

- 新しいサブ CA の署名証明書が IdM データベースに追加されたことを確認します。

```
[root@ipaserver ~]# certutil -d /etc/pki/pki-tomcat/alias/ -L

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

caSigningCert cert-pki-ca               CTu,Cu,Cu
Server-Cert cert-pki-ca                 u,u,u
auditSigningCert cert-pki-ca            u,u,Pu
caSigningCert cert-pki-ca ba83f324-5e50-4114-b109-acca05d6f1dc u,u,u
ocspSigningCert cert-pki-ca             u,u,u
subsystemCert cert-pki-ca               u,u,u
```

注記

新しいサブ CA 証明書は、証明書システムインスタンスがインストールされているすべてのレプリカに自動的に転送されます。

20.1.2. IdM WebUI からのサブ CA の削除

以下の手順に従って、IdM WebUI で軽量のサブ CA を削除します。

注記

- サブ CA を削除すると、そのサブ CA の失効確認は機能しなくなります。**notAfter** の有効期限が近づいていて、そのサブ CA が発行した証明書がなくなった場合に限り、サブ CA を削除してください。
- サブ CA が発行した期限切れではない証明書がまだ存在する場合に限り、サブ CA を無効にしてください。サブ CA により発行された証明書がすべて期限切れになると、そのサブ CA を削除できます。
- IdM CA を無効にしたり、削除したりすることはできません。

前提条件

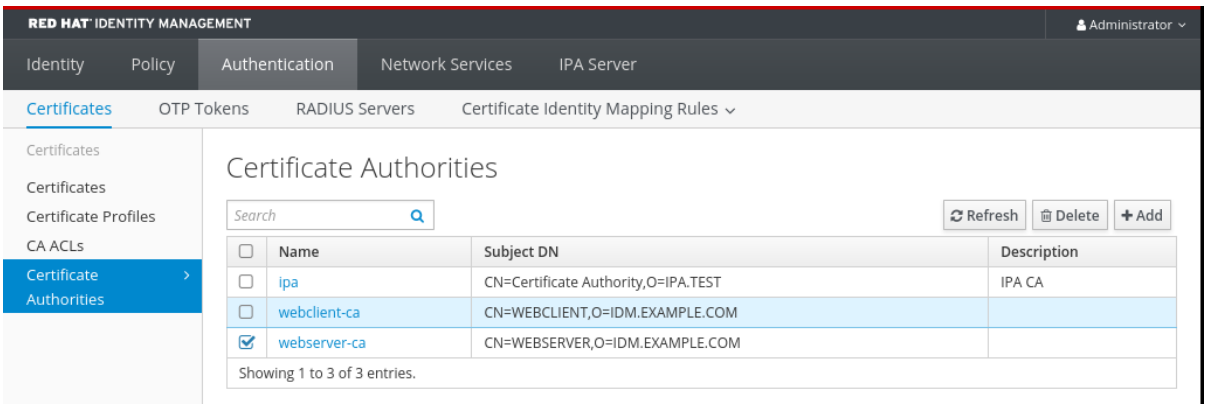
- 管理者の認証情報を取得していることを確認している。
- IdM CLI でサブ CA を無効にしている。IdM CLI からのサブ CA の無効化 参照

手順

1. IdM WebUI で、**Authentication** タブを開き、**Certificates** サブタブを選択します。

2. **Certificate Authorities** を選択します。
3. 削除するサブ CA を選択し、**Delete** をクリックします。

図20.1 IdM Web UI でのサブ CA の削除



4. **Delete** をクリックして確定します。

サブ CA が **Certificate Authorities** のリストから削除されます。

20.1.3. IdM CLI からのサブ CA の作成

以下の手順に従って、IdM CLI を使用して **webserver-ca** および **webclient-ca** という名前の新しいサブ CA を作成します。

前提条件

- 管理者の認証情報を取得していることを確認している。
- CA サーバーである IdM サーバーにログインしている。

手順

1. **ipa ca-add** コマンドを入力し、サブ CA **webserver-ca** の名前とそのサブジェクト識別名 (DN) を指定します。

```
[root@ipaserver ~]# ipa ca-add webserver-ca --
subject="CN=WEBSERVER,O=IDM.EXAMPLE.COM"
-------------------
Created CA "webserver-ca"
-------------------
  Name: webserver-ca
  Authority ID: ba83f324-5e50-4114-b109-acca05d6f1dc
  Subject DN: CN=WEBSERVER,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IDM.EXAMPLE.COM
```

名前

CA の名前

認証局 ID

CA 用に自動作成される個別 ID。

発行先 DN

サブジェクト識別名 (DN)サブジェクト DN は、IdM CA インフラストラクチャー内で一意である必要があります。

発行者 DN

サブ CA 証明書を発行した親 CA。サブ CA はすべて、IdM のルート CA の子として作成されます。

2. Web クライアントに証明書を発行するサブ CA **webclient-ca** を作成します。

```
[root@ipaserver ~]# ipa ca-add webclient-ca --
subject="CN=WEBCLIENT,O=IDM.EXAMPLE.COM"
-------------------
Created CA "webclient-ca"
-------------------
  Name: webclient-ca
  Authority ID: 8a479f3a-0454-4a4d-8ade-fd3b5a54ab2e
  Subject DN: CN=WEBCLIENT,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IDM.EXAMPLE.COM
```

3. **ipa-certupdate** コマンドを実行して、**webserver-ca** および **webclient-ca** のサブ CA 証明書の **certmonger** 追跡リクエストを作成します。

```
[root@ipaserver ~]# ipa-certupdate
```

重要

サブ CA の作成後に **ipa-certupdate** コマンドを実行し忘れ、サブ CA 証明書が期限切れになった場合、そのサブ CA が発行したエンドエンティティー証明書は、エンドエンティティー証明書の有効期限が切れていなくても無効と見なされます。

検証手順

- 新しいサブ CA の署名証明書が IdM データベースに追加されたことを確認します。

```
[root@ipaserver ~]# certutil -d /etc/pki/pki-tomcat/alias/ -L

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

caSigningCert cert-pki-ca               CTu,Cu,Cu
Server-Cert cert-pki-ca                 u,u,u
auditSigningCert cert-pki-ca            u,u,Pu
caSigningCert cert-pki-ca ba83f324-5e50-4114-b109-acca05d6f1dc u,u,u
ocspSigningCert cert-pki-ca             u,u,u
subsystemCert cert-pki-ca               u,u,u
```

注記

新しいサブ CA 証明書は、証明書システムインスタンスがインストールされているすべてのレプリカに自動的に転送されます。

20.1.4. IdM CLI からのサブ CA の無効化

以下の手順に従って、IdM CLI からサブ CA を無効にします。サブ CA が発行した期限切れでない証明書が依然として存在する場合は、証明書を削除しないでください。ただし、無効にすることはできます。サブ CA を削除すると、そのサブ CA に対する取消しの確認が機能しなくなります。

前提条件

- 管理者の認証情報を取得していることを確認している。

手順

1. **ipa ca-find** コマンドを実行して、削除するサブ CA の名前を確認します。

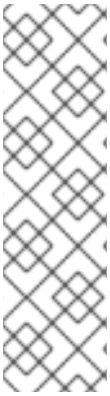
```
[root@ipaserver ~]# ipa ca-find
----------
3 CAs matched
----------
  Name: ipa
  Description: IPA CA
  Authority ID: 5195deaf-3b61-4aab-b608-317aff38497c
  Subject DN: CN=Certificate Authority,O=IPA.TEST
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

  Name: webclient-ca
  Authority ID: 605a472c-9c6e-425e-b959-f1955209b092
  Subject DN: CN=WEBCLIENT,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

 Name: webserver-ca
  Authority ID: 02d537f9-c178-4433-98ea-53aa92126fc3
  Subject DN: CN=WEBSERVER,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST
----------------------------
Number of entries returned 3
----------------------------
```

2. **ipa ca-disable** コマンドを実行して、サブ CA (この例では **webserver-ca**) を無効にします。

```
ipa ca-disable webserver-ca
--------------------------
Disabled CA "webserver-ca"
--------------------------
```

20.1.5. IdM CLI からのサブ CA の削除

以下の手順に従って、IdM CLI から軽量のサブ CA を削除します。



注記

- サブ CA を削除すると、そのサブ CA の失効確認は機能しなくなります。**notAfter** の有効期限が近づいていて、そのサブ CA が発行した証明書がなくなった場合に限り、サブ CA を削除してください。
- サブ CA が発行した期限切れではない証明書がまだ存在する場合に限り、サブ CA を無効にしてください。サブ CA により発行された証明書がすべて期限切れになると、そのサブ CA を削除できます。
- IdM CA を無効にしたり、削除したりすることはできません。

前提条件

- 管理者の認証情報を取得していることを確認している。

手順

1. サブ CA および CA のリストを表示するには、**ipa ca-find** コマンドを実行します。

```
# ipa ca-find
-------------
3 CAs matched
-------------
  Name: ipa
  Description: IPA CA
  Authority ID: 5195deaf-3b61-4aab-b608-317aff38497c
  Subject DN: CN=Certificate Authority,O=IPA.TEST
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

  Name: webclient-ca
  Authority ID: 605a472c-9c6e-425e-b959-f1955209b092
  Subject DN: CN=WEBCLIENT,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

 Name: webserver-ca
  Authority ID: 02d537f9-c178-4433-98ea-53aa92126fc3
  Subject DN: CN=WEBSERVER,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST
----------------------------
Number of entries returned 3
----------------------------
```

2. **ipa ca-disable** コマンドを実行して、サブ CA (この例では **webserver-ca**) を無効にします。

```
# ipa ca-disable webserver-ca
--------------------------
Disabled CA "webserver-ca"
--------------------------
```

3. サブ CA (この例では **webserver-ca**) を削除します。

```
# ipa ca-del webserver-ca
-------------------------
Deleted CA "webserver-ca"
```

```
-------------------------
```

検証

- **ipa ca-find** を実行して、CA およびサブ CA のリストを表示します。**webserver-ca** がリストに表示されなくなりました。

```
# ipa ca-find
-------------
2 CAs matched
-------------
  Name: ipa
  Description: IPA CA
  Authority ID: 5195deaf-3b61-4aab-b608-317aff38497c
  Subject DN: CN=Certificate Authority,O=IPA.TEST
  Issuer DN: CN=Certificate Authority,O=IPA.TEST

  Name: webclient-ca
  Authority ID: 605a472c-9c6e-425e-b959-f1955209b092
  Subject DN: CN=WEBCLIENT,O=IDM.EXAMPLE.COM
  Issuer DN: CN=Certificate Authority,O=IPA.TEST
----------------------------
Number of entries returned 2
----------------------------
```

20.2. IDM WEB UI からのサブ CA 証明書のダウンロード

前提条件

- IdM 管理者の認証情報を取得していることを確認している。

手順

1. **Authentication** メニューで、**Certificates** > **Certificates** をクリックします。

   図20.2 証明書リストに含まれるサブ CA 証明書

☐	268173326	CN=WEBSERVER,O=IDM.EXAMPLE.COM	ipa	VALID
☐	268238849	CN=idm_user,O=IDM.EXAMPLE.COM	ipa	VALID

2. サブ CA 証明書のシリアル番号をクリックして、証明書情報ページを開きます。
3. 証明書情報ページで、**Actions** > **Download** をクリックします。
4. CLI で、サブ CA 証明書を **/etc/pki/tls/private/** ディレクトリーに移動します。

```
# mv path/to/the/downloaded/certificate /etc/pki/tls/private/sub-ca.crt
```

20.3. WEB サーバーおよびクライアント認証用の CA ACL の作成

認証局のアクセス制御リスト (CA ACL) ルールは、どのユーザー、サービス、またはホストにどのプロファイルを使用して証明書を発行するかを定義します。CA ACL は、プロファイル、プリンシパル、およびグループを関連付けることで、特定のプロファイルを使用した証明書をプリンシパルまたはグルー

