Red Hat Directory Server 12

Red Hat Directory Server のセキュリティー保護

Directory Server のセキュリティーの強化

Last Updated: 2024-05-09

Red Hat Directory Server 12 Red Hat Directory Server のセキュリティー保護

Directory Server のセキュリティーの強化

法律上の通知

概要

Red Hat Directory Server を使用して LDAP サービスのセキュリティーを向上させます。たとえば、クライアントと Directory Server 間の接続を暗号化し、暗号化された属性を Directory Server データベースに保存できます。レプリケーション変更ログの暗号化、認証の設定、その他のセキュリティータスクの実行も可能です。

目次

RED HAT ドキュメントへのフィードバック (英語のみ) 4

第1章 DIRECTORY SERVER への TLS 暗号化接続の有効化 5
1.1. DIRECTORY SERVER への暗号化接続のさまざまなオプション 5
1.2. DIRECTORY SERVER で NSS データベースをアンロックする方法 5
1.3. コマンドラインで DIRECTORY SERVER への TLS 暗号化接続の有効化 6
1.4. WEB コンソールを使用した DIRECTORY SERVER への TLS 暗号化接続の有効化 8
1.5. 証明書の有効期限が切れた場合に DIRECTORY SERVER の挙動の管理 10
1.6. NSS データベースのパスワードの変更 11
1.7. NSS データベースのパスワードを要求せずにインスタンスを起動するパスワードファイルの作成 12
1.8. DIRECTORY SERVER が使用する CA 証明書の RED HAT ENTERPRISE LINUX のトラストストアへの追加 13

第2章 サポートされる TLS プロトコルバージョンの設定 15
2.1. コマンドラインを使用した最小および最大の TLS プロトコルバージョンの設定 15
2.2. WEB コンソールを使用した最小および最大の TLS プロトコルバージョンの設定 16

第3章 暗号化接続に必要な LDAPS または STARTTLS 18
3.1. コマンドラインで DIRECTORY SERVER が LDAPS または STARTTLS で暗号化した接続のみを受け入れるように設定 18
3.2. WEB コンソールを使用して、LDAPS または STARTTLS で暗号化した接続のみを受け入れるように DIRECTORY SERVER を設定 18

第4章 DIRECTORY SERVER がサポートする暗号のリストの更新 20
4.1. デフォルトの暗号と利用可能な暗号の違い 20
4.2. 弱い暗号 20
4.3. コマンドラインを使用した DIRECTORY SERVER がサポートする暗号化の設定 20
4.4. WEB コンソールを使用した DIRECTORY SERVER がサポートする暗号化の設定 21

第5章 CA 信頼フラグの変更 23
5.1. コマンドラインを使用した CA 信頼フラグの変更 23
5.2. WEB コンソールを使用した CA 信頼フラグの変更 24

第6章 TLS 証明書の更新 25
6.1. コマンドラインでの TLS 証明書の更新 25

第7章 証明書ベースの認証の設定 27
7.1. 証明書ベースの認証の設定 27
7.2. ユーザーへの証明書の追加 28

第8章 証明書ベースの認証を使用したマルチサプライヤーレプリケーションの設定 30
8.1. 証明書ベースの認証による複製合意で使用するためのアカウントとバインドグループの準備 30
8.2. 一時レプリケーションマネージャーアカウントを使用した新しいサーバーの初期化 31
8.3. 証明書ベースの認証を使用したマルチサプライヤーレプリケーションの設定 32

第9章 レプリケーション変更ログの暗号化 35
9.1. コマンドラインを使用した CHANGELOG の暗号化 35

第10章 グループのメンバーが DIRECTORY SERVER をバックアップすることの許可、およびグループメンバーの 1 つとしてのバックアップの実行 37
10.1. グループが DIRECTORY SERVER をバックアップすることの許可 37
10.2. 通常ユーザーとしてのバックアップの実行 38

第11章 グループのメンバーがデータをエクスポートすることの許可、およびグループメンバーの 1 つとしてのエクスポートの実行 40
11.1. グループによるデータエクスポートの許可 40

11.2. 通常ユーザーとしてのエクスポートの実行 41

第12章 アクセス制御手順の管理 43
12.1. ACI 配置 43
12.2. ACI の構造 44
12.3. ACI 評価 44
12.4. ACI の制限 45
12.5. DIRECTORY SERVER がレプリケーショントポロジーで ACI を処理する方法 45
12.6. ACI の表示、追加、削除、および更新 45
12.7. ACI ターゲットの定義 46
12.8. ターゲットルールの高度な使用方法 53
12.9. ACI パーミッションの定義 55
12.10. ACI バインドルールの定義 58

第13章 DIRECTORY SERVER を FIPS モードで実行する 74
13.1. FIPS モードの有効化 74
13.2. 関連情報 74

第14章 パスワードベースのアカウントロックアウトポリシーの設定 75
14.1. 設定された最大試行に到達するか、超過する際にアカウントをロックするかどうかの設定 75
14.2. コマンドラインでパスワードベースのアカウントロックアウトポリシーの設定 76
14.3. WEB コンソールでパスワードベースのアカウントロックアウトポリシーの設定 78

第15章 匿名バインドの無効化 80
15.1. コマンドラインでの匿名バインドの無効化 80
15.2. WEB コンソールでの匿名バインドの無効化 80

第16章 レプリケーション環境のすべてのサーバー間でのアカウントのロックアウト属性の同期 82
16.1. レプリケーション環境で DIRECTORY SERVER がパスワードおよびアカウントのロックアウトポリシーを処理する方法 82
16.2. アカウントロックアウト属性をレプリケートするための DIRECTORY SERVER の設定 82

第17章 時間ベースのアカウントロックアウトポリシーの設定 85
17.1. 最後に成功したログインで一定時間アカウントを自動的に無効にする 85
17.2. アカウントを作成してから一定時間、アカウントを自動的に無効にする 87
17.3. パスワードの有効期限が切れてから一定時間アカウントを自動的に無効にする 89
17.4. アカウントのステータス (アクティブかどうか) とパスワードの有効期限の両方でアカウントを自動的に無効にする 91

第18章 非アクティブ制限に達したアカウントを再度有効にする 93
18.1. アカウントポリシープラグインによって非アクティブ化されたアカウントを再度有効にする 93

第19章 ロックアウトポリシーを設定せずに最終ログイン時間を追跡する 94
19.1. 最終ログイン時刻を記録するようにアカウントポリシープラグインを設定する 94

第20章 DIRECTORY MANAGER アカウントにアクセス制御を設定する 96
20.1. DIRECTORY MANAGER アカウントのアクセス制御 96
20.2. コマンドラインを使用した ROOTDN アクセス制御プラグインの設定 96
20.3. WEB コンソールを使用して ROOTDN アクセス制御プラグインを設定する 97

第21章 属性暗号化の管理 99
21.1. DIRECTORY SERVER が属性の暗号化に使用するキー 99
21.2. コマンドラインを使用して属性暗号化を有効にする 99
21.3. WEB コンソールを使用して属性暗号化を有効にする 100
21.4. 属性暗号化の有効化後の一般的な考慮事項 101
21.5. 属性暗号化に使用される TLS 証明書の更新 102

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに対するご意見をお聞かせください。ドキュメントの改善点があればお知らせください。これを行うには、以下を行います。

- 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。
- Bugzilla からのフィードバック送信 (アカウントが必要)
  1. Bugzilla の Web サイトに移動します。
  2. Component として **Documentation** を使用します。
  3. **Description** フィールドに、ドキュメントの改善に向けたご提案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。
  4. **Submit Bug** をクリックします。

第1章 DIRECTORY SERVER への TLS 暗号化接続の有効化

デフォルトでは、Red Hat Directory Server は暗号化なしで LDAP サービスを提供します。セキュリティーを改善するには、Directory Server で TLS を設定して、レプリケーション環境内のクライアントや他のホストを有効にして、暗号化された接続を使用できます。これらのユーザーは、ポート 389 で **STARTTLS** コマンドを使用し、またはセキュアな接続にポート 636 で LDAPS プロトコルを使用できます。

バインド識別名 (DN) およびパスワード、または証明書ベースの認証を使用して、簡易認証で TLS を使用できます。

Directory Server の暗号化サービスは、Mozilla Network Security Services (NSS) (TLS およびベース暗号化機能のライブラリー) によって提供されます。NSS には、連邦情報処理標準 (FIPS) 140-2 認定であるソフトウェアベースの暗号化トークンが含まれています。

1.1. DIRECTORY SERVER への暗号化接続のさまざまなオプション

暗号化された接続を使用して Directory Server に接続するには、以下のプロトコルとフレームワークを使用できます。

LDAPS

LDAPS プロトコルを使用すると、接続は暗号化を使用して開始し、成功または失敗します。ただし、暗号化されていないデータはネットワーク経由で送信されません。このため、暗号化されていない LDAP で **STARTTLS** を使用する代わりに、LDAPS の使用が推奨されます。

LDAP 上の STARTTLS

クライアントは LDAP プロトコルで暗号化されていない接続を確立し、**STARTTLS** コマンドを送信します。コマンドに成功すると、それ以降の通信はすべて暗号化されます。

警告

STARTTLS コマンドが失敗し、クライアントが接続をキャンセルしないと、認証情報を含むすべてのデータが暗号化されずにネットワーク上に送信されます。

SASL

Simple Authentication and Security Layer (SASL) フレームワークを使用すると、Kerberos などの外部認証方法を使用してユーザーを認証できます。

1.2. DIRECTORY SERVER で NSS データベースをアンロックする方法

Directory Server は、証明書署名要求 (CSR)、秘密鍵、および証明書をネットワークセキュリティーサービス (NSS) データベースに保存します。新規インスタンスをインストールすると、インストーラーは NSS データベースを自動的に作成し、無作為にパスワードで保護します。インストーラーは、このパスワードを以下のファイルに保存します。

- **/etc/dirsrv/slapd-instance_name/pwdfile.txt**: **dsconf tls** コマンドは、このファイルを使用して NSS データベースにアクセスします。

- **/etc/dirsrv/slapd-instance_name/pin.txt**: このファイルには、Directory Server の起動時に NSS データベースを自動的にアンロックするトークンとパスワードが含まれます。
  - インスタンスを起動するたびに、Directory Server が NSS データベースのパスワードを要求するようにするには、このファイルを削除します。
  - パスワードを要求せずにインスタンスを自動的に起動するようにするには、NSS データベースパスワードを変更する場合にこのファイルを保存して更新します。

/etc/dirsrv/slapd-instance_name/pin.txt ファイルが存在しない場合は、暗号化を有効にして Directory Server を起動し、NSS データベースにパスワードを設定すると、動作は以下のようになります。

- **systemctl** ユーティリティーまたは **dsctl** ユーティリティーが **ns-slapd** Directory Server プロセスを開始すると、**systemd** サービスはパスワードを要求して、自動的に **systemd-tty-ask-password-agent** ユーティリティーに入力を渡します。

```
# dsctl instance_name start
Enter PIN for Internal (Software) Token: (press TAB for no echo)
```

- まれに、**ns-slapd** Directory Server プロセスが **systemctl** ユーティリティーまたは **dsctl** ユーティリティーで開始されず、プロセスがターミナルから切り離されていると、**ns-slapd** は、**wall** コマンドを使用してすべてのターミナルにメッセージを送信します。

```
Broadcast message from root@server (Fri 2021-01-01 06:00:00 CET):

Password entry required for 'Enter PIN for Internal (Software) Token:' (PID 1234).
Please enter password with the systemd-tty-ask-password-agent tool!
```

パスワードを入力するには、次のコマンドを実行します。

```
# systemd-tty-ask-password-agent
Enter PIN for Internal (Software) Token:
```

関連情報

- NSS データベースのパスワードの変更

1.3. コマンドラインで DIRECTORY SERVER への TLS 暗号化接続の有効化

TLS による暗号化または証明書ベースの認証を使用するには、Network Security Services (NSS) データベースで証明書を管理する必要があります。インスタンスを作成すると、**dscreate** ユーティリティーは **/etc/dirsrv/slapd-instance_name/** ディレクトリーにこのデータベースを自動的に作成し、強力なパスワードで保護します。

手順

1. プライベートキーおよび証明書署名要求 (CSR) を作成します。外部ユーティリティーを使用して作成する場合は、この手順を省略します。
   - ホストが 1 つの名前のみで到達可能である場合は、以下を実行します。

```
# dsctl instance_name tls generate-server-cert-csr -s
"CN=server.example.com,O=example_organization"
```

- 複数の名前でホストにアクセスできる場合は、以下を行います。

```
# dsctl instance_name tls generate-server-cert-csr -s
"CN=server.example.com,O=example_organization" server.example.com
server.example.net
```

最後のパラメーターとしてホスト名を指定した場合、このコマンドは **DNS:server.example.com, DNS:server.example.net** エントリーで SAN (Subject Alternative Name) 拡張を CSR に追加します。

-s subject パラメーターで指定した文字列は、RFC 1485 に従って有効なサブジェクト名である必要があります。サブジェクトの **CN** フィールドが必要で、サーバーの完全修飾ドメイン名 (FQDN) の 1 つに設定する必要があります。このコマンドは、**/etc/dirsrv/slapd-instance_name/Server-Cert.csr** ファイルに CSR を保存します。

2. 認証局 (CA) に CSR を送信し、発行した証明書を取得します。詳細は、CA のドキュメントを参照してください。

3. CA が発行するサーバー証明書を NSS データベースにインポートします。

- **dsctl tls generate-server-cert-csr** コマンドを使用して秘密鍵を作成した場合は、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security certificate
add --file /root/instance_name.crt --name "server-cert" --primary-cert
```

--name _certificate_nickname パラメーターで設定した証明書の名前を書き留めておきます。これは後のステップで必要になります。

- 外部ユーティリティーを使用して秘密鍵を作成した場合は、サーバー証明書および秘密鍵をインポートします。

```
# dsctl instance_name tls import-server-key-cert /root/server.crt /root/server.key
```

このコマンドでは、最初にサーバー証明書へのパスを指定してから、秘密鍵へのパスを指定する必要があります。このメソッドは、証明書のニックネームを **Server-Cert** に設定します。

4. CA 証明書を NSS データベースにインポートします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ca-certificate
add --file /root/ca.crt --name "Example CA"
```

5. CA 証明書の信頼フラグを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ca-certificate
set-trust-flags "Example CA" --flags "CT,,"
```

これにより、Directory Server が、TLS による暗号化および証明書ベースの認証に対して CA を信頼するように設定します。

6. TLS を有効にし、LDAPS ポートを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-securePort=636 nsslapd-security=on
```

7. **firewalld** サービスで LDAPS ポートを開きます。

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

8. RSA 暗号ファミリーを有効にし、NSS データベースセキュリティーデバイスおよびサーバー証明書名を設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security rsa set --tls-allow-rsa-certificates on --nss-token "internal (software)" --nss-cert-name Server-Cert
```

デフォルトでは、NSS データベースのセキュリティーデバイスの名前は **internal (software)** です。

9. オプション: プレーンテキストの LDAP ポートを無効にします。

```
# dsconf inst security disable_plain_port
```

10. インスタンスの再起動

```
# dsctl instance_name restart
```

検証

- LDAPS プロトコルを使用して Directory Server への接続を確立します。たとえば、クエリーを実行します。

```
# ldapsearch -H ldaps://server.example.com:636 -D "cn=Directory Manager" -W -b "dc=example,dc=com" -x -s base
```

コマンドが失敗し、**ldap_sasl_bind(SIMPLE): Can't contact LDAP server (-1)** エラーが発生した場合は、デバッグレベル 1 でコマンドを再実行します。

```
# ldapsearch -H ldaps://server.example.com:636 -D "cn=Directory Manager" -W -b "dc=example,dc=com" -x -s base -d 1
```

次のステップ

- Directory Server が使用する CA 証明書を Red Hat Enterprise Linux のトラストストアに追加
- オプション: Change the password of the NSS database
- オプション: Directory Server がサポートする暗号化のリストの更新

関連情報

- コマンドラインを使用した CA 信頼フラグの変更

1.4. WEB コンソールを使用した DIRECTORY SERVER への TLS 暗号化接続の有効化

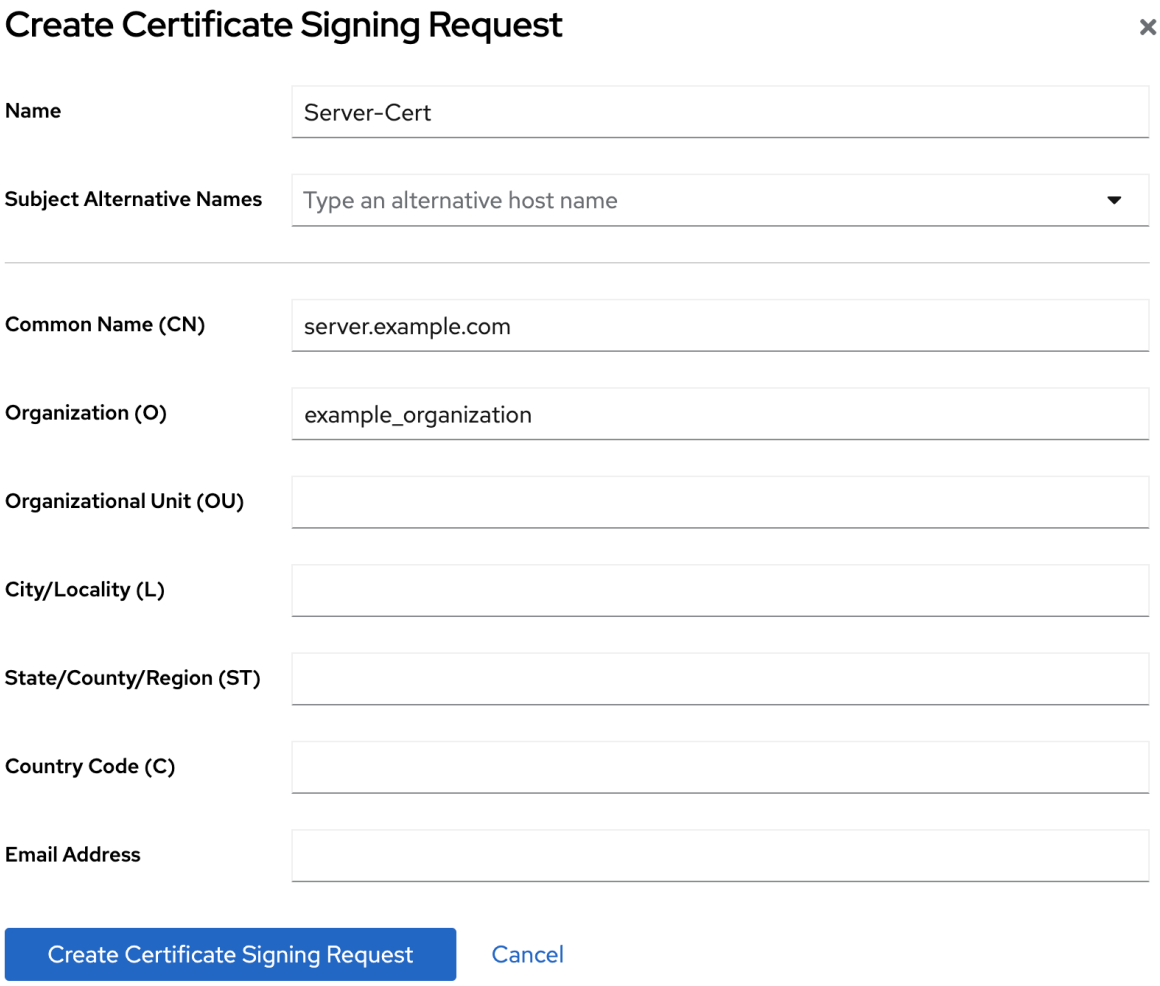
Web コンソールを使用して TLS 暗号化を設定できます。

前提条件

- Web コンソールでインスタンスにログインしている。

手順

1. **Server → Security → Certificate Management → Certificate Signing Request** に移動し、**Create Certificate Signing Request** をクリックします。
2. 証明書署名要求 (CSR)、共通名 (CN)、および組織 (O) の名前を設定します。



   ホストに複数の名前でアクセスできる場合は、**Subject Alternative Names** フィールドに代替名を設定します。

3. **Create Certificate Signing Request** をクリックします。
4. CSR テキストを表示し、これをコピーします。
   a. 表示する CSR の **Node options** アイコンをクリックして、**View CSR** を選択します。
   b. CSR コンテンツをコピーします。
5. 認証局 (CA) に CSR ファイルを送信し、発行した証明書を取得します。詳細は、CA のドキュメントを参照してください。
6. CA から証明書を取得したら、**Server → Security → Certificate Management → TLS Certificates** に移動し、**Add Server Certificate** をクリックします。

7. サーバー証明書に一意のニックネームを設定し、発行された証明書をアップロードし、**Add Certificate** をクリックします。
後のステップには証明書のニックネームが必要であるため、証明書のニックネームを覚えておいてください。

8. **Server → Security → Certificate Management → Trusted Certificate Authorities** に移動し、**Add CA Certificate** をクリックします。

9. CA 証明書に一意のニックネームを設定し、CA 証明書ファイルをアップロードし、**Add Certificate** をクリックします。

10. オプション：Directory Server インスタンスのインストール時に TLS 暗号化を有効にしなかった場合は、有効にします。

    a. **Server →Security Settings** に移動し、セキュリティースイッチを有効にします。

    b. ポップアップウィンドウで、**Enable Security** をクリックします。

    c. **Security Setting** ページで **Save Configuration** をクリックします。

11. **Security Configuration** ページで **Server Certificate Name** を設定します。

    a. **Server → Security → Security Configuration** の順に移動します。

    b. **Server Certificate Name** ドロップダウンリストでサーバー証明書のニックネームを選択し、**Save Configuration** をクリックします。

    c. オプション: ドロップダウンリストに証明書のニックネームが表示されない場合は、**Security Settings** ページを更新して、前の手順を再度実行します。

12. オプション: **636** 以外の LDAPS ポートを使用する場合は、**Server → Server Settings** に移動し、LDAPS ポートを設定して、**Save** をクリックします。

13. **firewalld** サービスで LDAPS ポートを開きます。

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

14. オプション: **Server → Security Security → Configuration** に移動し、**Require Secure Connections** チェックボックスを選択し、**Save Configuration** をクリックします。
Directory Server は、プレーンテキストの LDAP ポートを無効にします。

15. 右上隅の **Actions** をクリックし、**Restart Instance** を選択します。

次のステップ

- Directory Server が使用する CA 証明書を Red Hat Enterprise Linux のトラストストアに追加
- オプション: Change the password of the NSS database
- オプション: Directory Server がサポートする暗号化のリストの更新

1.5. 証明書の有効期限が切れた場合に DIRECTORY SERVER の挙動の管理

デフォルトでは、暗号化が有効で、証明書の有効期限が切れると、Directory Server は警告をログに記録し、サービスを起動します。この動作を変更するには、**nsslapd-validate-cert** パラメーターを設定します。以下の値を設定できます。

- **warn**: Directory Server が起動し、期限切れの証明書に関する警告を **/var/log/dirsrv/slapd-instance_name/error** ログファイルに記録します。これはデフォルト設定です。
- **on**: Directory Server は証明書を検証します。証明書の有効期限が切れると、インスタンスは起動できません。
- **off**: Directory Server は証明書の有効期限を検証しません。インスタンスが起動し、警告は記録されません。

前提条件

- TLS 暗号化を設定している。

手順

- 以下のコマンドを使用して、**nsslapd-validate-cert** パラメーターを変更します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-validate-cert=<value>
```

