Red Hat Enterprise Linux 8

さまざまな種類のサーバーのデプロイメント

Web サーバーとリバースプロキシー、ネットワークファイルサービス、データベースサーバー、メールトランスポートエージェント、およびプリンターのセットアップと設定

Last Updated: 2024-06-26

Red Hat Enterprise Linux 8 さまざまな種類のサーバーのデプロイメント

Web サーバーとリバースプロキシー、ネットワークファイルサービス、データベースサーバー、メールトランスポートエージェント、およびプリンターのセットアップと設定

法律上の通知

概要

Red Hat Enterprise Linux 8 上で Apache HTTP Web サーバーまたは NGINX Web サーバーを設定して実行します。Samba や NFS などのネットワークファイルサービスを設定して使用します。MariaDB、MySQL、または PostgreSQL データベースサーバーをインストールして設定し、データをバックアップして移行します。Postfix メールトランスポートエージェントをデプロイして設定します。CUPS サーバーを使用した印刷を設定します。

目次

RED HAT ドキュメントへのフィードバック (英語のみ)

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

Jira からのフィードバック送信 (アカウントが必要)

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

第1章 APACHE HTTP WEB サーバーの設定

1.1. APACHE HTTP WEB サーバーの概要

Web サーバー は、Web 経由でクライアントにコンテンツを提供するネットワークサービスです。これは通常 Web ページを指しますが、他のドキュメントも当てはまります。Web サーバーは、**ハイパーテキスト転送プロトコル** (**HTTP**) を使用するため、HTTP サーバーとも呼ばれます。

Apache HTTP Server (**httpd**) は、Apache Software Foundation が開発したオープンソースの Web サーバーです。

Red Hat Enterprise Linux の以前のリリースからアップグレードする場合は、適切に **httpd** サービス設定を更新する必要があります。本セクションでは、新たに追加された機能の一部と、以前の設定ファイルの更新を説明します。

1.2. APACHE HTTP SERVER への主な変更点

Apache HTTP Server が、RHEL 7 のバージョン 2.4.6 から、RHEL 8 のバージョン 2.4.37 に更新されました。この更新バージョンには新機能がいくつか含まれていますが、外部モジュールの設定および Application Binary Interface (ABI) のレベルでは、RHEL 7 バージョンとの後方互換性を維持します。

新機能は次のとおりです。

- **httpd** モジュール含まれる **mod_http2** パッケージにより、**HTTP/2** に対応するようになりました。
- systemd ソケットのアクティベーションが対応します。詳細は、man ページの **httpd.socket(8)** を参照してください。
- 新しいモジュールが複数追加されています。
  - **mod_proxy_hcheck** - プロキシーのヘルスチェックモジュール
  - **mod_proxy_uwsgi** - Web Server Gateway Interface (WSGI) プロキシー
  - **mod_proxy_fdpass** - クライアントのソケットを別のプロセスに渡す
  - **mod_cache_socache** - HTTP キャッシュ (例: memcache バックエンドを使用)
  - **mod_md** - ACME プロトコルの SSL/TLS 証明書サービス
- 以下のモジュールはデフォルトで読み込まれるようになりました。
  - **mod_request**
  - **mod_macro**
  - **mod_watchdog**
- 新しいサブパッケージ **httpd-filesystem** が追加されています。これには、**Apache HTTP Server** の基本的なディレクトリーレイアウト (ディレクトリーの適切な権限を含む) が含まれます。
- インスタンス化されたサービスのサポート **httpd@.service** が導入されました。詳細は、man ページの **httpd.service** を参照してください。

- 新しい **httpd-init.service** が **%post script** に置き換わり、自己署名の鍵ペア **mod_ssl** を作成します。

- (**Let's Encrypt** などの証明書プロバイダーで使用するため) 自動証明書管理環境 (ACME) プロトコルを使用した、TLS 証明書の自動プロビジョニングおよび更新に、**mod_md** パッケージで対応するようになりました。

- Apache HTTP Server が、**PKCS#11** モジュールを利用して、ハードウェアのセキュリティートークンから、TLS 証明書および秘密鍵を直接読み込むようになりました。これにより、**mod_ssl** 設定で、**PKCS#11** URL を使用して、**SSLCertificateKeyFile** ディレクティブおよび **SSLCertificateFile** ディレクティブに、TLS 秘密鍵と、必要に応じて TLS 証明書をそれぞれ指定できるようになりました。

- **/etc/httpd/conf/httpd.conf** ファイルの新しい **ListenFree** ディレクティブに対応するようになりました。
  Listen ディレクティブと同様、**ListenFree** は、サーバーがリッスンする IP アドレス、ポート、または IP アドレスとポートの組み合わせに関する情報を提供します。ただし、**ListenFree** を使用すると、**IP_FREEBIND** ソケットオプションがデフォルトで有効になります。したがって、**httpd** は、ローカルではない IP アドレス、または今はまだ存在していない IP アドレスにバインドすることもできます。これにより、**httpd** がソケットをリッスンできるようになり、**httpd** がバインドしようとするときに、基になるネットワークインターフェイスまたは指定した動的 IP アドレスを起動する必要がなくなります。

  ListenFree ディレクティブは、現在 RHEL 8 でのみ利用できます。

  ListenFree の詳細は、以下の表を参照してください。

  表1.1 ListenFree ディレクティブの構文、状態、およびモジュール

構文	状態	モジュール
ListenFree [IP-address:]portnumber [protocol]	MPM	event、worker、prefork、mpm_winnt、mpm_netware、mpmt_os2

その他の主な変更点は次の通りです。

- 以下のモジュールが削除されました。
  - **mod_file_cache**
  - **mod_nss**
    代わりに **mod_ssl** を使用します。**mod_nss** からの移行に関する詳細は、「Apache Web Server 設定で秘密鍵と証明書を使用できるように NSS データベースからの証明書のエクスポート」 を参照してください。
  - **mod_perl**

- RHEL 8 の **Apache HTTP Server** が使用するデフォルトの DBM 認証データベースのデフォルトタイプが、**SDBM** から **db5** に変更になりました。

- **Apache HTTP Server** の **mod_wsgi** モジュールが Python 3 に更新されました。WSGI アプリケーションは Python 3 でしか対応していないため、Python 2 から移行する必要があります。

- **Apache HTTP Server** を使用してデフォルトで設定されたマルチプロセッシングモジュール (MPM) は、マルチプロセスのフォークモデル (**prefork** として知られています) から、高パフォーマンスのマルチスレッドモデル **event** に変更しました。
スレッドセーフではないサードパーティーのモジュールは、交換または削除する必要があります。設定した MPM を変更するには、**/etc/httpd/conf.modules.d/00-mpm.conf** ファイルを編集します。詳細は、man ページの **httpd.service(8)** を参照してください。

- suEXEC によりユーザーに許可される最小 UID および GID はそれぞれ 1000 および 500 です (以前は 100 および 100 でした)。

- **/etc/sysconfig/httpd** ファイルは、**httpd** サービスへの環境変数の設定に対応するインターフェイスではなくなりました。systemd サービスに、**httpd.service(8)** の man ページが追加されています。

- **httpd** サービスを停止すると、デフォルトで自動停止が使用されます。

- **mod_auth_kerb** モジュールが、**mod_auth_gssapi** モジュールに置き換わりました。

1.3. 設定の更新

Red Hat Enterprise Linux 7 で使用されている **Apache HTTP Server** バージョンから設定ファイルを更新するには、以下のいずれかのオプションを選択します。

- **/etc/sysconfig/httpd** を使用して環境変数を設定する場合は、代わりに systemd ドロップインファイルを作成します。

- サードパーティーのモジュールを使用する場合は、そのモジュールがスレッド化 MPM と互換性があることを確認してください。

- suexec を使用する場合は、ユーザーおよびグループの ID が新しい最小値に合致していることを確認します。

以下のコマンドを使用すると、設定に誤りがないかどうかを確認できます。

```
# apachectl configtest
Syntax OK
```

1.4. APACHE 設定ファイル

デフォルトでは、**httpd** は起動後に設定ファイルを読み取ります。次の表に、設定ファイルの場所のリストを示します。

表1.2 httpd サービスの設定ファイル

パス	詳細
/etc/httpd/conf/httpd.conf	主要設定ファイル。
/etc/httpd/conf.d/	主要設定ファイル内に含まれている設定ファイル用の補助ディレクトリー。

パス	詳細
/etc/httpd/conf.modules.d/	Red Hat Enterprise Linux にパッケージ化されたインストール済みの動的モジュールを読み込む設定ファイルの補助ディレクトリー。デフォルト設定では、この設定ファイルが最初に処理されます。

デフォルト設定はほとんどの状況に適していますが、その他の設定オプションを使用することもできます。変更を有効にするには、まず Web サーバーを再起動します。

設定に誤りがないことを確認するには、シェルプロンプトで以下のコマンドを実行します。

```
# apachectl configtest
Syntax OK
```

間違いからの復元を容易にするため、編集する前にオリジナルファイルのコピーを作成します。

1.5. HTTPD サービスの管理

本セクションでは、**httpd** サービスを起動、停止、および再起動する方法を説明します。

前提条件

- Apache HTTP Server がインストールされている。

手順

- **httpd** サービスを起動するには、以下を入力します。

```
# systemctl start httpd
```

- **httpd** サービスを停止するには、以下を入力します。

```
# systemctl stop httpd
```

- **httpd** サービスを再起動するには、以下を入力します。

```
# systemctl restart httpd
```

1.6. シングルインスタンスの APACHE HTTP SERVER 設定

シングルインスタンスの Apache HTTP Server を設定して、静的 HTML コンテンツを提供できます。

Web サーバーに関連付けられた全ドメインにサーバーから同じコンテンツを提供する必要がある場合は、この手順に従います。異なるドメインに異なるコンテンツを提供する場合は、名前ベースの仮想ホストを設定します。詳細は Apache 名ベースの仮想ホストの設定 を参照してください。

手順

1. **httpd** パッケージをインストールします。

```
# yum install httpd
```

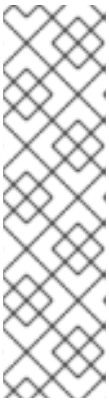
2. **firewalld** を使用する場合は、ローカルのファイアウォールで TCP ポート **80** を開きます。

```
# firewall-cmd --permanent --add-port=80/tcp
# firewall-cmd --reload
```

3. **httpd** サービスを有効にして起動します。

```
# systemctl enable --now httpd
```

4. 必要に応じて、HTML ファイルを **/var/www/html/** ディレクトリーに追加します。



注記

/var/www/html/ にコンテンツを追加する場合には、**httpd** を実行するユーザーが、デフォルトでファイルとディレクトリーを読み取れるようにする必要があります。コンテンツの所有者は、 **root** ユーザーおよび **root** ユーザーグループ、または管理者別のユーザーまたはグループのいずれかになります。コンテンツの所有者が **root** ユーザーおよび **root** ユーザーグループの場合には、他のユーザーがファイルを読み取れるようにする必要があります。すべてのファイルとディレクトリーの SELinux コンテキストは **httpd_sys_content_t** である必要があります。これはデフォルトで **/var/www** ディレクトリー内の全コンテンツに適用されます。

検証手順

- Web ブラウザーで **http://server_IP_or_host_name/** に接続します。
  /var/www/html/ ディレクトリーが空であるか、**index.html** または **index.htm** ファイルが含まれていない場合は、Apache が **Red Hat Enterprise Linux Test Page** を表示します。**/var/www/html/** に異なる名前の HTML ファイルが含まれる場合は、**http://server_IP_or_host_name/example.html** など、そのファイル名に URL を指定して読み込むことができます。

関連情報

- Apache マニュアル: Apache HTTP サーバーマニュアルのインストール
- **httpd.service(8)** man ページを参照してください。

1.7. APACHE 名前ベースの仮想ホストの設定

名前ベースの仮想ホストを使用すると、Apache は、サーバーの IP アドレスに解決されるドメイン別に異なるコンテンツを提供できます。

別々のドキュメントルートディレクトリーを使用して、**example.com** ドメインと **example.net** ドメインの両方に仮想ホストを設定できます。どちらの仮想ホストも静的 HTML コンテンツを提供します。

前提条件

- クライアントおよび Web サーバーは、**example.com** および **example.net** ドメインを Web サーバーの IP アドレスに解決します。
  これらのエントリーは DNS サーバーに手動で追加する必要がある点に注意してください。

手順

1. **httpd** パッケージをインストールします。

```
# yum install httpd
```

2. **/etc/httpd/conf/httpd.conf** ファイルを編集します。

   a. **example.com** ドメイン向けに以下の仮想ホスト設定を追加します。

```
<VirtualHost *:80>
    DocumentRoot "/var/www/example.com/"
    ServerName example.com
    CustomLog /var/log/httpd/example.com_access.log combined
    ErrorLog /var/log/httpd/example.com_error.log
</VirtualHost>
```

   これらの設定は以下を設定します。

   - **<VirtualHost *:80>** ディレクティブの全設定は、この仮想ホストに固有のものです。
   - **DocumentRoot** は、仮想ホストの Web コンテンツへのパスを設定します。
   - **ServerName** は、この仮想ホストがコンテンツを提供するドメインを設定します。複数のドメインを設定するには、**ServerAlias** パラメーターを設定に追加し、追加のドメインをスペース区切りで、このパラメーターに指定します。
   - **CustomLog** は、仮想ホストのアクセスログへのパスを設定します。
   - **ErrorLog** は、仮想ホストのエラーログへのパスを設定します。

注記

Apache は、**ServerName** および **ServerAlias** パラメーターに設定したドメインどれにも一致しない要求の場合でも、設定で最初に検出された仮想マシンを使用します。これには、サーバーの IP アドレス対してに送信される要求も含まれます。

3. **example.net** ドメイン向けに同様の仮想ホスト設定を追加します。

```
<VirtualHost *:80>
    DocumentRoot "/var/www/example.net/"
    ServerName example.net
    CustomLog /var/log/httpd/example.net_access.log combined
    ErrorLog /var/log/httpd/example.net_error.log
</VirtualHost>
```

4. 両方の仮想ホストのドキュメントルートを作成します。

```
# mkdir /var/www/example.com/
# mkdir /var/www/example.net/
```

5. **DocumentRoot** パラメーターのパスが **/var/www/** 内にない設定を行う場合は、両方のドキュメントルートに **httpd_sys_content_t** コンテキストを設定します。

```
# semanage fcontext -a -t httpd_sys_content_t "/srv/example.com(/.*)?"
# restorecon -Rv /srv/example.com/
# semanage fcontext -a -t httpd_sys_content_t "/srv/example.net(/.\*)?"
# restorecon -Rv /srv/example.net/
```

以下のコマンドは、**/srv/example.com/** および **/srv/example.net/** ディレクトリーに **httpd_sys_content_t** コンテキストを設定します。

policycoreutils-python-utils パッケージをインストールして **restorecon** コマンドを実行する必要があります。

6. **firewalld** を使用する場合は、ローカルのファイアウォールでポート **80** を開きます。

```
# firewall-cmd --permanent --add-port=80/tcp
# firewall-cmd --reload
```

7. **httpd** サービスを有効にして起動します。

```
# systemctl enable --now httpd
```

検証手順

1. 仮想ホストのドキュメントルートごとに異なるサンプルファイルを作成します。

```
# echo "vHost example.com" > /var/www/example.com/index.html
# echo "vHost example.net" > /var/www/example.net/index.html
```

2. ブラウザーを使用して **http://example.com** に接続します。Web サーバーは、**example.com** 仮想ホストからのサンプルファイルを表示します。

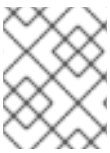
3. ブラウザーを使用して **http://example.net** に接続します。Web サーバーは、**example.net** 仮想ホストからのサンプルファイルを表示します。

関連情報

- Apache HTTP Server マニュアルのインストール - 仮想ホスト

1.8. APACHE HTTP WEB サーバーの KERBEROS 認証の設定

Apache HTTP Web サーバーで Kerberos 認証を実行するために、RHEL 8 は **mod_auth_gssapi** Apache モジュールを使用します。Generic Security Services API (**GSSAPI**) は、Kerberos などのセキュリティーライブラリーを使用する要求を行うアプリケーションのインターフェイスです。**gssproxy** サービスでは、**httpd** サーバーに特権の分離を実装できます。これにより、セキュリティーの観点からこのプロセスが最適化されます。



注記

削除した **mod_auth_kerb** モジュールは、**mod_auth_gssapi** モジュールに置き換わります。

前提条件

- **httpd** パッケージおよび **gssproxy** パッケージがインストールされている。

- Apache Web サーバーが設定され、**httpd** サービスが実行している。

1.8.1. IdM 環境で GSS-Proxy の設定

この手順では、Apache HTTP Web サーバーで Kerberos 認証を実行するように **GSS-Proxy** を設定する方法を説明します。

手順

1. サービスプリンシパルを作成し、HTTP/<SERVER_NAME>@realm プリンシパルの **keytab** ファイルへのアクセスを有効にします。

```
# ipa service-add HTTP/<SERVER_NAME>
```

2. **/etc/gssproxy/http.keytab** ファイルに保存されているプリンシパルの **keytab** を取得します。

```
# ipa-getkeytab -s $(awk '/^server =/ {print $3}' /etc/ipa/default.conf) -k
/etc/gssproxy/http.keytab -p HTTP/$(hostname -f)
```

このステップでは、パーミッションを 400 に設定すると、**root** ユーザーのみが **keytab** ファイルにアクセスできます。**apache** ユーザーは異なります。

3. 以下の内容で **/etc/gssproxy/80-httpd.conf** ファイルを作成します。

```
[service/HTTP]
  mechs = krb5
  cred_store = keytab:/etc/gssproxy/http.keytab
  cred_store = ccache:/var/lib/gssproxy/clients/krb5cc_%U
  euid = apache
```

4. **gssproxy** サービスを再起動して、有効にします。

```
# systemctl restart gssproxy.service
# systemctl enable gssproxy.service
```

関連情報

- **gssproxy(8)** の man ページ
- **gssproxy-mech(8)** の man ページ
- **gssproxy.conf(5)** の man ページ

1.8.2. Apache HTTP Web サーバーが共有するディレクトリーに Kerberos 認証の設定

この手順では、**/var/www/html/private/** ディレクトリーに Kerberos 認証を設定する方法を説明します。

前提条件

- **gssproxy** サービスが設定され、実行されている。

手順

1. **/var/www/html/private/** ディレクトリーを保護するように **mod_auth_gssapi** を設定します。

```
<Location /var/www/html/private>
  AuthType GSSAPI
  AuthName "GSSAPI Login"
  Require valid-user
</Location>
```

2. システムユニット設定のドロップインファイルを作成します。

```
# systemctl edit httpd.service
```

3. 次のパラメーターをシステムのドロップインファイルに追加します。

```
[Service]
Environment=GSS_USE_PROXY=1
```

4. **systemd** 設定をリロードします。

```
# systemctl daemon-reload
```

5. **httpd** サービスを再起動します。

```
# systemctl restart httpd.service
```

検証手順

1. Kerberos チケットを取得します。

```
# kinit
```

2. ブラウザーで、保護されているディレクトリーの URL を開きます。

1.9. APACHE HTTP サーバーで TLS 暗号化の設定

デフォルトでは、Apache は暗号化されていない HTTP 接続を使用してクライアントにコンテンツを提供します。本セクションでは、TLS 暗号化を有効にし、Apache HTTP Server で頻繁に使用される暗号化関連の設定を行う方法を説明します。

前提条件

- Apache HTTP Server がインストールされ、実行している。

1.9.1. Apache HTTP Server への TLS 暗号化の追加

example.com ドメインの Apache HTTP サーバーで TLS 暗号化を有効にすることができます。

前提条件

- Apache HTTP Server がインストールされ、実行している。
- 秘密鍵が **/etc/pki/tls/private/example.com.key** ファイルに保存されている。

秘密鍵および証明書署名要求 (CSR) を作成する方法と、認証局 (CA) からの証明書を要求する方法は、CA のドキュメントを参照してください。または、お使いの CA が ACME プロトコルに対応している場合は、**mod_md** モジュールを使用して、TLS 証明書の取得およびプロビジョニングを自動化できます。

- TLS 証明書は **/etc/pki/tls/certs/example.com.crt** ファイルに保存されます。別のパスを使用する場合は、この手順で対応する手順を調整します。
- 認証局証明書は **/etc/pki/tls/certs/ca.crt** に保存されています。別のパスを使用する場合は、この手順で対応する手順を調整します。
- クライアントおよび Web サーバーは、サーバーのホスト名を Web サーバーの IP アドレスに対して解決します。

手順

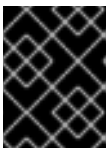
1. **mod_ssl** パッケージをインストールします。

```
# yum install mod_ssl
```