プが要求できるようにします。

たとえば、管理者は CA ACL を使用して、ロンドンオフィス関連のグループに所属するユーザーだけが、ロンドンのオフィスから作業する社員向けのプロファイルを使用するように限定できます。

20.3.1. IdM CLI での CA ACL の表示

IdM デプロイメントで利用可能な認証局のアクセス制御リスト (CA ACL) の一覧と、特定の CA ACL の詳細を表示するには、次の手順に従います。

手順

1. IdM 環境内のすべての CA ACL を表示するには、**ipa caacl-find** コマンドを入力します。

```
$ ipa caacl-find
-----------------
1 CA ACL matched
-----------------
  ACL name: hosts_services_caIPAserviceCert
  Enabled: TRUE
```

2. CA ACL の詳細を表示するには、**ipa caacl-show** コマンドを入力して、CA ACL 名を指定します。たとえば、CA ACL **hosts_services_caIPAserviceCert** の詳細を表示するには、次のコマンドを実行します。

```
$ ipa caacl-show hosts_services_caIPAserviceCert
  ACL name: hosts_services_caIPAserviceCert
  Enabled: TRUE
  Host category: all
  Service category: all
  CAs: ipa
  Profiles: caIPAserviceCert
  Users: admin
```

20.3.2. webserver-ca で発行される証明書を使用して Web クライアントを認証する Web サーバー用の CA ACL の作成

システム管理者が **HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM** サービス証明書の要求時に、サブ CA **webserver-ca** および **caIPAserviceCert** プロファイルを使用するように要求する CA ACL を作成するには、次の手順に従います。ユーザーが別のサブ CA または別のプロファイルの証明書を要求すると、その要求は失敗します。唯一の例外は、別に一致する CA ACL があり、その ACL が有効な場合です。利用可能な CA ACL を表示するには、IdM CLI で CA ACL の表示 を参照してください。

前提条件

- **HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM** サービスが IdM に含まれている。
- IdM 管理者の認証情報を取得していることを確認している。

手順

1. **ipa caacl** コマンドを使用して CA ACL を作成し、その名前を指定します。

```
$ ipa caacl-add TLS_web_server_authentication
----------------------------------------
Added CA ACL "TLS_web_server_authentication"
----------------------------------------
  ACL name: TLS_web_server_authentication
  Enabled: TRUE
```

2. **ipa caacl-mod** コマンドを使用して CA ACL を変更し、CA ACL の説明を指定します。

```
$ ipa caacl-mod TLS_web_server_authentication --desc="CAACL for web servers
authenticating to web clients using certificates issued by webserver-ca"
----------------------------------------------
Modified CA ACL "TLS_web_server_authentication"
----------------------------------------------
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued
by webserver-ca
  Enabled: TRUE
```

3. **webserver-ca** サブ CA を CA ACL に追加します。

```
$ ipa caacl-add-ca TLS_web_server_authentication --ca=webserver-ca
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued
by webserver-ca
  Enabled: TRUE
  CAs: webserver-ca
-------------------------
Number of members added 1
-------------------------
```

4. **ipa caacl-add-service** を使用して、プリンシパルで証明書を要求できるサービスを指定します。

```
$ ipa caacl-add-service TLS_web_server_authentication --
service=HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued
by webserver-ca
  Enabled: TRUE
  CAs: webserver-ca
  Services: HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM
-------------------------
Number of members added 1
-------------------------
```

5. **ipa caacl-add-profile** コマンドを使用して、要求された証明書の証明書プロファイルを指定します。

```
$ ipa caacl-add-profile TLS_web_server_authentication --
certprofiles=caIPAserviceCert
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued
by webserver-ca
```

```
  Enabled: TRUE
  CAs: webserver-ca
  Profiles: caIPAserviceCert
  Services: HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM
----------------------------
Number of members added 1
----------------------------
```

新たに作成した CA ACL は直接使用できます。CA ACL はデフォルトで作成後に有効になります。

注記

CA ACL は、特定のプリンシパルまたはグループから送信される要求への対応が許可されているのは、どの CA またはプロファイルの組み合わせであるかを指定することが目的です。CA ACL は、証明書の検証や信頼には適用されず、発行された証明書の使用方法にも影響はありません。

20.3.3. webclient-ca が発行する証明書を使用して Web サーバーに対して認証するユーザーの Web ブラウザー用に CA ACL を作成する手順

システム管理者が証明書の要求時に サブ CA **webclient-ca** と **IECUserRoles** プロファイルを使用する必要がある CA ACL を作成するには、次の手順に従います。ユーザーが別のサブ CA または別のプロファイルの証明書を要求すると、その要求は失敗します。唯一の例外は、別に一致する CA ACL があり、その ACL が有効な場合です。利用可能な CA ACL を表示するには、IdM CLI で CA ACL の表示 を参照してください。

前提条件

- IdM 管理者の認証情報を取得していることを確認している。

手順

1. **ipa caacl** コマンドを使用して CA ACL を作成し、その名前を指定します。

```
$ ipa caacl-add TLS_web_client_authentication
-----------------------------------------
Added CA ACL "TLS_web_client_authentication"
-----------------------------------------
  ACL name: TLS_web_client_authentication
  Enabled: TRUE
```

2. **ipa caacl-mod** コマンドを使用して CA ACL を変更し、CA ACL の説明を指定します。

```
$ ipa caacl-mod TLS_web_client_authentication --desc="CAACL for user web
browsers authenticating to web servers using certificates issued by webclient-ca"
-----------------------------------------
Modified CA ACL "TLS_web_client_authentication"
-----------------------------------------
  ACL name: TLS_web_client_authentication
  Description: CAACL for user web browsers authenticating to web servers using certificates
issued by webclient-ca
  Enabled: TRUE
```

3. **webclient-ca** サブ CA を CA ACL に追加します。

```
$ ipa caacl-add-ca TLS_web_client_authentication --ca=webclient-ca
  ACL name: TLS_web_client_authentication
  Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
  Enabled: TRUE
  CAs: webclient-ca
-------------------------
Number of members added 1
-------------------------
```

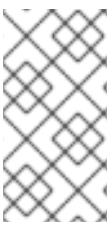
4. **ipa caacl-add-profile** コマンドを使用して、要求された証明書の証明書プロファイルを指定します。

```
$ ipa caacl-add-profile TLS_web_client_authentication --certprofiles=IECUserRoles
  ACL name: TLS_web_client_authentication
  Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
  Enabled: TRUE
  CAs: webclient-ca
  Profiles: IECUserRoles
-------------------------
Number of members added 1
-------------------------
```

5. **ipa caacl-mod** コマンドを使用して CA ACL を変更し、CA ACL がすべての IdM ユーザーに適用されるように指定します。

```
$ ipa caacl-mod TLS_web_client_authentication --usercat=all
-----------------------------------------------
Modified CA ACL "TLS_web_client_authentication"
-----------------------------------------------
  ACL name: TLS_web_client_authentication
  Description: CAACL for user web browsers authenticating to web servers using certificates issued by webclient-ca
  Enabled: TRUE
  User category: all
  CAs: webclient-ca
  Profiles: IECUserRoles
```

新たに作成した CA ACL は直接使用できます。CA ACL はデフォルトで作成後に有効になります。



注記

CA ACL は、特定のプリンシパルまたはグループから送信される要求への対応が許可されているのは、どの CA またはプロファイルの組み合わせであるかを指定することが目的です。CA ACL は、証明書の検証や信頼には適用されず、発行された証明書の使用方法にも影響はありません。

20.4. CERTMONGER を使用したサービスの IDM 証明書の取得

ブラウザーと、IdM クライアントで実行している Web サービスとの間の通信が安全で暗号化されてい

ることを確認するには、TLS 証明書を使用します。サブ CA **webserver-ca** が発行する証明書を信頼し、その他の IdM サブ CA は信頼しないように制限して Web ブラウザーを設定する場合にはサブ CA **webserver-ca** から Web サービスの TLS 証明書を取得します。

以下の手順に従って、**certmonger** を使用して、IdM クライアントで実行しているサービス (**HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM**) の IdM 証明書を取得します。

certmonger 使用して証明書を自動的に要求するということは、**certmonger** 更新の期限が切れたときに証明書を管理および更新することを意味します。

certmonger がサービス証明書を要求したときの動作の視覚的な表現については、サービス証明書を要求する certmonger の通信フロー を参照してください。

前提条件

- Web サーバーが、IdM クライアントとして登録されている。
- 手順を実行している IdM クライアントへのルートアクセス権限がある。
- 証明書を要求しているサービスは、前もって IdM に用意する必要はない。

手順

1. **HTTP** サービスが稼働している IdM クライアント **my_company.idm.example.com** で、以下を指定する **HTTP/my_company.idm.example.com@IDM.EXAMPLE.COM** プリンシパルに対応するサービスの証明書を要求します。
   - 証明書は、ローカルの **/etc/pki/tls/certs/httpd.pem** ファイルに保存されます。
   - 秘密鍵は、ローカルの **/etc/pki/tls/private/httpd.key** ファイルに保存されます。
   - サブ CA **webserver-ca** は、発行元の認証局になります。
   - **SubjectAltName** の extensionRequest が、 **my_company.idm.example.com** の DNS 名の署名要求に追加されます。

     ```
     # ipa-getcert request -K HTTP/my_company.idm.example.com -k
     /etc/pki/tls/private/httpd.key -f /etc/pki/tls/certs/httpd.pem -g 2048 -D
     my_company.idm.example.com -X webserver-ca -C "systemctl restart httpd"
     New signing request "20190604065735" added.
     ```

     上記のコマンドでは、以下のようになります。

     - **ipa-getcert request** コマンドは、証明書が IdM CA から取得することを示しています。**ipa-getcert request** コマンドは、**getcert request -c IPA** のショートカットです。
     - **-g** オプションは、生成先のキーのサイズ (設定されていない場合) を指定します。
     - **-D** オプションは、要求に追加する DNS 値 **SubjectAltName** を指定します。
     - **-X** オプションは、証明書の発行者が、**ipa** ではなく **webserver-ca** でなければならないことを指定します。
     - **-C** オプションは、証明書の取得後に **httpd** サービスを再起動するように **certmonger** に指示します。

- 特定のプロファイルで証明書を発行するように指定する場合は、**-T** オプションを使用します。

注記

RHEL 8 は、RHEL 7 で使用されるものとは異なる SSL モジュール (Apache) を使用します。SSL モジュールは、NSS ではなく OpenSSL に依存しています。このため、RHEL 8 では、NSS データベースを使用して **HTTPS** 証明書と秘密鍵を保存することができません。

2. 必要に応じて、リクエストの状況を確認するには、次のコマンドを実行します。

```
# ipa-getcert list -f /etc/pki/tls/certs/httpd.pem
Number of certificates and requests being tracked: 3.
Request ID '20190604065735':
    status: MONITORING
    stuck: no
    key pair storage: type=FILE,location='/etc/pki/tls/private/httpd.key'
    certificate: type=FILE,location='/etc/pki/tls/certs/httpd.crt'
    CA: IPA
    issuer: CN=WEBSERVER,O=IDM.EXAMPLE.COM

[...]
```

この出力は、要求が **MONITORING** 状況であることを表しています。これは、証明書が取得されていることを示しています。キーペアと証明書の場所は、要求された場所です。