1.6. NSS データベースのパスワードの変更

ネットワークセキュリティーサービス (NSS) データベースのパスワードを変更できます。たとえば、権限のない人にパスワードが知られるようになったときに変更します。

前提条件

- 現在の NSS データベースのパスワードを知っている必要がある。
  Directory Server の起動時に自動的にデータベースのロックを解除するためのパスワードファイルを使用している場合、パスワードは、**/etc/dirsrv/slapd-instance_name/pin.txt** に、プレーンテキストで暗号化されずに保存されている。

手順

1. 以下のコマンドを使用して NSS データベースのパスワードを変更します。

```
# certutil -d /etc/dirsrv/slapd-instance_name/ -W
Enter Password or Pin for "NSS Certificate DB":
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:
Password changed successfully.
```

2. NSS データベースのパスワードを要求せずに、パスワードファイルを使用して Directory Server を自動的に起動する場合は、以前のパスワードを **/etc/dirsrv/slapd-instance_name/pin.txt** 内の新しいパスワードに置き換えます。

- NSS ソフトウェア暗号モジュールを使用する場合は、以下になります。

  ```
  Internal (Software) Token:password
  ```

- Hardware Security Module (HSM) を使用する場合:

  ```
  name_of_the_token:password
  ```

検証

- パスワードの入力が必要な NSS データベースで操作を実行します。たとえば、インスタンスの秘密鍵をリスト表示します。

  ```
  # certutil -d /etc/dirsrv/slapd-instance_name/ -K
  certutil: Checking token "NSS Certificate DB" in slot "NSS User Private Key and Certificate Services"
  Enter Password or Pin for "NSS Certificate DB":
  < 0> rsa      72cb03f87381abfbb6b9e78234e2e4502ad1bfc0   NSS Certificate DB:Server-Cert
  ```

  新しいパスワードの入力後にコマンドにより想定される出力が表示される場合には、パスワードの変更に成功しました。

関連情報

- Directory Server で NSS データベースをアンロックする方法

1.7. NSS データベースのパスワードを要求せずにインスタンスを起動するパスワードファイルの作成

新規インスタンスの作成時に、インストーラーは **/etc/dirsrv/slapd-instance_name/pin.txt** ファイルを自動的に作成し、ネットワークセキュリティーサービス (NSS) パスワードを要求せずに Directory Server が起動するようにします。ただし、このファイルを削除すると再作成できます。

警告

このパスワードはプレーンテキストで保存されます。サーバーがセキュアでない環境で実行している場合は、パスワードファイルを使用しないでください。

前提条件

- NSS データベースのパスワードを知っている必要がある。

手順

1. 以下の内容で **/etc/dirsrv/slapd-instance_name/pin.txt** ファイルを作成します。

   - NSS ソフトウェア暗号モジュールを使用する場合は、以下になります。

     ■

```
Internal (Software) Token:password
```

- Hardware Security Module (HSM) を使用する場合:

```
name_of_the_token:password
```

2. ファイルの権限を設定します。

```
# chown dirsrv:root /etc/dirsrv/slapd-instance_name/pin.txt
# chmod 400 /etc/dirsrv/slapd-instance_name/pin.txt
```

検証

- インスタンスを再起動します。

```
# dsctl instance_name restart
```

システムが NSS データベースのパスワードを要求しない場合、Directory Server はパスワードファイルを使用します。

関連情報

- Directory Server で NSS データベースをアンロックする方法

1.8. DIRECTORY SERVER が使用する CA 証明書の RED HAT ENTERPRISE LINUX のトラストストアへの追加

Directory Server で TLS 暗号化を有効にすると、CA が発行した証明書を使用するようにインスタンスを設定します。クライアントが LDAPS プロトコルまたは LDAP 上の **STARTTLS** コマンドを使用してサーバーへの接続を確立する場合、Directory Server はこの証明書を使用して接続を暗号化します。クライアントユーティリティーは CA 証明書を使用して、サーバーの証明書が有効であるかどうかを確認します。デフォルトでは、これらのユーティリティーは、サーバーの証明書を信頼していない場合に接続を取り消します。

例1.1 クライアントユーティリティーが CA 証明書を使用しない場合の接続エラーの可能性

- **dsconf**

```
# dsconf -D "cn=Directory Manager" ldaps://server.example.com:636 config get
Error: {'desc': "Can't contact LDAP server", 'info': 'error:1416F086:SSL
routines:tls_process_server_certificate:certificate verify failed (self signed certificate in
certificate chain)'}
```

- **ldapsearch**

```
# ldapsearch -H ldaps://server.example.com:636 -D "cn=Directory Manager" -W -b
"dc=example,dc=com" -x
Enter LDAP Password:
ldap_sasl_bind(SIMPLE): Can't contact LDAP server (-1)
```

Red Hat Enterprise Linux でクライアントユーティリティーを有効にして Directory Server が使用する証明書を検証するには、オペレーティングシステムのトラストストアに CA 証明書を追加します。

前提条件

- ネットワークセキュリティーサービス (NSS) データベースのパスワードを知っている。Directory Server インスタンスのインストール時に生成されたパスワードを使用する場合は、**/etc/dirsrv/slapd-instance_name/pwdfile.txt** ファイルでこのパスワードをプレーンテキストで特定する。

手順

1. Directory Server が使用する CA 証明書のローカルコピーがない場合は、以下を実行します。

   a. サーバーのネットワークセキュリティーサービス (NSS) データベースで証明書をリスト表示します。

   ```
   # certutil -d /etc/dirsrv/slapd-instance_name/ -L

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

   Example CA                              C,,
   Server-Cert                             u,u,u
   ```

   b. NSS データベースの CA 証明書のニックネームを使用して、CA 証明書をエクスポートします。

   ```
   # certutil -d /etc/dirsrv/slapd-instance_name/ -L -n "Example CA" -a > /tmp/ds-ca.crt
   ```

2. CA 証明書を **/etc/pki/ca-trust/source/anchors/** ディレクトリーにコピーします。

   ```
   # cp /tmp/ds-ca.crt /etc/pki/ca-trust/source/anchors/
   ```

3. CA 信頼データベースを再構築します。

   ```
   # update-ca-trust
   ```

検証

- LDAPS プロトコルを使用して Directory Server への接続を確立します。たとえば、クエリーを実行します。

  ```
  # ldapsearch -H ldaps://server.example.com:636 -D "cn=Directory Manager" -W -b "dc=example,dc=com" -x -s base
  ```

関連情報

- **update-ca-trust(8)** の man ページ

第2章 サポートされる TLS プロトコルバージョンの設定

Red Hat Enterprise Linux 9 では、システム全体の暗号化ポリシープロファイルはすべて最低要件として TLS 1.2 が定義されています。そのため、この TLS バージョンは Directory Server の最小値でもあります。ただし、TLS の新しいバージョンをサポートするクライアントのみがある場合は、少なくともより高いプロトコルバージョンを設定してセキュリティーを強化できます。

2.1. コマンドラインを使用した最小および最大の TLS プロトコルバージョンの設定

コマンドラインを使用して、最小および最大の TLS プロトコルの両方を設定できます。

警告

最大 TLS プロトコルを設定しないでください。設定した場合には、クライアントはデフォルトの標準よりも弱い TLS プロトコルを使用する必要がある場合があります。最大 TLS バージョンを設定しないと、Directory Server は、サポート範囲内で最も強力なバージョンを常に使用します。

前提条件

- Directory Server で TLS 暗号化を有効にしている。

手順

1. 必要に応じて、Directory Server で現在有効な TLS プロトコルを表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security get | egrep -i "sslVersionMin|sslVersionMax"
sslversionmin: TLS1.2
sslversionmax: TLS1.3
```

2. 最小 TLS プロトコルを設定します。たとえば、TLS 1.3 に設定するには、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security set --tls-protocol-min="TLS1.3"
```

パラメーターを TLS 1.2 よりも小さい値に設定することはできません。これは、RHEL システム全体の暗号化ポリシープロファイルの最小値です。

3. 非推奨: サポートされている最大の TLS プロトコルを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security set --tls-protocol-max="TLS1.3"
```

--tls-protocol-max を **--tls-protocol-min** よりも小さい値に設定すると、Directory Server は最大プロトコルを最小値と同じ値に設定します。

サポートされている最も強力な暗号化プロトコルを常にサポートされる最大 TLS バージョンとして使用するには、**--tls-protocol-max** を設定しないでください。

4. インスタンスを再起動します。

```
# dsctl instance_name restart
```

検証

1. サポートされる TLS プロトコルを表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security get | egrep -i "sslVersionMin|sslVersionMax"
sslversionmin: TLS1.3
sslversionmax: TLS1.3
```

2. **openssl** ユーティリティーを使用して、特定の TLS プロトコルを使用してセキュアなクライアント接続を確立します。

```
# echo | openssl s_client -connect server.example.com:636 -tls1_3
...
New, TLSv1.3, Cipher is TLS_AES_128_GCM_SHA256
...
```

2.2. WEB コンソールを使用した最小および最大の TLS プロトコルバージョンの設定

Web コンソールを使用して最小および最大の TLS プロトコルの両方を設定できます。

警告

最大 TLS プロトコルを設定しないでください。設定した場合には、クライアントはデフォルトの標準よりも弱い TLS プロトコルを使用する必要がある場合があります。最大 TLS バージョンを設定しないと、Directory Server は、サポート範囲内で最も強力なバージョンを常に使用します。

前提条件

- Directory Server で TLS 暗号化を有効にしている。
- Web コンソールで Directory Server インスタンスにログインしている。

手順

1. **Server → Security** の順に移動します。
2. **Minimum TLS Version** フィールドで、最低限必要な TLS プロトコルを設定します。

3. 非推奨: **Maximum TLS Version** フィールドでサポート対象範囲で最大の TLS プロトコルに設定します。
4. **Save settings** をクリックします。
5. 右上隅の **Actions** をクリックし、**Restart Instance** を選択します。

検証

- **openssl** ユーティリティーを使用して、特定の TLS プロトコルを使用してセキュアなクライアント接続を確立します。

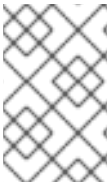
```
# echo | openssl s_client -connect server.example.com:636 -tls1_3
...
New, TLSv1.3, Cipher is TLS_AES_128_GCM_SHA256
...
```

第3章 暗号化接続に必要な LDAPS または STARTTLS

ネットワーク経由で暗号化されていないパスワードを送信するのを防ぐために、サーバーへの接続時に LDAPS または STARTTLS 暗号化を使用できるように Directory Server を設定できます。

3.1. コマンドラインで DIRECTORY SERVER が LDAPS または STARTTLS で暗号化した接続のみを受け入れるように設定

デフォルトでは、Directory Server はバインド DN と暗号化されていない接続を介したパスワードを使用した認証を許可します。これはセキュリティーリスクです。証明書ベースの認証や SASL などの別のセキュアなメカニズムを使用できないとします。この場合は、TLS または STARTTLS を使用してサーバーに対して認証する際に暗号化された接続を要求するように Directory Server を設定できます。



注記

バインド操作のセキュアな接続を必要とするのは、認証されたバインドにのみ適用されます。匿名および認証されていないバインドなどのパスワードのないバインド操作は、標準の接続で続行できます。

前提条件

- レプリカ合意などの既存のサーバー間の接続がセキュアなバインドを使用するように設定している。

手順

1. **nsslapd-require-secure-binds** 設定パラメーターを **on** に設定します。

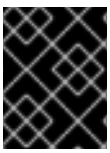
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-require-secure-binds=on
```

2. オプション: LDAPS を使用する場合は、プレーンテキストの LDAP ポートを無効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security
disable_plain_port
```

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```



重要

この機能を有効にすると、すべての接続に必要です。たとえば、これには、レプリカ合意、同期、データベースのチェーンが含まれます。

関連情報

- 認証方法に基づいたアクセスの定義

3.2. WEB コンソールを使用して、LDAPS または STARTTLS で暗号化した接続のみを受け入れるように DIRECTORY SERVER を設定

デフォルトでは、Directory Server はバインド DN と暗号化されていない接続を介したパスワードを使用した認証を許可します。これはセキュリティーリスクです。証明書ベースの認証や SASL などの別のセキュアなメカニズムを使用できないとします。この場合は、TLS または STARTTLS を使用してサーバーに対して認証する際に暗号化された接続を要求するように Directory Server を設定できます。

注記

バインド操作のセキュアな接続を必要とするのは、認証されたバインドにのみ適用されます。匿名および認証されていないバインドなどのパスワードのないバインド操作は、標準の接続で続行できます。

前提条件

- レプリカ合意などの既存のサーバー間の接続がセキュアなバインドを使用するように設定している。
- Web コンソールでインスタンスにログインしている。

手順

1. **Server → Security Security → Configuration** に移動し、**Require Secure Connections** オプションを選択し、**Save Configuration** をクリックします。
2. オプション: LDAPS を使用する場合は、**Server → Server Settings → General Settings** に移動し、**LDAP Port** を **0** に設定してプレーンテキストの LDAP ポートを無効にします。**Save** をクリックします。
3. 右上隅の **Actions** をクリックし、**Restart Instance** を選択します。

重要

この機能を有効にすると、すべての接続に必要です。たとえば、これには、レプリカ合意、同期、データベースのチェーンが含まれます。

関連情報

- 証明書ベースの認証の設定

第4章 DIRECTORY SERVER がサポートする暗号のリストの更新

暗号化された接続を確立するには、Directory Server とクライアントの両方に、少なくとも 1 つの共通の暗号が必要です。たとえば、Directory Server でデフォルトで有効化されていない暗号がレガシーアプリケーションで必要な場合は、有効にできます。

4.1. デフォルトの暗号と利用可能な暗号の違い

設定で個別の暗号をリスト表示する代わりに、**nsSSL3Ciphers** パラメーターで以下のキーワードのいずれかを使用できます。

- **デフォルト**: ネットワークセキュリティーサービス (NSS) で有効なデフォルトの暗号を参照してください。リストを表示するには、以下を入力します。

  ```
  # /usr/lib64/nss/unsupported-tools/listsuites | grep -B1 --no-group-separator "Enabled"
  ```

  デフォルト のキーワードは、**nsSSL3Ciphers** パラメーターのデフォルト値です。

- **All**: Directory Server で対応しているすべての暗号を参照します。リストを表示するには、以下を入力します。

  ```
  # dsconf -D "cn=Directory Manager" ldap://server.example.com security ciphers list --supported
  ```

  特定の暗号のみを有効にする場合は、**all** キーワードを使用します。たとえば、**nsSSL3Ciphers** を **-all,+TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384** に設定すると、Directory Server がすべての暗号を無効にし、**TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384** のみを有効にします。

4.2. 弱い暗号

デフォルトでは、Directory Server は弱い暗号を拒否し、Directory Server がそれに対応するように設定する必要があります。

暗号は、以下の場合に弱いとみなされます。

- これらはエクスポート可能です。
  エクスポートする暗号には、暗号名に **EXPORT** というラベルが付いています。たとえば、**TLS_RSA_EXPORT_WITH_RC4_40_MD5** の場合は以下のようになります。

- この暗号は対称的であり、**3DES** アルゴリズムよりも弱いです。
  対称暗号は、暗号化と復号化の両方に同じ暗号鍵を使用します。

- キーの長さは 128 ビットより短いです。

4.3. コマンドラインを使用した DIRECTORY SERVER がサポートする暗号化の設定

Directory Server で対応している暗号のリストを更新するには、**nsSSL3Ciphers** パラメーターを更新します。

前提条件

- Directory Server で TLS 暗号化を有効にしている。

手順

1. 有効な暗号のリストを表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ciphers list
default
```

default キーワードは、ネットワークセキュリティーサービス (NSS) で有効な暗号のみが有効になっていることを示します。

2. 弱い暗号を有効にする必要がある場合は、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security set --allow-insecure-ciphers on
```

3. **nsSSL3Ciphers** パラメーターを更新します。たとえば、**TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384** および **TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384** 暗号のみを有効にするには、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ciphers set --
"-
all,+TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,+TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384"
```

-- を使用して、シェルが **-all** の **-** 文字をコマンドのオプションとして解釈しないようにします。**-all** をエスケープするために \ 文字を使用しないでください。エラーが発生し、別の暗号が選択される可能性があるためです。

4. インスタンスを再起動します。

```
# dsctl instance_name restart
```

検証

- 有効な暗号のリストを表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ciphers list
default
+TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
+TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
```

関連情報

- デフォルトの暗号と利用可能な暗号の違い
- 弱い暗号

4.4. WEB コンソールを使用した DIRECTORY SERVER がサポートする暗号化の設定

Directory Server Web コンソールの **Cipher Preferences** メニューで暗号を設定できます。

前提条件

- Directory Server で TLS 暗号化を有効にしている。
- Web コンソールでインスタンスにログインしている。

手順

1. 弱い暗号を有効にする必要がある場合は、以下を行います。
   a. **Server → Security → Security Configuration** の順に移動します。
   b. **Allow Weak Ciphers** を選択します。
   c. **Save settings** をクリックします。
2. **Server → Security → Cipher Preferences** の順に移動します。
3. 暗号の設定を更新します。たとえば、**TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384** と **TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384** 暗号のみを有効にするには、以下を実行します。
   a. **Cipher Suite** フィールドで **No Ciphers** を選択します。
   b. **Allow Specific Ciphers** フィールドに **TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384** を入力します。
4. **Save settings** をクリックします。
5. **Actions → Restart Instance** をクリックします。

検証

- **Server → Security → Cipher Preferences** の順に移動します。**Enabled Ciphers** リストには、有効な暗号が表示されます。

第5章 CA 信頼フラグの変更

認証局 (CA) の信頼フラグは、DirectoryServer が CA 証明書を信頼するシナリオを定義します。たとえば、フラグを設定して、TLS 接続の証明書をサーバーへ信頼し、証明書ベースの認証用に設定します。

5.1. コマンドラインを使用した CA 信頼フラグの変更

以下の信頼フラグを認証局 (CA) 証明書に設定できます。

- **C**: 信頼される CA
- **T**: 信頼される CA クライアント認証
- **c**: 有効な CA
- **P**: 信頼されるピア
- **p**: 有効なピア
- **u**: 秘密鍵

TLS, email, object signing の 3 つのカテゴリーでコンマ区切りの信頼フラグを指定します

たとえば、TLS 暗号化および証明書ベースの認証の CA を信頼するには、信頼フラグを **CT** に設定します。

前提条件

- CA 証明書をネットワークセキュリティーサービス (NSS) データベースにインポートしている。

手順

1. 以下のコマンドを使用して、CA 証明書の信頼フラグを変更します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ca-certificate set-trust-flags "Example CA" --flags "trust_flags"
```

検証

- NSS データベースのすべての証明書を表示します。

```
# certutil -d /etc/dirsrv/slapd-instance_name/ -L

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

Example CA                                                   CT,,
```

関連情報

- **certutil(1)** の man ページ

5.2. WEB コンソールを使用した CA 信頼フラグの変更

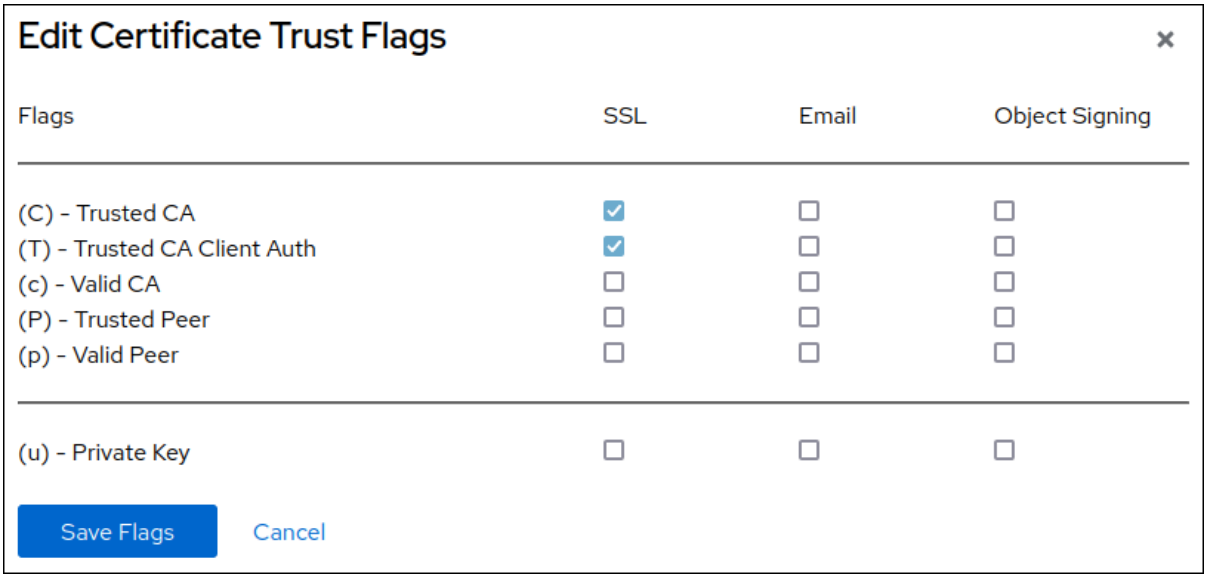
Web コンソールを使用して CA 信頼フラグを変更できます。

前提条件

- CA 証明書をネットワークセキュリティーサービス (NSS) データベースにインポートしている。

手順

1. **Server → Security → Certificate Management → Trusted Certificate Authorities** に移動します。
2. CA 証明書の横にある **...** アイコンをクリックして、**Edit Trust Flags** を選択します。
3. 信頼フラグを選択します。



4. **Save**をクリックします。

検証

1. **Server → Security → Certificate Management → Trusted Certificate Authorities** に移動します。
2. CA 証明書の横にある **>** をクリックして、信頼フラグを表示します。

第6章 TLS 証明書の更新

TLS 証明書には有効期限 (日時) があります。継続してセキュアな接続を提供するには、有効期限が切れる前に Directory Server のサーバー証明書を更新します。

6.1. コマンドラインでの TLS 証明書の更新

TLS サーバー証明書の有効期限が切れる前に、以下の手順に従います。

前提条件

- 属性の暗号化が設定されていない。
- TLS 証明書の有効期限がまもなく切れる。

手順

1. プライベートキーおよび証明書署名要求 (CSR) を作成します。外部ユーティリティーを使用して作成する場合は、この手順を省略します。

   - ホストが 1 つの名前のみで到達可能である場合は、以下を実行します。

     ```
     # dsctl instance_name tls generate-server-cert-csr -s
     "CN=server.example.com,O=example_organization"
     ```

   - 複数の名前でホストにアクセスできる場合は、以下を行います。

     ```
     # dsctl instance_name tls generate-server-cert-csr -s
     "CN=server.example.com,O=example_organization" server.example.com
     server.example.net
     ```

     最後のパラメーターとしてホスト名を指定した場合、このコマンドは **DNS:server.example.com, DNS:server.example.net** エントリーで SAN (Subject Alternative Name) 拡張を CSR に追加します。

   -s subject パラメーターで指定した文字列は、RFC 1485 に従って有効なサブジェクト名である必要があります。サブジェクトの **CN** フィールドが必要で、サーバーの完全修飾ドメイン名 (FQDN) の 1 つに設定する必要があります。このコマンドは、**/etc/dirsrv/slapd-instance_name/Server-Cert.csr** ファイルに CSR を保存します。

2. 認証局 (CA) に CSR を送信し、発行した証明書を取得します。詳細は、CA のドキュメントを参照してください。

3. CA 証明書とサーバー証明書の両方を **/root/** ディレクトリーに保存します。

4. 以下のオプションのいずれかを使用して、CA が発行するサーバー証明書を NSS データベースにインポートします。

   - **dsctl tls generate-server-cert-csr** コマンドを使用して秘密鍵を作成した場合は、以下を入力します。

     ```
     # dsconf -D "cn=Directory Manager" ldap://server.example.com security certificate
     add --file /root/instance_name.crt --name "server-cert" --primary-cert
     ```

--name _certificate_nickname パラメーターで設定した証明書の名前を書き留めておきます。これは後のステップで必要になります。

- 外部ユーティリティーを使用して秘密鍵を作成した場合は、サーバー証明書および秘密鍵をインポートします。