2. **/etc/httpd/conf.d/ssl.conf** ファイルを編集し、以下の設定を **<VirtualHost _default_:443>** ディレクティブに追加します。

   a. サーバー名を設定します。

```
ServerName example.com
```



重要

サーバー名は、証明書の **Common Name** フィールドに設定されているエントリーと一致している必要があります。

   a. 必要に応じて、証明書の **Subject Alt Names** (SAN) フィールドに追加のホスト名が含まれる場合に、これらのホスト名にも TLS 暗号化を提供するように **mod_ssl** を設定できます。これを設定するには、**ServerAliases** パラメーターと対応する名前を追加します。

```
ServerAlias www.example.com server.example.com
```

   b. 秘密鍵、サーバー証明書、および CA 証明書へのパスを設定します。

```
SSLCertificateKeyFile "/etc/pki/tls/private/example.com.key"
SSLCertificateFile "/etc/pki/tls/certs/example.com.crt"
SSLCACertificateFile "/etc/pki/tls/certs/ca.crt"
```

3. セキュリティー上の理由から、**root** ユーザーのみが秘密鍵ファイルにアクセスできるように設定します。

```
# chown root:root /etc/pki/tls/private/example.com.key
# chmod 600 /etc/pki/tls/private/example.com.key
```

警告

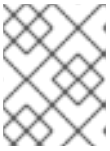
秘密鍵に権限のないユーザーがアクセスした場合は、証明書を取り消し、新しい秘密鍵を作成し、新しい証明書を要求します。そうでない場合は、TLS 接続が安全ではなくなります。

4. **firewalld** を使用する場合は、ローカルのファイアウォールでポート **443** を開きます。

```
# firewall-cmd --permanent --add-port=443/tcp
# firewall-cmd --reload
```

5. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```



注記

パスワードで秘密鍵ファイルを保護した場合は、**httpd** サービスの起動時に毎回このパスワードを入力する必要があります。

検証手順

- ブラウザーを使用して、**https://example.com** に接続します。

関連情報

- SSL/TLS 暗号化
- RHEL 8 における TLS のセキュリティー上の検討事項

1.9.2. Apache HTTP サーバーでサポートされる TLS プロトコルバージョンの設定

デフォルトでは、RHEL の Apache HTTP Server は、最新のブラウザーにも互換性のある安全なデフォルト値を定義するシステム全体の暗号化ポリシーを使用します。たとえば、**DEFAULT** ポリシーでは、**TLSv1.2** および **TLSv1.3** プロトコルバージョンのみが Apache で有効になるように定義します。

Apache HTTP Server がサポートする TLS プロトコルのバージョンを手動で設定できます。たとえば、環境が特定の TLS プロトコルバージョンのみを有効にする必要がある場合には、以下の手順に従います。

- お使いの環境のクライアントで、セキュリティーの低い **TLS1** (TLSv1.0) プロトコルまたは **TLS1.1** プロトコルも使用できるようにする必要がある場合。
- Apache が **TLSv1.2** プロトコルまたは **TLSv1.3** プロトコルのみに対応するように設定する場合。

前提条件

- Apache HTTP Server への TLS 暗号化の追加 で説明されているとおり、TLS 暗号化がサーバーで有効になります。

手順

1. **/etc/httpd/conf/httpd.conf** ファイルを編集し、TLS プロトコルバージョンを設定する **<VirtualHost>** ディレクティブに以下の設定を追加します。たとえば、**TLSv1.3** プロトコルのみを有効にするには、以下を実行します。

```
SSLProtocol -All TLSv1.3
```

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. 以下のコマンドを使用して、サーバーが **TLSv1.3** に対応していることを確認します。

```
# openssl s_client -connect example.com:443 -tls1_3
```

2. 以下のコマンドを使用して、サーバーが **TLSv1.2** に対応していないことを確認します。

```
# openssl s_client -connect example.com:443 -tls1_2
```

サーバーがプロトコルに対応していない場合には、このコマンドは以下のエラーを返します。

```
140111600609088:error:1409442E:SSL routines:ssl3_read_bytes:tlsv1 alert protocol
version:ssl/record/rec_layer_s3.c:1543:SSL alert number 70
```

3. 必要に応じて、他の TLS プロトコルバージョンのコマンドを繰り返し実行します。

関連情報

- **update-crypto-policies(8)** の man ページ
- Using system-wide cryptographic policies .
- **SSLProtocol** パラメーターの詳細については、Apache マニュアルの **mod_ssl** のドキュメント Apache HTTP サーバーマニュアルのインストール を参照してください。

1.9.3. Apache HTTP サーバーで対応している暗号の設定

デフォルトでは、Apache HTTP サーバーは、安全なデフォルト値を定義するシステム全体の暗号化ポリシーを使用します。これは、最近のブラウザーとも互換性があります。システム全体の暗号化で使用可能な暗号化のリストは、**/etc/crypto-policies/back-ends/openssl.config** ファイルを参照してください。

Apache HTTP Server がサポートする暗号を手動で設定できます。お使いの環境で特定の暗号が必要な場合は、以下の手順に従います。

前提条件

- Apache HTTP Server への TLS 暗号化の追加 で説明されているとおり、TLS 暗号化がサーバーで有効になります。

手順

1. **/etc/httpd/conf/httpd.conf** ファイルを編集し、TLS 暗号を設定する **<VirtualHost>** ディレクティブに **SSLCipherSuite** パラメーターを追加します。

```
SSLCipherSuite
"EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH:!SHA1:!SHA256"
```

この例では、**EECDH+AESGCM**、**EDH+AESGCM**、**AES256+EECDH**、および **AES256+EDH** 暗号のみを有効にし、**SHA1** および **SHA256** メッセージ認証コード (MAC) を使用するすべての暗号を無効にします。

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. Apache HTTP Server が対応する暗号化のリストを表示するには、以下を行います。
   a. **nmap** パッケージをインストールします。

```
# yum install nmap
```

   b. **nmap** ユーティリティーを使用して、対応している暗号を表示します。

```
# nmap --script ssl-enum-ciphers -p 443 example.com
...
PORT    STATE SERVICE
443/tcp open  https
| ssl-enum-ciphers:
|   TLSv1.2:
|     ciphers:
|       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (ecdh_x25519) - A
|       TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (dh 2048) - A
|       TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (ecdh_x25519) - A
...
```

関連情報

- **update-crypto-policies(8)** の man ページ
- Using system-wide cryptographic policies .
- SSLCipherSuite

1.10. TLS クライアント証明書認証の設定

クライアント証明書認証を使用すると、管理者は、証明書で認証したユーザーのみが Web サーバーのリソースにアクセスできるようにすることが可能です。**/var/www/html/Example/** ディレクトリーにクライアント証明書認証を設定できます。

Apache HTTP Server が TLS 1.3 プロトコルを使用する場合、特定のクライアントには追加の設定が必要です。たとえば、Firefox で、**about:config** メニューの **security.tls.enable_post_handshake_auth** パラメーターを **true** に設定します。詳細は、Transport Layer Security version 1.3 in Red Hat Enterprise Linux 8 を参照してください。

前提条件

- Apache HTTP Server への TLS 暗号化の追加 で説明されているとおり、TLS 暗号化がサーバーで有効になります。

手順

1. **/etc/httpd/conf/httpd.conf** ファイルを編集し、以下の設定をクライアント認証を設定する **<VirtualHost>** ディレクティブに追加します。

```
<Directory "/var/www/html/Example/">
  SSLVerifyClient require
</Directory>
```

SSLVerifyClient require の設定では、**/var/www/html/Example/** ディレクトリーのコンテンツにクライアントがアクセスする前に、サーバーがクライアント証明書を正常に検証する必要があることを定義します。

2. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. **curl** ユーティリティーを使用して、クライアント認証なしで **https://example.com/Example/** URL にアクセスします。

```
$ curl https://example.com/Example/
curl: (56) OpenSSL SSL_read: error:1409445C:SSL routines:ssl3_read_bytes:tlsv13 alert certificate required, errno 0
```

このエラーは、Web サーバーにクライアント証明書認証が必要であることを示しています。

2. クライアントの秘密鍵と証明書、および CA 証明書を **curl** に指定して、クライアント認証で同じ URL にアクセスします。

```
$ curl --cacert ca.crt --key client.key --cert client.crt https://example.com/Example/
```

要求に成功すると、**curl** は **/var/www/html/Example/** ディレクトリーに保存されている **index.html** ファイルを表示します。

関連情報

- mod_ssl 設定

1.11. MODSECURITY を使用した WEB サーバー上の WEB アプリケーションの保護

ModSecurity は、Apache、Nginx、IIS などのさまざまな Web サーバーでサポートされているオープンソースの Web アプリケーションファイアウォール (WAF) であり、Web アプリケーションのセキュリティーリスクを軽減します。ModSecurity は、サーバーを設定するためのカスタマイズ可能なルールセットを提供します。

mod_security-crs パッケージには、クロス Web サイトスクリプティング、不正なユーザーエージェント、SQL インジェクション、トロイの木馬、セッションハイジャック、およびその他の不正使用に対するルールを含むコアルールセット (CRS) が含まれています。

1.11.1. Apache 用 ModSecurity Web ベースアプリケーションファイアウォールのデプロイ

ModSecurity をデプロイして、Web サーバー上で Web ベースアプリケーションの実行に関連するリスクを軽減するには、Apache HTTP サーバー用の **mod_security** および **mod_security_crs** パッケージをインストールします。**mod_security_crs** パッケージは、ModSecurity Web ベースのアプリケーションファイアウォール (WAF) モジュールのコアルールセット (CRS) を提供します。

手順

1. **mod_security**、**mod_security_crs**、および **httpd** パッケージをインストールします。

```
# yum install -y mod_security mod_security_crs httpd
```

2. **httpd** サーバーを起動します。

```
# systemctl restart httpd
```

検証

1. ModSecurity Web ベースアプリケーションファイアウォールが Apache HTTPサーバーで有効になっていることを確認します。

```
# httpd -M | grep security
 security2_module (shared)
```

2. **/etc/httpd/modsecurity.d/activated_rules/** ディレクトリーに **mod_security_crs** によって提供されるルールが含まれていることを確認します。

```
# ls /etc/httpd/modsecurity.d/activated_rules/
...
REQUEST-921-PROTOCOL-ATTACK.conf
REQUEST-930-APPLICATION-ATTACK-LFI.conf
...
```

関連情報

- Red Hat JBoss Core Services ModSecurity ガイド
- An introduction to web application firewalls for Linux sysadmins

1.11.2. ModSecurity へのカスタムルールの追加

ModSecurity コアルールセット (CRS) に含まれるルールがシナリオに適合せず、追加の攻撃の可能性を

防ぎたい場合は、カスタムルールを ModSecurity Web ベースアプリケーションファイアウォールで使用されるルールセットに追加できます。次の例は、単純なルールの追加を示しています。より複雑なルールを作成するには、ModSecurity Wiki Web サイトのリファレンスマニュアルを参照してください。

前提条件

- ModSecurity for Apache がインストールされ、有効になっている。

手順

1. 任意のテキストエディターで **/etc/httpd/conf.d/mod_security.conf** ファイルを開きます。以下はその例です。

```
# vi /etc/httpd/conf.d/mod_security.conf
```

2. **SecRuleEngine On** で始まる行の後に、次のサンプルルールを追加します。

```
SecRule ARGS:data "@contains evil" "deny,status:403,msg:'param data contains evil data',id:1"
```

前のルールでは、**data** パラメーターに **evil** の文字列が含まれている場合、ユーザーによるリソースの使用を禁止しています。

3. 変更を保存し、エディターを終了します。
4. **httpd** サーバーを再起動します。

```
# systemctl restart httpd
```

検証

1. **test.html** ページを作成します。

```
# echo "mod_security test" > /var/www/html/test.html
```

2. **httpd** サーバーを再起動します。

```
# systemctl restart httpd
```

3. HTTP リクエストの **GET** 変数に悪意のあるデータが含まれない **test.html** をリクエストします。

```
$ curl http://localhost/test.html?data=good

mod_security test
```

4. HTTP リクエストの **GET** 変数に悪意のあるデータが含まれる **test.html** をリクエストします。

```
$ curl localhost/test.html?data=xxxevilxxx

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
```

```
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You do not have permission to access this resource.</p>
</body></html>
```

5. **/var/log/httpd/error_log** ファイルを確認し、**param data containing an evil data** メッセージでアクセスを拒否するログエントリーを見つけます。

```
[Wed May 25 08:01:31.036297 2022] [:error] [pid 5839:tid 139874434791168] [client
::1:45658] [client ::1] ModSecurity: Access denied with code 403 (phase 2). String match
"evil" at ARGS:data. [file "/etc/httpd/conf.d/mod_security.conf"] [line "4"] [id "1"] [msg "param
data contains evil data"] [hostname "localhost"] [uri "/test.html"] [unique_id
"Yo4amwIdsBG3yZqSzh2GuwAAAIY"]
```

関連情報

- ModSecurity Wiki

1.12. APACHE HTTP SERVER のマニュアルのインストール

Apache HTTP Server のマニュアルをインストールできます。このマニュアルには、以下のような詳細なドキュメントが含まれます。

- 設定パラメーターおよびディレクティブ
- パフォーマンスチューニング
- 認証設定
- モジュール
- コンテンツのキャッシュ
- セキュリティーに関するヒント
- TLS 暗号化の設定

マニュアルをインストールした後は、Web ブラウザーを使用して表示できます。

前提条件

- Apache HTTP Server がインストールされ、実行している。

手順

1. **httpd-manual** パッケージをインストールします。

```
# yum install httpd-manual
```

2. 必要に応じて、デフォルトでは、Apache HTTP Server に接続するすべてのクライアントはマニュアルを表示できます。**192.0.2.0/24** サブネットなど、特定の IP 範囲へのアクセスを制限するには、**/etc/httpd/conf.d/manual.conf** ファイルを編集し、**Require ip 192.0.2.0/24** 設定を **<Directory "/usr/share/httpd/manual">** ディレクティブに追加します。

```
<Directory "/usr/share/httpd/manual">
...
    Require ip 192.0.2.0/24
...
</Directory>
```

3. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

検証手順

1. Apache HTTP Server のマニュアルを表示するには、Web ブラウザーで **http://host_name_or_IP_address/manual/** に接続します。

1.13. APACHE モジュールの操作

httpd サービスはモジュラーアプリケーションであり、多数の **動的共有オブジェクト** (DSO) で拡張できます。**動的共有オブジェクト** は、必要に応じて実行時に動的にロードまたはアンロードできるモジュールです。これらのモジュールは **/usr/lib64/httpd/modules/** ディレクトリーにあります。

1.13.1. DSO モジュールのロード

管理者は、サーバーがロードするモジュールを設定することにより、サーバーに含める機能を選択できます。特定の DSO モジュールを読み込むには、**LoadModule** ディレクティブを使用します。別のパッケージが提供するモジュールは、多くの場合、**/etc/httpd/conf.modules.d/** ディレクトリーに独自の設定ファイルがあることに注意してください。

前提条件

- **httpd** パッケージをインストールしている。

手順

1. **/etc/httpd/conf.modules.d/** ディレクトリーの設定ファイルでモジュール名を検索します。

```
# grep mod_ssl.so /etc/httpd/conf.modules.d/*
```

2. モジュール名が見つかった設定ファイルを編集し、モジュールの **LoadModule** ディレクティブをコメント解除します。

```
LoadModule ssl_module modules/mod_ssl.so
```

3. RHEL パッケージがモジュールを提供していないなどの理由でモジュールが見つからなかった場合は、次のディレクティブを使用して **/etc/httpd/conf.modules.d/30-example.conf** などの設定ファイルを作成します。

```
LoadModule ssl_module modules/<custom_module>.so
```

4. **httpd** サービスを再起動します。

```
# systemctl restart httpd
```

1.13.2. カスタム Apache モジュールのコンパイル

独自のモジュールを作成し、モジュールのコンパイルに必要なインクルードファイル、ヘッダーファイル、および **APache eXtenSion** (**apxs**) ユーティリティーを含む **httpd-devel** パッケージを使用してビルドできます。

前提条件

- **httpd-devel** パッケージがインストールされている。

手順

- 次のコマンドでカスタムモジュールをビルドします。

```
# apxs -i -a -c module_name.c
```

検証手順

- DSO モジュールのロード で説明されている方法でモジュールをロードします。

1.14. APACHE WEB SERVER 設定で秘密鍵と証明書を使用できるように NSS データベースからの証明書のエクスポート

RHEL 8 では Apache Web Server に **mod_nss** モジュールが提供されなくなります。Red Hat は **mod_ssl** モジュールの使用を推奨します。たとえば、RHEL 7 から RHEL 8 へ Web サーバーを移行したなどして、秘密鍵と証明書を Network Security Services (NSS) データベースに保存する場合は、以下の手順に従って、Privacy Enhanced Mail (PEM) 形式の鍵および証明書を抽出します。次に Apache HTTP サーバーでの TLS 暗号化の設定 で説明されているとおり、**mod_ssl** 設定でファイルを使用できます。

この手順では、NSS データベースが **/etc/httpd/alias/** に保存され、エクスポートした秘密鍵と証明書を **/etc/pki/tls/** ディレクトリーに保存することを前提としています。

前提条件

- 秘密鍵、証明書、および認証局 (CA) の証明書は NSS データベースに保存されます。

手順

1. NSS データベースの証明書をリスト表示します。

```
# certutil -d /etc/httpd/alias/ -L
Certificate Nickname          Trust Attributes
                              SSL,S/MIME,JAR/XPI

Example CA                    C,,
Example Server Certificate    u,u,u
```

   次の手順では、証明書のニックネームが必要です。

2. 秘密鍵を抽出するには、鍵を PKCS #12 ファイルに一時的にエクスポートする必要があります。

a. 秘密鍵に関連付けられた証明書のニックネームを使用して、鍵を PKCS #12 ファイルにエクスポートします。

```
# pk12util -o /etc/pki/tls/private/export.p12 -d /etc/httpd/alias/ -n "Example Server Certificate"
Enter password for PKCS12 file: password
Re-enter password: password
pk12util: PKCS12 EXPORT SUCCESSFUL
```

PKCS #12 ファイルにパスワードを設定する必要があります。次の手順では、このパスワードが必要です。

b. PKCS #12 ファイルから秘密鍵をエクスポートします。

```
# openssl pkcs12 -in /etc/pki/tls/private/export.p12 -out /etc/pki/tls/private/server.key -nocerts -nodes
Enter Import Password: password
MAC verified OK
```

c. PKCS #12 の一時ファイルを削除します。

```
# rm /etc/pki/tls/private/export.p12
```

3. **/etc/pki/tls/private/server.key** にパーミッションを設定し、**root** ユーザーのみがこのファイルにアクセスできるようにします。

```
# chown root:root /etc/pki/tls/private/server.key
# chmod 0600 /etc/pki/tls/private/server.key
```

4. NSS データベースのサーバー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/httpd/alias/ -L -n "Example Server Certificate" -a -o /etc/pki/tls/certs/server.crt
```

5. **/etc/pki/tls/certs/server.crt** にパーミッションを設定し、**root** ユーザーのみがこのファイルにアクセスできるようにします。

```
# chown root:root /etc/pki/tls/certs/server.crt
# chmod 0600 /etc/pki/tls/certs/server.crt
```

6. 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/httpd/alias/ -L -n "Example CA" -a -o /etc/pki/tls/certs/ca.crt
```

7. Apache HTTP サーバーでの TLS 暗号化の設定 に従い、Apache Web サーバーを設定します。

- **SSLCertificateKeyFile** パラメーターを **/etc/pki/tls/private/server.key** に設定します。
- **SSLCertificateFile** パラメーターを **/etc/pki/tls/certs/server.crt** に設定します。
- **SSLCACertificateFile** パラメーターを **/etc/pki/tls/certs/ca.crt** に設定します。

関連情報

- **certutil(1)** man ページ
- **pk12util(1)** の man ページ
- **pkcs12(1ssl)** の man ページ

1.15. 関連情報

- **httpd(8)**
- **httpd.service(8)**
- **httpd.conf(5)**
- **apachectl(8)**
- GSS-Proxy を使用した Apache httpd の操作。
- PKCS #11 で暗号化ハードウェアを使用するようにアプリケーションを設定

第2章 NGINX の設定および設定

NGINX は、次のように使用できる高パフォーマンスなモジュラーサーバーです。

- Web サーバー
- リバースプロキシー
- ロードバランサー

本セクションでは、このシナリオで NGINX を行う方法を説明します。

2.1. NGINX のインストールおよび準備

Red Hat は、アプリケーションストリームを使用して NGINX の異なるバージョンを提供します。以下を実行できます。

- ストリームを選択し、NGINX をインストールします。
- ファイアウォールで必要なポートを開きます。
- **nginx** サービスの有効化および開始

デフォルト設定を使用すると、NGINX はポート **80** の Web サーバーとして実行され、**/usr/share/nginx/html/** ディレクトリーからコンテンツを提供します。

前提条件

- RHEL 8 がインストールされている。
- ホストが Red Hat カスタマーポータルにサブスクライブしている。
- **firewalld** サービスが有効化され、開始されている。

手順

1. 利用可能な NGINX モジュールストリームを表示します。

```
# yum module list nginx
Red Hat Enterprise Linux 8 for x86_64 - AppStream (RPMs)
Name       Stream       Profiles       Summary
nginx      1.14 [d]     common [d]     nginx webserver
nginx      1.16         common [d]     nginx webserver
...

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
```

2. デフォルト以外のストリームをインストールする場合は、そのストリームを選択します。

```
# yum module enable nginx:stream_version
```