20.5. サービス証明書を要求する CERTMONGER の通信フロー

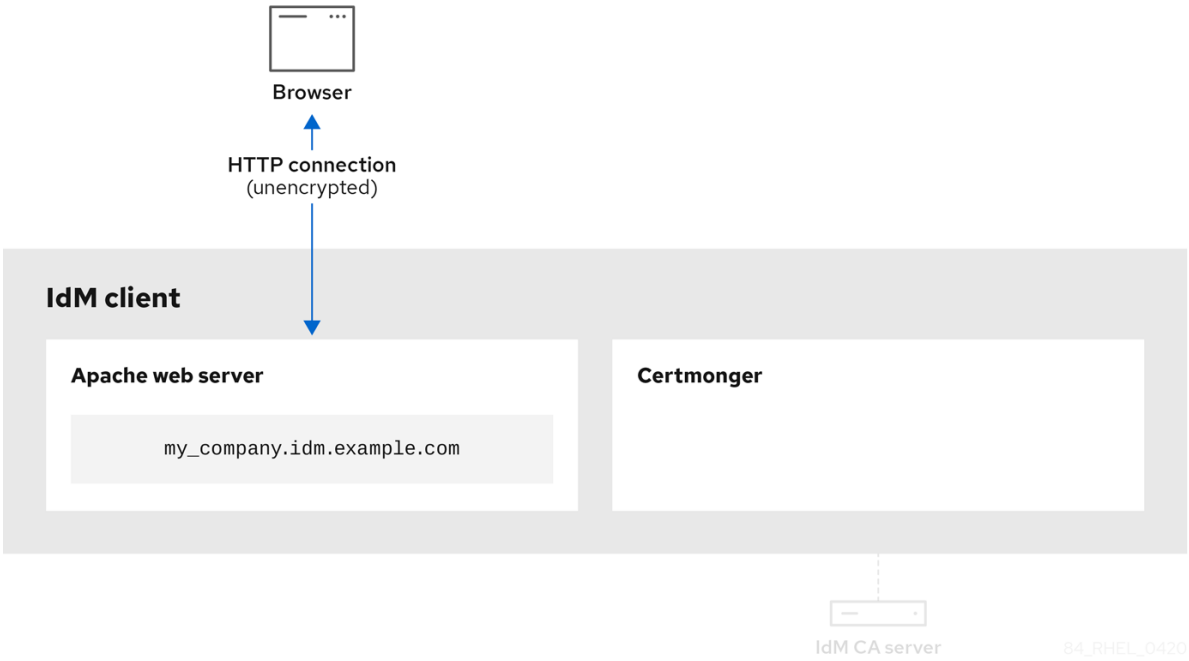
これらの図では、**certmonger** が Identity Management (IdM) 認証局 (CA) サーバーからサービス証明書を要求したときに何が起こるかを段階をおって紹介しています。シーケンスは次の図で設定されています。

- 暗号化されていない通信
- サービス証明書を要求する certmonger
- サービス証明書を発行する IdM CA
- Certmonger によるサービス証明書の適用
- 古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger

この図では、サブ CA **webserver-ca** は汎用 **IdM CA サーバー** で表現されます。

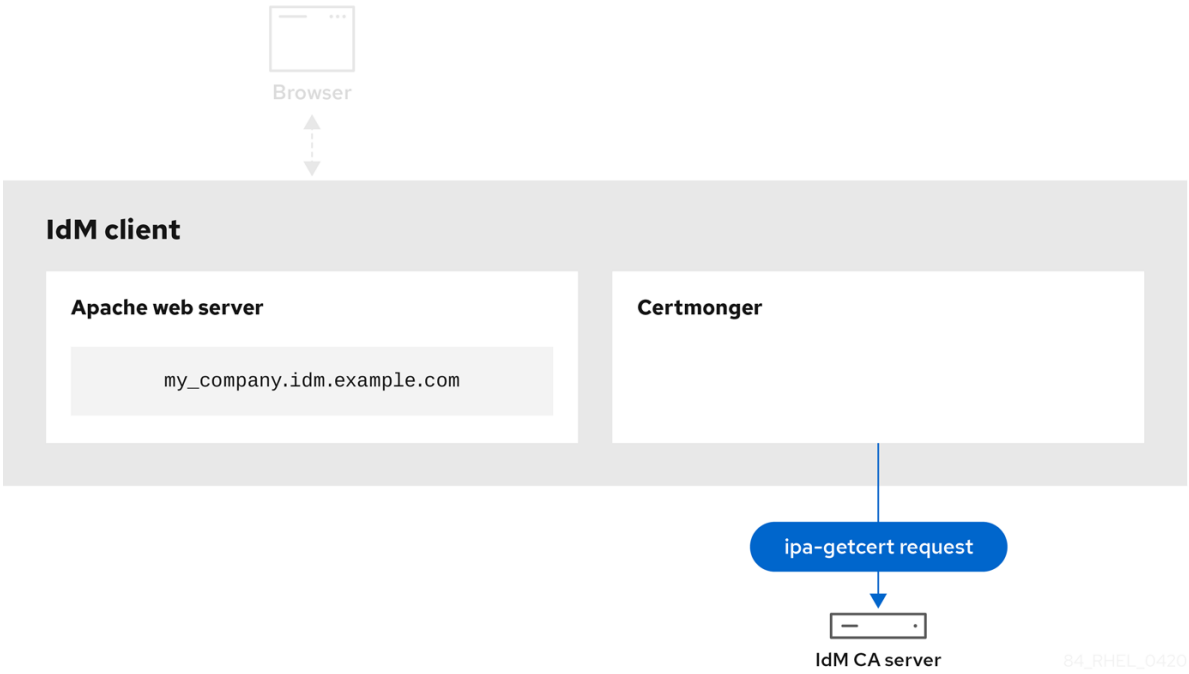
暗号化されていない通信 は、初期状態を示しています。HTTPS 証明書がないと、Web サーバーとブラウザー間の通信は暗号化されません。

図20.3 暗号化されていない通信



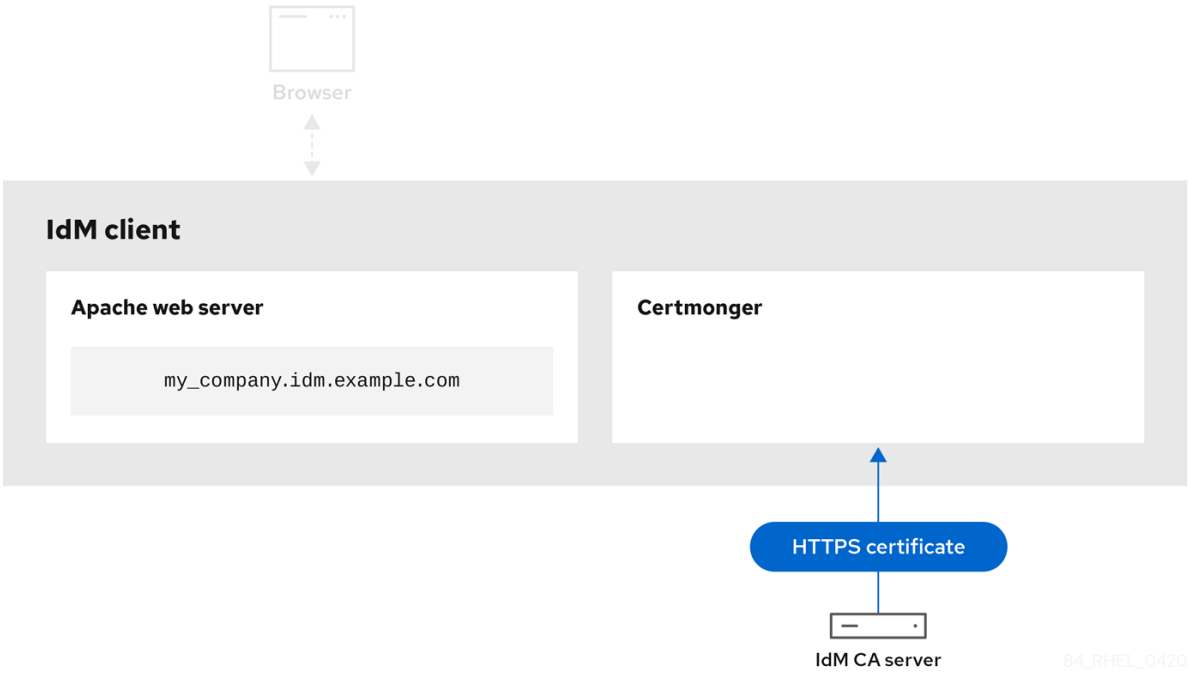
サービス証明書を要求する certmonger は、システム管理者が **certmonger** を使用して Apache Web サーバーの HTTPS 証明書を手動で要求していることを示しています。Web サーバー証明書を要求する場合、certmonger は CA と直接対話しないことに注意してください。IdM 経由でプロキシーが設定されます。

図20.4 サービス証明書を要求する certmonger



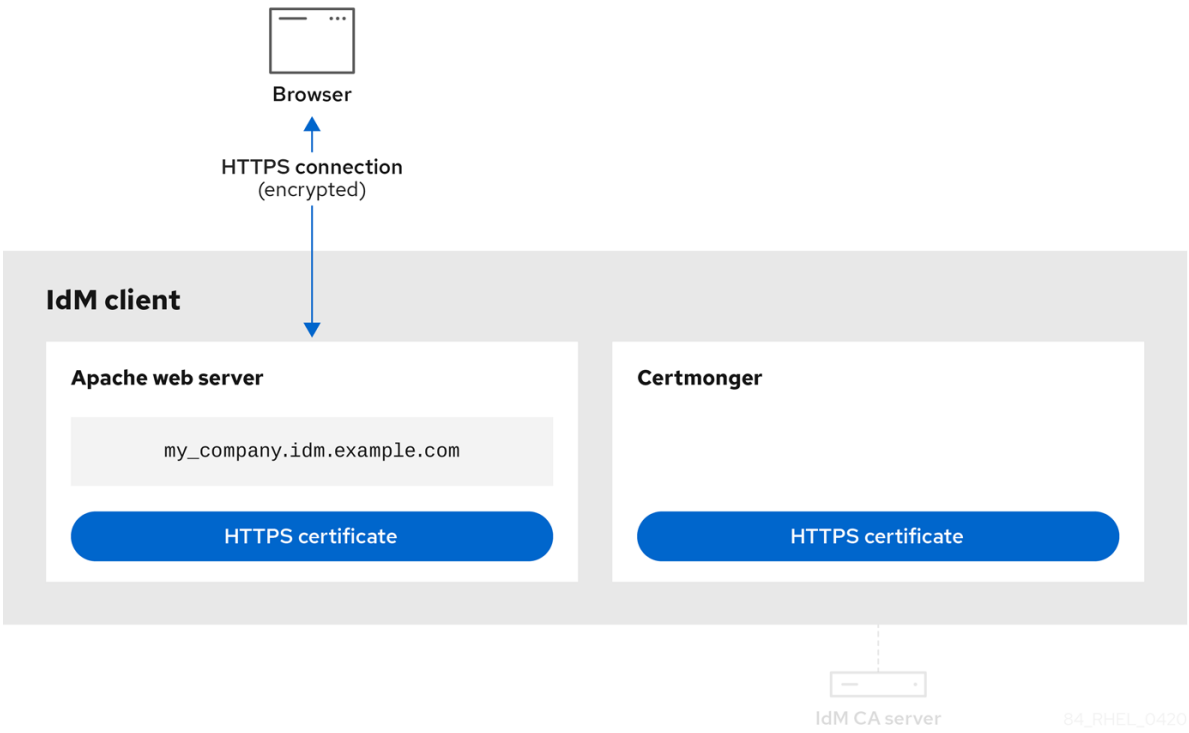
サービス証明書を発行する IdM CA は、Web サーバーで HTTPS 証明書を発行する IdM CA を示しています。