```
# dsctl instance_name tls import-server-key-cert /root/server.crt /root/server.key
```

このコマンドでは、最初にサーバー証明書へのパスを指定してから、秘密鍵へのパスを指定する必要があります。このメソッドは、証明書のニックネームを **Server-Cert** に設定します。

5. CA 証明書を NSS データベースにインポートします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ca-certificate add --file /root/ca.crt --name "Example CA"
```

6. CA 証明書の信頼フラグを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ca-certificate set-trust-flags "Example CA" --flags "CT,,"
```

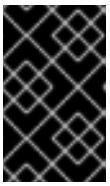
これにより、Directory Server が、TLS による暗号化および証明書ベースの認証に対して CA を信頼するように設定します。

7. インスタンスを停止します。

```
# dsctl instance_name stop
```

8. **/etc/dirsrv/slapd-instance_name/dse.ldif** ファイルを編集し、属性を含む以下のエントリーを削除します。

- **cn=AES,cn=encrypted attribute keys,cn=database_name,cn=ldbm database,cn=plugins,cn=config**
- **cn=3DES,cn=encrypted attribute keys,cn=database_name,cn=ldbm database,cn=plugins,cn=config**



重要

全データベースのエントリーを削除します。**nsSymmetricKey** 属性を含むエントリーが **/etc/dirsrv/slapd-instance_name/dse.ldif** ファイルに残されると、Directory Server は起動に失敗します。

9. インスタンスを起動します。

```
# dsctl instance_name start
```

第7章 証明書ベースの認証の設定

Directory Server は、LDAP クライアントの証明書ベースの認証と、レプリケーショントポロジーなどのサーバー間接続をサポートしています。

設定に応じて、クライアントは証明書を使用して認証できるか、認証する必要があります。サーバーは、証明書の subject フィールドの属性に基づいて証明書を検証した後、ディレクトリー内でユーザーを検索します。検索でユーザーエントリーを 1 つだけ返すと、Directory Server はこのユーザーを使用してすべての操作を行います。必要に応じて、認証に使用される証明書が、ユーザーエントリーの **userCertificate** 属性に保存されている Distinguished Encoding Rules (DER) 形式の証明書と一致するように設定できます。

証明書ベースの認証を使用する利点:

- 効率の向上: 証明書データベースのパスワードを使用して認証し、その後のすべてのバインド操作または認証操作でその証明書を使用する方が、バインド識別名 (DN) とパスワードを繰り返し提供するよりも効率的です。
- セキュリティーの向上: 証明書ベースの認証は、証明書ベースの認証では公開鍵の暗号化が使用されるため、証明書以外のバインド操作よりも安全です。攻撃者は、ネットワーク全体でバインド認証情報を傍受できません。証明書やデバイスが失われた場合は、PIN がなければ役に立たないため、フィッシング攻撃などのサードパーティーの干渉の影響を受けません。

7.1. 証明書ベースの認証の設定

前提条件

- Directory Server で TLS 暗号化を有効にしている。
- ネットワークセキュリティーサービス (NSS) データベースで認証局 (CA) 証明書の **CT** フラグを設定します。

手順

1. **/etc/dirsrv/slapd-instance_name/certmap.conf** ファイルを作成し、証明書から Directory Server ユーザーへ情報をマッピングします。

```
certmap default         default
default:DNComps         dc
default:FilterComps     mail,cn
default:VerifyCert      on

certmap example         cn=Example CA
example:DNComps
```

この設定では、**cn=Example CA** によって発行された証明書の場合、この発行者の **DNComps** パラメーターが空に設定されているため、Directory Server は証明書のサブジェクトからベース DN を生成しません。さらに、**FilterComps** と **VerifyCert** の設定は、デフォルトのエントリーから継承されます。

cn=Example CA とは異なる発行者 DN を持つ証明書は、デフォルトエントリーの設定を使用し、証明書のサブジェクトの cn 属性に基づいてベース DN を生成します。これにより、ディレクトリー全体を検索せずに、Directory Server が特定の DN で検索を開始できます。

すべての証明書について、Directory Server は、証明書のサブジェクトの **mail** 属性および **cn** 属性を使用して検索フィルターを生成します。ただし、件名に **mail** 属性が存在しない場合、Directory Server は自動的に証明書の e 属性の値を件名に使用します。

2. 証明書ベースの認証を有効にします。たとえば、証明書ベースの認証をオプションとして設定するには、次のように入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security set --tls-client-auth="allowed"
```

--tls-client-auth=required オプションを使用して、証明書ベースの認証を必須として設定します。

3. オプション: 証明書ベースの認証を必須として設定した場合は、**nsslapd-require-secure-binds** パラメーターを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-require-secure-binds=on
```

この設定により、ユーザーは暗号化されていない接続を使用して証明書ベースの認証をバイパスできなくなります。

4. オプション: Directory Server がバインド要求の認証情報の代わりに証明書の ID を使用する必要がある場合は、**EXTERNAL** 簡易認証およびセキュリティー層 (SASL) メカニズムを使用するように Directory Server を設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-force-sasl-external=on
```

この設定では、Directory Server は証明書内の ID 以外のバインド方法を無視します。

5. インスタンスを再起動します。

```
# dsctl instance_name restart
```

次のステップ:

- 認証証明書がユーザーの **userCertificate** 属性に格納されている証明書と一致する必要があるように Directory Server を設定した場合は、証明書をユーザーエントリーに追加します。詳細は、「ユーザーへの証明書の追加」 を参照してください。

関連情報

- Directory Server への TLS 暗号化接続の有効化
- CA 信頼フラグの変更
- **certmap.conf(5)** man ページ

7.2. ユーザーへの証明書の追加

証明書ベースの認証を設定すると、認証に使用される証明書がユーザーの **userCertificate** バイナリー属性に格納されている証明書と一致する必要があるようにサーバーを設定できます。この機能を有効にした場合は、影響を受けるユーザーの証明書をディレクトリーエントリーに追加する必要があります。

前提条件

- Directory Server で証明書ベースの認証を有効にした。
- サーバーによって信頼されている認証局 (CA) によって発行されたクライアント証明書がある。
- クライアント証明書は識別符号化規則 (DER) 形式である。
- クライアント証明書は、サーバーの **/etc/dirsrv/slapd-instance_name/certmap.conf** で設定された要件を満たしている。

手順

1. 証明書が DER 形式でない場合は、変換します。たとえば、証明書を Privacy Enhanced Mail (PEM) から DER に変換するには、次のように入力します。

```
# openssl x509 -in /home/user_name/certificate.pem -out
/home/user_name/certificate.der -outform DER
```

2. ユーザーの **userCertificate** 属性に証明書を追加します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldaps://server.example.com -x

dn: uid=user_name,ou=People,dc=example,dc=com
changetype: modify
add: userCertificate
userCertificate:< file:///home/user_name/example.der
```

検証

1. 証明書ベースの認証を使用してユーザーとして認証します。

   a. CA 証明書、ユーザーキー、およびユーザー証明書の対応するパスに、以下の環境変数を設定します。

```
LDAPTLS_CACERT=/home/user_name/CA.crt
LDAPTLS_KEY=/home/user_name/user.key
LDAPTLS_CERT=/home/user_name/user.der
```

   または、現在のユーザーの **~/.ldaprc** ファイルで **TLS_CACERT**、**TLS_KEY**、および **TLS_CERT** パラメーターを設定します。

   b. サーバーに接続します。

```
# ldapwhoami -H ldaps://server.example.com -Y EXTERNAL
dn: uid=example,ou=people,dc=example,dc=com
```

関連情報

- **ldap.conf(5)** man ページの **TLS OPTIONS** セクション

第8章 証明書ベースの認証を使用したマルチサプライヤーレプリケーションの設定

2 つの Directory Server インスタンス間のレプリケーションを設定する場合、バインド DN とパスワードを使用してレプリケーションパートナーを認証する代わりに、証明書ベースの認証を使用できます。

これを行うには、新しいサーバーをレプリケーショントポロジーに追加し、証明書ベースの認証を使用して新しいホストと既存のサーバーの間にレプリケーションアグリーメントを設定します。

重要

証明書ベースの認証には、TLS 暗号化接続が必要です。

8.1. 証明書ベースの認証による複製合意で使用するためのアカウントとバインドグループの準備

複製合意で証明書ベースの認証を使用するには、まずアカウントを準備し、クライアント証明書をこれらのアカウントの **userCertificate** 属性に保存します。さらに、この手順では、後で複製合意で使用するバインドグループを作成します。

この手順は、既存のホスト **server1.example.com** で実行します。

前提条件

- Directory Server で TLS 暗号化を有効にしています。
- **/root/server1.der** ファイルと **/root/server2.der** ファイルにクライアント証明書を識別名エンコーディングルール (DER) 形式で保存しました。
  クライアント証明書の詳細と、認証局 (CA) から証明書を要求する方法については、CA のドキュメントを参照してください。

手順

1. **ou=services** エントリーが存在しない場合は作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldaps://server1.example.com -x

dn: ou=services,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: services
```

2. **cn=server1,ou=services,dc=example,dc=com** および **cn=server1,ou=services,dc=example,dc=com** など、両方のサーバーのアカウントを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldaps://server1.example.com -x

dn: cn=server1,ou=services,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: inetOrgPerson
sn: server1
```

```
cn: server1
userPassword: password
userCertificate:< file:///root/server1.der

adding new entry "cn=server1,ou=services,dc=example,dc=com"

dn: cn=server2,ou=services,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: inetOrgPerson
sn: server2
cn: server2
userPassword: password
userCertificate:< file:///root/server2.der

adding new entry "cn=server2,ou=services,dc=example,dc=com"
```

3. **cn=repl_servers,dc=groups,dc=example,dc=com** などのグループを作成します。

```
# dsidm -D "cn=Directory Manager" ldaps://server1.example.com -b
"dc=example,dc=com" group create --cn "repl_servers"
```

4. 2 つのレプリケーションアカウントをメンバーとしてグループに追加します。

```
# dsidm -D "cn=Directory Manager" ldaps://server1.example.com -b
"dc=example,dc=com" group add_member repl_servers
"cn=server1,ou=services,dc=example,dc=com"

# dsidm -D "cn=Directory Manager" ldaps://server1.example.com -b
"dc=example,dc=com" group add_member repl_servers
"cn=server2,ou=services,dc=example,dc=com"
```

関連情報

- Directory Server への TLS 暗号化接続の有効化

8.2. 一時レプリケーションマネージャーアカウントを使用した新しいサーバーの初期化

証明書ベースの認証では、ディレクトリーに格納されている証明書が使用されます。ただし、新しいサーバーを初期化する前は、**server2.example.com** のデータベースは空であり、関連付けられた証明書を持つアカウントは存在しません。したがって、データベースの初期化前に、証明書を使用してレプリケーションすることはできません。この問題は、**server2.example.com** を一時的なレプリケーションマネージャーアカウントで初期化することで解決できます。

前提条件

- Directory Server インスタンスを **server2.example.com** にインストールした。
- **dc=example,dc=com** 接尾辞のデータベースが存在する。
- **server1.example.com** と **server2.example.com** の両方のサーバーの Directory Server で TLS 暗号化を有効にした。

手順

1. **server2.example.com** で、**dc=example,dc=com** 接尾辞のレプリケーションを有効にします。

```
# dsconf -D "cn=Directory Manager" ldaps://server2.example.com replication enable --suffix "dc=example,dc=com" --role "supplier" --replica-id 2 --bind-dn "cn=replication manager,cn=config" --bind-passwd "password"
```

このコマンドは、**server2.example.com** ホストを **dc=example,dc=com** 接尾辞のサプライヤーとして設定し、このホストのレプリカ ID を **2** に設定します。さらに、このコマンドは、指定されたパスワードで一時的な **cn=replication manager,cn=config** ユーザーを作成し、このアカウントが接尾辞の変更をこのホストに複製できるようにします。

トポロジー内のすべてのサプライヤーの接尾辞については、レプリカ ID は **1** から **65534** の間の一意の整数である必要があります。

2. **server1.example.com** 上で:

   a. レプリケーションを有効にします。

```
# dsconf -D "cn=Directory Manager" ldaps://server1.example.com replication enable --suffix="dc=example,dc=com" --role="supplier" --replica-id="1"
```

   b. 認証用に直前の手順で一時的なアカウントを使用する一時的なレプリカ合意を作成します。

```
# dsconf -D "cn=Directory Manager" ldaps://server1.example.com repl-agmt create --suffix="dc=example,dc=com" --host="server1.example.com" --port=636 --conn-protocol=LDAPS --bind-dn="cn=Replication Manager,cn=config" --bind-passwd="password" --bind-method=SIMPLE --init temporary_agreement
```

検証

1. 初期化が成功したことを確認します。

```
# dsconf -D "cn=Directory Manager" ldaps://server1.example.com repl-agmt init-status --suffix "dc=example,dc=com" temporary_agreement
Agreement successfully initialized.
```

関連情報

- Red Hat Directory Server のインストール
- Directory Server への TLS 暗号化接続の有効化

8.3. 証明書ベースの認証を使用したマルチサプライヤーレプリケーションの設定

証明書ベースの認証を使用するマルチサプライヤーレプリケーション環境では、レプリカは証明書を使用して相互に認証します。

前提条件

- **server1.example.com** と **server2.example.com** の両方のホストで証明書ベースの認証を設定します。
- Directory Server は、クライアント証明書を発行する認証局 (CA) を信頼します。
- クライアント証明書は、サーバーの **/etc/dirsrv/slapd-instance_name/certmap.conf** で設定された要件を満たしています。

手順

1. **server1.example.com** 上で:
   a. 一時的なレプリカ合意を削除します。

```
# dsconf -D "cn=Directory Manager" ldaps://server1.example.com repl-agmt delete
--suffix="dc=example,dc=com" temporary_agreement
```

   b. **cn=repl_servers,dc=groups,dc=example,dc=com** バインドグループをレプリケーション設定に追加します。

```
# dsconf -D "cn=Directory Manager" ldaps://server1.example.com replication set --
suffix="dc=example,dc=com" --repl-bind-group
"cn=repl_servers,dc=groups,dc=example,dc=com"
```

   c. バインドグループの変更を自動的にチェックするように Directory Server を設定します。

```
# dsconf -D "cn=Directory Manager" ldaps://server1.example.com replication set --
suffix="dc=example,dc=com" --repl-bind-group-interval=0
```

2. **server2.example.com** 上で:
   a. 一時的なレプリケーションマネージャーアカウントを削除します。

```
# dsconf -D "cn=Directory Manager" ldaps://server2.example.com replication
delete-manager --suffix="dc=example,dc=com" --name="Replication Manager"
```

   b. **cn=repl_servers,dc=groups,dc=example,dc=com** バインドグループをレプリケーション設定に追加します。

```
# dsconf -D "cn=Directory Manager" ldaps://server2.example.com replication set --
suffix="dc=example,dc=com" --repl-bind-group
"cn=repl_servers,dc=groups,dc=example,dc=com"
```

   c. バインドグループの変更を自動的にチェックするように Directory Server を設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server2.example.com replication set --
suffix="dc=example,dc=com" --repl-bind-group-interval=0
```

   d. 証明書ベースの認証を使用して複製合意を作成します。

```
dsconf -D "cn=Directory Manager" ldaps://server2.example.com repl-agmt create --
suffix="dc=example,dc=com" --host="server1.example.com" --port=636 --conn-
protocol=LDAPS --bind-method="SSLCLIENTAUTH" --init server2-to-server1
```

3. **server1.example.com** で、証明書ベースの認証を使用して複製合意を作成します。

```
dsconf -D "cn=Directory Manager" ldaps://server1.example.com repl-agmt create --
suffix="dc=example,dc=com" --host="server2.example.com" --port=636 --conn-
protocol=LDAPS --bind-method="SSLCLIENTAUTH" --init server1-to-server2
```

検証

1. 初期化が成功したことを各サーバーで確認します。

```
# dsconf -D "cn=Directory Manager" ldaps://server1.example.com repl-agmt init-status
--suffix "dc=example,dc=com" server1-to-server2
Agreement successfully initialized.

# dsconf -D "cn=Directory Manager" ldaps://server2.example.com repl-agmt init-status
--suffix "dc=example,dc=com" server2-to-server1
Agreement successfully initialized.
```

関連情報

- 証明書ベースの認証の設定
- CA 信頼フラグの変更

第9章 レプリケーション変更ログの暗号化

攻撃者がサーバーのファイルシステムにアクセスできる場合は、レプリケーション changelog を暗号化してインスタンスのセキュリティーを強化します。

changelog 暗号化は、サーバーの TLS 暗号化キーと同じ PIN を使用してキーのロックを解除します。サーバーの起動時に PIN を手動で入力するか、PIN ファイルを使用する必要があります。

Directory Server は、無作為に生成された対称暗号キーを使用して、changelog を暗号化および復号化します。サーバーは、設定された暗号ごとに異なるキーを使用します。これらの鍵は、サーバーの TLS 証明書から公開鍵を使用してラップされ、生成したラップ済みキーがサーバーの設定ファイル内に保存されます。属性暗号化の効果的な強度は、ラップに使用されるサーバーの TLS キーの強度と同じです。サーバーの秘密鍵と PIN にアクセスできないと、ラップ済みのコピーから対称キーを復旧することができません。

9.1. コマンドラインを使用した CHANGELOG の暗号化

レプリケーショントポロジーのセキュリティーを向上させるには、サプライヤーおよびハブの changelog を暗号化します。この手順では、**dc=example,dc=com** 接尾辞に対して changelog 暗号化を有効にする方法を説明します。

前提条件

- サーバーの TLS 暗号化が有効になっている。
- ホストは、レプリケーショントポロジー内のサプライヤーまたはハブです。

手順

1. changelog(例: **/tmp/changelog.ldif** ファイル) をエクスポートします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication export-changelog to-ldif -o /tmp/changelog.ldif -r "dc=example,dc=com"
```

2. **dc=example,dc=com** 接尾辞の changelog 暗号化を有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication --suffix "dc=example,dc=com" --encrypt
```

3. **/tmp/changelog.ldif** ファイルから changelog をインポートします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication import-changelog from-ldif -r "dc=example,dc=com" /tmp/changelog.ldif
```

4. インスタンスを再起動します。

```
# dsctl instance_name restart
```

検証

1. エントリーの更新など、LDAP ディレクトリーに変更を加えます。
2. インスタンスを停止します。

```
# dsctl instance_name stop
```

3. 接尾辞とそれに対応するデータベースをリスト表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend suffix list
dc=example,dc=com (userroot)
```

changelog の暗号化を有効にするデータベースの名前を書き留めておきます。

4. 以下のコマンドを実行して、changelog の一部を表示します。

```
# dbscan -f /var/lib/dirsrv/slapd-instance_name/db/userroot/replication_changelog.db |
tail -50
```

changelog が暗号化されている場合は、暗号化されたデータのみが表示されます。

5. インスタンスを起動します。

```
# dsctl instance_name start
```

関連情報

- Directory Server への TLS 暗号化接続の有効化

第10章 グループのメンバーが DIRECTORY SERVER をバックアップすることの許可、およびグループメンバーの 1 つとしてのバックアップの実行

グループのメンバーに、インスタンスをバックアップして、そのバックアップを実施するパーミッションを設定できます。バックアップスクリプトまたは cron ジョブに **cn=Directory Manager** の認証情報を設定する必要がなくなるため、セキュリティーが向上します。また、グループを変更して、バックアップのパーミッションを簡単に許可し、取り消すことができます。

10.1. グループが DIRECTORY SERVER をバックアップすることの許可

この手順を使用して、**cn=backup_users,ou=groups,dc=example,dc=com** グループを追加し、このグループのメンバーがバックアップタスクを作成するのを許可します。

前提条件

- エントリー **ou=groups,dc=example,dc=com** がデータベースに存在する。

手順

1. **cn=backup_users,ou=groups,dc=example,dc=com** グループを作成します。

```
# dsidm -D "cn=Directory manager" ldap://server.example.com -b
"dc=example,dc=com" group create --cn backup_users
```

2. **cn=backup_users,ou=groups,dc=example,dc=com** グループのメンバーによるバックアップタスクの作成を許可するアクセス制御手順 (ACI) を追加します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com

dn: cn=config
changetype: modify
add: aci
aci: (target = "ldap:///cn=backup,cn=tasks,cn=config")(targetattr="*")
 (version 3.0 ; acl "permission: Allow backup_users
  group to create backup tasks" ; allow (add, read, search) groupdn
  = "ldap:///cn=backup_users,ou=groups,dc=example,dc=com";)
-
add: aci
aci: (target = "ldap:///cn=config")(targetattr = "nsslapd-bakdir ||
  objectClass") (version 3.0 ; acl "permission: Allow backup_users
  group to access bakdir attribute" ; allow (read,search)
  groupdn = "ldap:///cn=backup_users,ou=groups,dc=example,dc=com";)
```

3. ユーザーを作成します。

   a. ユーザーアカウントを作成します。

```
# dsidm -D "cn=Directory manager" ldap://server.example.com -b
"dc=example,dc=com" user create --uid="example" --cn="example" --
uidNumber="1000" --gidNumber="1000" --homeDirectory="/home/example/" --
displayName="Example User"
```

b. ユーザーアカウントのパスワードを設定します。

```
# dsidm -D "cn=Directory manager" ldap://server.example.com -b
"dc=example,dc=com" account reset_password
"uid=example,ou=People,dc=example,dc=com" "password"
```

4. **uid=example,ou=People,dc=example,dc=com** ユーザーを **cn=backup_users,ou=groups,dc=example,dc=com** グループに追加します。

```
# dsidm -D "cn=Directory manager" ldap://server.example.com -b
"dc=example,dc=com" group add_member backup_users
uid=example,ou=People,dc=example,dc=com
```

検証

- **cn=config** エントリーに設定された ACI を表示します。