3. **nginx** パッケージをインストールします。

```
# yum install nginx
```

4. NGINX がファイアウォールでサービスを提供するポートを開きます。たとえば、**firewalld** で HTTP (ポート 80) および HTTPS (ポート 443) のデフォルトポートを開くには、次のコマンドを実行します。

```
# firewall-cmd --permanent --add-port={80/tcp,443/tcp}
# firewall-cmd --reload
```

5. **nginx** サービスがシステムの起動時に自動的に起動するようにします。

```
# systemctl enable nginx
```

6. 必要に応じて、**nginx** サービスを起動します。

```
# systemctl start nginx
```

デフォルト設定を使用しない場合は、この手順を省略し、サービスを起動する前に NGINX を適切に設定します。

検証手順

1. **yum** ユーティリティーを使用して、**nginx** パッケージがインストールされていることを確認します。

```
# yum list installed nginx
Installed Packages
nginx.x86_64    1:1.14.1-9.module+el8.0.0+4108+af250afe    @rhel-8-for-x86_64-appstream-rpms
```

2. NGINX がサービスを提供するポートが firewalld で開いていることを確認します。

```
# firewall-cmd --list-ports
80/tcp 443/tcp
```

3. **nginx** サービスが有効になっていることを確認します。

```
# systemctl is-enabled nginx
enabled
```

関連情報

- Subscription Manager の詳細は、Subscription Manager の使用および設定 ガイドを参照してください。
- アプリケーションストリーム、モジュール、およびインストールパッケージの詳細は、ユーザー空間コンポーネントのインストール、管理、および削除 を参照してください。
- ファイアウォールの設定に関する詳細は、ネットワークのセキュリティー保護 を参照してください。

2.2. ドメインごとに異なるコンテンツを提供する WEB サーバーとしての NGINX の設定

デフォルトでは、NGINX は Web サーバーとして機能し、サーバーの IP アドレスに関連付けられた全ドメイン名のクライアントに、同じコンテンツを提供します。この手順では、NGINX を設定する方法を説明します。

- **/var/www/example.com/** ディレクトリーのコンテンツで、**example.com** ドメインに対するリクエストに対応する。
- **/var/www/example.net/** ディレクトリーのコンテンツで、**example.net** ドメインに対するリクエストに対応する。
- その他の全リクエスト (たとえば、サーバーの IP アドレスまたはサーバーの IP アドレスに関連付けられたその他のドメイン) に **/usr/share/nginx/html/** ディレクトリーのコンテンツを指定します。

前提条件

- NGINX がインストールされている
- クライアントおよび Web サーバーは、**example.com** および **example.net** ドメインを Web サーバーの IP アドレスに解決します。
  これらのエントリーは DNS サーバーに手動で追加する必要がある点に注意してください。

手順

1. **/etc/nginx/nginx.conf** ファイルを編集します。
   a. デフォルトでは、**/etc/nginx/nginx.conf** ファイルには catch-all 設定がすでに含まれています。設定からこの部分を削除した場合は、以下の **server** ブロックを **/etc/nginx/nginx.conf** ファイルの **http** ブロックに追加し直します。

```
server {
    listen       80 default_server;
    listen       [::]:80 default_server;
    server_name  _;
    root         /usr/share/nginx/html;
}
```

   これらの設定は以下を設定します。

   - **listen** ディレクティブは、サービスがリッスンする IP アドレスとポートを定義します。この場合、NGINX は IPv4 と IPv6 の両方のアドレスのポート **80** でリッスンします。**default_server** パラメーターは、NGINX がこの **server** ブロックを IP アドレスとポートに一致するリクエストのデフォルトとして使用していることを示します。
   - **server_name** パラメーターは、この **server** ブロックに対応するホスト名を定義します。**server_name** を _ に設定すると、この **server** ブロックのホスト名を受け入れるように NGINX を設定します。
   - **root** ディレクティブは、この **server** ブロックの Web コンテンツへのパスを設定します。

   b. **example.com** ドメインの同様の **server** ブロックを **http** ブロックに追加します。

```
server {
    server_name  example.com;
    root         /var/www/example.com/;
```

```
    access_log   /var/log/nginx/example.com/access.log;
    error_log    /var/log/nginx/example.com/error.log;
}
```

- **access_log** ディレクティブは、このドメインに別のアクセスログファイルを定義します。
- **error_log** ディレクティブは、このドメインに別のエラーログファイルを定義します。

c. **example.net** ドメインの同様の **server** ブロックを **http** ブロックに追加します。

```
server {
    server_name  example.net;
    root         /var/www/example.net/;
    access_log   /var/log/nginx/example.net/access.log;
    error_log    /var/log/nginx/example.net/error.log;
}
```

2. 両方のドメインのルートディレクトリーを作成します。

```
# mkdir -p /var/www/example.com/
# mkdir -p /var/www/example.net/
```

3. 両方のルートディレクトリーに **httpd_sys_content_t** コンテキストを設定します。

```
# semanage fcontext -a -t httpd_sys_content_t "/var/www/example.com(/.*)?"
# restorecon -Rv /var/www/example.com/
# semanage fcontext -a -t httpd_sys_content_t "/var/www/example.net(/.\*)?"
# restorecon -Rv /var/www/example.net/
```

これらのコマンドは、**/var/www/example.com/** ディレクトリーおよび **/var/www/example.net/** ディレクトリーに **httpd_sys_content_t** コンテキストを設定します。

policycoreutils-python-utils パッケージをインストールして **restorecon** コマンドを実行する必要があります。

4. 両方のドメインのログディレクトリーを作成します。

```
# mkdir /var/log/nginx/example.com/
# mkdir /var/log/nginx/example.net/
```

5. **nginx** サービスを再起動します。

```
# systemctl restart nginx
```

検証手順

1. 仮想ホストのドキュメントルートごとに異なるサンプルファイルを作成します。

```
# echo "Content for example.com" > /var/www/example.com/index.html
# echo "Content for example.net" > /var/www/example.net/index.html
# echo "Catch All content" > /usr/share/nginx/html/index.html
```

2. ブラウザーを使用して **http://example.com** に接続します。Web サーバーは、**/var/www/example.com/index.html** ファイルからのサンプルコンテンツを表示します。

3. ブラウザーを使用して **http://example.net** に接続します。Web サーバーは、**/var/www/example.net/index.html** ファイルからのサンプルコンテンツを表示します。

4. ブラウザーを使用して **http://IP_address_of_the_server** に接続します。Web サーバーは、**/usr/share/nginx/html/index.html** ファイルからのサンプルコンテンツを表示します。

2.3. NGINX WEB サーバーへの TLS 暗号化の追加

example.com ドメインの NGINX Web サーバーで TLS 暗号化を有効にすることができます。

前提条件

- NGINX がインストールされている。
- 秘密鍵が **/etc/pki/tls/private/example.com.key** ファイルに保存されている。秘密鍵および証明書署名要求 (CSR) を作成する方法と、認証局 (CA) からの証明書を要求する方法は、CA のドキュメントを参照してください。
- TLS 証明書は **/etc/pki/tls/certs/example.com.crt** ファイルに保存されます。別のパスを使用する場合は、この手順で対応する手順を調整します。
- CA 証明書がサーバーの TLS 証明書ファイルに追加されている。
- クライアントおよび Web サーバーは、サーバーのホスト名を Web サーバーの IP アドレスに対して解決します。
- ポート **443** が、ローカルのファイアウォールで開放されている。

手順

1. **/etc/nginx/nginx.conf** ファイルを編集し、設定の **http** ブロックに以下の **server** ブロックを追加します。

```
server {
    listen              443 ssl;
    server_name         example.com;
    root                /usr/share/nginx/html;
    ssl_certificate     /etc/pki/tls/certs/example.com.crt;
    ssl_certificate_key /etc/pki/tls/private/example.com.key;
}
```

2. セキュリティー上の理由から、**root** ユーザーのみが秘密鍵ファイルにアクセスできるように設定します。

```
# chown root:root /etc/pki/tls/private/example.com.key
# chmod 600 /etc/pki/tls/private/example.com.key
```

警告

秘密鍵に権限のないユーザーがアクセスした場合は、証明書を取り消し、新しい秘密鍵を作成し、新しい証明書を要求します。そうでない場合は、TLS 接続が安全ではなくなります。

3. **nginx** サービスを再起動します。

```
# systemctl restart nginx
```

検証手順

- ブラウザーを使用して、**https://example.com** に接続します。

関連情報

- RHEL 9 における TLS のセキュリティー上の検討事項

2.4. HTTP トラフィックのリバースプロキシーとしての NGINX の設定

NGINX Web サーバーは、HTTP トラフィックのリバースプロキシーとして機能するように設定できます。たとえば、この機能を使用すると、リモートサーバーの特定のサブディレクトリーに要求を転送できます。クライアント側からは、クライアントはアクセス先のホストからコンテンツを読み込みます。ただし、NGINX は実際のコンテンツをリモートサーバーから読み込み、クライアントに転送します。

この手順では、Web サーバーの **/example** ディレクトリーへのトラフィックを、URL **https://example.com** に転送する方法を説明します。

前提条件

- NGINX は、NGINX のインストールと準備 の説明に従ってインストールされます。
- 必要に応じて、TLS 暗号化がリバースプロキシーで有効になっている。

手順

1. **/etc/nginx/nginx.conf** ファイルを編集し、リバースプロキシーを提供する **server** ブロックに以下の設定を追加します。

```
location /example {
    proxy_pass https://example.com;
}
```

location ブロックでは、NGINX が **/example** ディレクトリー内の全要求を **https://example.com** に渡すことを定義します。

2. SELinux ブール値パラメーター **httpd_can_network_connect** を **1** に設定して、SELinux が NGINX がトラフィックを転送できるように設定します。

```
# setsebool -P httpd_can_network_connect 1
```

3. **nginx** サービスを再起動します。

```
# systemctl restart nginx
```

検証手順

- ブラウザーを使用して **http://host_name/example** に接続すると、**https://example.com** の内容が表示されます。

2.5. NGINX の HTTP ロードバランサーとしての設定

NGINX リバースプロキシー機能を使用してトラフィックを負荷分散できます。この手順では、HTTP ロードバランサーとして NGINX を設定して、アクティブな接続数が最も少ないサーバーがどれかを基にして、要求を異なるサーバーに送信する方法を説明します。どちらのサーバーも利用できない場合には、この手順でフォールバックを目的とした 3 番目のホストも定義します。

前提条件

- NGINX は、NGINX のインストールと準備 の説明に従ってインストールされます。

手順

1. **/etc/nginx/nginx.conf** ファイルを編集し、以下の設定を追加します。

```
http {
    upstream backend {
        least_conn;
        server server1.example.com;
        server server2.example.com;
        server server3.example.com backup;
    }

    server {
        location / {
            proxy_pass http://backend;
        }
    }
}
```

backend という名前のホストグループの **least_conn** ディレクティブは、アクティブな接続数が最も少ないサーバーがどれかを基にして、NGINX が要求を **server1.example.com** または **server2.example.com** に送信することを定義します。NGINX は、他の 2 つのホストが利用できない場合は、**server3.example.com** のみをバックアップとして使用します。

proxy_pass ディレクティブを **http://backend** に設定すると、NGINX はリバースプロキシーとして機能し、**backend** ホストグループを使用して、このグループの設定に基づいて要求を配信します。

least_conn 負荷分散メソッドの代わりに、以下を指定することができます。

- ラウンドロビンを使用し、サーバー全体で要求を均等に分散する方法はありません。

- **ip_hash**: クライアントの IPv4 アドレスのオクテットの内、最初の 3 つ、または IPv6 アドレス全体から計算されたハッシュに基づいて、あるクライアントアドレスから同じサーバーに要求を送信します。
- **hash**: ユーザー定義のキーに基づいてサーバーを判断します。これは、文字列、変数、または両方の組み合わせになります。**consistent** パラメーターは、ユーザー定義のハッシュ化された鍵の値に基づいて、NGINX がすべてのサーバーに要求を分散するように設定します。
- **random**: 無作為に選択されたサーバーに要求を送信します。

2. **nginx** サービスを再起動します。

```
# systemctl restart nginx
```

2.6. 関連情報

- 公式の NGINX のドキュメントについては、https://nginx.org/en/docs/ を参照してください。Red Hat はこのドキュメントを管理しておらず、インストールした NGINX バージョンで機能しない可能性があることに注意してください。
- PKCS #11 で暗号化ハードウェアを使用するようにアプリケーションを設定

第3章 SAMBA をサーバーとして使用

Samba は、Red Hat Enterprise Linux にサーバーメッセージブロック (SMB) プロトコルを実装します。SMB プロトコルは、ファイル共有、共有プリンターなど、サーバーのリソースにアクセスするのに使用されます。また、Samba は、Microsoft Windows が使用する分散コンピューティング環境のリモートプロシージャコール (DCE RPC) のプロトコルを実装します。

Samba は以下のように実行できます。

- Active Directory (AD) または NT4 ドメインメンバー
- スタンドアロンサーバー
- NT4 プライマリードメインコントローラー (PDC) またはバックアップドメインコントローラー (BDC)

注記

Red Hat は、NT4 ドメインに対応する Windows バージョンの既存のインストールでのみ、PDC モードおよび BDC モードをサポートします。Red Hat では、新しい Samba NT4 ドメインを設定しないことを推奨します。これは、Windows 7 および Windows Server 2008 R2 以降の Microsoft オペレーティングシステムが NT4 ドメインに対応していないためです。

Red Hat は、Samba を AD ドメインコントローラー (DC) として実行することはサポートしていません。

インストールモードとは関係なく、必要に応じてディレクトリーやプリンターを共有できます。これにより、Samba がファイルサーバーおよびプリントサーバーとして機能できるようになります。

3.1. さまざまな SAMBA サービスおよびモードについて

samba パッケージは複数のサービスを提供します。環境と設定するシナリオに応じて、これらのサービスが 1 つ以上必要となり、Samba をさまざまなモードで設定します。

3.1.1. Samba サービス

Samba は以下のサービスを提供します。

smbd

このサービスは、SMB プロトコルを使用してファイル共有およびプリントサービスを提供します。また、サービスは、リソースのロックと、接続ユーザーの認証を担当します。ドメインメンバーを認証するには、**smbd** に **winbindd** が必要です。**smb systemd** サービスが起動し、**smbd** デーモンが停止します。
smbd サービスを使用するには、**samba** パッケージをインストールします。

nmbd

このサービスは、NetBIOS over IPv4 プロトコルを使用してホスト名および IP 解決を提供します。名前解決に加え、**nmbd** サービスで SMB ネットワークを参照して、ドメイン、作業グループ、ホスト、ファイル共有、およびプリンターを探すことができます。このため、サービスはこの情報をブロードキャストクライアントに直接報告するか、ローカルまたはマスターのブラウザーに転送します。**nmb systemd** サービスは、**nmbd** デーモンを起動し、停止します。

最近の SMB ネットワークは、クライアントおよび IP アドレスの解決に DNS を使用することに注意してください。Kerberos の場合は、稼働中の DNS 設定が必要です。

nmbd サービスを使用するには、**samba** パッケージをインストールします。

winbindd

このサービスは、ローカルシステムの AD または NT4 のドメインユーザーおよびグループを使用する Name Service Switch (NSS) のインターフェイスを提供します。これにより、たとえばドメインユーザーを、Samba サーバーにホストされるサービスや他のローカルサービスに認証できます。**winbind systemd** サービスは、**winbindd** デーモンを開始および停止します。
Samba をドメインメンバーとして設定する場合は、**smbd** サービスの前に **winbindd** を起動する必要があります。そうしないと、ドメインユーザーおよびグループはローカルシステムで使用できなくなります。

winbindd サービスを使用するには、**samba-winbind** パッケージをインストールします。

重要

Red Hat は、ドメインユーザーおよびグループをローカルシステムに提供するために、Samba を、**winbindd** サービスを使用するサーバーとして実行することのみをサポートします。Windows アクセス制御リスト (ACL) のサポート、NT LAN Manager (NTLM) のフォールバックがないなど、特定の制限により、SSSD に対応しません。

3.1.2. Samba セキュリティーサービス

/etc/samba/smb.conf ファイルの **[global]** セクションの **security** パラメーターは、Samba がサービスに接続しているユーザーを認証する方法を管理します。Samba をインストールするモードに応じて、パラメーターは異なる値に設定する必要があります。

AD ドメインメンバーに、security = ads を設定する。

このモードでは、Samba は Kerberos を使用して AD ユーザーを認証します。
Samba をドメインメンバーとして設定する方法の詳細については、Samba を AD ドメインメンバーサーバーとして設定 を参照してください。

スタンドアロンサーバーで、security = user を設定する。

このモードでは、Samba がローカルデータベースを使用して接続ユーザーを認証します。
Samba をスタンドアロンサーバーとしてセットアップする方法の詳細については、スタンドアロンサーバーとしての Samba の設定 を参照してください。

NT4 PDC または BDC に security = user を設定する。

Samba は、このモードでは、ユーザーをローカルまたは LDAP データベースに認証します。

NT4 ドメインメンバーで、security = domain を設定する。

Samba は、このモードでは、NT4 PDC または BDC にユーザーを接続する認証を行います。このモードは、AD ドメインメンバーには使用できません。
Samba をドメインメンバーとして設定する方法の詳細については、Samba を AD ドメインメンバーサーバーとして設定 を参照してください。

関連情報

- **smb.conf(5)** man ページの **security** パラメーター

3.1.3. Samba サービスおよび Samba クライアントユーティリティーが設定を読み込み、再読み込みするシナリオ

以下は、Samba サービスおよびユーティリティーによる設定の読み込み、再読み込み時について説明します。

- Samba サービスは、設定を再読み込みする時:
  - 3 分ごとに自動更新
  - 手動要求の場合に **smbcontrol all reload-config** コマンドを実行するとします。
- Samba クライアントユーティリティーは、起動時にのみ設定を読み取ります。

security などの特定のパラメーターの適用には、**smb** サービスの再起動が必要です。再読み込みだけでは十分ではないことに注意してください。

関連情報

- **smb.conf(5)** man ページの **How configuration changes are apply** セクション
- **smbd(8)**、**nmbd(8)**、および **winbindd(8)** man ページ

3.1.4. 安全な方法での Samba 設定の編集

Samba サービスは、3 分ごとに設定を自動的に再読み込みします。**testparm** ユーティリティーでの設定の検証前にサービスが変更を再読み込みしないように、安全な方法で Samba 設定を編集できます。

前提条件

- Samba がインストールされている。

手順

1. **/etc/samba/smb.conf** ファイルのコピーを作成します。

```
# cp /etc/samba/smb.conf /etc/samba/samba.conf.copy
```

2. コピーして作成したファイルを編集し、必要な変更を加えます。
3. **/etc/samba/samba.conf.copy** ファイルの設定を確認します。

```
# testparm -s /etc/samba/samba.conf.copy
```

testparm がエラーを報告した場合は、修正してもう一度コマンドを実行します。

4. **/etc/samba/smb.conf** ファイルを新しい設定に上書きします。

```
# mv /etc/samba/samba.conf.copy /etc/samba/smb.conf
```

5. Samba サービスが設定を自動的に再読み込みするか、手動で設定を再読み込みするまで待ちます。

```
# smbcontrol all reload-config
```

関連情報

- Samba サービスおよび Samba クライアントユーティリティーが設定を読み込み、再読み込みするシナリオ

3.2. TESTPARM ユーティリティーを使用した SMB.CONF ファイルの検証

testparm ユーティリティーは、**/etc/samba/smb.conf** ファイルの Samba 設定が正しいことを確認します。このユーティリティーは、無効なパラメーターおよび値を検出しますが、ID マッピングなどの間違った設定も検出します。**testparm** が問題を報告しないと、Samba サービスは **/etc/samba/smb.conf** ファイルを正常に読み込みます。**testparm** は、設定されたサービスが利用可能であること、または期待通りに機能するかを確認できないことに注意してください。

重要

Red Hat では、このファイルの変更後に毎回 **testparm** を使用して、**/etc/samba/smb.conf** ファイルを検証することが推奨されます。

前提条件

- Samba をインストールしている。
- **/etc/samba/smb.conf** ファイルが存在する。

手順

1. **root** ユーザーで **testparm** ユーティリティーを実行します。

```
# testparm
Load smb config files from /etc/samba/smb.conf
rlimit_max: increasing rlimit_max (1024) to minimum Windows limit (16384)
Unknown parameter encountered: "log levell"
Processing section "[example_share]"
Loaded services file OK.
ERROR: The idmap range for the domain * (tdb) overlaps with the range of DOMAIN (ad)!

Server role: ROLE_DOMAIN_MEMBER

Press enter to see a dump of your service definitions

# Global parameters
[global]
 ...

[example_share]
 ...
```

   上記の出力例では、存在しないパラメーターと間違った ID マッピングの設定が報告されます。

2. **testparm** が設定内の間違ったパラメーター、値、またはその他のエラーを報告する場合は、問題を修正してから再度ユーティリティーを実行してください。

3.3. SAMBA をスタンドアロンサーバーとして設定

Samba は、ドメインのメンバーではないサーバーとして設定できます。このインストールモードでは、Samba はユーザーを中央 DC ではなくローカルデータベースに認証します。また、ゲストアクセスを有効にして、ユーザーが、認証なしで 1 つまたは複数のサービスに接続できるようにすることもできます。