図20.5 サービス証明書を発行する IdM CA



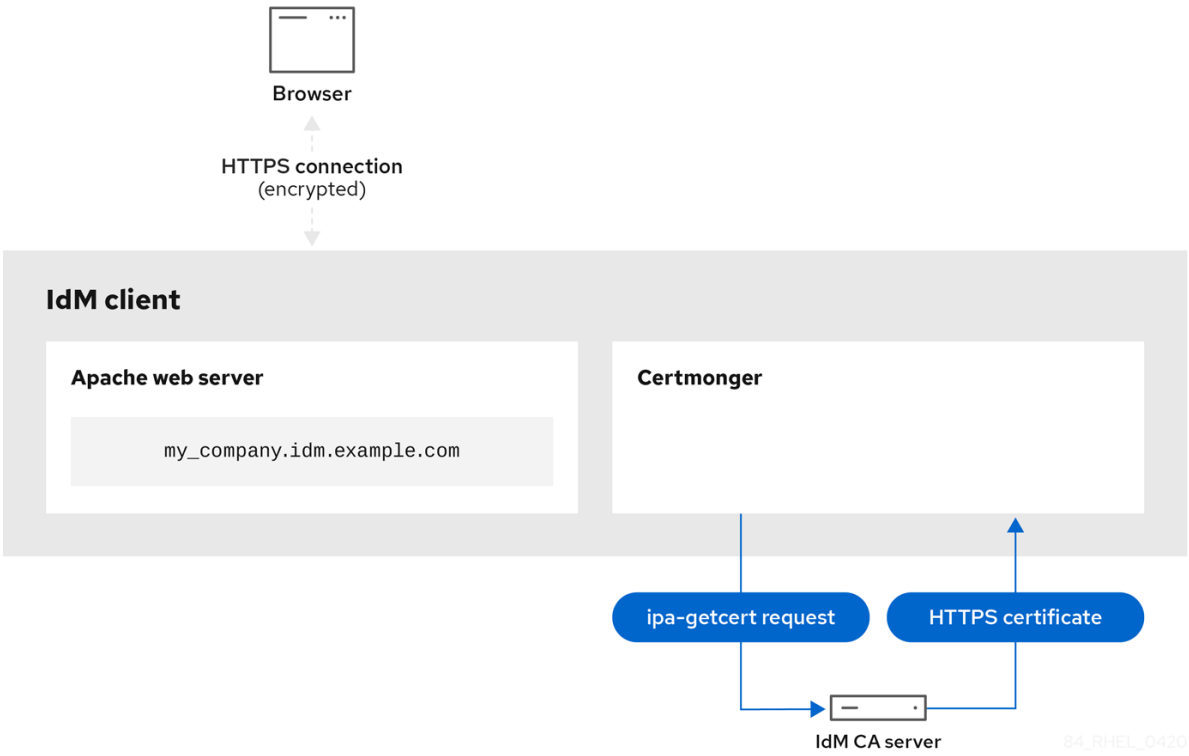
Certmonger によるサービス証明書の適用 は、**certmonger** が HTTPS 証明書を Id M クライアントの適切な場所に配置し、指示された場合は **httpd** サービスを再起動することを示します。その後、Apache サーバーは HTTPS 証明書を使用して、Apache サーバーとブラウザー間のトラフィックを暗号化します。

図20.6 Certmonger によるサービス証明書の適用



古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger は、証明書の有効期限が切れる前に、**certmonger** が Id MCA からのサービス証明書の更新を自動的に要求していることを示しています。IdM CA は、新しい証明書を発行します。

図20.7 古い証明書が有効期限に近づいているときに新しい証明書を要求する certmonger



20.6. シングルインスタンスの APACHE HTTP SERVER 設定

シングルインスタンスの Apache HTTP Server を設定して、静的 HTML コンテンツを提供できます。

Web サーバーに関連付けられた全ドメインにサーバーから同じコンテンツを提供する必要がある場合は、この手順に従います。異なるドメインに異なるコンテンツを提供する場合は、名前ベースの仮想ホストを設定します。詳細は Apache 名ベースの仮想ホストの設定 を参照してください。

手順

1. **httpd** パッケージをインストールします。

```
# yum install httpd
```

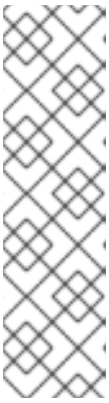
2. **firewalld** を使用する場合は、ローカルのファイアウォールで TCP ポート **80** を開きます。

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

3. **httpd** サービスを有効にして起動します。

```
# systemctl enable --now httpd
```

4. 必要に応じて、HTML ファイルを **/var/www/html/** ディレクトリーに追加します。



注記

/var/www/html/ にコンテンツを追加する場合には、**httpd** を実行するユーザーが、デフォルトでファイルとディレクトリーを読み取れるようにする必要があります。コンテンツの所有者は、 **root** ユーザーおよび **root** ユーザーグループ、または管理者別のユーザーまたはグループのいずれかになります。コンテンツの所有者が **root** ユーザーおよび **root** ユーザーグループの場合には、他のユーザーがファイルを読み取れるようにする必要があります。すべてのファイルとディレクトリーの SELinux コンテキストは **httpd_sys_content_t** である必要があります。これはデフォルトで **/var/www** ディレクトリー内の全コンテンツに適用されます。

検証手順

- Web ブラウザーで **http://my_company.idm.example.com/** または **http://server_IP/** に移動します。
  /var/www/html/ ディレクトリーが空であるか、**index.html** または **index.htm** ファイルが含まれていない場合は、Apache が **Red Hat Enterprise Linux Test Page** を表示します。**/var/www/html/** に異なる名前の HTML ファイルが含まれている場合は、**http://server_IP/example.html** や **http://my_company.idm.example.com/example.html** などの URL をファイルに指定してそのファイルを読み込むことができます。

関連情報

- Apache マニュアル: Apache HTTP Server マニュアルのインストール
- **httpd.service(8)** man ページを参照してください。

20.7. APACHE HTTP SERVER への TLS 暗号化の追加

idm.example.com ドメイン向けに、Apache HTTP Server **my_company.idm.example.com** で TLS 暗号化を有効にできます。

前提条件

- Apache HTTP Server **my_company.idm.example.com** をインストールして、実行している。
- certmonger を使用したサービスの IdM 証明書の取得 で説明されているように、**webserver-ca** サブ CA から TLS 証明書を取得し、**/etc/pki/tls/certs/httpd.pem** ファイルに保存しました。別のパスを使用する場合は、この手順で対応する手順を調整します。
- 対応する秘密鍵を **/etc/pki/tls/private/httpd.key** ファイルに保存している。別のパスを使用する場合は、この手順で対応する手順を調整します。
- CA 証明書 **webserver-ca** が **/etc/pki/tls/private/sub-ca.crt** ファイルに保存されている。別のパスを使用する場合は、この手順で対応する手順を調整します。
- クライアントおよび Web サーバー **my_company.idm.example.com** を使用してサーバーのホスト名が Web サーバーのホスト名に対して解決されている。

手順

1. **mod_ssl** パッケージをインストールします。

```
# yum install mod_ssl
```

2. **/etc/httpd/conf.d/ssl.conf** ファイルを編集し、以下の設定を **<VirtualHost _default_:443>** ディレクティブに追加します。

   a. サーバー名を設定します。

   ```
   ServerName my_company.idm.example.com
   ```

   重要

   サーバー名は、証明書の **Common Name** フィールドに設定されているエントリーと一致している必要があります。

   a. 必要に応じて、証明書の **Subject Alt Names** (SAN) フィールドに追加のホスト名が含まれる場合に、これらのホスト名にも TLS 暗号化を提供するように **mod_ssl** を設定できます。これを設定するには、**ServerAliases** パラメーターと対応する名前を追加します。

   ```
   ServerAlias www.my_company.idm.example.com
   server.my_company.idm.example.com
   ```

   b. 秘密鍵、サーバー証明書、および CA 証明書へのパスを設定します。

   ```
   SSLCertificateKeyFile "/etc/pki/tls/private/httpd.key"
   SSLCertificateFile "/etc/pki/tls/certs/httpd.pem"
   SSLCACertificateFile "/etc/pki/tls/certs/ca.crt"
   ```

3. セキュリティー上の理由から、**root** ユーザーのみが秘密鍵ファイルにアクセスできるように設定します。

   ```
   # chown root:root /etc/pki/tls/private/httpd.key
   # chmod 600 //etc/pki/tls/private/httpd.key
   ```

   警告

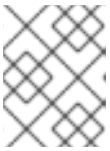
   秘密鍵に権限のないユーザーがアクセスした場合は、証明書を取り消し、新しい秘密鍵を作成し、新しい証明書を要求します。そうでない場合は、TLS 接続が安全ではなくなります。

4. **firewalld** を使用する場合は、ローカルのファイアウォールでポート **443** を開きます。

   ```
   # firewall-cmd --permanent --add-port=443/tcp
   # firewall-cmd --reload
   ```

5. **httpd** サービスを再起動します。

   ```
   # systemctl restart httpd
   ```



注記

パスワードで秘密鍵ファイルを保護した場合は、**httpd** サービスの起動時に毎回このパスワードを入力する必要があります。

- ブラウザーを使用して **https://my_company.idm.example.com** に接続します。

関連情報

- SSL/TLS 暗号化
- RHEL 8 における TLS のセキュリティー上の検討事項

20.8. APACHE HTTP サーバーでサポートされる TLS プロトコルバージョンの設定

デフォルトでは、RHEL の Apache HTTP Server は、最新のブラウザーにも互換性のある安全なデフォルト値を定義するシステム全体の暗号化ポリシーを使用します。たとえば、**DEFAULT** ポリシーでは、**TLSv1.2** および **TLSv1.3** プロトコルバージョンのみが Apache で有効になるように定義します。

Apache HTTP Server **my_company.idm.example.com** がサポートする TLS プロトコルのバージョンを手動で設定できます。たとえば、環境が特定の TLS プロトコルバージョンのみを有効にする必要がある場合には、以下の手順に従います。

- お使いの環境のクライアントで、セキュリティーの低い **TLS1** (TLSv1.0) プロトコルまたは **TLS1.1** プロトコルも使用できるようにする必要がある場合。
- Apache が **TLSv1.2** プロトコルまたは **TLSv1.3** プロトコルのみに対応するように設定する場合。

前提条件

- TLS 暗号化は、Apache HTTP Server への TLS 暗号化の追加 で説明されているように、**my_company.idm.example.com** サーバーで有効になっています。

手順

1. **/etc/httpd/conf/httpd.conf** ファイルを編集し、TLS プロトコルバージョンを設定する **<VirtualHost>** ディレクティブに以下の設定を追加します。たとえば、**TLSv1.3** プロトコルのみを有効にするには、以下を実行します。

```
SSLProtocol -All TLSv1.3
```

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. 以下のコマンドを使用して、サーバーが **TLSv1.3** に対応していることを確認します。

```
# openssl s_client -connect example.com:443 -tls1_3
```

2. 以下のコマンドを使用して、サーバーが **TLSv1.2** に対応していないことを確認します。

```
# openssl s_client -connect example.com:443 -tls1_2
```

サーバーがプロトコルに対応していない場合には、このコマンドは以下のエラーを返します。

```
140111600609088:error:1409442E:SSL routines:ssl3_read_bytes:tlsv1 alert protocol
version:ssl/record/rec_layer_s3.c:1543:SSL alert number 70
```

3. 必要に応じて、他の TLS プロトコルバージョンのコマンドを繰り返し実行します。

関連情報

- **update-crypto-policies(8)** の man ページ
- Using system-wide cryptographic policies.
- **SSLProtocol** パラメーターの詳細については、Apache マニュアルの **mod_ssl** のドキュメント Apache HTTP Server マニュアルのインストール を参照してください。