```
# ldapsearch -o ldif-wrap=no -LLLx -D "cn=directory manager" -W -H
ldap://server.example.com -b cn=config aci=* aci -s base
dn: cn=config
aci: (target = "ldap:///cn=backup,cn=tasks,cn=config")(targetattr="*")(version 3.0 ; acl
"permission: Allow backup_users group to create backup tasks" ; allow (add, read, search)
groupdn = "ldap:///cn=backup_users,ou=groups,dc=example,dc=com";)
aci: (target = "ldap:///cn=config")(targetattr = "nsslapd-bakdir || objectClass")(version 3.0 ; acl
"permission: Allow backup_users group to access bakdir attribute" ; allow (read,search)
groupdn = "ldap:///cn=backup_users,ou=groups,dc=example,dc=com";)
...
```

10.2. 通常ユーザーとしてのバックアップの実行

cn=Directory Manager ではなく、通常のユーザーとしてバックアップを実行できます。

前提条件

- **cn=backup_users,ou=groups,dc=example,dc=com** グループのメンバーがデータをバックアップするのを許可している。
- バックアップの実行に使用するユーザーが **cn=backup_users,ou=groups,dc=example,dc=com** グループのメンバーである。

手順

- 以下の方法のいずれかを使用してバックアップタスクを作成します。
  - **dsconf backup create** コマンドの使用:

```
# dsconf -D "uid=example,ou=People,dc=example,dc=com"
ldap://server.example.com backup create
```

  - タスクの手動での作成:

```
# ldapadd -D "uid=example,ou=People,dc=example,dc=com" -W -H
ldap://server.example.com
```

```
dn: cn=backup-2021_07_23_12:55_00,cn=backup,cn=tasks,cn=config
changetype: add
objectClass: extensibleObject
nsarchivedir: /var/lib/dirsrv/slapd-instance_name/bak/backup-2021_07_23_12:55_00
nsdatabasetype: ldbm database
cn: backup-2021_07_23_12:55_00
```

検証

- バックアップが作成されたことを確認します。

```
# ls -l /var/lib/dirsrv/slapd-instance_name/bak/
total 0
drwx------. 3 dirsrv dirsrv 108 Jul 23 12:55 backup-2021_07_23_12_55_00
...
```

関連情報

- グループが Directory Server をバックアップすることの許可

第11章 グループのメンバーがデータをエクスポートすることの許可、およびグループメンバーの 1 つとしてのエクスポートの実行

グループのメンバーに、データをエクスポートするパーミッションを設定できます。スクリプトに **cn=Directory Manager** の認証情報を設定する必要がなくなるため、セキュリティーが向上します。また、グループを変更して、エクスポートのパーミッションを簡単に許可し、取り消しすことができます。

11.1. グループによるデータエクスポートの許可

この手順を使用して、**cn=export_users,ou=groups,dc=example,dc=com** グループを追加し、このグループのメンバーがエクスポートタスクを作成することを許可します。

手順

1. **cn=export_users,ou=groups,dc=example,dc=com** グループを作成します。

```
# dsidm -D "cn=Directory manager" ldap://server.example.com -b
"dc=example,dc=com" group create --cn export_users
```

2. **cn=export_users,ou=groups,dc=example,dc=com** グループのメンバーがエクスポートタスクを作成することを許可するアクセス制御手順 (ACI) を追加します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com

dn: cn=config
changetype: modify
add: aci
aci: (target = "ldap:///cn=export,cn=tasks,cn=config")
 (targetattr="*") (version 3.0 ; acl "permission:
  Allow export_users group to export data" ;
  allow (add, read, search) groupdn
  = "ldap:///cn=export_users,ou=groups,dc=example,dc=com";)
-
add: aci
aci: (target = "ldap:///cn=config")(targetattr =
  "objectclass || cn || nsslapd-suffix || nsslapd-ldifdir")
  (version 3.0 ; acl "permission: Allow export_users
  group to access ldifdir attribute" ; allow
  (read,search) groupdn = "ldap:///cn=export_users,ou=groups,dc=example,dc=com";)
```

3. ユーザーを作成します。

   a. ユーザーアカウントを作成します。

```
# dsidm -D "cn=Directory manager" ldap://server.example.com -b
"dc=example,dc=com" user create --uid="example" --cn="example" --
uidNumber="1000" --gidNumber="1000" --homeDirectory="/home/example/" --
displayName="Example User"
```

   b. ユーザーアカウントのパスワードを設定します。

```
# dsidm -D "cn=Directory manager" ldap://server.example.com -b
"dc=example,dc=com" account reset_password
"uid=example,ou=People,dc=example,dc=com" "password"
```

4. **uid=example,ou=People,dc=example,dc=com** ユーザーを **cn=export_users,ou=groups,dc=example,dc=com** グループに追加します。

```
# dsidm -D "cn=Directory manager" ldap://server.example.com -b
"dc=example,dc=com" group add_member export_users
uid=example,ou=People,dc=example,dc=com
```

検証

- **cn=config** エントリーに設定された ACI を表示します。

```
# ldapsearch -o ldif-wrap=no -LLLx -D "cn=directory manager" -W -H
ldap://server.example.com -b cn=config aci=* aci -s base
dn: cn=config
aci: (target = "ldap:///cn=export,cn=tasks,cn=config")(targetattr="*")(version 3.0 ; acl
"permission: Allow export_users group to export data" ; allow (add, read, search) groupdn =
"ldap:///cn=export_users,ou=groups,dc=example,dc=com";)
aci: (target = "ldap:///cn=config")(targetattr = "objectclass || cn || nsslapd-suffix || nsslapd-
ldifdir")(version 3.0 ; acl "permission: Allow export_users group to access ldifdir attribute" ;
allow (read,search) groupdn = "ldap:///cn=export_users,ou=groups,dc=example,dc=com";)
...
```

11.2. 通常ユーザーとしてのエクスポートの実行

cn=Directory Manager ではなく、通常のユーザーとしてエクスポートを実行できます。

前提条件

- **cn=export_users,ou=groups,dc=example,dc=com** グループのメンバーがデータをエクスポートすることを許可している。
- エクスポートの実行に使用するユーザーが **cn=export_users,ou=groups,dc=example,dc=com** グループのメンバーである。

手順

- 以下の方法のいずれかを使用してエクスポートタスクを作成します。
  - **dsconf backend export** コマンドの使用:

```
# dsconf -D "uid=example,ou=People,dc=example,dc=com"
ldap://server.example.com backend export userRoot
```

  - タスクの手動での作成:

```
# ldapadd -D "uid=example,ou=People,dc=example,dc=com" -W -H
ldap://server.example.com

dn: cn=userRoot-2021_07_23_12:55_00,cn=export,cn=tasks,cn=config
```

```
changetype: add
objectClass: extensibleObject
nsFilename: /var/lib/dirsrv/slapd-instance_name/ldif/None-userroot-
2021_07_23_12:55_00.ldif
nsInstance: userRoot
cn: export-2021_07_23_12:55_00
```

検証

- バックアップが作成されたことを確認します。

```
# ls -l /var/lib/dirsrv/slapd-instance_name/ldif/*.ldif
total 0
-rw-------. 1 dirsrv dirsrv 10306 Jul 23 12:55 None-userroot-2021_07_23_12_55_00.ldif
...
```

関連情報

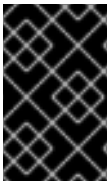
- グループがデータをエクスポートすることの許可

第12章 アクセス制御手順の管理

Directory Server が要求を受信すると、bind 操作でユーザーによって提供される認証情報、およびディレクトリーに定義されているアクセス制御の手順 (ACI) を使用し、要求されたエントリーまたは属性へのアクセスを許可または拒否します。サーバーは、**read**、**write**、**search**、**compare** などのアクションのパーミッションを許可または拒否できます。ユーザーに付与されたパーミッションレベルは、指定される認証情報によって異なります。

Directory Server のアクセス制御により、ACI が適用される場合に正確なルールを設定できます。

- ディレクトリー全体、サブツリー、または特定のエントリーの場合
- 特定のユーザー、特定のユーザーまたはグループに属するすべてのユーザー、またはディレクトリー内のすべてのユーザーの場合
- IP アドレス、IP 範囲、または DNS 名などの特定の場所。
  ロードバランサーは場所固有のルールに影響を及ぼす可能性があることに注意してください。



重要

複雑な ACI の読み取りと理解は難しくなります。1 つの複雑な ACI の代わりに、同じ効果を達成するために複数の単純なルールを作成できます。ただし、ACI が多いほど、ACI 処理のコストも増加します。

12.1. ACI 配置

Directory Server は、アクセス制御命令 (ACI) をディレクトリーエントリーの多値 **aci** 操作属性に保存します。ACI を設定するには、**aci** 属性を対応するディレクトリーエントリーに追加します。Directory Server は ACI を適用します。

- ACI を含むエントリー (子エントリーがない場合) にのみ適用されます。たとえば、クライアントが **uid=user_name,ou=People,dc=example,dc=com** オブジェクトへのアクセスを必要とし、ACI が **dc=example,dc=com** にのみ設定されており、子エントリーには設定されていない場合は、この ACI のみが適用されます。

  注記

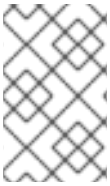
  add 権限を持つ ACI は、今後作成される子エントリーにも適用されます。

- ACI を含むエントリーと、(子エントリーがある場合は) その下のすべてのエントリーへ。これにより、サーバーが指定のエントリーに対するアクセスパーミッションを評価すると、リクエストされたディレクトリー接尾辞と、エントリー自体の ACI との間のすべてのエントリーについて ACI を検証します。
  たとえば、ACI は **dc=example,dc=com** および **ou=People,dc=example,dc=com** エントリーに設定されます。ACI が設定されていない **uid=user_name,ou=People,dc=example,dc=com** オブジェクトにクライアントがアクセスする場合、Directory Server はまず **ou=People,dc=example,dc=com** エントリー上の ACI を検証します。この ACI がアクセスを許可する場合は、評価は停止し、アクセスを許可します。そうでない場合は、Directory Server は **ou=People,dc=example,dc=com** 上の ACI を検証します。この ACI がクライアントを正常に承認すると、オブジェクトにアクセスできます。

注記

rootDSE エントリーに設定された ACI はこのエントリーにのみ適用されます。

エントリーで作成された ACI は、そのエントリーに直接適用するのではなく、以下のサブツリーの一部のエントリーまたはすべてのエントリーに適用できます。この方法の利点は、一般的な ACI をディレクトリーツリーの上位において、下位にあるエントリーに影響を与えることができることです。たとえば、**inetOrgPerson** オブジェクトクラスを含むエントリーをターゲットにする ACI は、**organizationalUnit** エントリーまたは locality エントリーのレベルで作成できます。



注記

一般的なルールを高レベルのブランチポイントに配置し、ディレクトリーツリー内の ACI の数を最小限にします。より具体的なルールの範囲を制限するには、できるだけ早くリーフエントリーに配置します。

12.2. ACI の構造

aci 属性は以下の構文を使用します。

> (**target_rule**) (version 3.0; acl "**ACL_name**"; **permission_rule bind_rules**;)

- **target_rule** は、アクセスを制御するためのエントリー、属性、またはエントリーと属性のセットを指定します。
- **バージョン 3.0** は、アクセス制御手順 (ACI) バージョンを特定する必須文字列です。
- **acl "ACL name"** は ACI を記述する名前または文字列を設定します。
- **permission_rule** は、**read**、**write** など、どの権限を許可または拒否されるかを設定します。
- **bind_rules** は、アクセスを許可または拒否するために、バインド時に一致するルールを指定します。

パーミッションとバインドルールのペアはアクセス制御ルールと呼ばれます。

特定のターゲットに複数のアクセス制御を効率的に設定するには、ターゲットごとに複数のアクセス制御ルールを設定します。

> (**target_rule**)(version 3.0; acl "**ACL_name**"; **permission_rule bind_rules**; **permission_rule bind_rules**; ... ;)

12.3. ACI 評価

特定のエントリーに対するアクセス権を評価するには、サーバーによりエントリー自体に存在するアクセス制御の手順 (ACI) のリストと、親エントリーにある ACI のリストが Directory Server に保存されている最上位のエントリーに再び作成されます。ACI は、特定のインスタンス用のデータベース全体で評価されますが、異なるインスタンスもすべて評価されます。

Directory Server は、ディレクトリーツリー内の配置ではなく、ACI のセマンティクスに基づいてこのリストを評価します。これは、ディレクトリーツリーのルートに近い ACI が、ディレクトリーツリーのリーフに近い ACI よりも優先されないことを意味しています。

Directory Server では、ACI の **deny** パーミッションは **allow** パーミッションよりも優先されます。たとえば、ディレクトリーのルートレベルで書き込みパーミッションを拒否する場合は、他の ACI がこのパーミッションを付与していても、ユーザーはディレクトリーに書き込むことができません。特定のユーザーにディレクトリーへの書き込みパーミッションを付与するには、ユーザーがそのディレクトリーに書き込むことができるように、元の拒否ルールに例外を追加する必要があります。

注記

ACI を改善するには、**deny** ルールの代わりに、粒度の細かい **allow** ルールを使用します。

12.4. ACI の制限

アクセス制御手順 (ACI) を設定する場合は、以下の制限が適用されます。

- ディレクトリーデータベースが複数のサーバーに分散されている場合は、ACI で使用できるキーワードに以下の制限が適用されます。
  - **groupdn** キーワードを使用したグループエントリーに依存する ACI は、グループエントリーと同じサーバーに置く必要があります。
    グループが動的の場合、グループのすべてのメンバーに、サーバーのエントリーが必要です。静的グループのメンバーエントリーは、リモートサーバーに配置できます。
  - **roledn** キーワードを使用したロール定義に依存する ACI は、ロール定義エントリーと同じサーバーにある必要があります。ロールを持つすべてのエントリーは、同じサーバーに配置する必要があります。

  ただし、ターゲットエントリーに保存されている値を、たとえば **userattr** キーワードを使用して、bind ユーザーのエントリーに保存されている値と一致させることができます。この場合、通常、バインドユーザーに ACI を格納するサーバーにエントリーがない場合でも、アクセスが評価されます。
- 以下の ACI キーワードでは、Class of Service (CoS) 属性などの仮想属性を使用することはできません。
  - **targetfilter**
  - **targattrfilters**
  - **userattr**
- アクセス制御ルールは、ローカルサーバーでのみ評価されます。たとえば、ACI キーワードの LDAP URL にサーバーのホスト名を指定すると、URL は無視されます。

12.5. DIRECTORY SERVER がレプリケーショントポロジーで ACI を処理する方法

アクセス制御手順 (ACI) は、エントリーの **aci** 属性に保存されます。したがって、ACI を含むエントリーが複製されたデータベースの一部である場合、ACI は複製されます。

ACI は、受信 LDAP 要求を解決するサーバーで常に評価されます。コンシューマーサーバーが更新要求を受け取ると、サプライヤーサーバーに参照を返してから、その要求がサプライヤーでサービスを提供できるかどうかを評価します。

12.6. ACI の表示、追加、削除、および更新

ldapsearch ユーティリティーを使用して検索、および **ldapmodify** ユーティリティーを使用して、アクセス制御手順 (ACI) を追加、削除、および更新できます。

ACI の表示

たとえば、**dc=example,dc=com** およびサブエントリーに設定された ACI を表示するには、以下のコマンドを実行します。

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -x -b
"dc=example,dc=com" -s sub '(aci=*)' aci
```

ACI の追加

たとえば、ACI を **ou=People,dc=example,dc=com** エントリーに追加するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="userPassword") (version 3.0; acl
  "Allow users updating their password";
  allow (write) userdn= "ldap:///self";)
```

ACI の削除

ACI を削除するには、以下を実行します。

- 1 つの **aci** 属性がエントリーに設定されているか、エントリーからすべての ACI を削除する場合は、以下を実行します。

  ```
  # ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

  dn: ou=People,dc=example,dc=com
  changetype: delete
  delete: aci
  ```

- 複数の ACI がエントリーに存在し、特定の ACI を削除する場合は、実際の ACI を指定します。

  ```
  # ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

  dn: ou=People,dc=example,dc=com
  changetype: modify
  delete: aci
  aci: (targetattr="userPassword") (version 3.0; acl "Allow users
    updating their password"; allow (write) userdn= "ldap:///self";)
  ```

ACI の更新

ACI を更新するには、以下を実行します。

- 既存の ACI を削除します。
- 更新された設定で新しい ACI を追加します。

12.7. ACI ターゲットの定義

アクセス制御手順 (ACI) のターゲットルールは、Directory Server が ACI を適用するエントリーを定義します。ターゲットを設定しない場合、ACI は **aci** 属性が含まれるエントリーと以下のエントリーに適用されます。

ACI では、以下の強調表示された部分がターゲットルールになります。

> (**target_rule**)(version 3.0; acl "**ACL_name**"; **permission_rule bind_rules**;)

複雑な ACI の場合、Directory Server は ACI で異なるキーワードを持つ複数のターゲットルールをサポートします。

> (**target_rule_1**)(**target_rule_2**)(...)(version 3.0; acl "**ACL_name**"; **permission_rule bind_rules**;)

複数のターゲットルールを指定した場合に、その順番は関係ありません。以下のキーワードはそれぞれ、ACI で一度だけ使用できることに注意してください。

- **target**
- **targetattr**
- **targetattrfilters**
- **targetfilter**
- **target_from**
- **target_to**

12.7.1. ターゲットルールの構文

ターゲットルールの一般的な構文は、以下のとおりです。

> (**keyword comparison_operator "expression"**)

- **keyword**: ターゲットの種類を設定します。
- **comparison_operator**: 有効な値は **=** および **!=** で、ターゲットが式で指定されたオブジェクトであるかを示します。

警告

セキュリティー上の理由から、Red Hat は、他のすべてのエントリーまたは属性で指定の操作を許可するため、**!=** 演算子を使用しないことを推奨します。以下に例を示します。

```
(targetattr != "userPassword");(version 3.0; acl "example"); allow (write)
... );
```

前の例では、ACI を設定する識別名 (DN) の下にある **userPassword** 属性以外の属性の設定、更新、または削除を行うことができます。ただし、これにより、ユーザーはこの属性への書き込みアクセスを許可する **aci** 属性を追加することもできます。

- **expression**: ターゲットを設定し、引用符で囲む必要があります。式自体は使用するキーワードによって異なります。

12.7.2. ディレクトリーエントリーのターゲット

識別名 (DN) およびその下のエントリーに基づいてアクセスを制御するには、アクセス制御手順 (ACI) の **target** キーワードを使用します。**target** キーワードを使用するターゲットルールは、DN を式として取ります。

```
(target comparison_operator "ldap:///distinguished_name")
```

注記

対象となる DN またはその上位 DN に、**target** キーワードで ACI を設定する必要があります。たとえば、ou=People,dc=example,dc=com をターゲットにする場合、ACI を ou=People,dc=example,dc=com または dc=example,dc=com のいずれかに設定する必要があります。

例12.1 target キーワードの使用

ou=People,dc=example,dc=com エントリーに保存されているユーザーを有効にして、独自のエントリー内の全属性を検索および表示するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (target = "ldap:///ou=People,dc=example,dc=com") (version 3.0;
 acl "Allow users to read and search attributes of own entry"; allow (search, read)
 (userdn = "ldap:///self");)
```

target キーワードでのワイルドカードの使用

* ワイルドカード文字ターゲットに複数のエントリーを使用できます。

以下のターゲットルールの例は、uid 属性が文字 **a** で始まる値に設定される **ou=People,dc=example,dc=com** のすべてのエントリーと一致します。

```
(target = "ldap:///uid=a*,ou=People,dc=example,dc=com")
```

ワイルドカードの位置に応じて、ルールは属性値だけでなく、完全な DN にも適用されます。そのため、ワイルドカードを DN の一部の代わりに使用できます。

例12.2 ワイルドカードを使用したディレクトリーエントリーのターゲット

次のルールは、**dc=example,dc=com** ツリー内のすべてのエントリーを対象とし、**uid** 属性が一致するもので、**dc=example,dc=com** エントリー自体に格納されているエントリーだけではありません。

```
(target = "ldap:///uid=user_name*,dc=example,dc=com")
```

以前のターゲットルールは、以下のような複数のエントリーと一致します。

- **uid=user_name,dc=example,dc=com**
- **uid=user_name,ou=People,dc=example,dc=com**
- **uid=user_name2,dc=example,dc=com**

重要

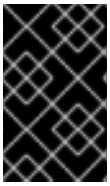
Directory Server は、DN の接尾辞部分でのワイルドカードをサポートしません。たとえば、ディレクトリーの接尾辞が **dc=example,dc=com** の場合は、**(target = "ldap:///dc=*.com")** などのように、この接尾辞でワイルドカード付きのターゲットは使用できません。

12.7.3. ターゲット属性

アクセス制御手順 (ACI) のアクセスを特定の属性に制限するには、**targetattr** キーワードを使用します。たとえば、このキーワードは以下を定義します。

- 読み取り操作では、どの属性がクライアントに返されるか
- 検索操作では、どのような属性が検索されるのか
- 書き込み操作では、どの属性がオブジェクトに書き込むことができるか
- add 操作では、新規オブジェクトの作成時に追加できる属性

特定の状況では、**targetattr** キーワードを使用して、他のターゲットキーワードを **targetattr** と組み合わせることで、ACI をセキュアにすることができます。ターゲットルールの高度な使用方法 を参照してください。



重要

read および **search** 操作では、デフォルトのターゲットは無属性です。**targetattr** キーワードのない ACI は、**add**、**delete** などの完全なエントリーに影響する ACI にのみ役に立ちます。

targetattr キーワードを使用するターゲットルールで複数の属性を分離するには、|| を使用します。

```
(targetattr comparison_operator "attribute_1 || attribute_2 || ...")
```

式に設定された属性はスキーマに定義する必要があります。

式に指定される属性は、ACI の作成先となるエントリーと、さらにターゲットルールによって制限されない場合は、それ以下のすべてのエントリーに適用されます。

例12.3 targetattr キーワードの使用

dc=example,dc=com に保存されているユーザーとすべてのサブエントリーで、独自のエントリー内の **userPassword** 属性を更新するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap::server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "userPassword") (version 3.0;
 acl "Allow users updating own userPassword";
 allow (write) (userdn = "ldap:///self");)
```

targetattr キーワードでのワイルドカードの使用

* ワイルドカード文字を使用すると、たとえば全属性をターゲットにすることができます。

```
(targetattr = "*")
```

警告

セキュリティー上の理由から、操作属性を含むすべての属性へのアクセスが許可されているため、**targetattr** ではワイルドカードを使用しないでください。たとえば、ユーザーがすべての属性を追加または変更できると、ユーザーは追加の ACI を作成し、独自の権限を増やす可能性があります。

12.7.4. LDAP フィルターを使用したエントリーと属性の対象

特定の基準に一致するエントリーのグループを対象にするには、LDAP フィルターで **targetfilter** キーワードを使用します。

```
(targetfilter comparison_operator "LDAP_filter")
```