3.3.1. スタンドアロンサーバーのサーバー設定の設定

Samba スタンドアロンサーバーのサーバー設定を設定できます。

手順

1. **samba** パッケージをインストールします。

```
# yum install samba
```

2. **/etc/samba/smb.conf** ファイルを編集して、以下のパラメーターを設定します。

```
[global]
 workgroup = Example-WG
 netbios name = Server
 security = user

 log file = /var/log/samba/%m.log
 log level = 1
```

この設定では、**Example-WG** ワークグループに、スタンドアロンサーバー (**Server**) を定義します。また、この設定により最小レベル (**1**) でのログ記録が可能になり、ログファイルは **/var/log/samba/** ディレクトリーに保存されます。Samba は、**log file** パラメーターの **%m** マクロを、接続しているクライアントの NetBIOS 名までデプロイメントします。これにより、クライアントごとに個別のログファイルが有効になります。

3. オプションで、ファイルまたはプリンターの共有を設定します。参照:
   - POSIX ACL で共有の設定
   - Windows ACL で共有の設定
   - プリントサーバーとしての Samba の設定
4. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

5. 認証が必要な共有を設定する場合は、ユーザーアカウントを作成します。
   詳細は ローカルユーザーアカウントの作成および有効化 を参照してください。
6. **firewall-cmd** ユーティリティーを使用して必要なポートを開き、ファイアウォール設定を再読み込みします。

```
# firewall-cmd --permanent --add-service=samba
# firewall-cmd --reload
```

7. **smb** サービスを有効にして起動します。

```
# systemctl enable --now smb
```

関連情報

- **smb.conf(5)** man ページ

3.3.2. ローカルユーザーアカウントの作成および有効化

共有への接続時にユーザーが認証を行えるようにするには、オペレーティングシステムと Samba データベースの両方で Samba ホストにアカウントを作成する必要があります。Samba では、ファイルシステムオブジェクトでアクセス制御リスト (ACL) を検証するオペレーティングシステムアカウントと、接続ユーザーの認証を行う Samba アカウントが必要です。

passdb backend = tdbsam のデフォルト設定を使用すると、Samba はユーザーアカウントを **/var/lib/samba/private/passdb.tdb** データベースに保存します。

example という名前のローカル Samba ユーザーを作成できます。

前提条件

- Samba が、スタンドアロンサーバーとしてインストールされている。

手順

1. オペレーティングシステムアカウントを作成します。

```
# useradd -M -s /sbin/nologin example
```

このコマンドは、ホームディレクトリーを作成せずに、**example** アカウントを追加します。アカウントが Samba への認証のみに使用される場合は、**/sbin/nologin** コマンドをシェルとして割り当て、アカウントがローカルでログインしないようにします。

2. オペレーティングシステムのアカウントにパスワードを設定して、これを有効にします。

```
# passwd example
Enter new UNIX password: password
Retype new UNIX password: password
passwd: password updated successfully
```

Samba は、オペレーティングシステムのアカウントに設定されたパスワードを使用して認証を行いません。ただし、アカウントを有効にするには、パスワードを設定する必要があります。アカウントが無効になると、そのユーザーが接続した時に Samba がアクセスを拒否します。

3. Samba データベースにユーザーを追加し、そのアカウントにパスワードを設定します。

```
# smbpasswd -a example
New SMB password: password
Retype new SMB password: password
Added user example.
```

このアカウントを使用して Samba 共有に接続する場合に、このパスワードを使用して認証を行います。

4. Samba アカウントを有効にします。

```
# smbpasswd -e example
Enabled user example.
```

3.4. SAMBA ID マッピングの理解および設定

Windows ドメインは、ユーザーおよびグループを一意のセキュリティー識別子 (SID) で区別します。ただし、Linux では、ユーザーおよびグループごとに一意の UID と GID が必要です。Samba をドメインメンバーとして実行する場合は、**winbindd** サービスが、ドメインユーザーおよびグループに関する情報をオペレーティングシステムに提供します。

winbindd サービスが、ユーザーおよびグループの一意の ID を Linux に提供するようにするには、**/etc/samba/smb.conf** ファイルで ID マッピングを設定する必要があります。

- ローカルデータベース (デフォルトドメイン)
- Samba サーバーがメンバーになっている AD または NT4 のドメイン
- ユーザーがこの Samba サーバーのリソースにアクセスする必要のある信頼ドメイン

Samba は、特定の設定に対して異なる ID マッピングバックエンドを提供します。最も頻繁に使用されるバックエンドは、以下の通りです。

バックエンド	ユースケース
tdb	* デフォルトドメインのみ
ad	AD ドメインのみ
rid	AD ドメインおよび NT4 ドメイン
autorid	AD、NT4、および * デフォルトのドメイン

3.4.1. Samba ID 範囲の計画

Linux の UID および GID を AD に保存するか、Samba がそれを生成するように設定するかに関係なく、各ドメイン設定には、他のドメインと重複しない一意の ID 範囲が必要です。

警告

重複する ID 範囲を設定すると、Samba が正常に機能しなくなります。

例3.1 一意の ID 範囲

以下は、デフォルト (*)、**AD-DOM**、および **TRUST-DOM** のドメインの非オーバーランディングの ID マッピング範囲を示しています。

```
[global]
...
idmap config * : backend = tdb
idmap config * : range = 10000-999999

idmap config AD-DOM:backend = rid
idmap config AD-DOM:range = 2000000-2999999

idmap config TRUST-DOM:backend = rid
idmap config TRUST-DOM:range = 4000000-4999999
```

重要

1 つのドメインに割り当てられるのは 1 つの範囲だけです。したがって、ドメイン範囲間で十分な容量を残しておきます。これにより、ドメインが拡大した場合に、後で範囲を拡張できます。

後で別の範囲をドメインに割り当てると、このユーザーおよびグループが作成したファイルおよびディレクトリーの所有権が失われます。

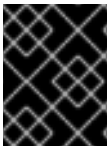
3.4.2. * デフォルトドメイン

ドメイン環境では、以下の各 ID マッピング設定を追加します。

- Samba サーバーがメンバーとなっているドメイン
- Samba サーバーにアクセスできる信頼された各ドメイン

ただし、Samba が、その他のすべてのオブジェクトに、デフォルトドメインから ID を割り当てます。これには以下が含まれます。

- ローカルの Samba ユーザーおよびグループ
- Samba の組み込みアカウントおよびグループ (**BUILTIN\Administrators** など)



重要

Samba が正常に機能できるようにするには、説明に従ってデフォルトのドメインを設定する必要があります。

割り当てられた ID を永続的に格納するには、デフォルトのドメインバックエンドを書き込み可能にする必要があります。

デフォルトドメインには、以下のいずれかのバックエンドを使用できます。

tdb

デフォルトのドメインを、**tdb** バックエンドを使用するように設定する場合は、ID 範囲を設定します。この ID 範囲には、将来作成されるオブジェクトや、定義されたドメイン ID マッピング設定には含まれないオブジェクトを追加できます。
たとえば、**/etc/samba/smb.conf** ファイルの **[global]** セクションで以下を設定します。

```
idmap config * : backend = tdb
idmap config * : range = 10000-999999
```

詳細は、TDB ID マッピングバックエンドの使用 を参照してください。

autorid

autorid バックエンドを使用するように、デフォルトのドメインを設定する場合、ドメイン用の ID マッピング設定を追加するかどうかは任意になります。
たとえば、**/etc/samba/smb.conf** ファイルの **[global]** セクションで以下を設定します。

```
idmap config * : backend = autorid
idmap config * : range = 10000-999999
```

詳細は、autorid ID マッピングバックエンドの使用 を参照してください。

3.4.3. tdb ID マッピングバックエンドの使用

winbindd サービスは、デフォルトで書き込み可能な **tdb** ID マッピングバックエンドを使用して、セキュリティー識別子 (SID)、UID、および GID のマッピングテーブルを格納します。これには、ローカルユーザー、グループ、組み込みプリンシパルが含まれます。

このバックエンドは、* デフォルトドメインにのみ使用してください。以下に例を示します。

```
idmap config * : backend = tdb
idmap config * : range = 10000-999999
```

関連情報

- * デフォルトドメイン

3.4.4. ad ID マッピングバックエンドの使用

ad ID マッピングバックエンドを使用するように Samba AD メンバーを設定できます。

ad ID マッピングバックエンドは、読み取り専用 API を実装し、AD からアカウントおよびグループの情報を読み取ります。これには、以下の利点があります。

- ユーザーとグループの全設定は、AD に集中的に保存されます。
- ユーザーおよびグループの ID は、このバックエンドを使用するすべての Samba サーバーで一貫しています。
- ID は、破損する可能性のあるローカルデータベースには保存されないため、ファイルの所有権は失われません。

注記

ad ID マッピングバックエンドは、一方向の信頼を使用する Active Directory ドメインに対応していません。一方向の信頼で Active Directory のドメインメンバーを設定する場合は、**tdb**、 **rid**、または **autorid** のいずれかの ID マッピングバックエンドを使用します。

ad バックエンドは、AD から以下の属性を読み込みます。

AD 属性名	オブジェクトの種類	マッピング先
sAMAccountName	ユーザーおよびグループ	オブジェクトのユーザー名またはグループ名
uidNumber	ユーザー	ユーザー ID (UID)
gidNumber	グループ	グループ ID (GID)
loginShell [a]	ユーザー	ユーザーのシェルのパス
unixHomeDirectory [a]	ユーザー	ユーザーのホームディレクトリーのパス
primaryGroupID [b]	ユーザー	プライマリーグループ ID

[a] **idmap config DOMAIN:unix_nss_info = yes** を設定している場合に限り、Samba がこの属性を読み込みます。

[b] **idmap config DOMAIN:unix_primary_group = yes** を設定している場合に限り、Samba がこの属性を読み込みます。

前提条件

- ユーザーおよびグループはいずれも、AD で一意の ID が設定され、ID が、**/etc/samba/smb.conf** ファイルで設定されている範囲内にある。ID が範囲外にあるオブジェクトは、Samba サーバーでは利用できません。
- ユーザーおよびグループには、AD ですべての必須属性が設定されている。必要な属性がないと、ユーザーまたはグループは Samba サーバーで使用できなくなります。必要な属性は、設定によって異なります。前提条件:
- Samba をインストールしている。
- ID マッピングを除く Samba 設定が **/etc/samba/smb.conf** ファイルにある。

手順

1. **/etc/samba/smb.conf** ファイルの **[global]** セクションを編集します。
   a. デフォルトドメイン (*) に ID マッピング設定が存在しない場合は追加します。以下に例を示します。

```
idmap config * : backend = tdb
idmap config * : range = 10000-999999
```

   b. AD ドメインの **ad** ID マッピングバックエンドを有効にします。

```
idmap config DOMAIN : backend = ad
```

   c. AD ドメインのユーザーおよびグループに割り当てられている ID の範囲を設定します。以下に例を示します。

```
idmap config DOMAIN : range = 2000000-2999999
```

重要

この範囲は、このサーバーの他のドメイン設定と重複させることはできません。また、この範囲には、今後割り当てられる ID がすべて収まる大きさを設定する必要があります。詳細は、Samba ID 範囲の計画 を参照してください。

d. Samba が AD から属性を読み取る際に RFC 2307 スキーマを使用するように設定します。

```
idmap config DOMAIN : schema_mode = rfc2307
```

e. Samba が、対応する AD 属性からログインシェルおよびユーザーホームディレクトリーのパスを読み取るようにする場合は、以下を設定します。

```
idmap config DOMAIN : unix_nss_info = yes
```

または、すべてのユーザーに適用される、ドメイン全体のホームディレクトリーのパスおよびログインシェルを統一して設定できます。以下に例を示します。

```
template shell = /bin/bash
template homedir = /home/%U
```

f. デフォルトでは、Samba は、ユーザーオブジェクトの **primaryGroupID** 属性を、Linux のユーザーのプライマリーグループとして使用します。または、代わりに **gidNumber** 属性に設定されている値を使用するように Samba を設定できます。

```
idmap config DOMAIN : unix_primary_group = yes
```

2. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

3. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

関連情報

- * デフォルトドメイン
- **smb.conf(5)** および **idmap_ad(8)** man ページ
- **smb.conf(5)** man ページの **VARIABLE SUBSTITUTIONS** セクション

3.4.5. rid ID マッピングバックエンドの使用

rid ID マッピングバックエンドを使用するように Samba ドメインメンバーを設定できます。

Samba は、Windows SID の相対識別子 (RID) を使用して、Red Hat Enterprise Linux で ID を生成できます。

注記

RID は、SID の最後の部分です。たとえば、ユーザーの SID が **S-1-5-21-5421822485-1151247151-421485315-30014** の場合、対応する RID は **30014** になります。

rid ID マッピングバックエンドは、AD ドメインおよび NT4 ドメインのアルゴリズムマッピングスキームに基づいてアカウントおよびグループの情報を計算する読み取り専用 API を実装します。バックエンドを設定する場合は、**idmap config DOMAIN : range** パラメーターで、RID の最小値および最大値を設定する必要があります。Samba は、このパラメーターで設定される RID の最小値および最大値を超えるユーザーまたはグループをマッピングしません。

重要

読み取り専用のバックエンドとして、**rid** は、**BUILTIN** グループなど、新しい ID を割り当てることができません。したがって、* デフォルトドメインにはこのバックエンドを使用しないでください。

rid バックエンドを使用した利点

- 設定された範囲内の RID があるドメインユーザーとグループはすべて、自動的にドメインメンバーで利用可能になります。
- ID、ホームディレクトリー、およびログインシェルを手動で割り当てる必要はありません。

rid バックエンドを使用した場合の短所

- すべてのドメインユーザーは、割り当てられた同じログインシェルとホームディレクトリーを取得します。ただし、変数を使用できます。
- 同じ ID 範囲設定で **rid** バックエンドを使用している Samba ドメインメンバーでは、ユーザー ID とグループ ID が同じになります。
- ドメインメンバーで個々のユーザーまたはグループを除外して、利用できないようにすることはできません。設定されている範囲外にあるユーザーとグループのみが除外されます。
- 異なるドメインのオブジェクトの RID が同じ場合は、**winbindd** サービスが ID の計算に使用する式に基づき、複数ドメインの環境で重複する ID が発生する場合があります。

前提条件

- Samba をインストールしている。
- ID マッピングを除く Samba 設定が **/etc/samba/smb.conf** ファイルにある。

手順

1. **/etc/samba/smb.conf** ファイルの **[global]** セクションを編集します。
   a. デフォルトドメイン (*) に ID マッピング設定が存在しない場合は追加します。以下に例を示します。

```
idmap config * : backend = tdb
idmap config * : range = 10000-999999
```

   b. ドメインの **rid** ID マッピングバックエンドを有効にします。

```
idmap config DOMAIN : backend = rid
```

c. 今後割り当てられるすべての RID が収まる大きさの範囲を設定します。以下に例を示します。

```
idmap config DOMAIN : range = 2000000-2999999
```

Samba は、そのドメインの RID がその範囲内にないユーザーおよびグループを無視します。

重要

この範囲は、このサーバーの他のドメイン設定と重複させることはできません。また、この範囲には、今後割り当てられる ID がすべて収まる大きさを設定する必要があります。詳細は、Samba ID 範囲の計画 を参照してください。

d. すべてのマッピングユーザーに割り当てられるシェルおよびホームディレクトリーのパスを設定します。以下に例を示します。

```
template shell = /bin/bash
template homedir = /home/%U
```

2. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

3. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

関連情報

- * デフォルトドメイン
- **smb.conf(5)** man ページの **VARIABLE SUBSTITUTIONS** セクション
- RID からのローカル ID の計算については、**idmap_rid(8)** man ページを参照してください。

3.4.6. autorid ID マッピングバックエンドの使用

autorid ID マッピングバックエンドを使用するように Samba ドメインメンバーを設定できます。

autorid バックエンドは、**rid** ID マッピングバックエンドと同様の動作をしますが、異なるドメインに対して自動的に ID を割り当てることができます。これにより、以下の状況で **autorid** バックエンドを使用できます。

- * デフォルトドメインのみ
- * デフォルトドメインと追加のドメインでは、追加のドメインごとに ID マッピング設定を作成する必要はありません。
- 特定のドメインのみ

注記

デフォルトドメインに **autorid** を使用する場合は、ドメイン用の ID マッピング設定を追加するかどうかは任意です。

このセクションの一部は、Samba Wiki に公開されているドキュメント idmap config autorid に掲載されています。ライセンスは、CC BY 4.0 にあります。著者および貢献者は、Wiki ページの history タブを参照してください。

autorid バックエンドを使用した利点

- 設定された範囲内に計算した UID と GID があるすべてのドメインユーザーおよびグループは、ドメインメンバーで自動的に利用可能になります。
- ID、ホームディレクトリー、およびログインシェルを手動で割り当てる必要はありません。
- 複数ドメイン環境内の複数のオブジェクトが同じ RID を持つ場合でも、重複する ID はありません。

短所

- Samba ドメインメンバー間では、ユーザー ID とグループ ID は同じではありません。
- すべてのドメインユーザーは、割り当てられた同じログインシェルとホームディレクトリーを取得します。ただし、変数を使用できます。
- ドメインメンバーで個々のユーザーまたはグループを除外して、利用できないようにすることはできません。計算された UID または GID が、設定された範囲外にあるユーザーとグループのみが除外されます。

前提条件

- Samba をインストールしている。
- ID マッピングを除く Samba 設定が **/etc/samba/smb.conf** ファイルにある。

手順

1. **/etc/samba/smb.conf** ファイルの **[global]** セクションを編集します。
   a. * デフォルトドメインの **autorid** ID マッピングバックエンドを有効にします。

      ```
      idmap config * : backend = autorid
      ```

   b. 既存および将来の全オブジェクトに ID を割り当てられる大きさの範囲を設定します。以下に例を示します。

      ```
      idmap config * : range = 10000-999999
      ```

      Samba は、このドメインで計算した ID が範囲内にないユーザーおよびグループを無視します。

警告

範囲を設定し、Samba がそれを使用して開始してからは、範囲の上限を小さくすることはできません。範囲にその他の変更を加えると、新しい ID 割り当てが発生し、ファイルの所有権が失われる可能性があります。

c. 必要に応じて、範囲サイズを設定します。以下に例を示します。

```
idmap config * : rangesize = 200000
```

Samba は、**idmap config * : range** パラメーターに設定されている範囲からすべての ID を取得するまで、各ドメインのオブジェクトにこの数の連続 ID を割り当てます。

注記

rangesize を設定する場合は、適宜範囲を調整する必要があります。この範囲は rangesize の倍数である必要があります。

d. すべてのマッピングユーザーに割り当てられるシェルおよびホームディレクトリーのパスを設定します。以下に例を示します。

```
template shell = /bin/bash
template homedir = /home/%U
```

e. 必要に応じて、ドメイン用の ID マッピング設定を追加します。個別のドメインの設定が利用できない場合、Samba は以前に設定した * デフォルトドメインの **autorid** バックエンド設定を使用して ID を計算します。

重要

この範囲は、このサーバーの他のドメイン設定と重複させることはできません。また、この範囲には、今後割り当てられる ID がすべて収まる大きさを設定する必要があります。詳細は、Samba ID 範囲の計画 を参照してください。

2. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

3. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

関連情報

- **idmap_autorid(8)** man ページの **THE MAPPING FORMULAS** セクション

- **idmap_autorid(8)** man ページの **rangesize** パラメーターの説明
- **smb.conf(5)** man ページの **VARIABLE SUBSTITUTIONS** セクション

3.5. SAMBA を AD ドメインメンバーサーバーとして設定

AD または NT4 のドメインを実行している場合は、Samba を使用して Red Hat Enterprise Linux サーバーをメンバーとしてドメインに追加し、以下を取得します。

- その他のドメインメンバーのドメインリソースにアクセスする
- **sshd** などのローカルサービスに対してドメインユーザーを認証する
- サーバーにホストされているディレクトリーおよびプリンターを共有して、ファイルサーバーおよびプリントサーバーとして動作する

3.5.1. RHEL システムの AD ドメインへの参加

Samba Winbind は、Red Hat Enterprise Linux (RHEL) システムを Active Directory (AD) に接続するための System Security Services Daemon (SSSD) の代替手段です。**realmd** を使用して Samba Winbind を設定することで、RHEL システムを AD ドメインに参加させることができます。

手順

1. AD で Kerberos 認証に非推奨の RC4 暗号化タイプが必要な場合は、RHEL でこの暗号のサポートを有効にします。

```
# update-crypto-policies --set DEFAULT:AD-SUPPORT
```

2. 以下のパッケージをインストールします。

```
# yum install realmd oddjob-mkhomedir oddjob samba-winbind-clients \
      samba-winbind samba-common-tools samba-winbind-krb5-locator
```

3. ドメインメンバーでディレクトリーまたはプリンターを共有するには、**samba** パッケージをインストールします。

```
# yum install samba
```

4. 既存の Samba 設定ファイル **/etc/samba/smb.conf** をバックアップします。

```
# mv /etc/samba/smb.conf /etc/samba/smb.conf.bak
```