20.9. APACHE HTTP サーバーで対応している暗号の設定

デフォルトでは、Apache HTTP サーバーは、安全なデフォルト値を定義するシステム全体の暗号化ポリシーを使用します。これは、最近のブラウザーとも互換性があります。システム全体の暗号化で使用可能な暗号化のリストは、**/etc/crypto-policies/back-ends/openssl.config** ファイルを参照してください。

Apache HTTP Server **my_company.idm.example.com** がサポートする暗号を手動で設定できます。お使いの環境で特定の暗号が必要な場合は、以下の手順に従います。

前提条件

- TLS 暗号化は、Apache HTTP Server への TLS 暗号化の追加 で説明されているように、**my_company.idm.example.com** サーバーで有効になっています。

手順

1. **/etc/httpd/conf/httpd.conf** ファイルを編集し、TLS 暗号を設定する **<VirtualHost>** ディレクティブに **SSLCipherSuite** パラメーターを追加します。

```
SSLCipherSuite
"EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH:!SHA1:!SHA256"
```

この例では、**EECDH+AESGCM**、**EDH+AESGCM**、**AES256+EECDH**、および **AES256+EDH** 暗号のみを有効にし、**SHA1** および **SHA256** メッセージ認証コード (MAC) を使用するすべての暗号を無効にします。

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. Apache HTTP Server が対応する暗号化のリストを表示するには、以下を行います。

   a. **nmap** パッケージをインストールします。

   ```
   # yum install nmap
   ```

   b. **nmap** ユーティリティーを使用して、対応している暗号を表示します。

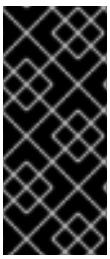
   ```
   # nmap --script ssl-enum-ciphers -p 443 example.com
   ...
   PORT    STATE SERVICE
   443/tcp open  https
   | ssl-enum-ciphers:
   |   TLSv1.2:
   |     ciphers:
   |       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (ecdh_x25519) - A
   |       TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (dh 2048) - A
   |       TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (ecdh_x25519) - A
   ...
   ```

関連情報

- **update-crypto-policies(8)** の man ページ
- Using system-wide cryptographic policies.
- Apache HTTP Server マニュアルのインストール - SSLCipherSuite

20.10. TLS クライアント証明書認証の設定

クライアント証明書認証を使用すると、管理者は、証明書を使用して認証したユーザーだけが Web サーバー **my_company.idm.example.com** のリソースにアクセスできるようにすることが可能です。**/var/www/html/Example/** ディレクトリーにクライアント証明書認証を設定できます。



重要

Apache サーバー **my_company.idm.example.com** が TLS 1.3 プロトコルを使用する場合は、一部のクライアントに追加の設定が必要です。たとえば、Firefox で、**about:config** メニューの **security.tls.enable_post_handshake_auth** パラメーターを **true** に設定します。詳細は、Transport Layer Security version 1.3 in Red Hat Enterprise Linux 8 を参照してください。

前提条件

- TLS 暗号化は、Apache HTTP Server への TLS 暗号化の追加 で説明されているように、**my_company.idm.example.com** サーバーで有効になっています。

手順

1. **/etc/httpd/conf/httpd.conf** ファイルを編集し、以下の設定をクライアント認証を設定する **<VirtualHost>** ディレクティブに追加します。

```
<Directory "/var/www/html/Example/">
  SSLVerifyClient require
</Directory>
```

SSLVerifyClient require の設定では、**/var/www/html/Example/** ディレクトリーのコンテンツにクライアントがアクセスする前に、サーバーがクライアント証明書を正常に検証する必要があることを定義します。

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. **curl** ユーティリティーを使用して、クライアント認証なしで URL **https://my_company.idm.example.com/Example/** にアクセスします。

```
$ curl https://my_company.idm.example.com/Example/
curl: (56) OpenSSL SSL_read: error:1409445C:SSL routines:ssl3_read_bytes:tlsv13 alert certificate required, errno 0
```

このエラーから、Web サーバー **my_company.idm.example.com** にクライアント証明書認証が必要であることが分かります。

2. クライアントの秘密鍵と証明書、および CA 証明書を **curl** に指定して、クライアント認証で同じ URL にアクセスします。

```
$ curl --cacert ca.crt --key client.key --cert client.crt
https://my_company.idm.example.com/Example/
```

要求に成功すると、**curl** は **/var/www/html/Example/** ディレクトリーに保存されている **index.html** ファイルを表示します。

関連情報

- Apache HTTP Server マニュアルのインストール - mod_ssl 設定

20.11. 新しいユーザー証明書を要求し、クライアントにエクスポート

Identity Management (IdM) 管理者は、Web ブラウザーを使用してサーバーにアクセスするユーザーに対して、特定の IdM サブ CA が発行する証明書での認証を求めるように IdM クライアントで実行中の Web サーバーを設定できます。特定の IdM サブ CA からユーザー証明書を要求し、ユーザーが Web ブラウザー経由で Web サーバーにアクセスするホストにその証明書と対応する秘密鍵をエクスポートするには、本セクションを実行するには、次の手順に従います。その後、ブラウザーに証明書と秘密鍵をインポート します。

手順

1. 必要に応じて、新しいディレクトリー (例: ~/**certdb**/) を作成し、証明書の一時データベースを作成します。要求されたら、NSS 証明書の DB パスワードを作成し、後続の手順で生成される証明書への鍵を暗号化します。

```
# mkdir ~/certdb/
```

```
# certutil -N -d ~/certdb/
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
```

2. 証明書署名要求 (CSR) を作成し、その出力をファイルにリダイレクトします。たとえば、**IDM.EXAMPLE.COM** レルムの **idm_user** ユーザーの **4096** ビット証明書に対して、**certificate_request.csr** という名前の CSR を作成する場合は、判別を簡単にするために、証明書の秘密鍵のニックネームを **idm_user** に設定し、発行先を **CN=idm_user,O=IDM.EXAMPLE.COM** に設定します。

```
# certutil -R -d ~/certdb/ -a -g 4096 -n idm_user -s "CN=idm_user,O=IDM.EXAMPLE.COM"
> certificate_request.csr
```

3. プロンプトが表示されたら、**certutil** を使用して一時データベースを作成したときに入力したパスワードを入力します。その後、止めるように言われるまで、ランダムにタイピングし続けます。

```
Enter Password or Pin for "NSS Certificate DB":

A random seed must be generated that will be used in the
creation of your key.  One of the easiest ways to create a
random seed is to use the timing of keystrokes on a keyboard.

To begin, type keys on the keyboard until this progress meter
is full.  DO NOT USE THE AUTOREPEAT FUNCTION ON YOUR KEYBOARD!


Continue typing until the progress meter is full:
```

4. 証明書要求ファイルをサーバーに送信します。新しく発行した証明書に関連付ける Kerberos プリンシパルと、証明書を保存する出力ファイルを指定し、必要に応じて証明書のプロファイルを指定します。証明書を発行する IdM サブ CA を指定します。たとえば、**idm_user@IDM.EXAMPLE.COM** プリンシパルの **IECUserRoles** プロファイル (ユーザーロール拡張を追加したプロファイル) の証明書を **webclient-ca** から取得して、**~/idm_user.pem** ファイルに保存するには、次のコマンドを実行します。

```
# ipa cert-request certificate_request.csr --principal=idm_user@IDM.EXAMPLE.COM --
profile-id=IECUserRoles --ca=webclient-ca --certificate-out=~/idm_user.pem
```

5. 証明書を NSS データベースに追加します。証明書が NSS データベースの秘密鍵に一致するように、CSR を作成する際に使用したニックネームを設定するには、**-n** オプションを使用します。**-t** オプションは信頼レベルを設定します。詳細は、certutil(1) man ページを参照してください。**-i** オプションは、入力証明書ファイルを指定します。たとえば、**idm_user** ニックネームを持つ証明書を NSS データベースに追加するには、次のコマンドを実行します。証明書は、**~/certdb/** データベースの **~/idm_user.pem** ファイルに保存されます。

```
# certutil -A -d ~/certdb/ -n idm_user -t "P,," -i ~/idm_user.pem
```

6. NSS データベースの鍵で、ニックネームが **(orphan)** と表示されていないことを確認します。たとえば、~/**certdb**/ データベースに保存されている証明書で、対応する鍵が存在することを確認するには、以下のコマンドを実行します。

```
# certutil -K -d ~/certdb/
< 0> rsa      5ad14d41463b87a095b1896cf0068ccc467df395   NSS Certificate DB:idm_user
```

7. 証明書を、NSS データベースから PKCS12 形式にエクスポートするには、**pk12util** コマンドを使用します。たとえば、NSS データベース **/root/certdb** から ~/**idm_user.p12** ファイルへ、**idm_user** ニックネームを持つ証明書をエクスポートする場合は、次のコマンドを実行します。

```
# pk12util -d ~/certdb -o ~/idm_user.p12 -n idm_user
Enter Password or Pin for "NSS Certificate DB":
Enter password for PKCS12 file:
Re-enter password:
pk12util: PKCS12 EXPORT SUCCESSFUL
```

8. **idm_user** の証明書認証を有効にするホストに、証明書を転送します。

```
# scp ~/idm_user.p12 idm_user@client.idm.example.com:/home/idm_user/
```

9. セキュリティー上の理由から、証明書が転送されたホストの、.pkcs12 ファイルが格納されているディレクトリーに、other グループがアクセスできないようにします。

```
# chmod o-rwx /home/idm_user/
```

10. セキュリティー上の理由から、一時 NSS データベースおよび .pkcs12 ファイルを、サーバーから削除します。

```
# rm ~/certdb/
# rm ~/idm_user.p12
```

20.12. 証明書認証を有効にするためのブラウザーの設定

Web UI を使用した Identity Management (IdM) へのログイン時に証明書で認証できるようにするには、ユーザーおよび関連の認証局 (CA) 証明書を Mozilla Firefox または Google Chrome ブラウザーにインポートする必要があります。ブラウザーが実行しているホスト自体は、IdM ドメインの一部である必要はありません。

IdM が Web UI への接続をサポートしているブラウザーは以下のとおりです。

- Mozilla Firefox 38 以降
- Google Chrome 46 以降

次の手順は、Mozilla Firefox 57.0.1 ブラウザーを設定する方法を説明します。

前提条件

- PKCS #12 形式で自由にブラウザーにインポートできる ユーザー証明書 がある。
- サブ CA 証明書をダウンロード し、PEM 形式で自由に使用できるようにしている。

手順

1. Firefox を開き、**設定** → **プライバシーとセキュリティー** に移動します。

図20.8 設定のプライバシーおよびセキュリティーセクション

2. **証明書を表示** をクリックします。

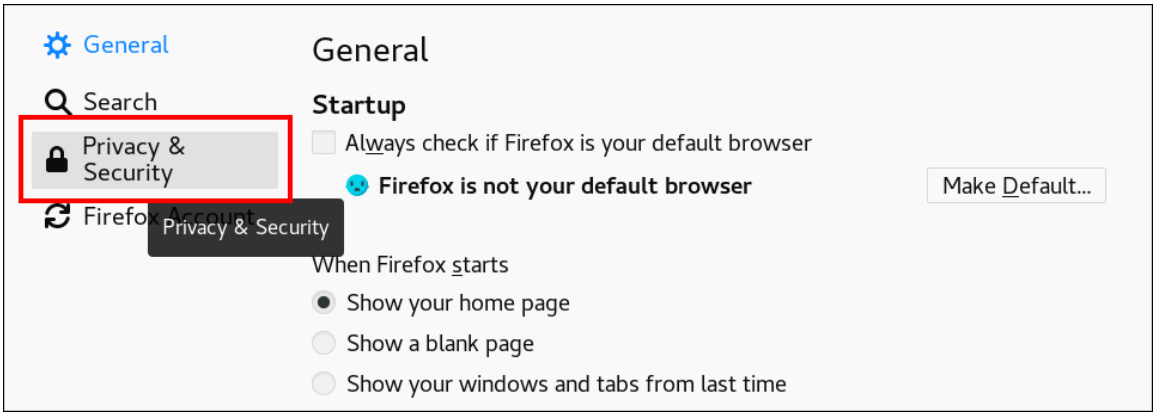
図20.9 プライバシーおよびセキュリティーで証明書を表示

3. **あなたの証明書** タブで、**インポート** をクリックします。PKCS12 形式のユーザー証明書を見つけて開きます。**OK** をクリックし、**OK** をクリックします。