フィルター式は、標準の LDAP 検索フィルターです。

例12.4 targetfilter キーワードの使用

department 属性が **Engineering** または **Sales** に設定されているすべてのエントリーを変更するために、**cn=Human Resources,dc=example,dc.com** グループのメンバーにパーミッションを付与するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetfilter = "(|(department=Engineering)(department=Sales)")
 (version 3.0; acl "Allow HR updating engineering and sales entries";
 allow (write) (groupdn = "ldap:///cn=Human Resources,dc=example,dc.com");)
```

targetfilter キーワードはエントリー全体を対象にします。これを **targetattr** キーワードと組み合わせると、アクセス制御の手順 (ACI) はターゲットエントリーの属性のサブセットにのみ適用されます。フィルターに一致するエントリーの個別属性のターゲット設定 を参照してください。

注記

LDAP フィルターは、ディレクトリーに分散されるエントリーおよび属性をターゲットにする場合に便利です。ただし、フィルターにはアクセスを管理するオブジェクトの名前を直接付けないため、結果が予測できないことがあります。フィルターが設定された ACI がターゲットとするエントリーのセットは、属性が追加または削除される際に変更する可能性が高くなります。したがって、ACI で LDAP フィルターを使用する場合は、**ldapsearch** 操作などで同じフィルターを使用して、正しいエントリーおよび属性を対象としていることを確認してください。

targetfilter キーワードでのワイルドカードの使用

targetfilter キーワードは、標準の LDAP フィルターと同様にワイルドカードをサポートします。たとえば、値が **adm** で始まるすべての uid 属性をターゲットにするには、次のコマンドを実行します。

```
(targetfilter = "(uid=adm*) ...)
```

12.7.5. LDAP フィルターを使用した属性値のターゲット

アクセス制御を使用すると、属性の特定値を対象にできます。つまり、ある属性の値がアクセス制御手順 (ACI) で定義されている基準を満たしていれば、その属性に対してパーミッションを付与したり、拒否したりすることができるのです。属性の値に基づいてアクセスを許可または拒否する ACI は、値ベースの ACI と呼ばれます。これは、**ADD** および **DEL** 操作にのみ適用されます。検索権限を持つユーザーは、特定の値で制限できません。

値ベースの ACI を作成するには、以下の構文で **targattrfilters** キーワードを使用します。

- 1 つの属性とフィルターの組み合わせが含まれる操作の場合:

  ```
  (targattrfilters="operation=attribute:filter")
  ```

- 複数の属性とフィルターの組み合わせのある操作の場合:

  (targattrfilters="**operation**=**attribute_1**:**filter_1** && **attribute_2**:**filter_2** ... && **attribute_m**:**filter_m**")

- 複数の属性とフィルターを組み合わせた 2 つの操作の場合。

  (targattrfilters="**operation_1**=**attribute_1_1**:**filter_1_1** && **attribute_1_2**:**filter_1_2** ... && **attribute_1_m**:**filter_1_m** , **operation_2**=**attribute_2_1**:**filter_2_1** && **attribute_2_2:filter_2_2** ... & **attribute_2_n**:**filter_2_n** ")

上記の構文の例では、オペレーションを **add** または **del** のいずれかに設定できます。**attribute:filter** の組み合わせは、フィルターと、フィルターが適用される属性を設定します。

以下では、フィルターを一致させる方法を説明します。

- エントリーを作成する際に、新しいエントリーの属性にフィルターが適用されると、その属性の各インスタンスがフィルターに一致する必要があります。
- エントリーとフィルターを削除するとエントリーの属性に適用される場合、その属性の各インスタンスはフィルターと一致する必要があります。
- エントリーを変更し、操作が属性を追加する場合は、その属性に適用される **add** フィルターが一致している必要があります。
- 操作が属性を削除すると、その属性に適用される **del** フィルターが一致している必要があります。エントリーに属性の個別の値が置き換えられる場合は、**add** および **del** フィルターの両方が一致する必要があります。

例12.5 targattrfilters キーワードの使用

Admin ロールを除く独自のエントリーにロールを追加できるようにする ACI を作成するには、値が **123** 接頭辞で始まる限り、**telephone** 属性を追加するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap::server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targattrfilters="add=nsroledn:(!(nsroledn=cn=Admin)) &&
 telephoneNumber:(telephoneNumber=123*)") (version 3.0;
 acl "Allow adding roles and telephone";
 allow (add) (userdn = "ldap:///self");)
```

12.7.6. ソースおよび宛先 DN のターゲット

特定の状況では、管理者がディレクトリーエントリーを移動できるようにします。アクセス制御手順 (ACI) で **target_from** および **target_to** キーワードを使用すると、ユーザーを有効にしなくても、操作の送信元および宛先を指定できます。

- ACI に設定される別のソースからエントリーを移動します。
- エントリーを ACI のセットとして別の宛先に移動するには、以下のコマンドを実行します。

- ソースの識別名 (DN) から既存のエントリーを削除します。
- 宛先 DN に新規エントリーを追加するには、以下を行います。

例12.6 target_from および target_to キーワードの使用

uid=user,dc=example,dc=com アカウントがユーザーアカウントを **cn=staging,dc=example,dc=com** エントリーから **cn=people,dc=example,dc=com** に移動するようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap:server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (target_from="ldap:///uid=*,cn=staging,dc=example,dc=com")
 (target_to="ldap:///cn=People,dc=example,dc=com")
 (version 3.0; acl "MODDN from"; allow (moddn))
 userdn="ldap:///uid=user,dc=example,dc=com";)
```

ACI は、それらが定義されているサブツリーにのみ適用されます。この例では、ACI は **dc=example,dc=com** サブツリーにのみ適用されます。

target_from または **target_to** キーワードが設定されていない場合は、ACI がソースまたは宛先と一致します。

12.8. ターゲットルールの高度な使用方法

複数のキーワードを組み合わせることで、複雑なターゲットルールを作成できます。本セクションでは、ターゲットルールの高度な使用例を紹介します。

12.8.1. グループの作成およびメンテナンスへのパーミッションの委譲

特定の状況では、管理者はパーミッションを他のアカウントまたはグループに委譲する必要があることがあります。ターゲットキーワードを組み合わせることで、この要求を解決するセキュアなアクセス制御手順 (ACI) を作成できます。

例12.7 グループの作成およびメンテナンスへのパーミッションの委譲

uid=user,ou=People,dc=example,dc=com" アカウントが ou=groups,dc=example,dc=com エントリーでグループを作成および更新できるようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (target = "ldap:///cn=*,ou=Groups,dc=example,dc=com")
 (targattrfilters="add=objectclass:(|(objectclas=top)(objectclass=groupOfUniqueNames)))
 (targetattr="cn || uniqueMember || objectClass")
 (version 3.0; acl "example"; allow (read, search, write, add)
 (userdn = "ldap:///uid=test,ou=People,dc=example,dc=com");)
```

前述の例は、セキュリティー上の理由から、特定の制限を追加します。**uid=test,ou=People,dc=example,dc=com** ユーザー:

- **top** オブジェクトクラスおよび **groupOfUniqueNames** オブジェクトクラスが含まれる必要があるオブジェクトを作成できます。
- **account** などの追加のオブジェクトクラスを追加できません。たとえば、ローカル認証に Directory Server アカウントを使用して、無効なユーザー ID (例: **root** ユーザーの **0**) を持つ新規ユーザーを作成できなくなります。

targetfilter ルールは、ACI エントリーが **groupofuniquenames** オブジェクトクラスを持つエントリーにのみ適用され、**targetattrfilter** ルールにより、他のオブジェクトクラスも追加されないようにします。

12.8.2. エントリーと属性の両方をターゲットに設定

target は、識別名 (DN) に基づいてアクセスを制御します。ただし、ワイルドカードと **targetattr** キーワードと組み合わせて使用する場合は、エントリーと属性の両方をターゲットにすることができます。

例12.8 エントリーと属性の両方をターゲットに設定

uid=user,ou=People,dc=example,dc.com ユーザーが、**dc=example,dc=com** サブツリー内のすべての組織単位でグループのメンバーを読み取り、検索できるようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///cn=*,dc=example,dc=com")(targetattr="member" || "cn") (version 3.0;
 acl "Allow uid=user to search and read members of groups";
 allow (read, search) (userdn = "ldap:///uid=user,ou=People,dc=example,dc.com");)
```

12.8.3. フィルターに一致するエントリーの個別属性のターゲット設定

2 つのターゲットルールで **targetattr** および **targetfilter** キーワードを組み合わせる場合は、フィルターに一致するエントリーの特定の属性をターゲットにすることができます。

例12.9 フィルターに一致するエントリーの個別属性のターゲット設定

department 属性が **Engineering** に設定されている全エントリーの **jpegPhoto** 属性および **manager** 属性を **cn=Engineering Admins,dc=example,dc=com** グループのメンバーが変更できるようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "jpegPhoto || manager")
```

```
(targetfilter = "(department=Engineering)") (version 3.0;
acl "Allow engineering admins updating jpegPhoto and manager of department members";
allow (write) (groupdn = "ldap:///cn=Engineering Admins,dc=example,dc=com");)
```

12.8.4. 単一ディレクトリーエントリーのターゲット設定

単一ディレクトリーエントリーを対象にするには、**targetattr** および **targetfilter** キーワードを組み合わせます。

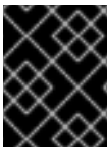
例12.10 単一ディレクトリーエントリーのターゲット設定

uid=user,ou=People,dc=example,dc=com ユーザーが u=Engineering,dc=example,dc=com エントリーで ou および cn 属性を読み取り、検索できるようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=Engineering,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "ou || cn")
 (targetfilter = "(ou=Engineering)") (version 3.0;
 acl "Allow uid=user to search and read engineering attributes";
 allow (read, search) (userdn = "ldap:///uid=user,ou=People,dc=example,dc=com");)
```

以前の例が **ou=Engineering,dc=example,dc=com** エントリーのみを対象にできるようにするには、**ou=Engineering,dc=example,dc=com** のサブエントリーは、**ou** 属性を **Engineering** に設定しないでください。



重要

ディレクトリーの構造が変更すると、これらの種類の ACI が失敗する可能性があります。

または、ターゲットエントリーに保存される属性値を使用して、バインド要求のユーザー入力に一致するバインドルールを作成できます。値の一致に基づくアクセスの定義 を参照してください。

12.9. ACI パーミッションの定義

パーミッションルールは、アクセス制御手順 (ACI) に関連付けられた権限と、アクセスを許可または拒否されるかどうかを定義します。

ACI では、以下の強調表示された部分はパーミッションルールになります。

(**target_rule**) (version 3.0; acl "**ACL_name**"; **permission_rule bind_rules**;)

12.9.1. パーミッションルールの構文

パーミッションルールの一般的な構文は、以下のとおりです。

permission (**rights**)

- **permission**: アクセス制御手順 (ACI) がパーミッションを許可するか、拒否するかを設定します。
- **rights**: ACI が許可または拒否する権限を設定します。User rights in permission rules を参照してください。

例12.11 パーミッションの定義

ou=People,dc=example,dc=com エントリーに保存されているユーザーが、独自のエントリー内の全属性を検索し、表示するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (target = "ldap:///ou=People,dc=example,dc=com") (version 3.0;
 acl "Allow users to read and search attributes of own entry"; allow (search, read)
 (userdn = "ldap:///self");)
```

12.9.2. パーミッションルールのユーザー権限

パーミッションルールの権限は、付与または拒否される操作を定義します。ACI では、以下の権限の 1 つまたは複数を設定できます。

表12.1 ユーザーの権利

権利	説明
read	ユーザーがディレクトリーデータを読み込めるかどうかを設定します。このパーミッションは、LDAP の検索操作にのみ適用されます。
write	属性を追加、変更、または削除してユーザーがエントリーを変更できるかどうかを設定します。このパーミッションは、LDAP の **modify** および **modrdn** 操作に適用されます。
add	ユーザーがエントリーを作成できるかどうかを設定します。このパーミッションは、LDAP の **add** 操作にのみ適用されます。
削除	ユーザーがエントリーを削除できるかどうかを設定します。このパーミッションは、LDAP の **delete** 操作にのみ適用されます。
search	ユーザーがディレクトリーデータを検索できるかどうかを設定します。検索結果の一部として返されたデータを表示するには、**search** および **read** 権限を付与します。このパーミッションは、LDAP の検索操作にのみ適用されます。
compare	ユーザーが提供したデータとディレクトリーに保存されているデータを比較できるかどうかを設定します。**compare** 権限では、ディレクトリーは問い合わせに対して成功または失敗のメッセージを返しますが、ユーザーはエントリーや属性の値を見ることはできません。このパーミッションは、LDAP の比較操作にのみ適用されます。

権利	説明
selfwrite	ユーザーがグループから独自の識別名 (DN) を追加または削除できるかどうかを設定します。この権限は、グループ管理にのみ使用されます。
proxy	指定した DN が他のエントリーの権限でターゲットにアクセスできるかどうかを設定します。**proxy** 権限は ACL の範囲内で付与され、その権限が付与されたユーザーやグループは、Directory Server のユーザーとしてコマンドを実行することができます。**proxy** 権限を特定のユーザーに制限することはできません。セキュリティー上の理由から、proxy 権限を使用する ACI は、ディレクトリーの最も対象となるレベルに設定してください。
all	**proxy** 以外のすべての権限を設定します。

12.9.3. LDAP 操作に必要な権限

> This section describes the rights you must grant to users depending on the type of LDAP operation you want to authorize them to perform.

- エントリーの追加:
  - 追加するエントリーの **add** パーミッションを付与します。
  - エントリーの各属性の値に **write** パーミッションを付与します。この権限はデフォルトで付与されますが、**targattrfilters** キーワードを使用して制限できます。
- エントリーの削除:
  - 削除するエントリーの **delete** パーミッションを付与します。
  - エントリーの各属性の値に **write** パーミッションを付与します。この権限はデフォルトで付与されますが、**targattrfilters** キーワードを使用して制限できます。
- エントリーの属性の変更:
  - 属性タイプで **write** パーミッションを付与します。
  - 各属性種別の値の **write** 権限を付与します。この権限はデフォルトで付与されますが、**targattrfilters** キーワードを使用して制限できます。
- エントリーの RDN の変更:
  - エントリーで **write** パーミッションを付与します。
  - 新しい RDN で使用される属性タイプの **write** パーミッションを付与します。
  - 古い RDN の削除に適した権限を付与する場合は、古い RDN で使用される属性タイプの **write** パーミッションを付与します。
  - 新しい RDN で使用される属性型の値に対して **write** 権限を付与します。この権限はデフォルトで付与されますが、**targattrfilters** キーワードを使用して制限できます。
- 属性の値を比較します。

- 属性タイプで **compare** パーミッションを付与します。

- エントリーの検索:
  - 検索フィルターで使用される各属性タイプの **search** パーミッションを付与します。
  - エントリーで使用される属性タイプの **read** パーミッションを付与します。

12.10. ACI バインドルールの定義

アクセス制御手順 (ACI) のバインドルールは、Directory Server が ACI を適用するのに必要なバインドパラメーターを定義します。たとえば、以下に基づいてバインドルールを設定できます。

- DNS
- グループメンバーシップまたは割り当てられたロール
- エントリーがバインドする場所
- バインド時に使用する必要のある認証の種類
- バインドが実行される回数または日数

ACI では、以下の強調表示された部分はバインドルールになります。

> (**target_rule**) (version 3.0; acl "**ACL_name**"; **permission_rule bind_rules**;)

12.10.1. バインドルールの構文

バインドルールの一般的な構文は以下のとおりです。

> **keyword comparison_operator "expression"**

- **keyword**: bind 操作のタイプを設定します。
- **comparison_operator**: 有効な値は **=** および **!=** で、ターゲットが式で指定されたオブジェクトであるかを示します。キーワードが追加の比較演算子に対応している場合は、該当するセクションで説明されます。
- **expression**: 式を設定し、引用符で囲む必要があります。式自体は使用するキーワードによって異なります。

12.10.2. ユーザーベースのアクセスの定義

userdn キーワードを使用すると、1 つまたは複数の DN に基づいてアクセスを許可または拒否でき、以下の構文を使用します。

> userdn **comparison_operator "ldap:///distinguished_name || ldap:///distinguished_name || ..."**

式の DN を以下のように設定します。

- DN: userdn キーワードでの DN の使用 を参照してください。
- LDAP フィルター: Using the userdn keyword with an LDAP filter を参照してください。

- **anyone** alias: Granting anonymous access を参照してください。
- **all** エイリアス: 認証済みユーザーへのアクセスの付与 を参照してください。
- **self** エイリアス: Enabling users to access their own entries を参照してください。
- **parent** エイリアス: Setting access for child entries of a user を参照してください。

注記

LDAP URL 内でホスト名またはポート番号を指定しないでください。URL は常にローカルサーバーに適用されます。

userdn キーワードでの DN の使用

userdn キーワードを識別名 (DN) に設定して、ACI を一致するエントリーのみに適用します。複数のエントリーを照合するには、DN で * ワイルドカードを使用します。

userdn キーワードを DN とともに使用するには、以下の構文を使用します。

```
userdn comparison_operator ldap:///distinguished_name
```

例12.12 userdn キーワードでの DN の使用

uid=admin,ou=People,dc=example,dc=com ユーザーが **ou=People,dc=example,dc=com** エントリーで他のすべてのユーザーの **manager** 属性を読み取るようにするには、以下を実行します。

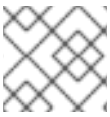
```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="manager") (version 3.0; acl "Allow uid=admin reading manager attribute";
 allow (search, read) userdn = "ldap:///uid=admin,ou=People,dc=example,dc=com";)
```

LDAP フィルターで userdn キーワードの使用

ユーザーへのパーミッションを動的に許可または拒否するには、LDAP フィルターで **userdn** キーワードを使用します。

```
userdn comparison_operator "ldap:///distinguished_name??scope?(filter)"
```



注記

LDAP フィルターは * ワイルドカードをサポートします。

例12.13 LDAP フィルターで userdn キーワードの使用

department 属性が **Human Resources** に設定されたユーザーを有効にするには、**ou=People,dc=example,dc=com** エントリーでユーザーの **homePostalAddress** 属性を更新します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="homePostalAddress") (version 3.0;
 acl "Allow HR setting homePostalAddress"; allow (write)
 userdn = "ldap:///ou=People,dc=example,dc=com??sub?(department=Human Resources)";)
```

匿名アクセスの付与

特定の状況では、管理者はディレクトリー内のデータへの匿名アクセスを設定します。匿名アクセスは、以下を指定してディレクトリーにバインドできることを意味します。

- バインド DN およびパスワードなし
- 有効なバインド DN およびパスワード

匿名アクセスを設定するには、bind ルールの **userdn** キーワードで **ldap:///anyone** 式を使用します。

```
userdn comparison_operator "ldap:///anyone"
```

例12.14 匿名アクセスの付与

認証なしですべてのユーザーが **ou=People,dc=example,dc=com** エントリーで **sn**、**givenName**、および **telephoneNumber** 属性を読み取りおよび検索できるようにするには、以下を行います。

```
# ldapmodify -D "cn=Directory Manager" -W -H __ldap://server.example.com -x`
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="sn" || targetattr="givenName" || targetattr = "telephoneNumber")
 (version 3.0; acl "Anonymous read, search for names and phone numbers";
 allow (read, search) userdn = "ldap:///anyone")
```

認証済みユーザーへのアクセスの付与

特定の状況では、管理者は匿名バインドを除き、Directory Server に正常にバインドできるユーザーにパーミッションを付与します。この機能を設定するには、bind ルールの **userdn** キーワードで **ldap:///all** 式を使用します。

```
userdn comparison_operator "ldap:///all"
```

例12.15 認証済みユーザーへのアクセスの付与

認証されたユーザーが自分自身をメンバーとして **ou=example,ou=groups,dc=example,dc=com** グループに追加およびグループから削除できるようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=example,ou=Groups,dc=example,dc=com
```

```
changetype: modify
add: aci
aci: (targetattr="member") (version 3.0;
 acl "Allow users to add/remove themselves from example group";
 allow (selfwrite) userdn = "ldap:///all")
```

ユーザーが空のエントリーにアクセスできるようにする

ユーザーの独自のエントリーへのアクセスを許可または拒否する ACI を設定するには、bind ルールの **userdn** キーワードで **ldap:///self** 式を使用します。

```
userdn comparison_operator "ldap:///self"
```

例12.16 ユーザーが空のエントリーにアクセスできるようにする

ou=People,dc=example,dc=com エントリーのユーザーが独自の **userPassword** 属性を更新できるようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="userPassword") (version 3.0;
 acl "Allow users updating their password";
 allow (write) userdn = "ldap:///self")
```

ユーザーの子エントリーへのアクセス設定

バインド DN がターゲットエントリーの親である場合にのみエントリーへのアクセスを許可または拒否されるように設定するには、bind ルールの **userdn** キーワードで **self:///parent** 式を使用します。

```
userdn comparison_operator "ldap:///parent"
```

例12.17 ユーザーの子エントリーへのアクセス設定

cn=user,ou=People,dc=example,dc=com ユーザーが独自のサブエントリー (**cn=example,cn=user,ou=People,dc=example,dc=com** など) の **manager** 属性を更新できるようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x
dn: cn=user,ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="manager") (version 3.0;
 acl "Allow cn=user to update manager attributes";
 allow (write) userdn = "ldap:///parent")
```

12.10.3. グループベースのアクセスの定義

グループベースのアクセス制御手順 (ACI) を使用すると、グループへのユーザーの追加、またはグループからのユーザーの削除により、アクセスを管理できます。グループメンバーシップに基づく ACI を設定するには、**groupdn** キーワードを使用します。ユーザーが指定された 1 つまたは複数のグループのメンバーである場合は、ACI が一致します。

groupdn キーワードを使用すると、Directory Server は以下の属性に基づいてグループメンバーシップを検証します。

- member
- uniqueMember
- memberURL
- memberCertificateDescription

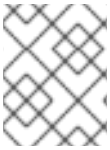
groupdn キーワードでルールをバインドするには、以下の構文を使用します。

```
groupdn comparison_operator "ldap:///distinguished_name || ldap:///distinguished_name || ..."
```

式の識別名 (DN) を次のように設定します。

- DN。groupdn キーワードでの DN の使用 を参照してください。
- LDAP フィルター。LDAP フィルターで groupdn キーワードの使用 を参照してください。

1 つのバインドルールに複数の DN を設定する場合は、認証されたユーザーがこれらのグループのいずれかのメンバーの場合、Directory Server は ACI を適用します。ユーザーを複数のグループのメンバーとして設定するには、複数の **groupdn** キーワードを使用して、ブール値 **and** 演算子を使用して組み合わせます。詳細は、Combining Bind Rules Using Boolean Operators を参照してください。



注記

LDAP URL 内でホスト名またはポート番号を指定しないでください。URL は常にローカルサーバーに適用されます。

groupdn キーワードでの DN の使用

ACI をグループのメンバーに適用するには、**groupdn** キーワードをグループの DN に設定します。

DN に設定された **groupdn** キーワードは、以下の構文を使用します。

```
groupdn comparison_operator ldap:///distinguished_name
```

例12.18 groupdn キーワードでの DN の使用

cn=example,ou=Groups,dc=example,dc=com グループのメンバーが **ou=People,dc=example,dc=com** のエントリーの manager 属性を検索および読み取るようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
```