5. ドメインに参加します。たとえば、ドメイン **ad.example.com** に参加するには、以下のコマンドを実行します。

```
# realm join --membership-software=samba --client-software=winbind ad.example.com
```

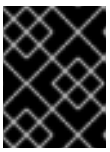
上記のコマンドを使用すると、**realm** ユーティリティーが自動的に以下を実行します。

- **ad.example.com** ドメインのメンバーシップに **/etc/samba/smb.conf** ファイルを作成します。

- ユーザーおよびグループの検索用の **winbind** モジュールを、**/etc/nsswitch.conf** ファイルに追加します。
- **/etc/pam.d/** ディレクトリーの PAM (プラグ可能な認証モジュール) 設定ファイルを更新します。
- **winbind** サービスを起動し、システムの起動時にサービスを起動できるようにします。

6. 必要に応じて、**/etc/samba/smb.conf** ファイルの別の ID マッピングバックエンド、またはカスタマイズした ID マッピングを設定します。詳細は、SambaID マッピングの理解と設定 を参照してください。
7. **winbind** サービスが稼働していることを確認します。

```
# systemctl status winbind
...
   Active: active (running) since Tue 2018-11-06 19:10:40 CET; 15s ago
```



重要

Samba がドメインのユーザーおよびグループの情報をクエリーできるようにするには、**smb** を起動する前に **winbind** サービスを実行する必要があります。

8. **samba** パッケージをインストールしてディレクトリーおよびプリンターを共有している場合は、**smb** サービスを有効化して開始します。

```
# systemctl enable --now smb
```

9. 必要に応じて、Active Directory へのローカルログインを認証する場合は、**winbind_krb5_localauth** プラグインを有効にします。MIT Kerberos 用のローカル承認プラグインの使用

検証手順

1. AD ドメインの AD 管理者アカウントなど、AD ユーザーの詳細を表示します。

```
# getent passwd "AD\administrator"
AD\administrator:*:10000:10000::/home/administrator@AD:/bin/bash
```

2. AD ドメイン内のドメインユーザーグループのメンバーをクエリーします。

```
# getent group "AD\Domain Users"
   AD\domain users:x:10000:user1,user2
```

3. オプションで、ファイルやディレクトリーに権限を設定する際に、ドメインのユーザーおよびグループを使用できることを確認します。たとえば、**/srv/samba/example.txt** ファイルの所有者を **AD\administrator** に設定し、グループを **AD\Domain Users** に設定するには、以下のコマンドを実行します。

```
# chown "AD\administrator":"AD\Domain Users" /srv/samba/example.txt
```

4. Kerberos 認証が期待どおりに機能することを確認します。
   a. AD ドメインメンバーで、**administrator@AD.EXAMPLE.COM** プリンシパルのチケットを

取得します。

```
# kinit administrator@AD.EXAMPLE.COM
```

b. キャッシュされた Kerberos チケットを表示します。

```
# klist
Ticket cache: KCM:0
Default principal: administrator@AD.EXAMPLE.COM

Valid starting       Expires              Service principal
01.11.2018 10:00:00  01.11.2018 20:00:00
krbtgt/AD.EXAMPLE.COM@AD.EXAMPLE.COM
        renew until 08.11.2018 05:00:00
```

5. 利用可能なドメインの表示:

```
# wbinfo --all-domains
BUILTIN
SAMBA-SERVER
AD
```

関連情報

- 非推奨の RC4 暗号化を使用しない場合は、AD で AES 暗号化タイプを有効にすることができます。詳細は、以下を参照してください。
- GPO を使用した Active Directory で AES 暗号化タイプの有効化
- **realm(8)** man ページ

3.5.2. MIT Kerberos 用のローカル承認プラグインの使用

winbind サービスは、Active Directory ユーザーをドメインメンバーに提供します。特定の状況では、管理者が、ドメインメンバーで実行している SSH サーバーなどのローカルサービスに対して、ドメインユーザーが認証を行えるようにします。Kerberos を使用してドメインユーザーを認証している場合は、**winbind** サービスを介して、**winbind_krb5_localauth** プラグインが Kerberos プリンシパルを Active Directory アカウントに正しくマッピングできるようにします。

たとえば、Active Directory ユーザーの **sAMAccountName** 属性を **EXAMPLE** に設定し、小文字のユーザー名でユーザーがログインしようとすると、Kerberos はユーザー名を大文字で返します。その結果、エントリーは認証の失敗に一致しません。

winbind_krb5_localauth プラグインを使用すると、アカウント名が正しくマッピングされます。これは GSSAPI 認証にのみ適用され、初期のチケット付与チケット (TGT) の取得には該当しません。

前提条件

- Samba が Active Directory のメンバーとして設定されている。
- Red Hat Enterprise Linux が、Active Directory に対してログイン試行を認証している。
- **winbind** サービスが実行している。

手順

/etc/krb5.conf ファイルを編集し、以下のセクションを追加します。

```
[plugins]
localauth = {
    module = winbind:/usr/lib64/samba/krb5/winbind_krb5_localauth.so
    enable_only = winbind
}
```

関連情報

- **winbind_krb5_localauth(8)** man ページ.

3.6. IDM ドメインメンバーでの SAMBA の設定

Red Hat Identity Management (IdM) ドメインに参加しているホスト上で Samba をセットアップできます。IdM のユーザー、および可能であれば、信頼された Active Directory (AD) ドメインのユーザーは、Samba が提供する共有およびプリンターサービスにアクセスできます。

重要

IdM ドメインメンバーで Samba を使用する機能は、テクノロジープレビュー機能で、特定の制限が含まれています。たとえば、IdM 信頼コントローラーは Active Directory グローバルカタログサービスをサポートしておらず、分散コンピューティング環境/リモートプロシージャコール (DCE/RPC) プロトコルを使用した IdM グループの解決をサポートしていません。結果として、AD ユーザーは、他の IdM クライアントにログインしている場合、IdM クライアントでホストされている Samba 共有とプリンターにのみアクセスできます。Windows マシンにログインしている AD ユーザーは、IdM ドメインメンバーでホストされている Samba 共有にアクセスできません。

IdM ドメインメンバーに Samba をデプロイしているお客様は、ぜひ Red Hat にフィードバックをお寄せください。

AD ドメインのユーザーが Samba によって提供される共有およびプリンターサービスにアクセスする必要がある場合は、AES 暗号化タイプが AD になっていることを確認してください。詳細は、GPO を使用した Active Directory での AES 暗号化タイプの有効化 を参照してください。

前提条件

- ホストは、クライアントとして IdM ドメインに参加している。
- IdM サーバーとクライアントの両方が RHEL 8.1 以降で実行されている必要がある。

3.6.1. Samba をドメインメンバーにインストールするための IdM ドメインの準備

IdM クライアントに Samba を設定する前に、IdM サーバーで **ipa-adtrust-install** ユーティリティーを使用して IdM ドメインを準備する必要があります。

注記

ipa-adtrust-install コマンドを自動的に実行するシステムは、AD 信頼コントローラーになります。ただし、**ipa-adtrust-install** は、IdM サーバーで 1 回のみ実行する必要があります。

前提条件

- IdM サーバーがインストールされている。
- パッケージをインストールし、IdM サービスを再起動するには、root 権限が必要です。

手順

1. 必要なパッケージをインストールします。

```
[root@ipaserver ~]# yum install ipa-server-trust-ad samba-client
```

2. IdM 管理ユーザーとして認証します。

```
[root@ipaserver ~]# kinit admin
```

3. **ipa-adtrust-install** ユーティリティーを実行します。

```
[root@ipaserver ~]# ipa-adtrust-install
```

統合 DNS サーバーとともに IdM がインストールされていると、DNS サービスレコードが自動的に作成されます。

IdM が統合 DNS サーバーなしで IdM をインストールすると、**ipa-adtrust-install** は、続行する前に DNS に手動で追加する必要があるサービスレコードのリストを出力します。

4. スクリプトにより、**/etc/samba/smb.conf** がすでに存在し、書き換えられることが求められます。

```
WARNING: The smb.conf already exists. Running ipa-adtrust-install will break your existing
Samba configuration.

Do you wish to continue? [no]: yes
```

5. このスクリプトは、従来の Linux クライアントが信頼できるユーザーと連携できるようにする互換性プラグインである **slapi-nis** プラグインを設定するように求めるプロンプトを表示します。

```
Do you want to enable support for trusted domains in Schema Compatibility plugin?
This will allow clients older than SSSD 1.9 and non-Linux clients to work with trusted users.

Enable trusted domains support in slapi-nis? [no]: yes
```

6. プロンプトが表示されたら、IdM ドメインの NetBIOS 名を入力するか、**Enter** を押して提案された名前を使用します。

```
Trust is configured but no NetBIOS domain name found, setting it now.
Enter the NetBIOS name for the IPA domain.
```

```
Only up to 15 uppercase ASCII letters, digits and dashes are allowed.
Example: EXAMPLE.

NetBIOS domain name [IDM]:
```

7. SID 生成タスクを実行して、既存ユーザーに SID を作成するように求められます。

```
Do you want to run the ipa-sidgen task? [no]: yes
```

これはリソースを集中的に使用するタスクであるため、ユーザー数が多い場合は別のタイミングで実行できます。

8. **(必要に応じて)** デフォルトでは、Windows Server 2008 以降での動的 RPC ポートの範囲は **49152-65535** として定義されます。ご使用の環境に異なる動的 RPC ポート範囲を定義する必要がある場合は、Samba が異なるポートを使用するように設定し、ファイアウォール設定でそのポートを開くように設定します。以下の例では、ポート範囲を **55000-65000** に設定します。

```
[root@ipaserver ~]# net conf setparm global 'rpc server dynamic port range' 55000-65000
[root@ipaserver ~]# firewall-cmd --add-port=55000-65000/tcp
[root@ipaserver ~]# firewall-cmd --runtime-to-permanent
```

9. **ipa** サービスを再起動します。

```
[root@ipaserver ~]# ipactl restart
```

10. **smbclient** ユーティリティーを使用して、Samba が IdM からの Kerberos 認証に応答することを確認します。

```
[root@ipaserver ~]# smbclient -L ipaserver.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry
   Sharename       Type      Comment
   ---------       ----      -------
   IPC$            IPC       IPC Service (Samba 4.15.2)
...
```

3.6.2. IdM クライアントでの Samba サーバーのインストールおよび設定

IdM ドメインに登録されたクライアントに Samba をインストールして設定できます。

前提条件

- IdM サーバーとクライアントの両方が RHEL 8.1 以降で実行されている必要がある。
- IdM ドメインは、ドメインメンバーに Samba をインストールするための IdM ドメインの準備 の説明に従って準備されます。
- IdM に AD で設定された信頼がある場合は、Kerberos の AES 暗号化タイプを有効にします。たとえば、グループポリシーオブジェクト (GPO) を使用して、AES 暗号化の種類を有効にします。詳細は、GPO を使用した Active Directory での AES 暗号化の有効化 を参照してください。

手順

1. **ipa-client-samba** パッケージをインストールします。

```
[root@idm_client]# yum install ipa-client-samba
```

2. **ipa-client-samba** ユーティリティーを使用して、クライアントを準備し、初期 Samba 設定を作成します。

```
[root@idm_client]# ipa-client-samba
Searching for IPA server...
IPA server: DNS discovery
Chosen IPA master: idm_server.idm.example.com
SMB principal to be created: cifs/idm_client.idm.example.com@IDM.EXAMPLE.COM
NetBIOS name to be used: IDM_CLIENT
Discovered domains to use:

 Domain name: idm.example.com
NetBIOS name: IDM
        SID: S-1-5-21-525930803-952335037-206501584
   ID range: 212000000 - 212199999

 Domain name: ad.example.com
NetBIOS name: AD
        SID: None
   ID range: 1918400000 - 1918599999

Continue to configure the system with these values? [no]: yes
Samba domain member is configured. Please check configuration at /etc/samba/smb.conf
and start smb and winbind services
```

3. デフォルトでは、**ipa-client-samba** は、ユーザーが接続したときにそのユーザーのホームディレクトリーを動的に共有するために、**/etc/samba/smb.conf** ファイルに **[homes]** セクションが自動的に追加されます。ユーザーがこのサーバーにホームディレクトリーがない場合、または共有したくない場合は、**/etc/samba/smb.conf** から次の行を削除します。

```
[homes]
   read only = no
```

4. ディレクトリーとプリンターを共有します。詳細は、以下を参照してください。

- POSIX ACL を使用した Samba ファイル共有の設定
- Windows ACL で共有の設定
- プリントサーバーとしての Samba の設定

5. ローカルファイアウォールで Samba クライアントに必要なポートを開きます。

```
[root@idm_client]# firewall-cmd --permanent --add-service=samba-client
[root@idm_client]# firewall-cmd --reload
```

6. **smb** サービスおよび **winbind** サービスを有効にして開始します。

```
[root@idm_client]# systemctl enable --now smb winbind
```

検証手順

samba-client パッケージがインストールされている別の IdM ドメインメンバーで次の検証手順を実行します。

- Kerberos 認証を使用して、Samba サーバー上の共有をリスト表示します。

```
$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry

    Sharename       Type      Comment
    ---------       ----      -------
    example         Disk
    IPC$            IPC       IPC Service (Samba 4.15.2)
...
```

関連情報

- **ipa-client-samba(1)** の man ページ

3.6.3. IdM が新しいドメインを信頼する場合は、ID マッピング設定を手動で追加

Samba では、ユーザーがリソースにアクセスする各ドメインの ID マッピング設定が必要です。IdM クライアントで実行している既存の Samba サーバーでは、管理者が Active Directory (AD) ドメインに新しい信頼を追加した後、ID マッピング設定を手動で追加する必要があります。

前提条件

- IdM クライアントで Samba を設定している。その後、IdM に新しい信頼が追加されている。
- Kerberos の暗号化タイプ DES および RC4 は、信頼できる AD ドメインで無効にしている。セキュリティー上の理由から、RHEL 8 はこのような弱い暗号化タイプに対応していません。

手順

1. ホストのキータブを使用して認証します。

```
[root@idm_client]# kinit -k
```

2. **ipa idrange-find** コマンドを使用して、新しいドメインのベース ID と ID 範囲のサイズの両方を表示します。たとえば、次のコマンドは **ad.example.com** ドメインの値を表示します。

```
[root@idm_client]# ipa idrange-find --name="AD.EXAMPLE.COM_id_range" --raw
---------------
1 range matched
---------------
  cn: AD.EXAMPLE.COM_id_range
  ipabaseid: 1918400000
  ipaidrangesize: 200000
  ipabaserid: 0
  ipanttrusteddomainsid: S-1-5-21-968346183-862388825-1738313271
  iparangetype: ipa-ad-trust
----------------------------
Number of entries returned 1
----------------------------
```

次の手順で、**ipabaseid** 属性および **ipaidrangesize** 属性の値が必要です。

3. 使用可能な最高の ID を計算するには、次の式を使用します。

```
maximum_range = ipabaseid + ipaidrangesize - 1
```

前の手順の値を使用すると、**ad.example.com** ドメインで使用可能な最大 ID は **1918599999** (1918400000 + 200000 - 1) です。

4. **/etc/samba/smb.conf** ファイルを編集し、ドメインの ID マッピング設定を **[global]** セクションに追加します。

```
idmap config AD : range = 1918400000 - 1918599999
idmap config AD : backend = sss
```

ipabaseid 属性の値を最小値として指定し、前の手順で計算された値を範囲の最大値として指定します。

5. **smb** サービスおよび **winbind** サービスを再起動します。

```
[root@idm_client]# systemctl restart smb winbind
```

検証手順

- Kerberos 認証を使用して、Samba サーバー上の共有をリスト表示します。

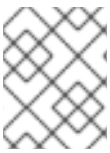
```
$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry

   Sharename       Type      Comment
   ---------       ----      -------
   example         Disk
   IPC$            IPC       IPC Service (Samba 4.15.2)
...
```

3.6.4. 関連情報

- Installing an Identity Management client

3.7. POSIX ACL を使用した SAMBA ファイル共有の設定

Samba は、Linux サービスとして、POSIX ACL との共有に対応します。**chmod** などのユーティリティーを使用して、Samba サーバーの権限をローカルに管理できます。拡張属性に対応するファイルシステムに共有が保存されている場合は、複数のユーザーおよびグループで ACL を定義できます。



注記

代わりにきめ細かい Windows ACL を使用する必要がある場合は、Windows ACL を使用する共有の設定 を参照してください。

このセクションの一部は、Samba Wiki に公開されているドキュメント Setting up a Share Using POSIX ACLs に掲載されています。ライセンスは、CC BY 4.0 にあります。著者および貢献者は、Wiki ページの history タブを参照してください。

3.7.1. POSIX ACL を使用する共有の追加

/srv/samba/example/ ディレクトリーのコンテンツを提供し、POSIX ACL を使用する **example** という名前の共有を作成できます。

前提条件

Samba が、以下のいずれかのモードで設定されている。

- スタンドアロンサーバー
- ドメインメンバー

手順

1. ディレクトリーが存在しない場合は作成します。以下に例を示します。

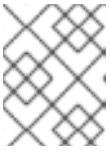
```
# mkdir -p /srv/samba/example/
```

2. SELinux を、**enforcing** モードで実行する場合は、そのディレクトリーに **samba_share_t** コンテキストを設定します。

```
# semanage fcontext -a -t samba_share_t "/srv/samba/example(/.*)?"
# restorecon -Rv /srv/samba/example/
```

3. ディレクトリーにファイルシステムの ACL を設定します。詳細は、以下を参照してください。
   - POSIX ACL を使用する Samba 共有での標準的な ACL の設定
   - POSIX ACL を使用する Samba 共有で拡張 ACL の設定
4. **/etc/samba/smb.conf** ファイルにサンプル共有を追加します。たとえば、共有の write-enabled を追加するには、次のコマンドを実行します。

```
[example]
 path = /srv/samba/example/
 read only = no
```



注記

ファイルシステムの ACL に関係なく、**read only = no** を設定しないと、Samba がディレクトリーを読み取り専用モードで共有します。

5. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

6. **firewall-cmd** ユーティリティーを使用して必要なポートを開き、ファイアウォール設定を再読み込みします。

```
# firewall-cmd --permanent --add-service=samba
# firewall-cmd --reload
```

7. **smb** サービスを再起動します。

```
# systemctl restart smb
```

3.7.2. POSIX ACL を使用する Samba 共有での標準的な Linux ACL の設定

Linux の標準 ACL は、所有者、グループ、その他の未定義ユーザーの権限の設定に対応します。ユーティリティーの **chown**、**chgrp**、および **chmod** を使用して ACL を更新できます。正確な制御が必要な場合は、より複雑な POSIX ACL を使用します。以下を参照してください。

Setting extended ACLs on a Samba share that uses POSIX ACLs .