4. IdM サブ CA が信頼された認証局として Firefox に認識されていることを確認するには、IdM WebUI からのサブ CA 証明書のダウンロード に保存した IdM サブ CA 証明書を信頼された認証局証明書としてインポートします。

   a. Firefox を起動し、設定に移動して、**プライバシーおよびセキュリティー**に移動します。

図20.10 設定のプライバシーおよびセキュリティーセクション



   b. **証明書を表示** をクリックします。

図20.11 プライバシーおよびセキュリティーで証明書を表示

c. **認証機関** タブで、**インポート** をクリックします。サブ CA 証明書を探して開きます。証明書を信頼し、Web サイトを識別したら、**OK** をクリックし、**OK** をクリックします。

第21章 関連する証明書の特定グループの迅速な無効化

システム管理者は任意で、関連する証明書の特定のグループをすばやく無効にできます。

- 特定の軽量の Identity Management (IdM) サブ CA が発行した証明書のみを信頼するようにアプリケーションを設計します。その後、これらの証明書を発行した Identity Management (IdM) サブ CA の証明書を取り消すだけで、この証明書をすべて無効にできます。IdM で軽量のサブ CA を作成して使用する方法の詳細は、関連する証明書の特定グループの迅速な無効化 を参照してください。

- 失効予定の IdM サブ CA が発行した全証明書がすぐに無効となっていることを確認するには、このような証明書に依存するアプリケーションが IdM OCSP レスポンダーを使用するように設定します。たとえば、Firefox ブラウザーが OCSP レスポンダーを使用するように設定するには、Firefox の設定で **Query OCSP responder servers to confirm the current validity of certificates** チェックボックスが選択されているようにします。
  IdM では、証明書失効リスト (CRL) が 4 時間ごとに更新されます。IdM サブ CA によって発行されたすべての証明書を無効にするには、IdM サブ CA 証明書を失効 させます。さらに、関連する CA ACL を無効 にして、IdM サブ CA の無効化 の検討も行ってください。サブ CA を無効にするとサブ CA が新しい証明書を発行できなくなりますが、サブ CA の署名キーが保持されるため、以前に発行した証明書に、オンライン証明書ステータスプロトコル (OCSP) の応答を生成することができます。

重要

お使いの環境で OCSP を使用する場合は、サブ CA を削除しないでください。サブ CA を削除するとサブ CA の署名キーが削除されるため、そのサブ CA が発行する証明書の OCSP 応答を生成できなくなります。

サブ CA の無効化が唯一推奨されるのは、署名キーを新しくしつつも、同じサブジェクト識別名 (DN) を使用して、サブ CA を新規作成するシナリオの場合のみです。

21.1. IDM CLI での CA ACL の無効化

IdM サービスまたは IdM サービスのグループを終了する場合は、対応する既存の CA ACL を無効にすることを検討してください。

Web サーバーが IdM クライアントで実行中の場合には IdM サブ CA (**webserver-ca**) 発行の証明書を要求するように制限する TLS_web_server_authentication CA ACL と、IdM ユーザーの場合には IdM サブ CA (**webclient-ca**) 発行のユーザー証明書を要求するように制限する TLS_web_client_authentication CA ACL を無効にするには、次の手順に従います。

手順

1. 必要に応じて、IdM 環境内の CA ACL をすべて表示するには、**ipa caacl-find** コマンドを入力します。

```
$ ipa caacl-find
-----------------
3 CA ACLs matched
-----------------
  ACL name: hosts_services_caIPAserviceCert
  Enabled: TRUE

  ACL name: TLS_web_server_authentication
  Enabled: TRUE
```

```
  ACL name: TLS_web_client_authentication
  Enabled: TRUE
```

2. 必要に応じて、CA ACL の詳細を表示するには、**ipa caacl-show** コマンドを入力して、CA ACL 名を指定します。

```
$ ipa caacl-show TLS_web_server_authentication
  ACL name: TLS_web_server_authentication
  Description: CAACL for web servers authenticating to web clients using certificates issued by webserver-ca
  Enabled: TRUE
  CAs: webserver-ca
  Profiles: caIPAserviceCert
  Services: HTTP/rhel8server.idm.example.com@IDM.EXAMPLE.COM
```

3. CA ACL を無効にするには、**ipa caacl-disable** コマンドを入力して、CA ACL 名を指定します。

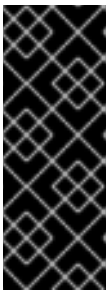
   - **TLS_web_server_authentication** CA ACL を無効にするには、以下を入力します。

```
$ ipa caacl-disable TLS_web_server_authentication
---------------------------------------------
Disabled CA ACL "TLS_web_server_authentication"
---------------------------------------------
```

   - **TLS_web_client_authentication** CA ACL を無効にするには、以下を入力します。

```
$ ipa caacl-disable TLS_web_client_authentication
---------------------------------------------
Disabled CA ACL "TLS_web_client_authentication"
---------------------------------------------
```

有効な CA ACL は **hosts_services_caIPAserviceCert** CA ACL のみです。



重要

CA ACL **hosts_services_caIPAserviceCert** を無効にする場合には、細心の注意を払ってください。プロファイルが **caIPAserviceCert** の **ipa** CA を IdM サーバーで使用できるように指定する別の CA ACL を用意せずに **hosts_services_caIPAserviceCert** を無効にすると、IdM の **HTTP** と **LDAP** 証明書の更新に失敗します。IdM **HTTP** 証明書および **LDAP** 証明書の期限が切れると、最終的には IdM システムに問題が発生します。

21.2. IDM サブ CA の無効化

IdM サブ CA の CA 証明書を無効にして、そのサブ CA によって発行されたすべての証明書を無効にした後、IdM サブ CA が必要なくなった場合は、IdM サブ CA を無効にすることを検討してください。サブ CA は後で再度有効にできます。

サブ CA を無効にするとサブ CA が新しい証明書を発行できなくなりますが、サブ CA の署名キーが保持されるため、以前に発行した証明書に、オンライン証明書ステータスプロトコル (OCSP) の応答を生成することができます。

前提条件

- IdM 管理者としてログインしている。

手順

- **ipa ca-disable** コマンドを入力し、サブ CA の名前を指定します。

```
$ ipa ca-disable webserver-CA
--------------------
Disabled CA "webserver-CA"
--------------------
```

第22章 ANSIBLE を使用した IDM 証明書の管理

ansible-freeipa ipacert モジュールを使用して、Identity Management (IdM) ユーザー、ホスト、およびサービスの SSL 証明書を要求、取り消し、および取得できます。保留された証明書を復元することもできます。

22.1. ANSIBLE を使用した IDM ホスト、サービス、ユーザーの SSL 証明書の要求

ansible-freeipa ipacert モジュールを使用して、Identity Management (IdM) ユーザー、ホスト、およびサービスの SSL 証明書を要求できます。その後、これらの証明書を使用して IdM に対する認証を行うことができます。

Ansible Playbook を使用して IdM 認証局 (CA) から HTTP サーバーの証明書を要求するには、この手順を完了します。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - **secret.yml** Ansible vault に **ipaadmin_password** が保存されている。
- IdM デプロイメントに統合 CA がある。

手順

1. ユーザー、ホスト、またはサービスの証明書署名要求 (CSR) を生成します。たとえば、**openssl** ユーティリティーを使用して client.idm.example.com で実行されている **HTTP** サービスの CSR を生成するには、次のように入力します。

   ```
   # openssl req -new -newkey rsa:2048 -days 365 -nodes -keyout new.key -out new.csr -subj '/CN=client.idm.example.com,O=IDM.EXAMPLE.COM'
   ```

   その結果、CSR は **new.csr** に保存されます。

2. 次の内容を含む Ansible Playbook ファイル **request-certificate.yml** を作成します。

   ```
   ---
   - name: Playbook to request a certificate
     hosts: ipaserver

     vars_files:
     - /home/user_name/MyPlaybooks/secret.yml

     tasks:
     - name: Request a certificate for a web server
       ipacert:
         ipaadmin_password: "{{ ipaadmin_password }}"
   ```

```
    state: requested
    csr: |
      -----BEGIN CERTIFICATE REQUEST-----

MIGYMEwCAQAwGTEXMBUGA1UEAwwOZnJlZWlwYSBydWxlcyEwKjAFBgMrZXADIQBs
      HlqIr4b/XNK+K8QLJKIzfvuNK0buBhLz3LAzY7QDEqAAMAUGAytlcANBAF4oSCbA
      5aIPukCidnZJdr491G4LBE+URecYXsPknwYb+V+ONnf5ycZHyaFv+jkUBFGFeDgU
      SYaXm/gF8cDYjQI=
      -----END CERTIFICATE REQUEST-----
    principal: HTTP/client.idm.example.com
  register: cert
```

証明書要求を **new.csr** の CSR に置き換えます。

3. 証明書を要求します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/request-
certificate.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの cert モジュール

22.2. ANSIBLE を使用して IDM ホスト、サービス、ユーザーの SSL 証明書を取り消す

ansible-freeipa ipacert モジュールを使用して、Identity Management (IdM) ユーザー、ホスト、およびサービスが IdM への認証に使用する SSL 証明書を取り消すことができます。

Ansible Playbook を使用して HTTP サーバーの証明書を取り消すには、この手順を完了します。証明書を取り消す理由は “keyCompromise” です。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - **secret.yml** Ansible vault に **ipaadmin_password** が保存されている。
  - **openssl x509 -noout -text -in <path_to_certificate>** コマンドを入力するなどして、証明書のシリアル番号を取得している。この例では、証明書のシリアル番号は 123456789 です。
- IdM デプロイメントに統合 CA がある。

手順

1. 次の内容を含む Ansible Playbook ファイル **revoke-certificate.yml** を作成します。

```
---
- name: Playbook to revoke a certificate
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Revoke a certificate for a web server
    ipacert:
      ipaadmin_password: "{{ ipaadmin_password }}"
      serial_number: 123456789
      revocation_reason: "keyCompromise"
      state: revoked
```

2. 証明書を取り消します。

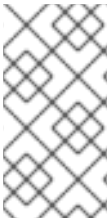
```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/revoke-
certificate.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの cert モジュール
- RFC 5280 の Reason Code

22.3. ANSIBLE を使用して IDM ユーザー、ホスト、およびサービスの SSL 証明書を復元する

ansible-freeipa ipacert モジュールを使用すると、Identity Management (IdM) ユーザー、ホスト、またはサービスが IdM への認証に以前に使用した、取り消された SSL 証明書を復元できます。



注記

復元できるのは、保留された証明書のみです。たとえば、秘密キーを紛失したかどうかがわからないなどの理由で、秘密キーを保留した可能性があります。ただし、キーを回復し、回復までの間に誰もそのキーにアクセスしていないと確信しているため、証明書を復元したいと考えています。

Ansible Playbook を使用して、IdM に登録されたサービスの証明書を保留から解放するには、この手順を完了します。この例では、HTTP サービスの証明書の保留を解除する方法について説明します。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。

- **secret.yml** Ansible vault に **ipaadmin_password** が保存されている。

- IdM デプロイメントに統合 CA がある。
- たとえば、**openssl x509 -noout -text -in path/to/certificate** コマンドを入力するなどして、証明書のシリアル番号を取得している。この例では、証明書のシリアル番号は **123456789** です。

手順

1. 次の内容を含む Ansible Playbook ファイル **restore-certificate.yml** を作成します。