```
aci: (targetattr="manager") (version 3.0;
 acl "Allow example group to read manager attribute";
 allow (search, read) groupdn = "ldap:///cn=example,ou=Groups,dc=example,dc=com";)
```

LDAP フィルターで groupdn キーワードの使用

groupdn キーワードを使用した LDAP フィルターを使用すると、ACI に一致させるために、認証されたユーザーがフィルター検索で返されるグループの少なくとも 1 つのメンバーでなければならないことを定義できます。

LDAP フィルターが含まれる **groupdn** キーワードは以下の構文を使用します。

```
groupdn comparison_operator "ldap:///distinguished_name??scope?(filter)"
```

注記

LDAP フィルターは * ワイルドカードをサポートします。

例12.19 LDAP フィルターで groupdn キーワードの使用

dc=example,dc=com のグループのメンバーや、**manager** 属性が **example** に設定されているサブツリーを有効にするには、**ou=People,dc=example,dc=com** のエントリーの **homePostalAddress** を更新します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="homePostalAddress") (version 3.0;
 acl "Allow manager=example setting homePostalAddress"; allow (write)
 userdn = "ldap:///dc=example,dc=com??sub?(manager=example)";)
```

12.10.4. 値の一致に基づくアクセスの定義

バインドルールの **userattr** キーワードを使用して、ディレクトリーとターゲットエントリーにバインドするのに使用されるエントリー間でどの属性が一致するかを指定します。

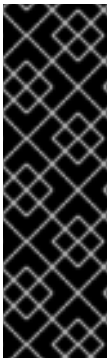
userattr キーワードは、以下の構文を使用します。

```
userattr comparison_operator "attribute_name#bind_type_or_attribute_value
```

詳細は、以下を参照してください。

- USERDN バインドタイプの使用
- GROUPDN バインドタイプの使用
- ROLEDN バインドタイプの使用
- SELFDN バインドタイプの使用

- LDAPURL バインドタイプの使用
- 継承による userattr キーワードの使用



重要

デフォルトでは、Directory Server は、作成したエントリーに対するアクセス権限を評価します。ただし、同じレベルのユーザーオブジェクトを防ぐために、Directory Server は、**userattr** キーワードを使用した場合に、アクセス制御手順 (ACI) を設定したエントリーに **add** パーミッションを付与しません。この動作を設定するには、**parent** キーワードとともに **userattr** キーワードを使用して、レベル 0 にもパーミッションを付与します。

継承の詳細は、Defining access based on value matching を参照してください。

USERDN バインドタイプの使用

バインディングユーザーの識別名 (DN) が属性に保存されている DN と一致する場合に ACI を適用するには、**USERDN** バインドタイプを使用します。

USERDN バインドタイプの **userattr** キーワードには、以下の構文を使用します。

```
userattr comparison_operator "attribute_name#USERDN"
```

例12.20 USERDN バインドタイプの使用

マネージャーに対し、すべての権限を独自の関連付けの **telephoneNumber** 属性に付与するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "telephoneNumber")
 (version 3.0; acl "Manager: telephoneNumber";
 allow (all) userattr = "manager#USERDN";)
```

前述の ACI は、**ou=People,dc=example,dc=com** のエントリーに対して操作を行ったユーザーの DN が、このエントリーの **manager** 属性に格納されている DN と一致すれば、真と評価されます。

GROUPDN バインドタイプの使用

バインディングユーザー DN が属性に設定されたグループのメンバーである場合に ACI を適用するには、**GROUPDN** バインドタイプを使用します。

GROUPDN バインドタイプの **userattr** キーワードには、以下の構文を使用します。

```
userattr comparison_operator "attribute_name#GROUPDN"
```

例12.21 GROUPDN バインドタイプの使用

ユーザーに、**ou=Social Committee,ou=Groups,dc=example,dc=com** エントリーを所有するグループエントリーを削除する権限を付与するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=Social Committee,ou=Groups,dc=example,dc=com
changetype: modify
add: aci
aci: (target="ou=Social Committee,ou=Groups,dc=example,dc=com)
 (targattrfilters="del=objectClass:(objectClass=groupOfNames)")
 (version 3.0; acl "Delete Group";
 allow (delete) userattr = "owner#GROUPDN";)
```

操作を実行するユーザーの DN が **owner** 属性で指定されたグループのメンバーである場合に、以前の ACI が true になります。

指定のグループは動的グループで、グループの DN はデータベースの任意の接尾辞にすることができます。しかし、このタイプの ACI をサーバーが評価するには、リソースを大量に必要とします。

ターゲットエントリーと同じ接尾辞の下にある静的グループを使用している場合は、パフォーマンスを改善するために以下の式を使用します。

```
userattr comparison_operator "ldap:///distinguished_name?attribute_name#GROUPDN"
```

ROLEDN バインドタイプの使用

バインディングユーザーが属性で指定されたロールに属する場合に ACI を適用するには、**ROLEDN** バインドタイプを使用します。

ROLEDN バインドタイプの **userattr** キーワードには、以下の構文を使用します。

```
userattr comparison_operator "attribute_name#ROLEDN"
```

例12.22 ROLEDN バインドタイプの使用

cn=Administrators,dc=example,dc=com ロールを持つユーザーが **ou=People,dc=example,dc=com** のエントリーの **manager** 属性を検索および読み取るようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (version 3.0; acl "Allow example role owners to read manager attribute";
 allow (search, read) userattr = manager#ROLEDN;)
```

指定のロールはデータベースの任意の接尾辞の下に置くことができます。フィルターされたロールも使用している場合、このタイプの ACI の評価は、サーバー上の多くのリソースを使用します。

静的ロール定義を使用し、ロールエントリーがターゲットエントリーと同じ接尾辞下にある場合は、パフォーマンスを向上させるために以下の式を使用します。

SELFDN バインドタイプの使用

SELFDN バインドタイプを使用すると、バインドされたユーザーの DN がエントリーの単一値属性に設定されている場合にパーミッションを付与できます。

SELFDN バインドタイプの **userattr** キーワードには、以下の構文を使用します。

```
userattr comparison_operator "attribute_name#SELFDN"
```

例12.23 SELFDN バインドタイプの使用

ユーザーが **ipatokenOwner** 属性にバインドユーザーの DN が設定された **ipatokenuniqueid=*,cn=otp,dc=example,dc=com** エントリーを追加できるようにするには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=otp,dc=example,dc=com
changetype: modify
add: aci
aci: (target = "ldap:///ipatokenuniqueid=*,cn=otp,dc=example,dc=com")
 (targetfilter = "(objectClass=ipaToken)")(version 3.0;
 acl "token-add-delete"; allow (add) userattr = "ipatokenOwner#SELFDN";)
```

LDAPURL バインドタイプの使用

バインド DN がターゲットエントリーの属性で指定されたフィルターと一致する場合に ACL を適用するには、**LDAPURL** バインドタイプを使用します。

LDAPURL バインドタイプの **userattr** キーワードには、以下の構文を使用します。

```
userattr comparison_operator "attribute_name#LDAPURL"
```

例12.24 LDAPURL バインドタイプの使用

ldap:///ou=People,dc=example,dc=com??one?(uid=user*) に設定した **aciurl** 属性が含まれるユーザーオブジェクトに読み取りパーミッションおよび検索パーミッションを付与するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*")
 (version 3.0; acl "Allow read,search "; allow (read,search)
 (userattr = "aciurl#LDAPURL);)
```

継承による userattr キーワードの使用

userattr キーワードを使用してターゲットエントリーにバインドするために使用されるエントリーを関連付ける場合、ACI は指定されたターゲットにのみ適用され、その下のエントリーには適用されません。特定の状況下では、管理者は ACI の適用範囲を、対象となるエントリーよりも数レベル広げたいと考えます。これは、**parent** キーワードを使用して、ACI を継承するターゲットよりも低いレベルの数を指定できます。

parent キーワードで **userattr** キーワードを使用する場合、構文は以下のようになります。

```
userattr comparison_operator
"parent[inheritance_level].attribute_name#bind_type_or_attribute_value
```

- **inheritance_level**: ターゲットが ACI を継承するレベルの数を指定します。ターゲットエントリーの下に、5 つのレベル (**0**、**1**、**2**、**3**、**4**) を追加できます。ゼロ (**0**) はターゲットエントリーを示します。
- **attribute_name**: **userattr** または **groupattr** のキーワードでターゲットとする属性。
- **bind_type_or_attribute_value**: **USERDN** などの属性値またはバインドタイプを設定します。

以下に例を示します。

```
userattr = "parent[0,1].manager#USERDN"
```

このバインドルールは、バインド DN がターゲットエントリーのマネージャー属性と一致する場合に true になります。バインドルールが true であるときに付与されるパーミッションは、ターゲットエントリーと、その下のすべてのエントリーに適用されます。

例12.25 継承による userattr キーワードの使用

ユーザーの DN が **owner** 属性に設定されている **cn=Profiles,dc=example,dc=com** エントリー、および **cn=mail,cn=Profiles,dc=example,dc=com** および **cn=news,cn=Profiles,dc=example,dc=com** を含む第 1 レベルの子エントリーの読み取りと検索を可能にするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x`

dn: cn=Profiles,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="*") (version 3.0; acl "Profile access",
 allow (read,search) userattr="parent[0,1].owner#USERDN" ;)
```

12.10.5. 特定の IP アドレスまたは範囲からのアクセスの定義

バインドルールの **ip** キーワードを使用すると、特定の IP アドレスまたは IP アドレスの範囲からのアクセスを許可または拒否できます。

ip キーワードでルールをバインドするには、以下の構文を使用します。

```
ip comparison_operator "IP_address_or_range"
```

例12.26 バインドルールでの IPv4 アドレス範囲の使用

192.0.2.0/24 ネットワークから **dc=example,dc=com** エントリーへのアクセスを拒否するには、以下のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x
```

```
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*") (version 3.0;acl "Deny 192.0.2.0/24"; deny (all)
 (userdn = "ldap:///anyone") and (ip != "192.0.2.");)
```

例12.27 バインドルールでの IPv6 アドレス範囲の使用

2001:db8::/64 ネットワークから **dc=example,dc=com** エントリーへのアクセスを拒否するには、以下のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*") (version 3.0;acl "Deny 2001:db8::/64"; deny (all)
 (userdn = "ldap:///anyone") and (ip != "2001:db8::");)
```

12.10.6. 特定のホストまたはドメインからアクセスの定義

バインドルールの **dns** キーワードを使用すると、特定のホストまたはドメインからのアクセスを許可または拒否できます。

警告

DNS を使用して Directory Server が完全修飾ドメイン名 (FQDN) への接続 IP アドレスを解決できない場合、サーバーはこのクライアントの **dns** バインディングルールを持つアクセス制御手順 (ACI) を適用しません。

クライアント IP アドレスが DNS を使用して解決できない場合は、代わりに **ip** キーワードおよび IP アドレスを使用してください。特定の IP アドレスまたは範囲からのアクセスの定義 を参照してください。

dns キーワードでルールをバインドするには、以下の構文を使用します。

```
dns comparison_operator "host_name_or_domain_name"
```

例12.28 特定のホストからのアクセスの定義

client.example.com ホストから dc=example,dc=com エントリーへのアクセスを拒否するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: dc=example,dc=com
```

```
changetype: modify
add: aci
aci: (targetattr = "*") (version 3.0;acl "Deny client.example.com"; deny (all)
 (userdn = "ldap:///anyone") and (dns != "client.example.com");)
```

例12.29 特定のドメインからアクセスの定義

example.com ドメイン内のすべてのホストから dc=example,dc=com エントリーへのアクセスを拒否するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "") (version 3.0;acl "Deny example.com"; deny (all) (userdn =
"ldap:///anyone") and (dns != ".example.com");)
```

12.10.7. 接続に一定レベルのセキュリティーの要求

接続のセキュリティーは、操作を処理するために必要な最低限の鍵の強度を設定する SSF (Security Strength Factor) によって決定されます。バインドルールで **ssf** キーワードを使用すると、接続が一定レベルのセキュリティーを使用する必要があります。これにより、パスワード変更などの操作を強制的に、暗号化された接続上で実行できます。

すべての操作の SSF 値は、TLS 接続と SASL バインドの間の値が高くなります。これは、サーバーが TLS で実行されるように設定され、レプリカ合意が SASL/GSSAPI に対して設定されている場合は、操作の SSF が利用可能な暗号化タイプがよりセキュアであることを意味します。

ssf キーワードでルールをバインドするには、以下の構文を使用します。

```
ssf comparison_operator key_strength
```

以下の比較演算子を使用できます。

- **=** (等しい)
- **!** (等しくない)
- **<** (より小さい)
- **>** (より大きい)
- **⇐** (より小さいか等しい)
- **>=** (より大きいか等しい)

key_strength パラメーターが **0** に設定されている場合、LDAP 操作にセキュアな操作は必要ありません。

例12.30 接続に一定レベルのセキュリティーの要求

dc=example,dc=com エントリーのユーザーが、SSF が 128 以上の場合にのみ、userPassword 属性を更新できるように設定する場合は、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "userPassword") (version 3.0;
 acl "Allow users updating own userPassword";
 allow (write) (userdn = "ldap:///self") and (ssf >= "128");)
```

12.10.8. 曜日の特定の日におけるアクセスの定義

バインドルールの **dayofweek** キーワードを使用すると、曜日に基づいてアクセスを許可または拒否できます。

注記

Directory Server はサーバー上で時間を使用してアクセス制御手順 (ACI) を評価しますが、クライアントの時間ではありません。

dayofweek キーワードでルールをバインドするには、以下の構文を使用します。

```
dayofweek comparison_operator "comma-separated_list_of_days"
```

例12.31 特定の曜日にアクセスの付与

毎週土曜日と日曜日のサーバーにバインドするために **uid=user,ou=People,dc=example,dc=com** ユーザーエントリーのアクセスを拒否するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (version 3.0; acl "Deny access on Saturdays and Sundays";
 deny (all)
 (userdn = "ldap:///uid=user,ou=People,dc=example,dc=com") and
 (dayofweek = "Sun,Sat");)
```

12.10.9. 特定の時刻におけるアクセスの定義

バインドルールで **timeofday** キーワードを使用すると、時間帯に基づいてアクセスを許可または拒否することができます。

注記

Directory Server はサーバー上で時間を使用してアクセス制御手順 (ACI) を評価しますが、クライアントの時間ではありません。

timeofday キーワードでルールをバインドするには、以下の構文を使用します。

```
timeofday comparison_operator "time"
```

以下の比較演算子を使用できます。

- **=** (等しい)
- **!** (等しくない)
- **<** (より小さい)
- **>** (より大きい)
- **⇐** (より小さいか等しい)
- **>=** (より大きいか等しい)

重要

timeofday キーワードには、24 時間形式で時間を指定する必要があります。

例12.32 特定の時刻におけるアクセスの定義

uid=user,ou=People,dc=example,dc=com ユーザーエントリーへのアクセスを拒否するには、6pm から 0am までのサーバーにバインドします。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
 aci: (version 3.0; acl "Deny access between 6pm and 0am";
 deny (all)
 (userdn = "ldap:///uid=user,ou=People,dc=example,dc=com") and
 (timeofday >= "1800" and timeofday < "2400");)
```

12.10.10. 認証方法に基づいたアクセスの定義

bind ルールの **authmethod** キーワードは、サーバーに接続する際にクライアントが使用する認証方法を設定し、アクセス制御手順 (ACI) を適用します。

authmethod キーワードでルールをバインドするには、以下の構文を使用します。

```
authmethod comparison_operator "authentication_method"
```

以下の認証方法を設定できます。

- **none**: 認証は不要で、匿名のアクセスを表します。これはデフォルトになります。
- **simple**: クライアントは、ディレクトリーにバインドするユーザー名とパスワードを提供する必要があります。

- **SSL**: クライアントは、データベース、スマートカード、または他のデバイスのいずれかで TLS 証明書を使用してディレクトリーにバインドする必要があります。証明書ベースの認証の詳細は、認証方法に基づいてアクセスの定義 を参照してください。
- **SASL**: クライアントは、Simple Authentication and Security Layer (SASL) 接続を介してディレクトリーにバインドする必要があります。bind ルールでこの認証方法を使用する場合は、**EXTERNAL** などの SASL メカニズムも指定します。

例12.33 EXTERNAL SASL 認証方法を使用した接続でのみアクセスのみの有効化

接続が証明書ベースの認証メソッドまたは SASL を使用していない場合にサーバーへのアクセスを拒否するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x`

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (version 3.0; acl "Deny all access without certificate"; deny (all)
 (authmethod = "none" or authmethod = "simple");)
```

12.10.11. ロールに基づくアクセスの定義

bind ルールの **roledn** キーワードを使用すると、1 つまたは複数のロールが設定されたユーザーへのアクセスを許可または拒否できます。

注記

Red Hat は、ロールの代わりにグループを使用することを推奨します。

roledn キーワードでルールをバインドするには、以下の構文を使用します。

```
roledn comparison_operator "ldap:///distinguished_name || ldap:///distinguished_name || ..."
```

識別名 (DN) にコンマが含まれている場合は、バックスラッシュでエスケープしてください。

例12.34 ロールに基づくアクセスの定義

nsRole 属性で **cn=Human Resources,ou=People,dc=example,dc=com** ロールを設定したユーザーが **ou=People,dc=example,dc=com** のエントリーの **manager** 属性を検索および読み取るようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="manager") (version 3.0;
 acl "Allow manager role to update manager attribute";
 allow (search, read) roledn = "ldap:///cn=Human Resources,ou=People,dc=example,dc=com";)
```

12.10.12. ブール演算子を使用したバインドルールの組み合わせ

複雑なバインドルールを作成する場合は、**AND**、**OR**、および **NOT** のブール値演算子を使用すると、複数のキーワードを組み合わせることができます。

バインドルールとブール演算子を組み合わせた構文は以下の通りです。

```
bind_rule_1 boolean_operator bind_rule_2...
```

例12.35 ブール演算子を使用したバインドルールの組み合わせ

cn=Administrators,ou=Groups,dc=example,com および **cn=Operators,ou=Groups,dc=example,com] group can [command]`read** の両方のグループのメンバーであるユーザーが、**ou=People,dc=example,dc=com** のエントリーを、**search**、**add**、**update**、および **delete** できるように設定するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///ou=People,dc=example,dc=com") (version 3.0;
 acl "Allow members of administrators and operators group to manage users";
 allow (read, search, add, write, delete)
 groupdn = "ldap:///cn=Administrators,ou=Groups,dc=example,com" AND
 groupdn = "ldap:///cn=Operators,ou=Groups,dc=example,com";)
```

Directory Server によるブール値演算子の評価方法

Directory Server は以下のルールを使用してブール値演算子を評価します。

- 左から右へのすべての式。
  以下の例では、**bind_rule_1** が最初に評価されます。

  ```
  (bind_rule_1) OR (bind_rule_2)
  ```

- 一番内側から外側に向かって、親表現が優先されます。
  以下の例では、**bind_rule_2** を最初に評価し、次に **bind_rule_3** を評価します。

  ```
  (bind_rule_1) OR ((bind_rule_2) AND (bind_rule_3))
  ```

- **AND** または **OR** 演算子の前に **NOT**。
  以下の例では、bind_rule_2 が最初に評価されます。

  ```
  (bind_rule_1) AND NOT (bind_rule_2)
  ```

  AND および **OR** 演算子には優先順位がありません。

第13章 DIRECTORY SERVER を FIPS モードで実行する

Directory Server は、連邦情報処理標準 (FIPS) 140-2 を完全にサポートします。Directory Server を FIPS モードで実行すると、セキュリティー関連の設定が変更されます。たとえば、SSL は自動的に無効になり、TLS 1.2 および 1.3 暗号化のみが使用されます。

13.1. FIPS モードの有効化

Directory Server を Federal Information Processing Standard (FIPS) モードで使用するには、RHEL および Directory Server でモードを有効にします。

前提条件

- RHEL で FIPS モードを有効にしました。

手順

1. ネットワークセキュリティーサービス (NSS) データベースの FIPS モードを有効にします。

```
# modutil -dbdir /etc/dirsrv/slapd-instance_name/ -fips true
```

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

検証

- NSS データベースで FIPS モードが有効になっていることを確認します。

```
# modutil -dbdir /etc/dirsrv/slapd-instance_name/ -chkfips true
FIPS mode enabled.
```

モジュールが FIPS モードの場合、このコマンドは **FIPS mode enabled** を返します。

13.2. 関連情報

- FIPS (Federal Information Processing Standard)
- FIPS モードへのシステムの切り替え

第14章 パスワードベースのアカウントロックアウトポリシーの設定

パスワードベースのアカウントのロックアウトポリシーにより、攻撃者はユーザーのパスワードを繰り返し推測できなくなります。アカウントロックアウトポリシーを設定して、指定した数のバインドの試行後にユーザーアカウントをロックできます。

パスワードベースのアカウントのロックアウトポリシーが設定されている場合、Directory Server はユーザーエントリーの以下の属性でロックアウト情報を維持します。

- **passwordRetryCount**: 失敗したバインドの試行回数を格納します。Directory Server は、ユーザーが **retryCountResetTime** の時間よりも後にディレクトリーに正常にバインドされると、値をリセットします。この属性は、ユーザーが初めてバインドに失敗すると表示されます。
- **retryCountResetTime**: **passwordRetryCount** 属性がリセットされるまでの時間を保存します。この属性は、ユーザーが初めてバインドに失敗すると表示されます。
- **accountUnlockTime**: ユーザーアカウントのロックが解除されてからの時間を保存します。この属性は、アカウントの初回ロック後に存在します。

14.1. 設定された最大試行に到達するか、超過する際にアカウントをロックするかどうかの設定

管理者は、Directory Server がログイン試行の失敗時にアカウントをロックすると、以下のいずれかの動作を設定できます。

- 上限を超えた場合、サーバーがアカウントをロックします。たとえば、制限が 3 回の試行に設定されていると、4 回目の試行 (**n+1**) の後にロックアウトが実行されます。これは、4 番目の試行に成功すると、Directory Server がアカウントをロックしないことを意味します。
  デフォルトでは、Directory Server は従来の LDAP クライアントが必要とするこのレガシーパスワードポリシーを使用します。
- 制限に達すると、サーバーがアカウントをロックします。たとえば、制限が 3 回の試行に設定されていると、4 回目の試行 (**n+1**) の後にロックアウトが実行されます。
  最新の LDAP クライアントは、多くの場合、この動作を想定しています。

この手順では、レガシーパスワードポリシーを無効にする方法を説明します。ポリシーの変更後に、Directory Server は設定された制限に到達したユーザーのログイン試行をブロックします。

前提条件

- アカウントロックアウトポリシーを設定している。

手順

- 制限に達すると、レガシーパスワードポリシーとロックアカウントを無効にするには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
passwordLegacyPolicy=off
```

検証

1. **passwordmaxfailure** 設定の値を表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy get
passwordmaxfailure
passwordmaxfailure: 2
```

2. **passwordmaxfailure** に設定された値よりも、無効なパスワードを使用したバインドを試みます。

```
# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Invalid credentials (49)

# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Invalid credentials (49)

# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Constraint violation (19)
 additional info: Exceed password retry limit. Please try later.
```

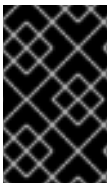
レガシーパスワードが無効になっていると、Directory Server は 2 回目の試行後にアカウントをロックし、さらに **ldap_bind: Constraint violation(19)** エラーでブロックされます。