以下の手順では、**/srv/samba/example/** ディレクトリーの所有者を **root** ユーザーに設定し、**Domain Users** グループに読み取りおよび書き込みの権限を付与して、他のすべてのユーザーのアクセスを拒否します。

前提条件

- ACL を設定する Samba 共有がある。

手順

```
# chown root:"Domain Users" /srv/samba/example/
# chmod 2770 /srv/samba/example/
```

注記

ディレクトリーで set-group-ID (SGID) ビットを有効にすると、新しいディレクトリーエントリーを作成したユーザーのプライマリーグループに設定する通常の動作の代わりに、すべての新しいファイルとサブディレクトリーのデフォルトグループが、そのディレクトリーグループのデフォルトグループに自動的に設定されます。

関連情報

- **chown(1)** および **chmod(1)** の man ページ

3.7.3. POSIX ACL を使用する Samba 共有での拡張 ACL の設定

共有ディレクトリーが保存されているファイルシステムが拡張 ACL に対応している場合は、それを使用して複雑な権限を設定できます。拡張 ACL には、複数のユーザーおよびグループの権限を指定できます。

拡張 POSIX ACL を使用すると、複数のユーザーおよびグループで複雑な ACL を設定できます。ただし、設定できるのは以下の権限のみです。

- アクセスなし
- 読み取りアクセス
- 書き込みアクセス

- 完全な制御

フォルダーの作成やデータの追加 など、詳細な Windows 権限が必要な場合は、Windows ACL を使用するように共有を設定します。Windows ACL で共有の設定

以下の手順では、共有で拡張 ACL を有効にする方法を説明します。また、拡張 ACL の設定例も含まれています。

前提条件

- ACL を設定する Samba 共有がある。

手順

1. **/etc/samba/smb.conf** ファイルの共有セクションで以下のパラメーターを有効にして、拡張 ACL の ACL 継承を有効にします。

```
inherit acls = yes
```

詳細は、man ページの **smb.conf(5)** のパラメーターの説明を参照してください。

2. **smb** サービスを再起動します。

```
# systemctl restart smb
```

3. ディレクトリーの ACL を設定します。以下に例を示します。

例3.2 拡張 ACL の設定

以下の手順は、**/srv/samba/example/** ディレクトリーに対して、**Domain Admins** グループに読み取り、書き込み、および実行の権限、**Domain Users** グループに対する読み取りおよび実行の権限を設定し、その他の全員のアクセスを拒否します。

1. ユーザーアカウントのプライマリーグループへの自動許可権限を無効にします。

```
# setfacl -m group::--- /srv/samba/example/
# setfacl -m default:group::--- /srv/samba/example/
```

ディレクトリーのプライマリーグループは、さらに動的な **CREATOR GROUP** プリンシパルにマッピングされます。Samba 共有で拡張 POSIX ACL を使用すると、このプリンシパルは自動的に追加され、削除できません。

2. ディレクトリーに権限を設定します。

a. **Domain Admins** グループに読み取り、書き込み、および実行の権限を付与します。

```
# setfacl -m group:"DOMAIN\Domain Admins":rwx /srv/samba/example/
```

b. **Domain Users** グループに読み取りおよび実行の権限を付与します。

```
# setfacl -m group:"DOMAIN\Domain Users":r-x /srv/samba/example/
```

c. **その他** の ACL エントリーに権限を設定し、その他の ACL エントリーに一致しないユーザーへのアクセスを拒否します。

```
# setfacl -R -m other::--- /srv/samba/example/
```

この設定は、このディレクトリーにのみ適用されます。Windows では、これらの ACL は **このフォルダーのみ** のモードにマッピングされます。

3. 前の手順で設定した権限を、このディレクトリーに作成した新規ファイルシステムのオブジェクトから継承できるようにするには、以下のコマンドを実行します。

```
# setfacl -m default:group:"DOMAIN\Domain Admins":rwx /srv/samba/example/
# setfacl -m default:group:"DOMAIN\Domain Users":r-x /srv/samba/example/
# setfacl -m default:other::--- /srv/samba/example/
```

この設定では、プリンシパルの **このフォルダーのみ** モードが、**このフォルダー、サブフォルダー、およびファイル** に設定されます。

Samba は、手順に設定されている権限を、以下の Windows ACL にマッピングします。

プリンシパル	アクセス	適用先
Domain\Domain Admins	完全な制御	このフォルダー、サブフォルダー、およびファイル
Domain\Domain Users	読み取りおよび実行	このフォルダー、サブフォルダー、およびファイル
Everyone[a]	なし	このフォルダー、サブフォルダー、およびファイル
owner (Unix User\owner)[b]	完全な制御	このフォルダーのみ
primary_group (Unix User\primary_group)[c]	なし	このフォルダーのみ
CREATOR OWNER[d] [e]	完全な制御	サブフォルダーおよびファイルのみ
CREATOR GROUP[e] [f]	なし	サブフォルダーおよびファイルのみ

[a] Samba は、このプリンシパルの権限を **その他** の ACL エントリーからマッピングします。

[b] Samba は、ディレクトリーの所有者をこのエントリーにマッピングします。

[c] Samba は、ディレクトリーのプライマリーグループをこのエントリーにマッピングします。

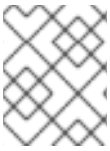
[d] 新規ファイルシステムオブジェクトでは、作成者はこのプリンシパルの権限を自動的に継承します。

[e] POSIX ACL を使用する共有では、このプリンシパルの設定または削除には対応していません。

[f] 新規ファイルシステムオブジェクトの場合、作成者のプライマリーグループは、自動的にこのプリンシパルの権限を継承します。

3.8. POSIX ACL を使用する共有への権限の設定

必要に応じて、Samba 共有へのアクセスを制限または許可するには、**/etc/samba/smb.conf** ファイルの共有のセクションに特定のパラメーターを設定します。



注記

共有ベースの権限は、ユーザー、グループ、またはホストが共有にアクセスできるかどうかを管理します。この設定は、ファイルシステムの ACL には影響しません。

共有ベースの設定を使用して共有へのアクセスを制限します。たとえば、特定のホストからのアクセスを拒否します。

前提条件

- POSIX ACL との共有が設定されている。

3.8.1. ユーザーおよびグループに基づいた共有アクセスの設定

ユーザーおよびグループに基づいたアクセス制御により、特定のユーザーおよびグループの共有へのアクセスを許可または拒否できます。

前提条件

- ユーザーまたはグループベースのアクセスを設定する Samba 共有がある。

手順

1. たとえば、**Domain Users** グループの全メンバーが、**ユーザー** アカウントのアクセスが拒否されている時に共有にアクセスできるようにするには、共有の設定に以下のパラメーターを追加します。

```
valid users = +DOMAIN\"Domain Users"
invalid users = DOMAIN\user
```

無効なユーザー パラメーターの優先度は、**有効なユーザー** パラメーターよりも高くなります。たとえば、**ユーザー** アカウントが **Domain Users** グループのメンバーである場合に上述の例を使用すると、このアカウントへのアクセスは拒否されます。

2. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

関連情報

- **smb.conf(5)** man ページ

3.8.2. ホストベースの共有アクセスの設定

ホストベースのアクセス制御により、クライアントのホスト名、IP アドレス、または IP 範囲に基づいて、共有へのアクセスを許可または拒否できます。

以下の手順では、IP アドレスの **127.0.0.1**、IP 範囲の **192.0.2.0/24**、およびホストの **client1.example.com** を有効にして共有にアクセスする方法と、**client2.example.com** ホストへのアクセスを拒否する方法を説明します。

前提条件

- ホストベースのアクセスを設定する Samba 共有がある。

手順

1. 以下のパラメーターを、**/etc/samba/smb.conf** ファイルの共有の設定に追加します。

```
hosts allow = 127.0.0.1 192.0.2.0/24 client1.example.com
hosts deny = client2.example.com
```

hosts deny パラメーターは、**hosts allow** よりも優先順位が高くなります。たとえば、**client1.example.com** が **hosts allow** パラメーターにリスト表示されている IP アドレスに解決すると、このホストへのアクセスは拒否されます。

2. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

関連情報

- **smb.conf(5)** man ページ

3.9. WINDOWS ACL で共有の設定

Samba は、共有およびファイルシステムオブジェクトへの Windows ACL の設定に対応します。これを使用すると、以下が可能になります。

- きめ細かな Windows ACL を使用する
- Windows を使用して共有権限およびファイルシステムの ACL を管理する

または、POSIX ACL を使用するように共有を設定することもできます。詳細は、POSIX ACL を使用する Samba ファイル共有の設定 を参照してください。

このセクションの一部は、Samba Wiki に公開されているドキュメント Setting up a Share Using Windows ACLs に掲載されています。ライセンスは、CC BY 4.0 にあります。著者および貢献者は、Wiki ページの history タブを参照してください。

3.9.1. SeDiskPrivilege 特権の付与

Windows ACL を使用する共有に対する権限を設定できるのは、**SeDiskOperatorPrivilege** 特権が付与されているユーザーおよびグループのみです。

手順

1. たとえば、**SeDiskOperatorPrivilege** 特権を **DOMAIN\Domain Admins** グループに付与するには、以下のコマンドを実行します。

```
# net rpc rights grant "DOMAIN\Domain Admins" SeDiskOperatorPrivilege -U
```

```
"DOMAIN\administrator"
Enter DOMAIN\administrator's password:
Successfully granted rights.
```

注記

ドメイン環境では、**SeDiskOperatorPrivilege** をドメイングループに付与します。これにより、ユーザーのグループメンバーシップを更新し、権限を集中的に管理できます。

2. **SeDiskOperatorPrivilege** が付与されているすべてのユーザーおよびグループをリスト表示するには、以下のコマンドを実行します。

```
# net rpc rights list privileges SeDiskOperatorPrivilege -U "DOMAIN\administrator"
Enter administrator's password:
SeDiskOperatorPrivilege:
  BUILTIN\Administrators
  DOMAIN\Domain Admins
```

3.9.2. Windows ACL サポートの有効化

Windows ACL に対応する共有を設定するには、Samba でこの機能を有効にする必要があります。

前提条件

- ユーザー共有が Samba サーバーに設定されている。

手順

1. すべての共有に対してグローバルに有効にするには、次の設定を **/etc/samba/smb.conf** ファイルの **[global]** セクションに追加します。

```
vfs objects = acl_xattr
map acl inherit = yes
store dos attributes = yes
```

または、共有のセクションに同じパラメーターを追加して、個別の共有に対して Windows ACL サポートを有効にできます。

2. **smb** サービスを再起動します。

```
# systemctl restart smb
```

3.9.3. Windows ACL を使用する共有の追加

/srv/samba/example/ ディレクトリーのコンテンツを共有し、Windows ACL を使用する **example** という名前の共有を作成できます。

手順

1. ディレクトリーが存在しない場合は作成します。以下に例を示します。

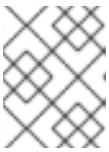
```
# mkdir -p /srv/samba/example/
```

2. SELinux を、**enforcing** モードで実行する場合は、そのディレクトリーに **samba_share_t** コンテキストを設定します。

```
# semanage fcontext -a -t samba_share_t "/srv/samba/example(/.)?"*
# restorecon -Rv /srv/samba/example/
```

3. **/etc/samba/smb.conf** ファイルにサンプル共有を追加します。たとえば、共有の write-enabled を追加するには、次のコマンドを実行します。

```
[example]
 path = /srv/samba/example/
 read only = no
```



注記

ファイルシステムの ACL に関係なく、**read only = no** を設定しないと、Samba がディレクトリーを読み取り専用モードで共有します。

4. すべての共有の **[global]** セクションで Windows ACL サポートを有効にしていない場合は、以下のパラメーターを **[example]** セクションに追加して、この共有に対してこの機能を有効にします。

```
vfs objects = acl_xattr
map acl inherit = yes
store dos attributes = yes
```

5. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

6. **firewall-cmd** ユーティリティーを使用して必要なポートを開き、ファイアウォール設定を再読み込みします。

```
# firewall-cmd --permanent --add-service=samba
# firewall-cmd --reload
```

7. **smb** サービスを再起動します。

```
# systemctl restart smb
```

3.9.4. Windows ACL を使用する共有の共有権限とファイルシステム ACL の管理

Windows ACL を使用する Samba 共有で共有権限およびファイルシステムの ACL を管理するには、**コンピューターの管理** などの Windows アプリケーションを使用します。詳細は、Windows のドキュメントを参照してください。または、**smbcacls** ユーティリティーを使用して ACL を管理します。

注記

Windows からファイルシステムの権限を変更するには、**SeDiskOperatorPrivilege** 権限が付与されたアカウントを使用する必要があります。

関連情報

- 「smbcacls で SMB 共有上の ACL の管理」
- 「SeDiskPrivilege 特権の付与」

3.10. SMBCACLS で SMB 共有上の ACL の管理

smbcacls ユーティリティーは、SMB 共有に保存されたファイルおよびディレクトリーの ACL をリスト表示、設定、および削除できます。**smbcacls** を使用して、ファイルシステムの ACL を管理できます。

- 高度な Windows ACL または POSIX ACL を使用するローカルまたはリモートの Samba サーバー
- Red Hat Enterprise Linux で、Windows でホストされる共有の ACL をリモートで管理

3.10.1. アクセス制御エントリー

ファイルシステムオブジェクトの各 ACL エントリーには、以下の形式のアクセス制御エントリー (ACE) が含まれます。

> **security_principal:access_right/inheritance_information/permissions**

> **例3.3 アクセス制御エントリー**
>
> **AD\Domain Users** グループに、Windows 上の **このフォルダー、サブフォルダー、およびファイル** に適用される **変更** 権限がある場合、ACL には以下の ACE が含まれます。
>
> > AD\Domain Users:ALLOWED/OI|CI/CHANGE

ACE には、以下が含まれます。

セキュリティープリンシパル
: セキュリティープリンシパルは、ACL の権限が適用されるユーザー、グループ、または SID です。

アクセス権
: オブジェクトへのアクセスが許可または拒否されるかどうかを定義します。値は **ALLOWED** または **DENIED** です。

継承情報
: 次の値を取ります。

表3.1 継承の設定

値	詳細	マップ先
OI	オブジェクトの継承	このフォルダーおよびファイル
CI	コンテナーの継承	このフォルダーおよびサブフォルダー
IO	継承のみ	ACE は、現在のファイルまたはディレクトリーには適用されません。
ID	継承済	親ディレクトリーから ACE が継承されました。

また、値は以下のように組み合わせることができます。

表3.2 継承設定の組み合わせ

値の組み合わせ	Windows の 適用先 設定にマップします。
OI|CI	このフォルダー、サブフォルダー、およびファイル
OI|CI|IO	サブフォルダーおよびファイルのみ
CI|IO	サブディレクトリーのみ
OI|IO	ファイルのみ

権限

この値は、Windows の権限または **smbcacls** エイリアスを表す 16 進値になります。

- 1 つ以上の Windows の権限を表す 16 進値。
  次の表に、Windows の高度な権限とそれに対応する値を 16 進法で表示します。

  表3.3 Windows の権限とそれに対応する smbcacls 値を 16 進法で設定

Windows の権限	16 進値
完全な制御	**0x001F01FF**
フォルダーのスキャンおよびファイルの実行	**0x00100020**
フォルダーのリスト表示 / データの読み取り	**0x00100001**
属性の読み取り	**0x00100080**
拡張属性の読み取り	**0x00100008**
ファイルの作成/データの書き込み	**0x00100002**

Windows の権限	16 進値
フォルダーの作成/データの追加	**0x00100004**
属性の書き込み	**0x00100100**
拡張属性の書き込み	**0x00100010**
サブフォルダーおよびファイルの削除	**0x00100040**
削除	**0x00110000**
権限の読み取り	**0x00120000**
権限の変更	**0x00140000**
所有権の取得	**0x00180000**

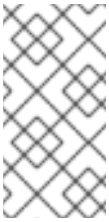
ビット単位の **OR** 演算を使用すると、複数の権限を 1 つの 16 進値として組み合わせることができます。詳細は、ACE マスク計算 を参照してください。

- **smbcacls** エイリアス。以下の表には、利用可能なエイリアスが表示されます。

表3.4 既存の smbcacls エイリアスとそれに対応する Windows の権限

smbcacls エイリアス	Windows の権限へのマッピング
-R	読み取り
READ	読み取りおよび実行
W	主な機能: ○ ファイルの作成/データの書き込み ○ フォルダーの作成/データの追加 ○ 属性の書き込み ○ 拡張属性の書き込み ○ 権限の読み取り
D	削除
%P	権限の変更
O	所有権の取得

smbcacls エイリアス	Windows の権限へのマッピング
X	スキャン / 実行
CHANGE	修正
FULL	完全な制御



注記

権限を設定する際に、1 文字のエイリアスを組み合わせることができます。たとえば、Windows の権限の **Read** および **Delete** を適用するように **RD** を設定できます。ただし、1 文字以外のエイリアスを複数組み合わせたり、エイリアスと 16 進値を組み合わせることはできません。

3.10.2. smbcacls を使用した ACL の表示

SMB 共有で ACL を表示するには、**smbcacls** ユーティリティーを使用します。**--add** などの操作パラメーターを付けずに **smbcacls** を実行すると、ユーティリティーは、ファイルシステムオブジェクトの ACL を表示します。

手順

たとえば、**//server/example** 共有のルートディレクトリーの ACL をリスト表示するには、以下のコマンドを実行します。

```
# smbcacls //server/example / -U "DOMAIN\administrator"
Enter DOMAIN\administrator's password:
REVISION:1
CONTROL:SR|PD|DI|DP
OWNER:AD\Administrators
GROUP:AD\Domain Users
ACL:AD\Administrator:ALLOWED/OI|CI/FULL
ACL:AD\Domain Users:ALLOWED/OI|CI/CHANGE
ACL:AD\Domain Guests:ALLOWED/OI|CI/0x00100021
```

コマンドの出力は以下のようになります。

- **REVISION** - セキュリティー記述子の内部 Windows NT ACL リビジョン
- **CONTROL** - セキュリティー記述子の制御
- **OWNER** - セキュリティー記述子の所有者の名前または SID
- **GROUP** - セキュリティー記述子のグループの名前または SID
- **ACL** エントリー。詳細は、アクセス制御エントリー を参照してください。

3.10.3. ACE マスク計算

ほとんどの場合、ACE を追加または更新するときは、既存の smbcacls エイリアスとそれに対応する Windows アクセス許可に リストされている **smbcacls** エイリアスを使用します。

ただし、Windows の権限と それに対応する smbcacls 値を 16 進法でリストしたように高度な Windows の権限 を設定する場合は、ビット単位の **OR** 操作を使用して正しい値を計算する必要があります。以下のシェルコマンドを使用して値を計算できます。

```
# echo $(printf '0x%X' $(( hex_value_1 | hex_value_2 | ... )))
```

例3.4 ACE マスクの計算

以下の権限を設定します。

- フォルダーのスキャン / ファイルの実行 (0x00100020)
- フォルダーのリスト表示 / データの読み取り (0x00100001)
- 属性の読み取り (0x00100080)

以前の権限の 16 進値を計算するには、以下を入力します。

```
# echo $(printf '0x%X' $(( 0x00100020 | 0x00100001 | 0x00100080 )))
0x1000A1
```

ACE を設定または更新する場合は、戻り値を使用します。

3.10.4. smbcacls を使用した ACL の追加、更新、および削除

smbcacls ユーティリティーに渡すパラメーターに応じて、ファイルまたはディレクトリーから ACL を追加、更新、および削除できます。

ACL の追加
このフォルダー、サブフォルダー、およびファイル の **CHANGE** 権限を **AD\Domain Users** グループに付与する //**server**/**example** 共有のルートに ACL を追加するには、以下のコマンドを実行します。

```
# smbcacls //server/example / -U "DOMAIN\administrator --add ACL:"AD\Domain Users":ALLOWED/OI|CI/CHANGE
```

ACL の更新
ACL の更新は、新しい ACL の追加に似ています。ACL を更新する場合は、**--modify** パラメーターと既存のセキュリティープリンシパルを使用して ACL を上書きします。**smbcacls** が ACL リスト内でセキュリティープリンシパルを検出すると、ユーティリティーは権限を更新します。これを行わないと、以下のエラーでコマンドが失敗します。

```
ACL for SID principal_name not found
```

たとえば、**AD\Domain Users** グループの権限を更新し、**このフォルダー、サブフォルダー、およびファイル** の **READ** に設定するには、以下のコマンドを実行します。

```
# smbcacls //server/example / -U "DOMAIN\administrator --modify ACL:"AD\Domain Users":ALLOWED/OI|CI/READ
```

ACL の削除
ACL を削除するには、正確な ACL を持つ **--delete** パラメーターを **smbcacls** ユーティリティーに渡します。以下に例を示します。

```
# smbcacls //server/example / -U "DOMAIN\administrator --delete ACL:"AD\Domain
Users":ALLOWED/OI|CI/READ
```

3.11. ユーザーが SAMBA サーバーのディレクトリーを共有できるようにする

Samba サーバーでは、root 権限なしでユーザーがディレクトリーを共有できるように設定できます。

3.11.1. ユーザーの共有機能の有効化

ユーザーがディレクトリーを共有できるようにするには、管理者が Samba でユーザー共有を有効にする必要があります。

たとえば、ローカルの **example** グループのメンバーのみがユーザー共有を作成できるようにするには、以下を実行します。

手順

1. ローカルの **example** グループが存在しない場合は作成します。

   ```
   # groupadd example
   ```

2. ユーザー共有の定義を保存し、その権限を正しく設定するために、Samba 用のディレクトリーを準備します。以下に例を示します。

   a. ディレクトリーを作成します。

      ```
      # mkdir -p /var/lib/samba/usershares/
      ```

   b. **example** グループの書き込み権限を設定します。

      ```
      # chgrp example /var/lib/samba/usershares/
      # chmod 1770 /var/lib/samba/usershares/
      ```

   c. このディレクトリーの他のユーザーが保存したファイルの名前変更や削除を禁止するように sticky ビットを設定します。

3. **/etc/samba/smb.conf** ファイルを編集し、以下を **[global]** セクションに追加します。

   a. ユーザー共有の定義を保存するように設定したディレクトリーのパスを設定します。以下に例を示します。

      ```
      usershare path = /var/lib/samba/usershares/
      ```

   b. このサーバーで Samba を作成できるユーザー共有の数を設定します。以下に例を示します。

      ```
      usershare max shares = 100
      ```

      usershare max shares パラメーターにデフォルトの **0** を使用すると、ユーザー共有が無効になります。

c. 必要に応じて、ディレクトリーの絶対パスのリストを設定します。たとえは、Samba が **/data** ディレクトリーおよび **/srv** ディレクトリーのサブディレクトリーの共有のみを許可するように設定するには、以下を設定します。

```
usershare prefix allow list = /data /srv
```

設定可能なユーザー共有関連のパラメーターのリストは、man ページの **smb.conf(5)** の **USERSHARES** セクションを参照してください。

4. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

5. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

これで、ユーザーが、ユーザー共有を作成できるようになりました。

3.11.2. ユーザー共有の追加

Samba でユーザー共有機能を有効にすると、ユーザーは **net usershare add** コマンドを実行して、**root** 権限なしで Samba サーバーのディレクトリーを共有できます。

net usershare add コマンドの構文:

net usershare add share_name path [[comment] | [ACLs]] [guest_ok=y|n]

重要

ユーザー共有の作成時に ACL を設定する場合は、ACL の前に comment パラメーターを指定する必要があります。空のコメントを設定するには、空の文字列を二重引用符で囲みます。