```
---
- name: Playbook to restore a certificate
  hosts: ipaserver
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Restore a certificate for a web service
    ipacert:
      ipaadmin_password: "{{ ipaadmin_password }}"
      serial_number: 123456789
      state: released
```

2. Playbook を実行します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/restore-
certificate.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの cert モジュール

22.4. ANSIBLE を使用して IDM ユーザー、ホスト、およびサービスの SSL 証明書を取得する

ansible-freeipa ipacert モジュールを使用すると、Identity Management (IdM) ユーザー、ホスト、またはサービスに対して発行された SSL 証明書を取得し、マネージドノード上のファイルに保存できます。

前提条件

- コントロールノードでは、
  - Ansible バージョン 2.14 以降を使用している。
  - **ansible-freeipa** パッケージをインストールしている。
  - **~/MyPlaybooks/** ディレクトリーに、IdM サーバーの完全修飾ドメイン名 (FQDN) を使用して Ansible インベントリーファイル を作成した。
  - **secret.yml** Ansible vault に **ipaadmin_password** が保存されている。

- **openssl x509 -noout -text -in <path_to_certificate>** コマンドを入力するなどして、証明書のシリアル番号を取得している。この例では、証明書のシリアル番号は 123456789 で、取得した証明書を保存するファイルは **cert.pem** です。

手順

1. 次の内容を含む Ansible Playbook ファイル **retrieve-certificate.yml** を作成します。

```
---
- name: Playbook to retrieve a certificate and store it locally on the managed node
  hosts: ipaserver

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml

  tasks:
  - name: Retrieve a certificate and save it to file 'cert.pem'
    ipacert:
      ipaadmin_password: "{{ ipaadmin_password }}"
      serial_number: 123456789
      certificate_out: cert.pem
      state: retrieved
```

2. 証明書を取得します。

```
$ ansible-playbook --vault-password-file=password_file -v -i
<path_to_inventory_directory>/hosts <path_to_playbooks_directory>/retrieve-
certificate.yml
```

関連情報

- **ansible-freeipa** アップストリームドキュメントの cert モジュール

第23章 IDM HEALTHCHECK を使用した証明書の検証

Identity Management (IdM) の Healthcheck ツールを使用し、**certmonger** によって維持されている IPA 証明書の問題を特定する方法について詳しく説明します。

詳細は IdM のヘルスチェック を参照してください。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

23.1. IDM 証明書の HEALTHCHECK テスト

Healthcheck ツールには、Identity Management (IdM) の certmonger が維持する証明書の状況を確認するさまざまなテストが含まれています。certmonger の詳細は、certmonger を使用してサービスの IdM 証明書の取得 を参照してください。

この一連のテストでは、有効期限、検証、信頼性、その他の問題を確認します。根本的な問題 1 つに対して、複数のエラーが発生する可能性があります。

すべての証明書テストを表示するには、**--list-sources** オプションを指定して **ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

すべてのテストは、**ipahealthcheck.ipa.certs** ソースの下にあります。

IPACertmongerExpirationCheck

このテストでは、**certmonger** の有効期限を確認します。
証明書の有効期限が切れている場合は、エラーが報告されます。

証明書の有効期限が間近な場合は、警告が表示されます。デフォルトでは、このテストは、証明書の有効期限が 28 日以内のものを対象としています。

/etc/ipahealthcheck/ipahealthcheck.conf ファイルで日数を設定できます。ファイルを開いた後、デフォルトセクションにある **cert_expiration_days** オプションを変更します。

注記

certmonger は、証明書の有効期限に関する独自のビューをロードして維持します。このチェックでは、ディスク上の証明書は検証されません。

IPACertfileExpirationCheck

このテストでは、証明書ファイルまたは NSS データベースを開けないかどうかを確認します。このテストでは、有効期限も確認します。そのため、エラーまたは警告出力の **msg** 属性をよく読んでください。このメッセージは問題を特定するものです。

注記

このテストでは、ディスク上の証明書が確認されます。証明書がない、読み取りができないなどの問題が発生した場合は、別のエラーが出力される可能性があります。

IPACertNSSTrust

このテストでは、NSS データベースに保存されている証明書の信頼を比較します。NSS データベースで期待される、追跡される証明書では、期待される値と信頼が比較されます。一致しないとエラーが発生します。

IPANSSChainValidation

このテストでは、NSS 証明書の証明書チェーンを検証します。テストでは、**certutil -V -u V -e -d [dbdir] -n [nickname]** を実行します。

IPAOpenSSLChainValidation

このテストでは、OpenSSL 証明書の証明書チェーンを検証します。**NSSChain** 検証と比較するために、実行する OpenSSL コマンドを以下に示します。

```
openssl verify -verbose -show_chain -CAfile /etc/ipa/ca.crt [cert file]
```

IPARAAgent

このテストでは、ディスク上の証明書を、**uid=ipara,ou=People,o=ipaca** の LDAP の同等のレコードと比較します。

IPACertRevocation

このテストでは、certmonger を使用して、証明書が取り消されていないことを確認します。したがって、テストでは certmonger でのみメンテナンスされる証明書に接続している問題を見つけることができます。

IPACertmongerCA

このテストでは、certmonger の認証局 (CA) の設定を検証します。IdM は、CA を使用しない証明書を発行できません。
certmonger は、CA ヘルパーのセットを維持します。IdM には、IPA という名前の CA があります。IPA は、IdM を介して証明書を発行し、ホストまたはサービスの証明書に対して、ホストまたはユーザーのプリンシパルとして認証します。

また、CA サブシステム証明書を更新する **dogtag-ipa-ca-renew-agent** および **dogtag-ipa-ca-renew-agent-reuse** があります。

注記

問題を確認するには、すべての IdM サーバーで上記のテストを実行します。

23.2. HEALTHCHECK ツールを使用した証明書のスクリーニング

Healthcheck ツールを使用して Identity Management (IdM) 証明書ヘルスチェックのスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck ツールには多くのテストが含まれているため、以下の方法で結果を短くすることができます。

- 成功したテストをすべて除外する - **--failures-only**
- 証明書テストのみを含める: **--source=ipahealthcheck.ipa.certs**

前提条件

- **root** ユーザーとして Healthcheck テストを実行する必要があります。

手順

- 証明書に関する警告、エラー、および重大な問題について Healthcheck を実行するには、次のコマンドを実行します。

```
# ipa-healthcheck --source=ipahealthcheck.ipa.certs --failures-only
```

テストに成功すると、空の括弧が表示されます。

```
[]
```

失敗したテストでは、以下の出力が表示されます。

```
{
  "source": "ipahealthcheck.ipa.certs",
  "check": "IPACertfileExpirationCheck",
  "result": "ERROR",
  "kw": {
    "key": 1234,
    "dbdir": "/path/to/nssdb",
    "error": [error],
    "msg": "Unable to open NSS database '/path/to/nssdb': [error]"
  }
}
```

上記の **IPACertfileExpirationCheck** テストは、NSS データベースを開くときに失敗しています。

関連情報

- **man ipa-healthcheck** を参照してください。

第24章 IDM HEALTHCHECK を使用したシステム証明書の検証

Healthcheck ツールを使用して Identity Management (IdM) のシステム証明書の問題を特定する方法について詳しく説明します。

詳細は IdM のヘルスチェック を参照してください。

前提条件

- Healthcheck ツールは、RHEL 8.1 以降でのみ利用できます。

24.1. システム証明書の HEALTHCHECK テスト

Healthcheck ツールには、システム (DogTag) 証明書を検証するさまざまなテストがあります。

すべてのテストを表示するには、**--list-sources** オプションを指定して **ipa-healthcheck** を実行します。

```
# ipa-healthcheck --list-sources
```

すべてのテストは、**ipahealthcheck.dogtag.ca** ソースの下にあります。

DogtagCertsConfigCheck
: このテストでは、NSS データベース内の CA (認証局) 証明書を、**CS.cfg** に保存されている同じ値と比較します。一致しない場合、CA は起動に失敗します。
具体的には、以下を確認します。

  - **ca.audit_signing.cert** の場合は **auditSigningCert cert-pki-ca**
  - **ca.ocsp_signing.cert** の場合は **ocspSigningCert cert-pki-ca**
  - **ca.signing.cert** の場合は **caSigningCert cert-pki-ca**
  - **ca.subsystem.cert** の場合は **subsystemCert cert-pki-ca**
  - **ca.sslserver.cert** の場合は **Server-Cert cert-pki-ca**

  Key Recovery Authority (KRA) がインストールされている場合は、以下を確認します。

  - **ca.connector.KRA.transportCert** の場合は **transportCert cert-pki-kra**

DogtagCertsConnectivityCheck
: このテストでは、接続性を検証します。このテストは、以下の確認を行う **ipa cert-show 1** コマンドと同等です。

  - Apache の PKI プロキシー設定
  - IdM が CA を検出できること
  - RA エージェントクライアント証明書
  - 要求に対する CA 返信の正確性

  このテストでは、**cert-show** を実行して CA から期待される結果 (証明書または not found) が返されることを確認する必要があるため、シリアル番号 #1 の証明書がチェックされます。

注記

問題を確認するには、すべての IdM サーバーで上記のテストを実行してください。

24.2. HEALTHCHECK を使用したシステム証明書のスクリーニング

Healthcheck ツールを使用して Identity Management (IdM) 証明書のスタンドアロン手動テストを実行するには、次の手順に従います。

Healthcheck ツールには多くのテストが含まれているため、Dogtag テスト (**--source=ipahealthcheck.dogtag.ca**) のみを含めることで結果を絞り込むことができます。

手順

- DogTag 証明書に限定して Healthcheck を実行するには、次のように入力します。

```
# ipa-healthcheck --source=ipahealthcheck.dogtag.ca
```

テストに成功すると、以下のようになります。

```
{
  "source: ipahealthcheck.dogtag.ca",
  "check: DogtagCertsConfigCheck",
  "result: SUCCESS",
  "uuid: 9b366200-9ec8-4bd9-bb5e-9a280c803a9c",
  "when: 20191008135826Z",
  "duration: 0.252280",
  "kw:" {
    "key": "Server-Cert cert-pki-ca",
    "configfile":  "/var/lib/pki/pki-tomcat/conf/ca/CS.cfg"
    }
}
```

テストに失敗すると、以下のようになります。