関連情報

- コマンドラインでパスワードベースのアカウントロックアウトポリシーの設定

14.2. コマンドラインでパスワードベースのアカウントロックアウトポリシーの設定

無効なパスワードでログインの繰り返しバインド試行をブロックするには、パスワードベースのアカウントのロックアウトポリシーを設定します。



重要

設定された最大試行に到達するか、超過した場合に Directory Server がアカウントをロックするかどうかの動作は、レガシーパスワードポリシーの設定によって異なります。

手順

1. オプション: レガシーパスワードポリシーを有効または無効にするかどうかを特定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config get
passwordLegacyPolicy
passwordLegacyPolicy: on
```

2. パスワードのロックアウトポリシーを有効にし、失敗の最大数を **2** に設定します。

```
# [command]`dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy
set --pwdlockout on --pwdmaxfailures=2
```

レガシーパスワードポリシーを有効にすると、3 番目のバインド試行に失敗した後に、Directory Server はアカウントをロックします (**--pwdmaxfailures** パラメーターの値 + 1)。

dsconf pwpolicy set コマンドは以下のパラメーターをサポートします。

- **--pwdlockout**: アカウントロックアウト機能を有効または無効にします。デフォルト: **off**
- **--pwdmaxfailures**: Directory Server がアカウントをロックするまでに許可されるバインド試行の最大数を設定します。デフォルト: **3**
  従来のパスワードポリシー設定が有効な場合は、このロックアウトが後で試行されることに注意してください。デフォルト: **3**
- **--pwdresetfailcount**: Directory Server がユーザーのエントリーの **passwordRetryCount** 属性をリセットするまでの時間を秒単位で設定します。デフォルト: **600** 秒 (10 分)
- **--pwdlockoutduration**: アカウントがロックされる時間を秒単位で設定します。**--pwdunlock** パラメーターを **off** に設定すると、このパラメーターは無視されます。デフォルト: **3600** 秒 (1 時間)
- **--pwdunlock**: 特定の時間が経過するとロックされたアカウントをアンロックするか、管理者が手動でアンロックするまで、無効になっているかを有効または無効にします。デフォルト: **on**

検証

- **--pwdmaxfailures** パラメーターに設定した値よりも 2 回無効なパスワードを使用したバインドを試みます。

```
# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Invalid credentials (49)

# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Invalid credentials (49)

# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Invalid credentials (49)

# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Constraint violation (19)
        additional info: Exceed password retry limit. Please try later.
```

レガシーパスワードを有効にすると、Directory Server は制限を超えた後にアカウントをロックし、さらに **ldap_bind: Constraint violation(19)** エラーでブロックされます。

関連情報

関連情報

- レガシーパスワードポリシーの設定

14.3. WEB コンソールでパスワードベースのアカウントロックアウトポリシーの設定

無効なパスワードでログインの繰り返しバインド試行をブロックするには、パスワードベースのアカウントのロックアウトポリシーを設定します。

重要

設定された最大試行に到達するか、超過した場合に Directory Server がアカウントをロックするかどうかの動作は、レガシーパスワードポリシーの設定によって異なります。

前提条件

- Web コンソールでインスタンスにログインしている。

手順

1. オプション: レガシーパスワードポリシーを有効または無効にするかどうかを特定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config get
passwordLegacyPolicy
passwordLegacyPolicy: on
```

   この設定は、Web コンソールでは利用できません。

2. **Database → Password Policies → Global Policy → Lockout** に移動します。
3. **Enable Account Lockout** を選択します。
4. ロックアウトを設定します。
   - **Number of Failed Logins That Locks out Account**: Directory Server がアカウントをロックする前に失敗したバインド試行の最大数を設定します。
   - **Time Until Failure Count Resets**: ユーザーのエントリーの **passwordRetryCount** 属性をリセットするまでの時間を秒単位で設定します。
   - **Time Until Account Unlocked**: アカウントがロックされるまでの時間を秒単位で設定します。**Do Not Lockout Account Forever** を無効にした場合、このパラメーターは無視されます。
   - **Do Not Lockout Account Forever**: ロックされたアカウントを一定時間後にロック解除するか、管理者が手動でロックを解除するまで無効のままにするかを有効または無効にします。
5. **Save** をクリックします。

検証

- **Number of Failed Logins That Locks out Account** で設定した値よりも 2 回多く、無効なパスワードを使用してバインドを試みてください。

```
# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Invalid credentials (49)

# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Invalid credentials (49)

# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Invalid credentials (49)

# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w invalid-password -b
"dc=example,dc=com" -x
ldap_bind: Constraint violation (19)
        additional info: Exceed password retry limit. Please try later.
```

レガシーパスワードを有効にすると、Directory Server は制限を超えた後にアカウントをロックし、さらに **ldap_bind: Constraint violation(19)** エラーでブロックされます。

関連情報

- レガシーパスワードポリシーの設定

第15章 匿名バインドの無効化

ユーザーが認証情報を指定せずに Directory Server への接続を試みた場合、この操作は **anonymous bind** (匿名バインド) と呼ばれます。匿名バインドは、ユーザーに最初の認証を要求しないため、ディレクトリー内の電話番号の検索など、検索と読み取り操作が簡素化されます。ただし、アカウントのないユーザーがデータにアクセスできるため、匿名バインドはセキュリティーリスクとなる可能性もあります。

警告

デフォルトでは、匿名バインドは Directory Server で検索および読み取り操作に対して有効になります。これにより、ユーザーエントリーだけでなく、ルート Directory Server エントリー (DSE) などの設定エントリーへの不正アクセスが可能になります。

15.1. コマンドラインでの匿名バインドの無効化

セキュリティーを強化するには、匿名バインドを無効にします。

手順

- **nsslapd-allow-anonymous-access** 設定パラメーターを **off** に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace
nsslapd-allow-anonymous-access=off
```

検証

- ユーザーアカウントを指定せずに検索を実行します。

```
# ldapsearch -H ldap://server.example.com -b "dc=example,dc=com" -x
ldap_bind: Inappropriate authentication (48)
 additional info: Anonymous access is not allowed
```

15.2. WEB コンソールでの匿名バインドの無効化

セキュリティーを強化するには、匿名バインドを無効にします。

前提条件

- Web コンソールでインスタンスにログインしている。

手順

1. **Server → Server Settings → Advanced Settings** に移動します。
2. **Allow Anonymous Access** パラメーターを **off** に設定します。

3. **Save** をクリックします。

検証

- ユーザーアカウントを指定せずに検索を実行します。

```
# ldapsearch -H ldap://server.example.com -b "dc=example,dc=com" -x
ldap_bind: Inappropriate authentication (48)
        additional info: Anonymous access is not allowed
```

第16章 レプリケーション環境のすべてのサーバー間でのアカウントのロックアウト属性の同期

Directory Server は、アカウントのロックアウト属性をローカルで保存します。複数のサーバーがある環境では、攻撃者がアカウントのロックアウト数に達するまで 1 台のサーバーへのログインを試み、その後さらに他のサーバーにログインを試みることを防ぐために、この属性のレプリケーションを設定してください。

16.1. レプリケーション環境で DIRECTORY SERVER がパスワードおよびアカウントのロックアウトポリシーを処理する方法

Directory Server は、パスワードおよびアカウントのロックアウトポリシーを次のように適用します。

- パスワードポリシーは、データプロバイダーに適用されます。
- アカウントロックアウトポリシーは、レプリケーショントポロジー内のすべてのサーバーに適用されます。

Directory Server は次パスワードポリシー属性をレプリケートします。

- **passwordMinAge**
- **passwordMaxAge**
- **passwordExp**
- **passwordWarning**

ただし、デフォルトでは、Directory Server は以下に示す一般的なアカウントロックアウト属性をレプリケートしません。

- **passwordRetryCount**
- **retryCountResetTime**
- **accountUnlockTime**

攻撃者がアカウントのロックアウト数に達するまで 1 台のサーバーへのログインを試み、その後さらに他のサーバーにログインを試みることを防ぐには、これらのアカウントロックアウト属性をレプリケートします。

関連情報

- アカウントロックアウト属性をレプリケートするための Directory Server の設定

16.2. アカウントロックアウト属性をレプリケートするための DIRECTORY SERVER の設定

passwordRetryCount、**retryCountResetTime**、または **accountUnlockTime** 属性を更新するアカウントロックアウトポリシーまたはパスワードポリシーを使用する場合は、これらの属性の値がすべてのサーバーで同じになるように Directory Server を設定してこれらの属性をレプリケートします。

この手順は、レプリケーショントポロジー内のすべてのサプライヤーで実行してください。

前提条件

- 前述の属性を1つ以上更新するアカウントロックアウトポリシーまたはパスワードポリシーを設定している。
- Directory Server をレプリケーション環境で使用している。

手順

1. パスワードポリシー属性のレプリケーションを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy set --
pwdisglobal="on"
```

2. 一部レプリケーションを使用する場合は、レプリケーションから除外される属性のリストを表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt get --suffix
"dc=example,dc=com" example-agreement | grep "nsDS5ReplicatedAttributeList"
```

デフォルト設定を使用すると、出力は表示されず、Directory Server はアカウントロックアウト属性をレプリケートします。ただし、次の例のように、除外される属性のリストが返された場合は、属性リストを確認してください。

```
nsDS5ReplicatedAttributeList: (objectclass=*) $ EXCLUDE accountUnlockTime
passwordRetryCount retryCountResetTime example1 example2
```

この例では、**accountUnlockTime**、**passwordRetryCount**、および **retryCountResetTime** ロックアウトポリシー属性が、その他の 2 つの属性とともにレプリケーションから除外されます。

3. 直前のコマンドの出力にアカウントロックアウト属性が表示された場合は、ロックアウトポリシー属性以外の属性のみを含めるように一部レプリケーション設定を更新します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt set --suffix
"dc=example,dc=com" --frac-list "example1 example2" example-agreement
```

検証

1. 無効なパスワードを使用して、ユーザーとして検索の実行を試みます。

```
# ldapsearch -H ldap://server.example.com -D
"uid=example,ou=People,dc=example,dc=com" -w "invalid-password" -b
"dc=example,dc=com" -x
ldap_bind: Invalid credentials (49)
```

2. ユーザーの **passwordRetryCount** 属性を表示します。

```
# ldapsearch -H ldap://server.example.com -D "cn=Directory Manager" -W -b
"uid=example,ou=People,dc=example,dc=com" -x passwordRetryCount
...
dn: uid=example,ou=People,dc=example,dc=com
passwordRetryCount: 1
```

3. レプリケーショントポロジー内の別のサーバーで直前のコマンドを実行します。**passwordRetryCount** 属性の値が同じである場合、Directory Server は属性をレプリケートします。

関連情報

- パスワードベースのアカウントロックアウトポリシーの設定

第17章 時間ベースのアカウントロックアウトポリシーの設定

アカウントポリシープラグインを使用して、次のようなさまざまな時間ベースのロックアウトポリシーを設定できます。

- 最後に成功したログインで一定時間アカウントを自動的に無効にする
- アカウントを作成してから一定時間、アカウントを自動的に無効にする
- パスワードの有効期限が切れてから一定時間アカウントを自動的に無効にする
- アカウントのステータス (アクティブかどうか) とパスワードの有効期限の両方でアカウントを自動的に無効にする

17.1. 最後に成功したログインで一定時間アカウントを自動的に無効にする

この手順に従って、21 日を超えてログインしない **dc=example,dc=com** エントリーの下のユーザーを非アクティブ化する時間ベースのロックアウトポリシーを設定します。

このアカウント非アクティブ機能は、たとえば、従業員が会社を辞め、管理者がアカウントの削除を忘れた場合に、Directory Server が一定時間後にアカウントを非アクティブ化することを保証します。

手順

1. アカウントポリシープラグインを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy enable
```

2. プラグイン設定エントリーを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy config-entry set "cn=config,cn=Account Policy Plugin,cn=plugins,cn=config" --always-record-login yes --state-attr lastLoginTime --alt-state-attr 1.1 --spec-attr acctPolicySubentry --limit-attr accountInactivityLimit
```

コマンドは、以下のオプションを使用します。

- **--always-record-login yes**: ログイン時間のログを有効にします。これは、**acctPolicySubentry** 属性が設定されていない場合でも、サービスクラス (Co S) またはアカウントポリシーを持つロールを使用するために必要です。
- **--state-attr last LoginTime**: アカウントポリシープラグインがユーザーの **last Login Time** 属性に最終ログイン時刻を保存するように設定します。
- **--alt-state-attr 1.1**: 代替属性を使用してプライマリー属性が存在しないかどうかをチェックすることを無効にします。デフォルトでは、Directory Server は代わりに **createTimestamp** 属性を使用します。ただし、これにより、アカウントに **lastLoginTime** 属性が設定されておらず、**createTimestamp** が設定された非アクティブ期間よりも古い場合、Directory Server は既存のユーザーを自動的にログアウトします。代替属性を無効にすると、Directory Server は、ユーザーが次回ログインするときに、**lastLoginTime** 属性をユーザーエントリーに自動的に追加します。

- **--spec-attr acctPolicySubentry**: **acctPolicySubentry** 属性が設定されているエントリーにポリシーを適用するように Directory Server を設定します。この属性は、CoS エントリーで設定します。
- **--limit-attr accountInactivityLimit**: アカウント非アクティブ化ポリシーエントリーの **accountInactivityLimit** 属性が非アクティブ時間を保存するように設定します。

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

4. アカウント非アクティブ化ポリシーエントリーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=Account Inactivation Policy,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: accountpolicy
accountInactivityLimit: 1814400
cn: Account Inactivation Policy
```

accountInactivityLimit 属性の値は、Directory Server が最後のログインから **1814400** 秒 (21 日) 後にアカウントを非アクティブ化するように設定します。

5. CoS テンプレートエントリーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=TemplateCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: cosTemplate
acctPolicySubentry: cn=Account Inactivation Policy,dc=example,dc=com
```

このテンプレートエントリーは、アカウントの非アクティブ化ポリシーを参照します。

6. CoS 定義エントリーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=DefinitionCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=TemplateCoS,dc=example,dc=com
cosAttribute: acctPolicySubentry default operational-default
```

この定義エントリーは CoS テンプレートエントリーを参照し、**acctPolicySubentry** 属性が各ユーザーエントリーに表示され、値が **cn=Account Inactivation Policy,dc=example,dc=com** 設定されます。

検証

1. **lastLoginTime** 属性を、設定した非アクティブ時間よりも古い値に設定します。

```
# ldapmodify -H ldap://server.example.com -x -D "cn=Directory Manager" -W

dn: uid=example,ou=People,dc=example,dc=com
changetype: modify
replace: lastLoginTime
lastLoginTime: 20210101000000Z
```

2. このユーザーとしてディレクトリーに接続してみてください。

```
# ldapsearch -H ldap://server.example.com -x -D
"uid=example,ou=People,dc=example,dc=com" -W -b "dc=example,dc=com"
ldap_bind: Constraint violation (19)
 additional info: Account inactivity limit exceeded. Contact system administrator to reset.
```

Directory Server がアクセスを拒否してこのエラーを返した場合、アカウントの非アクティブが機能します。

関連情報

- 非アクティブ制限に達したアカウントを再度有効にする

17.2. アカウントを作成してから一定時間、アカウントを自動的に無効にする

次の手順に従って、**dc=example,dc=com** エントリーのアカウントが管理者が作成してから 60 日後に期限切れになるように設定します。

たとえば、アカウントの有効期限機能を使用して、外部ワーカーのアカウントが作成されてから一定時間ロックされるようにします。

手順

1. アカウントポリシープラグインを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy
enable
```

2. プラグイン設定エントリーを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy
config-entry set "cn=config,cn=Account Policy Plugin,cn=plugins,cn=config" --always-
record-login yes --state-attr createTimestamp --alt-state-attr 1.1 --spec-attr
acctPolicySubentry --limit-attr accountInactivityLimit
```

コマンドは、以下のオプションを使用します。

- **--always-record-login yes**: ログイン時間のログを有効にします。これは、**acctPolicySubentry** 属性が設定されていない場合でも、サービスクラス (Co S) またはアカウントポリシーを持つロールを使用するために必要です。

- **--state-attr createTimestamp**: アカウントポリシープラグインが **createTimestamp** 属性の値を使用して、アカウントの有効期限が切れているかどうかを計算するように設定します。
- **--alt-state-attr 1.1**: 代替属性を使用してプライマリー属性が存在しないかどうかをチェックすることを無効にします。
- **--spec-attr acctPolicySubentry**: **acctPolicySubentry** 属性が設定されているエントリーにポリシーを適用するように Directory Server を設定します。この属性は、CoS エントリーで設定します。
- **--limit-attr accountInactivityLimit**: アカウントの有効期限ポリシーエントリーの **accountInactivityLimit** 属性に最大経過時間を保存するように設定します。

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

4. アカウントの有効期限ポリシーエントリーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=Account Expiration Policy,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: accountpolicy
accountInactivityLimit: 5184000
cn: Account Expiration Policy
```

accountInactivityLimit 属性の値は、アカウントが作成されてから **5184000** 秒 (60 日) で期限切れになるように設定します。

5. CoS テンプレートエントリーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=TemplateCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: cosTemplate
acctPolicySubentry: cn=Account Expiration Policy,dc=example,dc=com
```

このテンプレートエントリーは、アカウントの有効期限ポリシーを参照します。

6. CoS 定義エントリーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=DefinitionCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectclass: cosSuperDefinition
```

```
objectclass: cosPointerDefinition
cosTemplateDn: cn=TemplateCoS,dc=example,dc=com
cosAttribute: acctPolicySubentry default operational-default
```

この定義エントリーは CoS テンプレートエントリーを参照し、**acctPolicySubentry** 属性が各ユーザーエントリーに表示され、値が **cn=Account Expiration Policy,dc=example,dc=com** 設定されます。

検証

- **createTimestamp** 属性が 60 日以上前の値に設定されている **dc=example,dc=com** エントリーに格納されているユーザーとしてディレクトリーに接続してみてください。

```
# ldapsearch -H ldap://server.example.com -x -D "uid=example,dc=example,dc=com" -W -b "dc=example,dc=com"
ldap_bind: Constraint violation (19)
 additional info: Account inactivity limit exceeded. Contact system administrator to reset.
```

Directory Server がアクセスを拒否してこのエラーを返した場合、アカウントの有効期限が機能します。

関連情報

- 非アクティブ制限に達したアカウントを再度有効にする

17.3. パスワードの有効期限が切れてから一定時間アカウントを自動的に無効にする

この手順に従って、28 日を超えてパスワードを変更しない **dc=example,dc=com** エントリーの下のユーザーを非アクティブ化する時間ベースのロックアウトポリシーを設定します。

前提条件

- ユーザーは、エントリーに **passwordExpirationTime** 属性を設定する必要があります。

手順

1. パスワードの有効期限機能を有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace passwordExp=on
```

2. アカウントポリシープラグインを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy enable
```

3. プラグイン設定エントリーを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy config-entry set "cn=config,cn=Account Policy Plugin,cn=plugins,cn=config" --always-record-login yes --always-record-login-attr lastLoginTime --state-attr
```

```
non_existent_attribute --alt-state-attr passwordExpirationTime --spec-attr
acctPolicySubentry --limit-attr accountInactivityLimit
```

コマンドは、以下のオプションを使用します。

- **--always-record-login yes**: ログイン時間のログを有効にします。これは、**acctPolicySubentry** 属性が設定されていない場合でも、サービスクラス (Co S) またはアカウントポリシーを持つロールを使用するために必要です。
- **--always-record-login-attr lastLoginTime**: アカウントポリシープラグインがユーザーの **lastLoginTime** 属性に最終ログイン時刻を保存するように設定します。
- **--state-attr non_existent_attribute**: アカウントポリシーの評価に使用されるプライマリー時間属性を、存在しないダミー属性名に設定します。
- **--alt-state-attr `passwordExpirationTime**: チェックする代替属性として **passwordExpirationTime** 属性を使用するようにプラグインを設定します。
- **--spec-attr acctPolicySubentry**: **acctPolicySubentry** 属性が設定されているエントリーにポリシーを適用するように Directory Server を設定します。この属性は、CoS エントリーで設定します。
- **--limit-attr accountInactivityLimit**: アカウントポリシーエントリーの **accountInactivityLimit** 属性に、最後にパスワードを変更した後にアカウントが非アクティブ化された時刻を保存するように設定します。

4. インスタンスを再起動します。

```
# dsctl instance_name restart
```

5. アカウント非アクティブ化ポリシーエントリーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=Account Inactivation Policy,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: accountpolicy
accountInactivityLimit: 2419200
cn: Account Inactivation Policy
```

account Inactivity Limit 属性の値は、パスワードが変更されてから **2419200** 秒 (28 日) 後に Directory Server がアカウントを非アクティブ化するように設定します。

6. CoS テンプレートエントリーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=TemplateCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: cosTemplate
acctPolicySubentry: cn=Account Inactivation Policy,dc=example,dc=com
```

このテンプレートエントリーは、アカウントの非アクティブ化ポリシーを参照します。

7. CoS 定義エントリーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=DefinitionCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=TemplateCoS,dc=example,dc=com
cosAttribute: acctPolicySubentry default operational-default
```

この定義エントリーは CoS テンプレートエントリーを参照し、**acctPolicySubentry** 属性が各ユーザーエントリーに表示され、値が **cn=Account Inactivation Policy,dc=example,dc=com** 設定されます。

検証

1. ユーザーの **passwordExpirationTime** 属性を、設定した非アクティブ時間よりも古い値に設定します。

```
# ldapmodify -H ldap://server.example.com -x -D "cn=Directory Manager" -W

dn: uid=example,ou=People,dc=example,dc=com
changetype: modify
replace: passwordExpirationTime
passwordExpirationTime: 20210101000000Z
```

2. このユーザーとしてディレクトリーに接続してみてください。

```
# ldapsearch -H ldap://server.example.com -x -D
"uid=example,ou=People,dc=example,dc=com" -W -b "dc=example,dc=com"
ldap_bind: Constraint violation (19)
 additional info: Account inactivity limit exceeded. Contact system administrator to reset.
```

Directory Server がアクセスを拒否してこのエラーを返した場合、アカウントの非アクティブが機能します。

関連情報

- 非アクティブ制限に達したアカウントを再度有効にする

17.4. アカウントのステータス (アクティブかどうか) とパスワードの有効期限の両方でアカウントを自動的に無効にする

checkAllStateAttrs 設定を使用してユーザーの認証時に、アカウントのステータス (アクティブかどうか)、またパスワードの有効期限の両方を適用できます。デフォルトでは、プラグイン設定エントリーに **checkAllStateAttrs** が存在しない場合、またはこのパラメーターを **no** に設定した場合、プラグインは状態属性 **lastLoginTime** をチェックします。属性がエントリーに存在しない場合には、プラグインは別の状態属性をチェックします。

プラグインが **passwordExpirationtime** 属性に基づいて有効期限を処理するようにする場合は、メイン

の状態属性を存在しない属性に設定し、別の状態属性を **passwordExpirationtime** に設定できます。このパラメーターを有効にすると、主な状態属性をチェックし、アカウントに問題がある場合は、別の状態属性を確認します。