管理者が、**/etc/samba/smb.conf** ファイルの **[global]** セクションで **usershare allow guests = yes** に設定すると、ユーザーはユーザー共有上でのみゲストアクセスを有効にできることに注意してください。

例3.5 ユーザー共有の追加

ユーザーが、Samba サーバーで **/srv/samba/** ディレクトリーを共有する場合があります。共有には、**example** という名前を付け、コメントを設定しないようにし、ゲストユーザーがアクセスできるようにします。また、共有権限は、**AD\Domain Users** グループへのフルアクセスと、その他のユーザーへの読み取り権限を設定する必要があります。この共有を追加するには、そのユーザーで以下を実行します。

```
$ net usershare add example /srv/samba/ "" "AD\Domain Users":F,Everyone:R
guest_ok=yes
```

3.11.3. ユーザー共有の設定の更新

ユーザー共有の設定を更新するには、同じ共有名と新しい設定で **net usershare add** コマンドを使用して共有を上書きします。ユーザー共有の追加 を参照します。

3.11.4. 既存のユーザー共有に関する情報の表示

ユーザーは、Samba サーバーで **net usershare info** コマンドを実行して、ユーザーの共有および設定を表示できます。

前提条件

- ユーザー共有が Samba サーバーに設定されている。

手順

1. 任意のユーザーが作成したすべてのユーザー共有を表示するには、以下のコマンドを実行します。

```
$ net usershare info -l
[share_1]
path=/srv/samba/
comment=
usershare_acl=Everyone:R,host_name\user:F,
guest_ok=y
...
```

   コマンドを実行するユーザーが作成した共有のみをリスト表示するには、**-l** パラメーターを省略します。

2. 特定の共有に関する情報のみを表示するには、共有名またはワイルドカードをコマンドに渡します。たとえば、名前が **share_** で始まる共有の情報を表示する場合は、以下のコマンドを実行します。

```
$ net usershare info -l share_*
```

3.11.5. ユーザー共有のリスト表示

Samba サーバーで設定を行わずに利用可能なユーザー共有のみをリスト表示するには、**net usershare list** コマンドを使用します。

前提条件

- ユーザー共有が Samba サーバーに設定されている。

手順

1. 任意のユーザーが作成した共有をリスト表示するには、以下のコマンドを実行します。

```
$ net usershare list -l
share_1
share_2
...
```

   コマンドを実行するユーザーが作成した共有のみをリスト表示するには、**-l** パラメーターを省略します。

2. 特定の共有のみをリスト表示するには、共有名またはワイルドカードをコマンドに渡します。たとえば、名前が **share_** で始まる共有のみをリスト表示するには、以下のコマンドを実行します。

```
$ net usershare list -l share_*
```

3.11.6. ユーザー共有の削除

ユーザー共有を削除するには、共有を作成したユーザーまたは **root** ユーザーで、**net usershare delete** コマンドを実行します。

前提条件

- ユーザー共有が Samba サーバーに設定されている。

手順

```
$ net usershare delete share_name
```

3.12. 認証なしでアクセスを許可する共有の設定

特定の状況では、認証なしでユーザーが接続できるディレクトリーを共有します。これを設定するには、共有でゲストアクセスを有効にします。

警告

共有に認証を使用しないと、セキュリティーリスクとなる場合があります。

3.12.1. 共有へのゲストアクセスの有効化

共有でゲストアクセスが有効になっている場合、Samba はゲスト接続を、**guest account** パラメーターで設定したオペレーティングシステムアカウントにマッピングします。少なくとも以下のいずれかの条件が満たされると、ゲストユーザーはこの共有のファイルにアクセスできます。

- アカウントがファイルシステムの ACL にリスト表示されます。
- **その他** のユーザーの POSIX 権限はこれを許可します。

例3.6 ゲスト共有の権限

ゲストアカウントを **nobody** (デフォルト) にマッピングするように Samba を設定している場合は、下記の例の ACL が、以下を行うようになります。

- ゲストユーザーが **file1.txt** の読み込みを許可する
- ゲストユーザーによる **file2.txt** の読み込みおよび修正を許可する
- ゲストユーザーが **file3.txt** を読み込んだり修正しないようにする

```
-rw-r--r--. 1 root       root      1024 1. Sep 10:00 file1.txt
-rw-r-----. 1 nobody     root      1024 1. Sep 10:00 file2.txt
-rw-r-----. 1 root       root      1024 1. Sep 10:00 file3.txt
```

手順

1. **/etc/samba/smb.conf** ファイルを編集します。
   a. これが、このサーバーで設定した最初のゲスト共有である場合は、以下を行います。
      i. **[global]** セクションに **map to guest = Bad User** を設定し ます。

      ```
      [global]
              ...
              map to guest = Bad User
      ```

      この設定により、ユーザー名が存在しない限り、Samba は間違ったパスワードを使用したログイン試行を拒否します。指定したユーザー名がなく、ゲストアクセスが共有で有効になっている場合、Samba は接続をゲストのログインとして処理します。

      ii. デフォルトでは、Samba は、Red Hat Enterprise Linux の **nobody** アカウントにゲストアカウントをマッピングします。または、別のアカウントを設定することもできます。以下に例を示します。

      ```
      [global]
              ...
              guest account = user_name
      ```

      このパラメーターに設定するアカウントは、Samba サーバーにローカルに存在する必要があります。セキュリティー上の理由から、Red Hat は有効なシェルを割り当てていないアカウントを使用することを推奨しています。

   b. **guest ok = yes** の設定を、**[example]** 共有セクションに追加します。

   ```
   [example]
           ...
           guest ok = yes
   ```

2. **/etc/samba/smb.conf** ファイルを検証します。

   ```
   # testparm
   ```

3. Samba 設定を再読み込みします。

   ```
   # smbcontrol all reload-config
   ```

3.13. MACOS クライアント向けの SAMBA の設定

fruit 仮想ファイルシステム (VFS) の Samba モジュールは、Apple サーバーメッセージブロック (SMB) クライアントとの互換性を強化します。

3.13.1. MacOS クライアントにファイル共有を提供する Samba 設定の最適化

fruit モジュールは、Samba の MacOS クライアントとの互換性を強化します。Samba サーバーでホストされているすべての共有に対してモジュールを設定して、MacOS クライアント向けにファイル共有を最適化できます。

注記

fruit モジュールをシステム全体で有効にします。MacOS を使用するクライアントは、クライアントがサーバーへの最初の接続を確立する時に、サーバーメッセージブロックバージョン 2 (SMB2) Apple (AAPL) プロトコル拡張をネゴシエートします。クライアントが最初に AAPL 拡張機能を有効にせずに共有に接続すると、クライアントはサーバーの共有に拡張機能を使用しません。

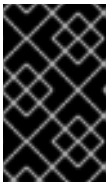
前提条件

- Samba が、ファイルサーバーとして設定されている。

手順

1. **/etc/samba/smb.conf** ファイルを編集し、**[global]** セクションの VFS モジュール **fruit** および **streams_xattr** を有効にします。

```
vfs objects = fruit streams_xattr
```



重要

streams_xattr を有効にする前に、**fruit** モジュールを有効にする必要があります。**fruit** モジュールは、別のデータストリーム (ADS) を使用します。このため、**streams_xattr** モジュールも有効にする必要があります。

2. 必要に応じて、共有で macOS Time Machine のサポートを提供する場合は、**/etc/samba/smb.conf** ファイルの共有設定に次の設定を追加します。

```
fruit:time machine = yes
```

3. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

4. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

関連情報

- **vfs_fruit(8)** の man ページ
- ファイル共有の設定
  - POSIX ACL を使用した Samba ファイル共有の設定
  - Windows ACL を使用する共有の設定

3.14. SMBCLIENT ユーティリティーを使用した SMB 共有へのアクセス

smbclient ユーティリティーを使用すると、コマンドラインの FTP クライアントと同様に、SMB サーバーのファイル共有にアクセスできます。たとえば、ファイルを共有にアップロードしたり、共有からダウンロードしたりできます。

前提条件

- **samba-client** パッケージがインストールされている。

3.14.1. smbclient 対話モードの動作

たとえば、**DOMAIN\user** アカウントを使用して **サーバー** でホストされる **example** 共有に認証するには、以下のコマンドを実行します。

```
# smbclient -U "DOMAIN\user" //server/example
Enter domain\user's password:
Try "help" to get a list of possible commands.
smb: \>
```

smbclient が共有に正常に接続すると、ユーティリティーはインタラクティブモードになり、以下のプロンプトが表示されます。

```
smb: \>
```

対話式シェルで利用可能なすべてのコマンドを表示するには、以下のコマンドを実行します。

```
smb: \> help
```

特定のコマンドのヘルプを表示するには、以下のコマンドを実行します。

```
smb: \> help command_name
```

関連情報

- **smbclient(1)** man ページ

3.14.2. 対話モードでの smbclient の使用

-c パラメーターを指定せずに **smbclient** を使用すると、ユーティリティーは対話モードを開始します。以下の手順では、SMB 共有に接続し、サブディレクトリーからファイルをダウンロードする方法を説明します。

手順

1. 共有に接続します。

   ```
   # smbclient -U "DOMAIN\user_name" //server_name/share_name
   ```

2. **/example/** ディレクトリーに移動します。

   ```
   smb: \> d /example/
   ```

3. ディレクトリー内のファイルをリスト表示します。

```
smb: \example\> ls
  .                   D        0  Thu Nov 1 10:00:00 2018
  ..                  D        0  Thu Nov 1 10:00:00 2018
  example.txt         N  1048576  Thu Nov 1 10:00:00 2018

        9950208 blocks of size 1024. 8247144 blocks available
```

4. **example.txt** ファイルをダウンロードします。

```
smb: \example\> get example.txt
getting file \directory\subdirectory\example.txt of size 1048576 as example.txt (511975,0
KiloBytes/sec) (average 170666,7 KiloBytes/sec)
```

5. 共有から切断します。

```
smb: \example\> exit
```

3.14.3. スクリプトモードでの smbclient の使用

-c パラメーターを **smbclient** に渡すと、リモートの SMB 共有でコマンドを自動的に実行できます。これにより、スクリプトで **smbclient** を使用できます。

以下の手順では、SMB 共有に接続し、サブディレクトリーからファイルをダウンロードする方法を説明します。

手順

- 以下のコマンドで共有に接続して **example** ディレクトリーに移動し、**example.txt** ファイルをダウンロードします。

```
# smbclient -U DOMAIN\user_name //server_name/share_name -c "cd /example/ ; get
example.txt ; exit"
```

3.15. プリントサーバーとしての SAMBA の設定

Samba をプリントサーバーとして設定すると、ネットワーク上のクライアントが Samba を使用して印刷できます。さらに、Windows クライアントは、(Samba サーバーが設定されている場合は) Samba サーバーからドライバーをダウンロードすることもできます。

このセクションの一部は、Samba Wiki に公開されているドキュメント Setting up Samba as a Print Server に掲載されています。ライセンスは、CC BY 4.0 にあります。著者および貢献者は、Wiki ページの history タブを参照してください。

前提条件

Samba が、以下のいずれかのモードで設定されている。

- スタンドアロンサーバー
- ドメインメンバー

3.15.1. Samba でのプリントサーバーのサポートの有効化

デフォルトでは、プリントサーバーサポートは Samba で有効になっていません。Samba をプリントサーバーとして使用するには、Samba を適切に設定する必要があります。

注記

印刷ジョブとプリンター操作には、リモートプロシージャコール (RPC) が必要です。デフォルトでは、Samba は RPC を管理するためにオンデマンドで **rpcd_spoolss** サービスを開始します。最初の RPC 呼び出し中、または CUPS でプリンターリストを更新するときに、Samba は CUPS からプリンター情報を取得します。これには、プリンターごとに約 1 秒かかる場合があります。そのため、プリンターが 50 台を超える場合は、**rpcd_spoolss** 設定を調整してください。

前提条件

- プリンターが CUPS サーバーで設定されている。
  CUPS でプリンターを設定する方法は、プリントサーバーの CUPS Web コンソール (https://**print_server_host_name**:631/help) で提供されているドキュメントを参照してください。

手順

1. **/etc/samba/smb.conf** ファイルを編集します。

   a. **[printers]** セクションを追加して、Samba で印刷バックエンドを有効にします。

   ```
   [printers]
           comment = All Printers
           path = /var/tmp/
           printable = yes
           create mask = 0600
   ```

   重要

   [printers] 共有名はハードコーディングされており、変更はできません。

   b. CUPS サーバーが別のホストまたはポートで実行されている場合は、**printers** セクションで設定を指定します。

   ```
   cups server = printserver.example.com:631
   ```

   c. 多数のプリンターがある場合は、待機秒数を CUPS に接続されているプリンターの数よりも大きい値に設定します。たとえば、100 台のプリンターがある場合は、**[global]** セクションに次のように設定します。

   ```
   rpcd_spoolss:idle_seconds = 200
   ```

   この設定が環境内でスケーリングされない場合は、**[global]** セクションで **rpcd_spoolss** ワーカーの数も増やします。

   ```
   rpcd_spoolss:num_workers = 10
   ```

デフォルトでは、**rpcd_spoolss** は 5 つのワーカーを開始します。

2. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

3. **firewall-cmd** ユーティリティーを使用して必要なポートを開き、ファイアウォール設定を再読み込みします。

```
# firewall-cmd --permanent --add-service=samba
# firewall-cmd --reload
```

4. **smb** サービスを再起動します。

```
# systemctl restart smb
```

サービスを再起動すると、Samba は CUPS バックエンドに設定したすべてのプリンターを自動的に共有します。特定のプリンターのみを手動で共有する場合は、特定のプリンターの手動共有 を参照してください。

検証

- 印刷ジョブを送信します。たとえば、PDF ファイルを印刷するには、次のように入力します。

```
# smbclient -Uuser //sambaserver.example.com/printer_name -c "print example.pdf"
```

3.15.2. 特定のプリンターの手動共有

Samba をプリントサーバーとして設定している場合、Samba は、デフォルトで CUPS バックエンドで設定されたプリンターをすべて共有します。以下の手順では、特定のプリンターのみを共有する方法を説明します。

前提条件

- Samba がプリントサーバーとして設定されている。

手順

1. **/etc/samba/smb.conf** ファイルを編集します。
   a. **[global]** セクションで、以下の設定で自動プリンター共有を無効にします。

```
load printers = no
```

   b. 共有するプリンターごとにセクションを追加します。たとえば、Samba で CUPS バックエンドで **example** という名前のプリンターを **Example-Printer** として共有するには、以下のセクションを追加します。

```
[Example-Printer]
        path = /var/tmp/
        printable = yes
        printer name = example
```

各プリンターに個別のスプールディレクトリーは必要ありません。**[printers]** セクションに設定したのと同じ spool ディレクトリーを、プリンターの **path** パラメーターに設定できます。

2. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

3. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

3.16. SAMBA プリントサーバーでの WINDOWS クライアント用の自動プリンタードライバーダウンロードの設定

Windows クライアント用に Samba プリントサーバーを実行している場合は、ドライバーをアップロードし、プリンターを事前設定できます。ユーザーがプリンターに接続すると、Windows により、ドライバーが自動的にクライアントにダウンロードされ、インストールされます。ユーザーがインストールするのに、ローカル管理者の権限を必要としません。また、Windows は、トレイの数などの事前設定済みのドライバー設定を適用します。

このセクションの一部は、Samba Wiki で公開されているドキュメント Setting up Automatic Printer Driver Downloads for Windows Clients に掲載されています。ライセンスは、CC BY 4.0 にあります。著者および貢献者は、Wiki ページの history タブを参照してください。

前提条件

- Samba がプリントサーバーとして設定されている。

3.16.1. プリンタードライバーに関する基本情報

本セクションでは、プリンタードライバーに関する一般的な情報を説明します。

対応しているドライバーモデルのバージョン
Samba は、Windows 2000 以降および Windows Server 2000 以降でサポートされているプリンタードライバーのモデルバージョン 3 のみに対応します。Samba は、Windows 8 および Windows Server 2012 で導入されたドライバーモデルのバージョン 4 には対応していません。ただし、これ以降の Windows バージョンは、バージョン 3 のドライバーにも対応しています。

パッケージ対応ドライバー
Samba は、パッケージ対応ドライバーに対応していません。

アップロードするプリンタードライバーの準備
Samba プリントサーバーにドライバーをアップロードする場合は、以下を行います。

- ドライバーが圧縮形式で提供されている場合は、ドライバーをデプロイメントします。
- 一部のドライバーでは、Windows ホストにドライバーをローカルにインストールするセットアップアプリケーションを起動する必要があります。特定の状況では、インストーラーはセットアップの実行中にオペレーティングシステムの一時フォルダーに個別のファイルを抽出します。アップロードにドライバーファイルを使用するには、以下のコマンドを実行します。

  a. インストーラーを起動します。

  b. 一時フォルダーから新しい場所にファイルをコピーします。

c. インストールをキャンセルします。

プリントサーバーへのアップロードをサポートするドライバーは、プリンターの製造元にお問い合わせください。

クライアントに 32 ビットおよび 64 ビットのプリンター用ドライバーを提供
32 ビットと 64 ビットの両方の Windows クライアントのプリンターにドライバーを提供するには、両方のアーキテクチャーに対して、同じ名前のドライバーをアップロードする必要があります。たとえば、32 ビットのドライバー **Example PostScript** および 64 ビットのドライバー **Example PostScript (v1.0)** をアップロードする場合は、その名前が一致しません。その結果、ドライバーのいずれかをプリンターに割り当てることしかできなくなり、両方のアーキテクチャーでそのドライバーが使用できなくなります。

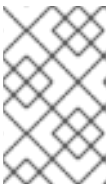
3.16.2. ユーザーがドライバーをアップロードおよび事前設定できるようにする

プリンタードライバーをアップロードおよび事前設定できるようにするには、ユーザーまたはグループに **SePrintOperatorPrivilege** 特権が付与されている必要があります。**printadmin** グループにユーザーを追加する必要があります。Red Hat Enterprise Linux に **samba** パッケージをインストールすると、このグループが自動的に作成されます。**printadmin** グループには、1000 未満で利用可能な一番小さい動的システムの GID が割り当てられます。

手順

1. たとえば、**SePrintOperatorPrivilege** 権限を **printadmin** グループに付与するには、以下のコマンドを 実行します。

```
# net rpc rights grant "printadmin" SePrintOperatorPrivilege -U
"DOMAIN\administrator"
Enter DOMAIN\administrator's password:
Successfully granted rights.
```



注記

ドメイン環境では、**SePrintOperatorPrivilege** をドメイングループに付与します。これにより、ユーザーのグループメンバーシップを更新し、権限を集中的に管理できます。

2. **SePrintOperatorPrivilege** が付与されているユーザーとグループのリストを表示するには、以下を実行します。

```
# net rpc rights list privileges SePrintOperatorPrivilege -U "DOMAIN\administrator"
Enter administrator's password:
SePrintOperatorPrivilege:
  BUILTIN\Administrators
  DOMAIN\printadmin
```

3.16.3. print$ 共有の設定

Windows のオペレーティングシステムは、プリントサーバーの共有 **print$** から、プリンタードライバーをダウンロードします。この共有名は Windows でハードコーディングされており、変更はできません。

以下の手順は、**/var/lib/samba/drivers/** ディレクトリーを **print$** として共有し、ローカルの **printadmin** グループのメンバーがプリンタードライバーをアップロードすることを有効にする方法を説明します。

手順

1. **[print$]** セクションを **/etc/samba/smb.conf** ファイルに追加します。

```
[print$]
        path = /var/lib/samba/drivers/
        read only = no
        write list = @printadmin
        force group = @printadmin
        create mask = 0664
        directory mask = 2775
```

以下の設定を使用します。

- **printadmin** グループのメンバーだけがプリンタードライバーを共有にアップロードできます。
- 新規に作成されたファイルおよびディレクトリーのグループは **printadmin** に設定されます。
- 新規ファイルの権限は **664** に設定されます。
- 新しいディレクトリーの権限は、**2775** に設定されます。

2. 全プリンター向けに 64 ビットドライバーのみをアップロードするには、**/etc/samba/smb.conf** ファイルの **[global]** セクションに以下の設定を含めます。

```
spoolss: architecture = Windows x64
```

この設定がないと、少なくとも 32 ビットバージョンでアップロードしたドライバーのみが表示されます。

3. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

4. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

5. **printadmin** グループが存在しない場合は作成します。

```
# groupadd printadmin
```

6. **SePrintOperatorPrivilege** 権限を、**printadmin** グループに付与します。

```
# net rpc rights grant "printadmin" SePrintOperatorPrivilege -U
"DOMAIN\administrator"
Enter DOMAIN\administrator's password:
Successfully granted rights.
```

7. SELinux を、**enforcing** モードで実行する場合は、そのディレクトリーに **samba_share_t** コンテキストを設定します。

```
# semanage fcontext -a -t samba_share_t "/var/lib/samba/drivers(/.*)?"

# restorecon -Rv /var/lib/samba/drivers/
```

8. **/var/lib/samba/drivers/** ディレクトリーに権限を設定します。

- POSIX ACL を使用する場合は、以下を設定します。

```
# chgrp -R "printadmin" /var/lib/samba/drivers/
# chmod -R 2775 /var/lib/samba/drivers/
```