```
{
  "source: ipahealthcheck.dogtag.ca",
  "check: DogtagCertsConfigCheck",
  "result: CRITICAL",
  "uuid: 59d66200-1447-4b3b-be01-89810c803a98",
  "when: 20191008135912Z",
  "duration: 0.002022",
  "kw:" {
    "exception": "NSDB /etc/pki/pki-tomcat/alias not initialized",
    }
}
```

関連情報

- **man ipa-healthcheck** を参照してください。

第25章 IDM が内部で使用する証明書について

Red Hat Identity Management (IdM) サーバーは、統合認証局 (CA) を使用してインストールすることも、CA を使用せずにインストールすることもできます。IdM へのアクセスおよび管理に必要な証明書は、CA が統合されているかどうかによって異なる方法で管理されます。

- 統合 CA: 証明書は **certmonger** によって自動的に作成され、追跡されます。**certmonger** は自動的に証明書を更新し、IdM サービスが引き続き有効であることを確認します。
- CA がない場合: 証明書はサードパーティーの認証局から要求されます。この場合は、有効期限を監視し、それらが更新されて、IdM サービスが引き続き有効であることを確認する必要があります。

25.1. IDM の内部証明書について

Red Hat Identity Management (IdM) は、LDAP サーバーと HTTP サーバーなど、ネットワークを使用してアクセスする多くのサービスを使用します。サーバー証明書を必要とする SSL/TLS ポートを使用して、これらのサービスにアクセスします。IdM サーバーのインストール時に、HTTP および LDAP サーバー証明書が必要です。

IdM のインストールおよび設定方法に応じて、複数の方法で証明書を取得できます。

- 自己署名または外部 CA が署名できる統合 CA の場合: IdM は、IdM が管理するユーザー、ホスト、およびサービスのすべての証明書を発行し、証明書ファイルを提供する必要はありません。
  certmonger は、証明書の有効期限を自動的に監視し、必要に応じて自動的に更新されます。
- 外部署名 CA の場合: インストールは複数の手順で実行されます。
  - CSR を生成するには、**--external-ca** オプションを指定してインストールを実行する必要があります。
  - CSR を外部 CA に送信し、発行した証明書および CA 証明書チェーンを PEM ファイルまたは Base64 でエンコードされた証明書として取得します。
  - 新たに発行した CA 証明書および CA チェーンファイルの場所と名前を指定して、IdM サーバーインストールを再度実行します。IdM 認証局は外部 CA の subCA として設定されており、この subCA は必要な HTTP および LDAP サーバー証明書を発行します。
    certmonger は、証明書の有効期限を自動的に監視し、必要に応じて自動的に更新されます。
- CA がない場合: サードパーティーの認証局から以下の証明書を要求する必要があります。
  - LDAP サーバー証明書
  - Apache サーバー証明書
  - PKINIT 証明書
  - LDAP および Apache のサーバー証明書を発行した CA の完全な CA 証明書チェーン
    これらの証明書は **certmonger** によって追跡されず、管理者には有効期限に達する前にそれらを更新する責任があります。

関連情報

- CA サービスの計画

25.2. IDM の内部の証明書

内部証明書は、IdM のインストール方法と、そのインストールに含まれるコンポーネントによって異なります。インストールによっては、以下の証明書がシステムに保存されている可能性があります。

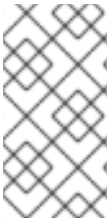
IdM CA 認証

IdM CA 証明書は、その他のすべての証明書に署名するために IdM によって使用されます。CA なしのインストールには存在しないことに注意してください。

caSigningCert	説明
ファイルシステムの場所	• **/etc/pki/pki-tomcat/alias** NSS データベースの **nickname=caSigningCert cert-pki-ca** • **/etc/ipa/nssdb/** および **/etc/ipa/ca.crt** の **nickname=REALM.NAME IPA CA** (LDAP から入力)
LDAP の場所	**cn=REALM.NAME IPA CA,cn=certificates,cn=ipa,cn=etc,dc=realm,dc=name and ou=authorities,ou=ca,o=ipaca**
Issuer	自己署名または外部 CA により署名されている
Subject	**O = REALM.NAME, CN = Certificate Authority** これはデフォルト値ですが、IdM サーバーのインストール時にカスタマイズできることに注意してください。
関連情報	**CA:true** 重大な制約が必要で、NSS データベースに **CT,C,C** 信頼フラグが必要です。

外部 CA 証明書

外部 CA を使用している場合は、IdM 証明書を検証するために IdM で外部 CA のチェーンが利用可能である必要があります。CA なしのインストールでは、HTTPD および LDAP 証明書を検証するために、LDAP や **/etc/ipa/ca.crt** ディレクトリーを含むさまざまな場所に、外部 CA 証明書が存在する必要があります。



注記

インストール時に自動的に行われるため、外部 CA 証明書を必要なすべての場所に手動で追加する必要はありません。ただし、外部 CA 証明書が後で更新された場合は、Renewing the IdM CA renewal server certificate using an external CA の手順に従って、新しい証明書が必要なすべての場所に追加されるようにします。

外部証明書	説明
ファイルシステムの場所	**/etc/pki/pki-tomcat/alias nssdb** および **/etc/ipa/ca.crt** のチェーンの一部として(LDAP から入力)
LDAP の場所	**cn=SUBJECT,cn=certificates,cn=ipa,cn=etc,dc=realm,dc=name and ou=authorities,ou=ca,o=ipaca**
Issuer	外部 CA 署名
Subject	外部 CA サブジェクト
関連情報	チェーン内のすべての証明書は DER 形式でなければならず、LDAP にインポートする必要があります。NSS データベースに **CT,C,C** 信頼フラグが必要です。

サブシステム CA 証明書

この証明書は、LDAP データベースに書き込むときに LDAP サーバーへの認証に使用されます。この証明書は CA なしのインストールには存在しません。

subsystemCert	説明
ファイルシステムの場所	**nickname=subsystemCert cert-pki-ca** in **/etc/pki/pki-tomcat/alias nssdb**
LDAP の場所	**uid=pkidbuser,ou=people,o=ipaca**
Issuer	IPA CA
Subject	**CN=CA Subsystem,O=REALM.NAME**
関連情報	LDAP のシリアルとブロブの不一致に注意してください。たとえば、**2;SERIAL;CN=Certificate Authority,O=REALM.NAME;CN=CA Subsystem,O=REALM.NAME** および **userCertificate** は、ファイルシステム上のものと一致する必要があります。

監査署名証明書

この証明書は、監査ログの署名に使用されます。CA なしのインストールには存在しないことに注意してください。

auditSigningCert	説明
ファイルシステムの場所	**nickname=auditSigningCert cert-pki-ca** in **/etc/pki/pki-tomcat/alias nssdb**
LDAP の場所	専用の LDAP の場所はありません。**ou=certificateRepository,ou=ca,o=ipaca** で共有されます。
Issuer	IPA CA
Subject	**CN=CA Audit,O=REALM.NAME**
関連情報	NSS データベースに **,,P** 信頼フラグが必要です。

OCSP 署名証明書

この証明書は、Online Certificate Status Protocol (OCSP) サービスを提供するために使用されます。CA なしのインストールには存在しないことに注意してください。

ocspSigningCert	説明
ファイルシステムの場所	**/etc/pki/pki-tomcat/alias nssdb** の **nickname=ocspSigningCert cert-pki-ca**
LDAP の場所	専用の LDAP の場所はありません。**ou=certificateRepository,ou=ca,o=ipaca** で共有されます。
Issuer	IPA CA
Subject	**CN=OCSP Subsystem,O=REALM.NAME**
関連情報	

Tomcat サーブレット証明書

この証明書は、クライアントが PKI に接続する場合に使用されます。このサーバー証明書はホストに固有であり、CA なしのインストールには存在しないことに注意してください。

Server-Cert	説明
ファイルシステムの場所	• Nickname=Server-Cert cert-pki-ca in /etc/pki/pki-tomcat/alias nssdb+
LDAP の場所	

Server-Cert	説明
Issuer	IPA CA
Subject	CN=$HOSTNAME,O=REALM.NAME
関連情報	

登録認証局の証明書

PKI に対して認証するために **certmonger** と IdM フレームワークによって使用される証明書。たとえば、**ipa cert-show 1** を実行すると、HTTPD は PKI と通信し、この証明書で認証します。CA なしのインストールには存在しません。

RA エージェント	説明
ファイルシステムの場所	**/var/lib/ipa/ra-agent.pem** (RHEL 7.4 より前は **/etc/httpd/alias**)
LDAP の場所	**uid=ipara,ou=people,o=ipaca**
Issuer	IPA CA
Subject	**CN=IPA RA,O=REALM.NAME**
関連情報	LDAP のシリアルとブロブの不一致に注意してください。たとえば、**2;SERIAL;CN=Certificate Authority,O=REALM.NAME;CN=IPA RA,O=REALM.NAME** および **userCertificate** は、ファイルシステム上のものと一致する必要があります。

HTTPD フロントエンド証明書

HTTPD フロントエンドが Web UI および API への接続を保護するために使用する証明書。存在している必要があります。

HTTPD	説明
ファイルシステムの場所	**/var/lib/ipa/certs/httpd.crt** (RHEL 8 より前は **/etc/httpd/alias**)
LDAP の場所	
Issuer	CA なしのインストールでの IPA CA または外部 CA
Subject	**CN=$HOSTNAME,O=REALM.NAME**

HTTPD	説明
関連情報	プリンシパル名を **otherName = 1.3.6.1.4.1.311.20.2.3;UTF8:HTTP/$HOSTNAME@REALM, DNS name = $HOSTNAME** とする **Certificate Subject Alt Name** エクステンションが含まれている必要があります。

LDAP TLS および STARTTLS 証明書

LDAP TLS および STARTTLS 接続に使用される証明書。存在している必要があります。

LDAP	説明
ファイルシステムの場所	**/etc/dirsrv/slapd-DOMAIN** NSS データベースの **nickname=Server-Cert** (**dse.ldif** の **nsSSLPersonalitySSL** に一致する他のニックネームの可能性もあります)
LDAP の場所	
Issuer	CA なしのインストールでの IPA CA または外部 CA
Subject	**CN=$HOSTNAME,O=REALM.NAME**
関連情報	プリンシパル名を **otherName = 1.3.6.1.4.1.311.20.2.3;UTF8:ldap/$HOSTNAME@REALM, DNS name = $HOSTNAME** として、**Certificate Subject Alt Name** エクステンションを含める必要があります

KDC 証明書

IdM KDC の PKINIT に使用される証明書。

KDC	説明
ファイルシステムの場所	**/var/kerberos/krb5kdc/kdc.crt**
LDAP の場所	
Issuer	CA なしのインストールでの IPA CA または外部 CA
Subject	**CN=$HOSTNAME,O=REALM.NAME**

KDC	説明
関連情報	プリンシパル名が **otherName = 1.3.6.1.4.1.311.20.2.3;UTF8:krbtgt/REALM@REALM, DNS name = $HOSTNAME** の拡張キー使用 **id-pkinit-KPkdc (1.3.6.1.5.2.3.5)** が必要です。

25.3. IDM 内部証明書の更新プロセス

デフォルトでは、**certmonger** は内部証明書を追跡し、更新をトリガーして、IdM CA に新しい証明書を発行するように要求します。

外部 CA を使用していて、内部証明書がこの CA によって発行された場合、それらは自動的に更新されません。この場合、証明書の有効期限を監視し、有効期限が切れる前に証明書を更新するようにしてください。更新プロセスは時間がかかるため、有効期限を慎重に追跡しないと、証明書の有効期限が切れ、一部のサービスが利用できなくなります。

警告

内部の Red Hat Identity Management (IdM) 証明書の有効期限が切れると、IdM は起動できません。

IdM CA 更新サーバーは、有効期限の 28 日前に共有内部証明書を更新します。**certmonger** はこの更新をトリガーし、新しい証明書を **cn=<nickname>,cn=ca_renewal,cn=ipa,cn=etc,$BASEDN** にアップロードします。**certmonger** は、他の IdM サーバーで更新プロセスもトリガーしますが、CA 以外の更新サーバーで実行するため、新しい証明書を要求せずに、LDAP から証明書をダウンロードします。**Server-Cert cert-pki-ca**、HTTP、LDAP、および PKINIT 証明書は、各レプリカに固有であり、サブジェクトのホスト名が含まれています。

注記

証明書の有効期限が切れる前に **getcert** を使用して共有証明書を手動で更新すると、更新プロセスは他のレプリカでトリガーされず、LDAP から更新された証明書のダウンロードを実行するために、他のレプリカで **getcert** を実行する必要があります。

25.4. 関連情報

- IdM CA 更新サーバーの使用
- IdM がオフライン時に期限切れのシステム証明書の更新
- IdM レプリカでまだ有効期限が切れていない場合の Web サーバーと LDAP サーバーの証明書の置き換え
- IdM デプロイメント全体で期限切れになった Web サーバーと LDAP サーバーの証明書を置き換える