これは、passwordExpirationtime がアクティブではない期間の制限を超えると、アカウントポリシープラグインがアカウントを完全に無効にするという点で、パスワードポリシーのパスワード有効期限とは異なります。パスワードポリシーの有効期限が切れても、ユーザーは引き続きログインしてパスワードを変更できます。アカウントポリシープラグインはユーザーの操作を完全にブロックするため、管理者はアカウントをリセットする必要があります。

手順

1. プラグイン設定エントリーを作成し、設定を有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy config-entry set "cn=config,cn=Account Policy Plugin,cn=plugins,cn=config" --always-record-login yes --state-attr lastLoginTime --alt-state-attr 1.1 --spec-attr acctPolicySubentry --limit-attr accountInactivityLimit --check-all-state-attrs yes
```

2. サーバーを再起動して、新しいプラグイン設定を読み込みます。

```
# dsctl instance_name restart
```

警告

checkAllStateAttrs 設定は、別の state 属性が **passwordExpiratontime** に設定されている場合にのみ機能するように設計されています。これを **createTimestamp** に設定すると、予想外の結果が発生し、エントリーがロックアウトされる可能性があります。

第18章 非アクティブ制限に達したアカウントを再度有効にする

Directory Server が非アクティブ制限に達したためにアカウントを非アクティブ化した場合、管理者はアカウントを再度有効にすることができます。

18.1. アカウントポリシープラグインによって非アクティブ化されたアカウントを再度有効にする

dsconf account unlock コマンドを使用するか、非アクティブ化されたユーザーの **lastLoginTime** 属性を手動で更新することにより、アカウントを再度有効にすることができます。

前提条件

- 非アクティブ化されたユーザーアカウント。

手順

- 次のいずれかの方法を使用して、アカウントを再アクティブ化します。
  - **dsconf account unlock** コマンドの使用:

    ```
    # dsidm -D "cn=Directory manager" ldap://server.example.com -b
    "dc=example,dc=com" account unlock
    "uid=example,ou=People,dc=example,dc=com"
    ```

  - ユーザーの **lastLoginTime** 属性を最近のタイムスタンプに設定するには、次のようにします。

    ```
    # ldapmodify -H ldap://server.example.com -x -D "cn=Directory Manager" -W

    dn: uid=example,ou=People,dc=example,dc=com
    changetype: modify
    replace: lastLoginTime
    lastLoginTime: 20210901000000Z
    ```

検証

- 再アクティブ化したユーザーとして認証します。たとえば、次の検索を実行します。

  ```
  # ldapsearch -H ldap://server.example.com -x -D
  "uid=example,ou=People,dc=example,dc=com" -W -b "dc=example,dc=com -s base"
  ```

  ユーザーが正常に認証できる場合、アカウントは再度アクティブ化されました。

第19章 ロックアウトポリシーを設定せずに最終ログイン時間を追跡する

アカウントポリシープラグインを使用すると、有効期限や非アクティブ期間を設定せずに、ユーザーのログイン時間を追跡できます。この場合、プラグインは **lastLoginTime** 属性をユーザーエントリーに追加します。

19.1. 最終ログイン時刻を記録するようにアカウントポリシープラグインを設定する

この手順に従って、ユーザーエントリーの **lastLoginTime** 属性にユーザーの最終ログイン時刻を記録します。

手順

1. アカウントポリシープラグインを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy enable
```

2. ログイン時間を記録するプラグイン設定エントリーを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy config-entry set "cn=config,cn=Account Policy Plugin,cn=plugins,cn=config" --always-record-login yes --state-attr lastLoginTime
```

   コマンドは、以下のオプションを使用します。

   - **--always-record-login yes**: ログイン時間のログ記録を有効にします。
   - **--state-attr lastLoginTime**: アカウントポリシープラグインがユーザーの **lastLoginTime** 属性に最終ログイン時刻を保存するように設定します。

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

検証

1. ユーザーとして Directory Server にログインします。たとえば、次の検索を実行します。

```
# ldapsearch -H ldap://server.example.com -x -D "uid=example,ou=People,dc=example,dc=com" -W -b "dc=example,dc=com"
```

2. 前の手順で使用したユーザーの **lastLoginTime** 属性を表示します。

```
# ldapsearch -H ldap://server.example.com -x -D "cn=Directory Manager" -W -b "uid=example,ou=people,dc=example,dc=com" lastLoginTime
...
dn: uid=example,ou=People,dc=example,dc=com
lastLoginTime: 20210913091435Z
```

lastLoginTime 属性が存在し、Directory Server がその値を更新した場合、最終ログイン時刻の記録が機能します。

第20章 DIRECTORY MANAGER アカウントにアクセス制御を設定する

未制約の管理ユーザーがあると、メンテナンスパースペクティブからは妥当になります。Directory Manager では、メンテナンスタスクを実行し、インシデントへの対応に高いレベルのアクセスが必要です。

ただし、Directory Manager ユーザーの権限により、管理ユーザーとして実行される攻撃による被害を防ぐために、一定レベルのアクセス制御を行うことを推奨します。

20.1. DIRECTORY MANAGER アカウントのアクセス制御

Directory Server は、通常のアクセス制御命令をディレクトリーツリーだけに適用します。Directory Manager アカウントの権限はハードコーディングされており、バインドルールでこのアカウントを使用することはできません。Directory Manager アカウントへのアクセスを制限するには、**RootDN Access Control** プラグインを使用します。

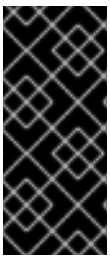
このプラグインの機能は、標準のアクセス制御命令 (ACI) とは異なります。たとえば、ターゲット (Directory Manager エントリー) や許可されたアクセス許可 (それらすべて) などの特定の情報が暗示されます。**RootDN Access Control** プラグインの目的は、場所または時間に基づいてディレクトリーマネージャーとしてログインできるユーザーを制限することにより、セキュリティーレベルを提供することであり、このユーザーができることを制限することではありません。

このため、プラグインの設定は以下のみをサポートします。

- 特定の日および特定の時間範囲でアクセスを許可または拒否する時間ベースのアクセス制御
- 定義された IP アドレス、サブネット、およびドメインからのアクセスを許可または拒否するための IP アドレス規則
- 特定のホスト、ドメイン、およびサブドメインからのアクセスを許可または拒否するホストアクセスルール

Directory Manager に設定できるアクセス制御規則は 1 つだけです。これはプラグインエントリーにあり、ディレクトリー全体に適用されます。

通常の ACI と同様に、拒否ルールは許可ルールよりも優先度が高くなります。



重要

Directory Manager アカウントに適切なレベルのアクセス権があることを確認してください。この管理ユーザーは、時間外に保守操作を実行したり、障害に対応したりする必要がある場合があります。この場合、制限が厳しすぎる時間または曜日のルールを設定すると、Directory Manager ユーザーがディレクトリーを適切に管理できなくなる可能性があります。

20.2. コマンドラインを使用した ROOTDN アクセス制御プラグインの設定

デフォルトでは、**RootDN Access Control** プラグインは無効になっています。Directory Manager アカウントのアクセス許可を制限するには、プラグインを有効にして設定します。

手順

1. **RootDN Access Control** プラグインを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin root-dn enable
```

2. バインドルールを設定します。たとえば、Directory Manager アカウントが IP アドレス **192.0.2.1** のホストから午前 6 時から午後 9 時までの間のみログインできるようにするには、次のように入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin root-dn set --open-time=0600 --close-time=2100 --allow-ip="192.0.2.1"
```

設定できるパラメーターの完全なリストとその説明については、次のように入力してください。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin root-dn set --help
```

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

検証

- 許可されていない、または許可された時間範囲外のホストから **cn=Directory Manager** としてクエリーを実行します。

```
[user@192.0.2.2]$ ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -x -b "dc=example,dc=com"
Enter LDAP Password:
ldap_bind: Server is unwilling to perform (53)
 additional info: RootDN access control violation
```

Directory Server がアクセスを拒否した場合、プラグインは期待どおりに動作します。

20.3. WEB コンソールを使用して ROOTDN アクセス制御プラグインを設定する

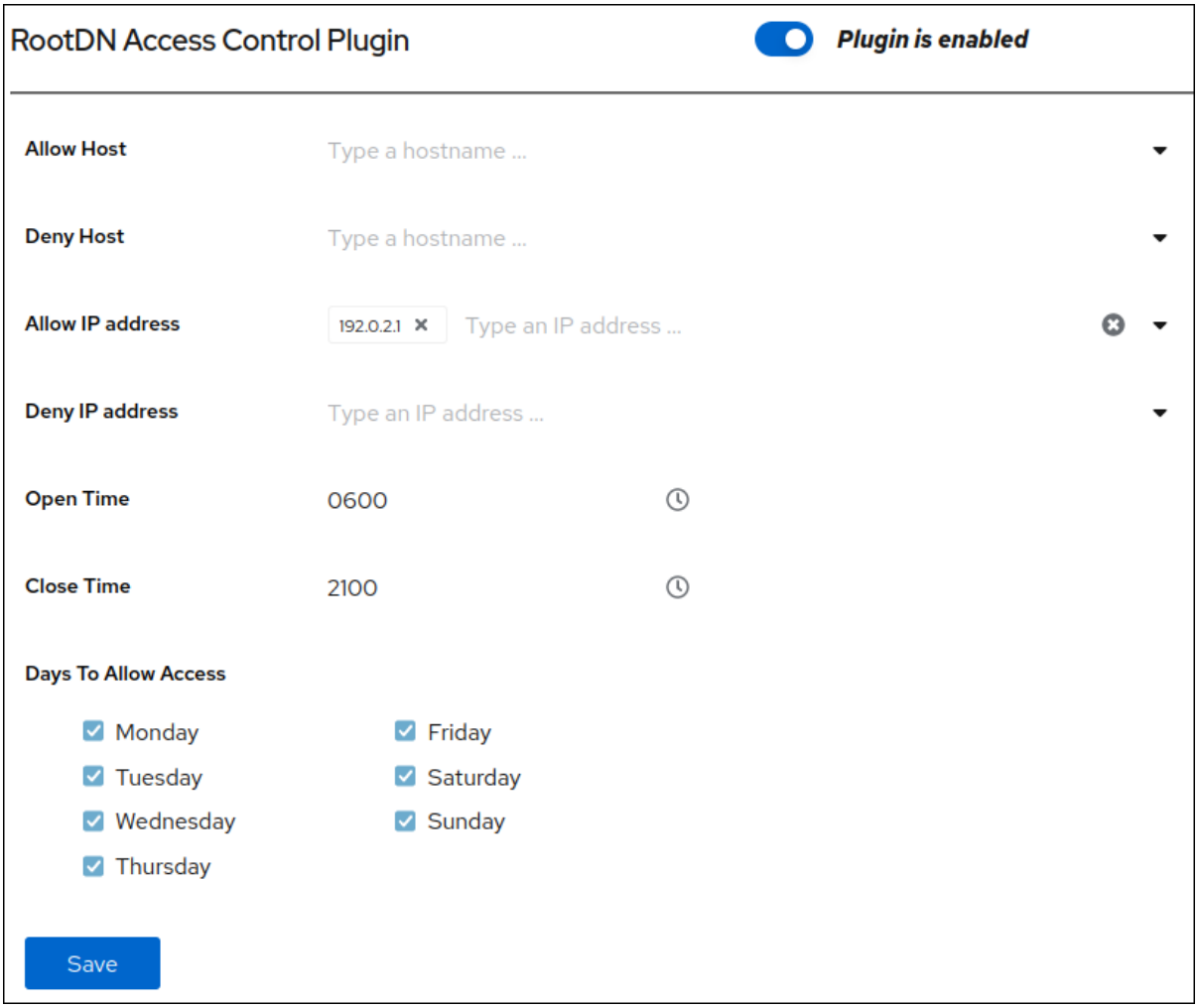
デフォルトでは、**RootDN Access Control** プラグインは無効になっています。Directory Manager アカウントのアクセス許可を制限するには、プラグインを有効にして設定します。

前提条件

- Web コンソールでインスタンスにログインしている。

手順

1. **Plugins → RootDN Access Control** に移動します。
2. プラグインを有効にします。
3. 要件に従ってフィールドに入力します。



4. **Save** をクリックします。
5. 右上隅の **Actions** をクリックし、**Restart Instance** を選択します。

検証

- 許可されていない、または許可された時間範囲外のホストから **cn=Directory Manager** としてクエリーを実行します。

```
[user@192.0.2.2]$ ldapsearch -D "cn=Directory Manager" -W -H
ldap://server.example.com -x -b "dc=example,dc=com"
Enter LDAP Password:
ldap_bind: Server is unwilling to perform (53)
        additional info: RootDN access control violation
```

Directory Server がアクセスを拒否した場合、プラグインは期待どおりに動作します。

第21章 属性暗号化の管理

Directory Server は、ディレクトリー内の機密データへのアクセスを保護するための多数のメカニズムを提供します。ただし、デフォルトでは、サーバーは暗号化されていないデータをデータベースに格納します。機密性の高い情報の場合、攻撃者がデータベースにアクセスできるという潜在的なリスクは、重大なリスクになる可能性があります。

属性の暗号化機能により、管理者は特定の属性を機密データ (政府識別番号など) と共に暗号化してデータベースに保存できます。接尾辞を有効にすると、これらの属性のすべてのインスタンス (インデックスデータも含む) が、データベース内のこの属性に格納されているすべてのエントリーに対して暗号化されます。接尾辞の属性暗号化を有効にできることに注意してください。サーバー全体でこの機能を有効にするには、サーバー上の各接尾辞の属性暗号化を有効にする必要があります。属性の暗号化は、**eq** および **pres** インデックス作成と完全に互換性があります。

重要

エントリーの識別名 (DN) 内で使用する属性は、効率的に暗号化できません。たとえば、**uid** 属性を暗号化するように設定した場合、値はエントリーでは暗号化されますが、DN では暗号化されません。

```
dn: uid=demo_user,ou=People,dc=example,dc=com
...
uid::Sf04P9nJWGU1qiW9JJCGRg==
```

21.1. DIRECTORY SERVER が属性の暗号化に使用するキー

属性の暗号化を使用するには、TLS を使用して暗号化された接続を設定する必要があります。Directory Server は、サーバーの TLS 暗号化キーと、属性の暗号化に同じ PIN 入力方法を使用します。

サーバーは、ランダムに生成された対称暗号鍵を使用して、属性データを暗号化および復号化します。サーバーは、サーバーの TLS 証明書の公開鍵を使用してこれらの鍵をラップします。結果として、属性暗号化の有効な強度は、サーバーの TLS キーの強度より高くすることはできません。

警告

サーバーの秘密鍵にアクセスできないと、ラップ済みのコピーから対称キーを復旧することができません。そのため、サーバーの証明書データベースを定期的にバックアップしてください。キーを紛失すると、データベースに保存されているデータを復号化および暗号化できなくなります。

21.2. コマンドラインを使用して属性暗号化を有効にする

この手順では、コマンドラインを使用して **userRoot** データベースの **telephoneNumber** 属性の属性暗号化を有効にする方法を示します。この手順を実行すると、サーバーはこの属性の既存の値と新しい値を AES 暗号化して格納します。

前提条件

- Directory Server で TLS 暗号化を有効にした。

手順

1. **userRoot** データベースをエクスポートします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend export -E
userRoot
```

サーバーは、エクスポートを **/var/lib/dirsrv/slapd-instance_name/ldif/** ディレクトリーの LDIF ファイルに保存します。**-E** オプションは、エクスポート中にすでに暗号化されている属性を復号化します。

2. **telephoneNumber** 属性の AES 暗号化を有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend attr-encrypt --
add-attr telephoneNumber dc=example,dc=com
```

3. インスタンスを停止します。

```
# dsctl instance_name stop
```

4. LDIF ファイルをインポートします。

```
# dsctl instance_name ldif2db --encrypted userRoot /var/lib/dirsrv/slapd-
instance_name/ldif/None-userroot-2022_01_24_10_28_27.ldif
```

--encrypted パラメーターを使用すると、スクリプトで属性を暗号化して、インポート中に暗号化を設定できます。

5. インスタンスを起動します。

```
# dsctl instance_name start
```

関連情報

- Directory Server への TLS 暗号化接続の有効化

21.3. WEB コンソールを使用して属性暗号化を有効にする

この手順では、Web コンソールを使用して **userRoot** データベースの **telephoneNumber** 属性の属性暗号化を有効にする方法を示します。この手順を実行すると、サーバーはこの属性の既存の値と新しい値を AES 暗号化して格納します。

Web コンソールのエクスポートおよびインポート機能は、暗号化された属性をサポートしていないことに注意してください。したがって、これらの手順はコマンドラインで実行する必要があります。

前提条件

- Directory Server で TLS 暗号化を有効にした。
- Web コンソールでインスタンスにログインしている。

手順

1. **userRoot** データベースをエクスポートします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend export -E
userRoot
```

サーバーは、エクスポートを **/var/lib/dirsrv/slapd-instance_name/ldif/** ディレクトリーの LDIF ファイルに保存します。**-E** オプションは、エクスポート中にすでに暗号化されている属性を復号化します。

2. Web コンソールで、**Database → Suffixes → suffix_entry → Encrypted Attributes** に移動します。

3. 暗号化する属性を入力し、**Add Attribute** をクリックします。

4. **Actions** メニューで、**Stop Instance** を選択します。

5. コマンドラインで、LDIF ファイルをインポートします。

```
# dsctl instance_name ldif2db --encrypted userRoot /var/lib/dirsrv/slapd-
instance_name/ldif/None-userroot-2022_01_24_10_28_27.ldif
```

--encrypted パラメーターを使用すると、スクリプトで属性を暗号化して、インポート中に暗号化を設定できます。

6. Web コンソールで **Actions** メニューを開き、**Start Instance** を選択します。

関連情報

- Directory Server への TLS 暗号化接続の有効化

21.4. 属性暗号化の有効化後の一般的な考慮事項

データベースにすでに存在するデータの暗号化を有効にした後、次の点を考慮してください。

- 暗号化されていないデータは、サーバーのデータベースページプールのバッキングファイルで保持できます。このデータを削除するには、以下を実行します。

  a. インスタンスを停止します。

  ```
  # dsctl instance_name stop
  ```

  b. **/var/lib/dirsrv/slapd-instance_name/db/guardian** ファイルを削除します。

  ```
  # **rm /var/lib/dirsrv/slapd-instance_name/db/guardian``
  ```

  c. インスタンスを起動します。

  ```
  # dsctl instance_name start
  ```

- 暗号化を有効にし、データが正常にインポートされた後に、暗号化されていないデータで LDIF ファイルを削除します。

- Directory Server はレプリケーションログファイルを暗号化しません。このデータを保護するには、レプリケーションログを暗号化されたディスクに保存します。

- サーバーのメモリー (RAM) のデータは暗号化されず、swap パーティションに一時的に保存できます。このデータを保護するには、暗号化されたスワップ領域を設定します。

重要

暗号化されていないデータを含むファイルを削除すると、このデータは特定の状況で復元できます。

21.5. 属性暗号化に使用される TLS 証明書の更新

属性の暗号化は、サーバーの TLS 証明書に基づいています。TLS 証明書を更新または置き換えた後に属性の暗号化が失敗しないようにするには、次の手順に従います。

前提条件

- 属性の暗号化を設定しました。
- TLS 証明書の有効期限がまもなく切れる。

手順

1. **userRoot** データベースをエクスポートします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend export -E
userRoot
```

サーバーは、エクスポートを **/var/lib/dirsrv/slapd-instance_name/ldif/** ディレクトリーの LDIF ファイルに保存します。**-E** オプションは、エクスポート中にすでに暗号化されている属性を復号化します。

2. プライベートキーおよび証明書署名要求 (CSR) を作成します。外部ユーティリティーを使用して作成する場合は、この手順を省略します。

   - ホストが 1 つの名前のみで到達可能である場合は、以下を実行します。

```
# dsctl instance_name tls generate-server-cert-csr -s
"CN=server.example.com,O=example_organization"
```

   - 複数の名前でホストにアクセスできる場合は、以下を行います。

```
# dsctl instance_name tls generate-server-cert-csr -s
"CN=server.example.com,O=example_organization" server.example.com
server.example.net
```

   最後のパラメーターとしてホスト名を指定した場合、このコマンドは **DNS:server.example.com, DNS:server.example.net** エントリーで SAN (Subject Alternative Name) 拡張を CSR に追加します。

-s subject パラメーターで指定した文字列は、RFC 1485 に従って有効なサブジェクト名である必要があります。サブジェクトの **CN** フィールドが必要で、サーバーの完全修飾ドメイン名 (FQDN) の 1 つに設定する必要があります。このコマンドは、**/etc/dirsrv/slapd-instance_name/Server-Cert.csr** ファイルに CSR を保存します。

3. 認証局 (CA) に CSR を送信し、発行した証明書を取得します。詳細は、CA のドキュメントを参照してください。

4. CA が発行するサーバー証明書を NSS データベースにインポートします。

   - **dsctl tls generate-server-cert-csr** コマンドを使用して秘密鍵を作成した場合は、以下を入力します。

     ```
     # dsconf -D "cn=Directory Manager" ldap://server.example.com security certificate add --file /root/instance_name.crt --name "server-cert" --primary-cert
     ```

     --name _certificate_nickname パラメーターで設定した証明書の名前を書き留めておきます。これは後のステップで必要になります。

   - 外部ユーティリティーを使用して秘密鍵を作成した場合は、サーバー証明書および秘密鍵をインポートします。

     ```
     # dsctl instance_name tls import-server-key-cert /root/server.crt /root/server.key
     ```

     このコマンドでは、最初にサーバー証明書へのパスを指定してから、秘密鍵へのパスを指定する必要があります。このメソッドは、証明書のニックネームを **Server-Cert** に設定します。

5. CA 証明書を NSS データベースにインポートします。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com security ca-certificate add --file /root/ca.crt --name "Example CA"
   ```

6. CA 証明書の信頼フラグを設定します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com security ca-certificate set-trust-flags "Example CA" --flags "CT,,"
   ```

   これにより、Directory Server が、TLS による暗号化および証明書ベースの認証に対して CA を信頼するように設定します。

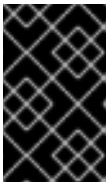
7. インスタンスを停止します。

   ```
   # dsctl instance_name stop
   ```

8. **/etc/dirsrv/slapd-instance_name/dse.ldif** ファイルを編集し、属性を含む以下のエントリーを削除します。

   - **cn=AES,cn=encrypted attribute keys,cn=database_name,cn=ldbm database,cn=plugins,cn=config**

   - **cn=3DES,cn=encrypted attribute keys,cn=database_name,cn=ldbm database,cn=plugins,cn=config**

   重要

   全データベースのエントリーを削除します。**nsSymmetricKey** 属性を含むエントリーが `/etc/dirsrv/slapd-**instance_name**/dse.ldi ファイルに残されると、Directory Server は起動に失敗します。

9. LDIF ファイルをインポートします。

```
# dsctl instance_name ldif2db --encrypted userRoot /var/lib/dirsrv/slapd-
instance_name/ldif/None-userroot-2022_01_24_10_28_27.ldif
```

--encrypted パラメーターを使用すると、スクリプトで属性を暗号化して、インポート中に暗号化を設定できます。

10. インスタンスを起動します。

```
# dsctl instance_name start
```