- Windows ACL を使用する場合は、以下を設定します。

プリンシパル	アクセス	適用先
CREATOR OWNER	完全な制御	サブフォルダーおよびファイルのみ
認証されたユーザー	読み取りおよび実行、フォルダーのコンテンツのリスト表示、読み取り	このフォルダー、サブフォルダー、およびファイル
printadmin	完全な制御	このフォルダー、サブフォルダー、およびファイル

Windows での ACL の設定に関する詳細は、Windows のドキュメントを参照してください。

関連情報

- ユーザーがドライバーをアップロードおよび事前設定できるようにする手順

3.16.4. クライアントが Samba プリントサーバーを信頼できるようにする GPO の作成

セキュリティー上の理由から、最新の Windows オペレーティングシステムでは、クライアントが、信頼できないサーバーから、パッケージ対応ではないプリンタードライバーをダウンロードできないようにします。プリントサーバーが AD のメンバーである場合は、Samba サーバーを信頼するために、ドメインに Group Policy Object (GPO) を作成できます。

前提条件

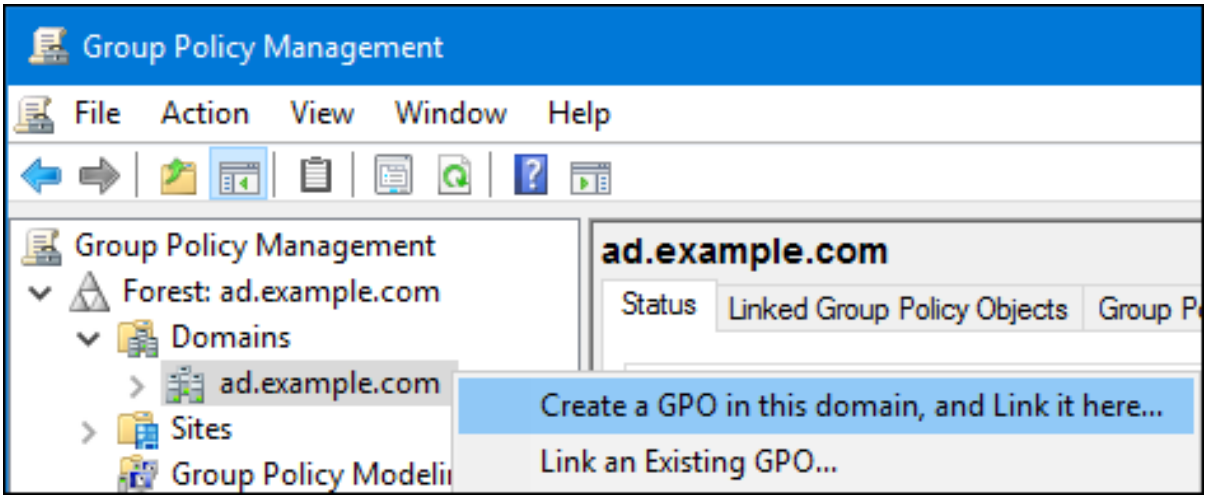
- Samba プリントサーバーが、AD ドメインのメンバーである。
- GPO の作成に使用する Windows コンピューターに、RSAT (Windows Remote Server Administration Tools) がインストールされている。詳細は、Windows のドキュメントを参照してください。

手順

1. AD ドメインの **管理者** ユーザーなど、グループポリシーの編集が可能なアカウントを使用して、Windows コンピューターにログインします。

2. **Group Policy Management Console** を開きます。

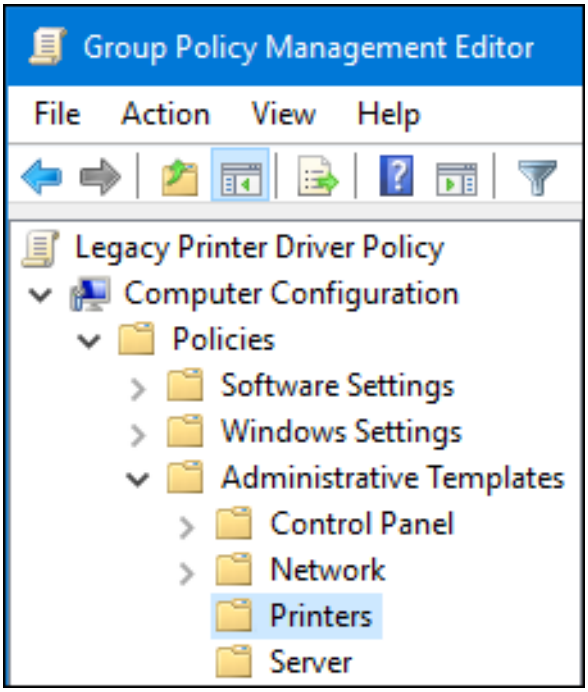
3. AD ドメインを右クリックし、**Create a GPO in this domain, and Link it here** を選択します。



4. **Legacy Printer Driver Policy** などの GPO の名前を入力して、**OK** をクリックします。新しい GPO がドメインエントリーの下に表示されます。

5. 新たに作成した GPO を右クリックして **Edit** を選択し、**Group Policy Management Editor** を開きます。

6. **Computer Configuration** → **Policies** → **Administrative Templates** → **Printers** の順にクリックします。

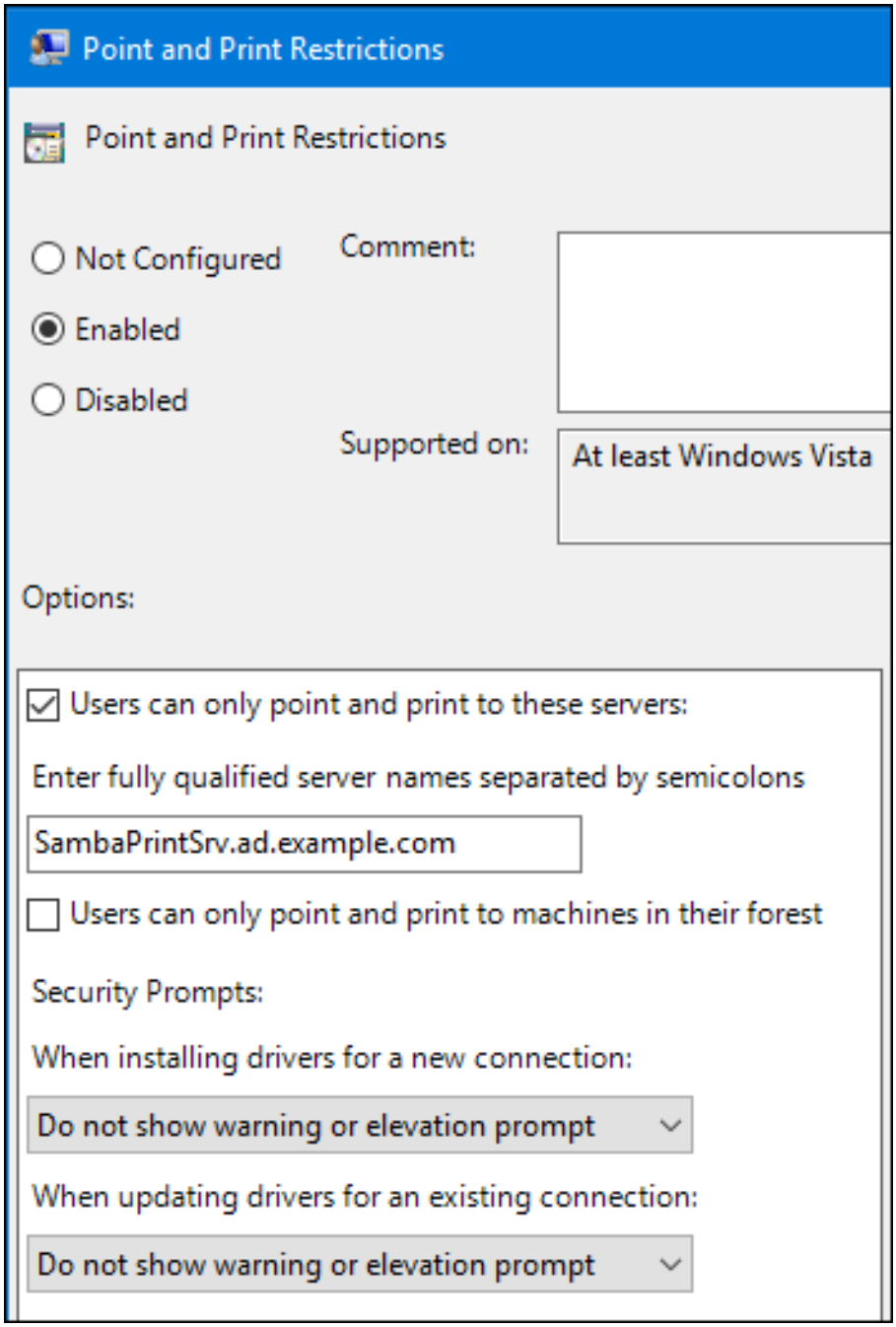


7. ウィンドウの右側で、**Point and Print Restriction** をダブルクリックして、ポリシーを編集します。

   a. ポリシーを有効にし、以下のオプションを設定します。

      i. **Users can only point and print to these servers** を選択し、このオプションの横にあるフィールドに、Samba プリントサーバーの完全修飾ドメイン名 (FQDN) を入力します。

      ii. **Security Prompts** の両チェックボックスで、**Do not show warning or elevation prompt** を選択します。

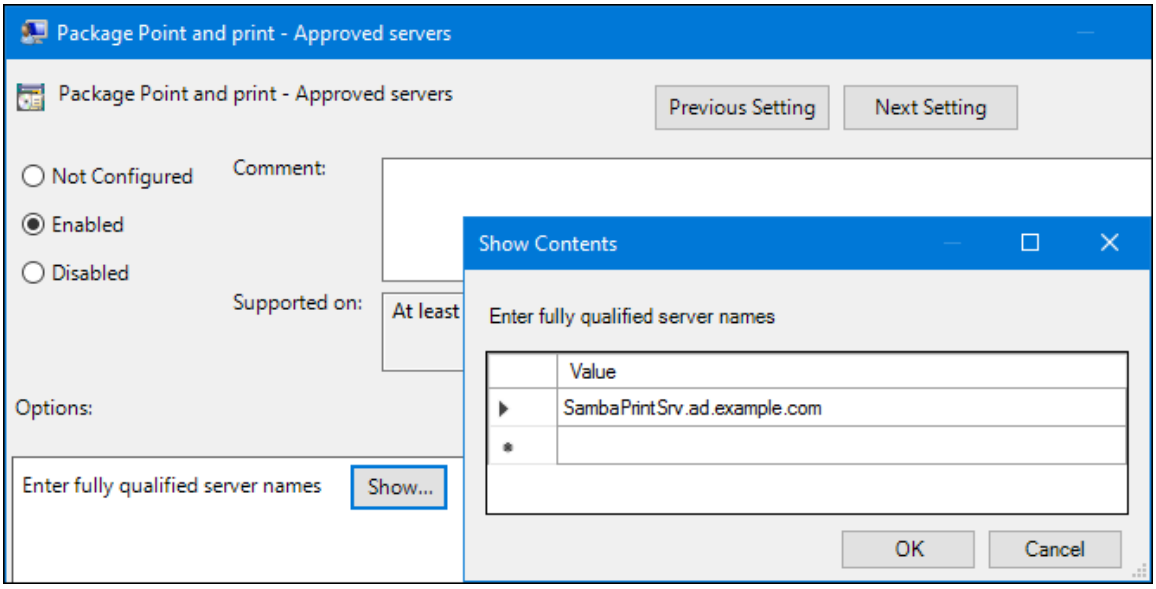


b. OK をクリックします。

8. **Package Point and Print - Approved servers** をダブルクリックして、ポリシーを編集します。

   a. ポリシーを有効にして、**Show** ボタンをクリックします。

   b. Samba プリントサーバーの FQDN を入力します。



c. **OK** をクリックして、**Show Contents** ウィンドウとポリシーのプロパティーウィンドウの両方を閉じます。

9. **Group Policy Management Editor** を閉じます。

10. **Group Policy Management Console** を閉じます。

Windows ドメインメンバーがこのグループポリシーを適用すると、ユーザーがプリンターに接続する際に、プリンタードライバーが Samba サーバーから自動的にダウンロードされます。

関連情報

- グループポリシーの使用については、Windows のドキュメントを参照してください。

3.16.5. ドライバーのアップロードおよびプリンターの事前設定

Windows クライアントで **Print Management** アプリケーションを使用してドライバーをアップロードし、Samba プリントサーバーでホストされるプリンターを事前設定します。詳細は、Windows のドキュメントを参照してください。

3.17. FIPS モードが有効なサーバーでの SAMBA の実行

本セクションでは、FIPS モードが有効な状態で Samba を実行する制限の概要を説明します。また、Samba を実行している Red Hat Enterprise Linux ホストで FIPS モードを有効にする手順も提供します。

3.17.1. FIPS モードでの Samba の使用制限

以下の Samba モードと機能は、指定された条件下で FIPS モードで動作します。

- Samba は、AES 暗号化を使用する Kerberos 認証を使用する Active Directory (AD) または Red Hat Identity Management (IdM) 環境でのみ、ドメインメンバーとして使用できます。
- Active Directory ドメインメンバーのファイルサーバーとして Samba を使用する。ただし、クライアントは Kerberos を使用してサーバーに対して認証する必要があります。

FIPS のセキュリティーが強化されているため、FIPS モードが有効な場合は、以下の Samba 機能およびモードは機能しません。

- RC4 暗号がブロックされていることによる NT LAN Manager (NTLM) 認証
- サーバーメッセージブロックバージョン 1 (SMB1) プロトコル
- NTLM 認証を使用することによるスタンドアロンファイルサーバーモード
- NT4- スタイルのドメインコントローラー
- NT4- スタイルのドメインメンバーRed Hat は、IdM がバックグラウンドで使用するプライマリードメインコントローラー (PDC) 機能のサポートを継続することに留意してください。
- Samba サーバーに対するパスワード変更Active Directory ドメインコントローラーに対して Kerberos を使用してパスワードの変更のみを実行できます。

以下の機能は FIPS モードでテストされていないため、Red Hat ではサポートされていません。

- プリントサーバーとしての Samba の実行

3.17.2. FIPS モードでの Samba の使用

Samba を実行する RHEL ホストで FIPS モードを有効にすることができます。

前提条件

- Samba が Red Hat Enterprise Linux ホストに設定されている。
- Samba は、FIPS モードでサポートされるモードで実行する。

手順

1. RHEL で FIPS モードを有効にします。

```
# fips-mode-setup --enable
```

2. サーバーを再起動します。

```
# reboot
```

3. **testparm** ユーティリティーを使用して、設定を確認します。

```
# testparm -s
```

コマンドがエラーや非互換性を表示する場合は、Samba が正常に機能するように修正してください。

関連情報

- 「FIPS モードでの Samba の使用制限」

3.18. SAMBA サーバーのパフォーマンスチューニング

特定の状況で Samba のパフォーマンスを向上させることができる設定と、パフォーマンスに悪影響を与える可能性がある設定について説明します。

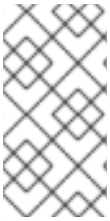
このセクションの一部は、Samba Wiki に公開されているドキュメント Performance Tuning に掲載されています。ライセンスは、CC BY 4.0 にあります。著者および貢献者は、Wiki ページの history タブを参照してください。

前提条件

- Samba が、ファイルサーバーまたはプリントサーバーとして設定されている。

3.18.1. SMB プロトコルバージョンの設定

新しい SMB バージョンごとに機能が追加され、プロトコルのパフォーマンスが向上します。最新の Windows および Windows Server オペレーティングシステムは、常に最新のプロトコルバージョンに対応しています。Samba がプロトコルの最新バージョンも使用している場合は、Samba に接続する Windows クライアントで、このパフォーマンス改善を活用できます。Samba では、server max protocol のデフォルト値が、対応している安定した SMB プロトコルの最新バージョンに設定されます。



注記

常に最新の安定した SMB プロトコルバージョンを有効にするには、**server max protocol** パラメーターを設定しないでください。このパラメーターを手動で設定する場合は、最新のプロトコルバージョンを有効にするために、それぞれ新しいバージョンの SMB プロトコルで設定を変更する必要があります。

次の手順では、**server max protocol** パラメーターでデフォルト値を使用する方法を説明します。

手順

1. **/etc/samba/smb.conf** ファイルの **[global]** セクションから、**server max protocol** パラメーターを削除します。
2. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

3.18.2. 大量のファイルを含むディレクトリーとの共有の調整

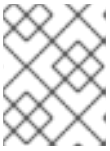
Linux は、大文字と小文字を区別するファイル名に対応しています。このため、Samba はファイルの検索時またはアクセス時に、大文字と小文字のファイル名のディレクトリーをスキャンする必要があります。小文字または大文字のいずれかでのみ新しいファイルを作成するように共有を設定すると、パフォーマンスが向上します。

前提条件

- Samba が、ファイルサーバーとして設定されている。

手順

1. 共有の全ファイルの名前を小文字に変更します。



注記

この手順の設定を使用すると、小文字以外の名前が付けられたファイルは表示されなくなります。

2. 共有のセクションに、以下のパラメーターを設定します。

```
case sensitive = true
default case = lower
preserve case = no
short preserve case = no
```

パラメーターの詳細は、man ページの **smb.conf(5)** を参照してください。

3. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

4. Samba 設定を再読み込みします。

```
# smbcontrol all reload-config
```

この設定が適用されと、この共有に新たに作成されるすべてのファイルの名前が小文字になります。この設定により、Samba はディレクトリーを大文字と小文字で分けたスキャンが不要になり、パフォーマンスが向上します。

3.18.3. パフォーマンスが低下する可能性のある設定

デフォルトでは、Red Hat Enterprise Linux のカーネルは、ネットワークパフォーマンスが高くなるように調整されています。たとえば、カーネルはバッファーサイズに自動チューニングメカニズムを使用しています。**/etc/samba/smb.conf** ファイルに **socket options** パラメーターを設定すると、このカーネル設定が上書きされます。その結果、このパラメーターの設定により、ほとんどの場合は、Samba ネットワークのパフォーマンスが低下します。

カーネルの最適化された設定を使用するには、**/etc/samba/smb.conf** の **[global]** セクションから **socket options** パラメーターを削除します。

3.19. SAMBA がデフォルトの SMB バージョンよりも前のバージョンのクライアントと互換対応するような設定

Samba は、サポート対象の最小サーバーメッセージブロック (SMB) バージョンに妥当で安全なデフォルト値を使用します。ただし、以前の SMB バージョンを必要とするクライアントがある場合は、Samba を設定してサポートできます。

3.19.1. Samba サーバーで対応している最小 SMB プロトコルバージョンの設定

Samba では、**/etc/samba/smb.conf** ファイルの **server min protocol** パラメーターは、Samba サーバーが対応する SMB (server message block) プロトコルの最小バージョンを定義します。SMB プロトコルの最小バージョンを変更できます。

注記

デフォルトでは、RHEL 8.2 以降の Samba では、SMB2 以降のプロトコルバージョンのみに対応します。Red Hat は、非推奨の SMB1 プロトコルを使用することは推奨されません。ただし、お使いの環境で SMB1 が必要な場合は、**server min protocol** パラメーターを手動で **NT1** に設定して、SMB1 を再度有効にできます。

前提条件

- Samba がインストールされ、設定されている。

手順

1. **/etc/samba/smb.conf** ファイルを編集し、**server min protocol** パラメーターを追加して、そのサーバーが対応する最小 SMB プロトコルバージョンに設定できます。たとえば、最小の SMB プロトコルバージョンを **SMB3** に設定するには、以下を追加します。

```
server min protocol = SMB3
```

2. **smb** サービスを再起動します。

```
# systemctl restart smb
```

関連情報

- **smb.conf(5)** man ページ

3.20. 頻繁に使用される SAMBA コマンドラインユーティリティー

本章では、Samba サーバーで作業する場合によく使用されるコマンドを説明します。

3.20.1. net ads join コマンドおよび net rpc join コマンドの使用

net ユーティリティーの **join** サブコマンドを使用すると、Samba を AD ドメインまたは NT4 ドメインに参加させることができます。ドメインに参加するには、**/etc/samba/smb.conf** ファイルを手動で作成し、必要に応じて PAM などの追加設定を更新する必要があります。

重要

Red Hat は、**realm** ユーティリティーを使用してドメインに参加させることを推奨します。**realm** ユーティリティーは、関連するすべての設定ファイルを自動的に更新します。

手順

1. 以下の設定で **/etc/samba/smb.conf** ファイルを手動で作成します。

   - AD ドメインメンバーの場合:

```
[global]
workgroup = domain_name
security = ads
```

```
passdb backend = tdbsam
realm = AD_REALM
```

- NT4 ドメインメンバーの場合:

```
[global]
workgroup = domain_name
security = user
passdb backend = tdbsam
```

2. **/etc/samba/smb.conf** ファイルの **[global]** セクションに、* デフォルトドメインおよび参加するドメイン用の ID マッピング設定を追加します。

3. **/etc/samba/smb.conf** ファイルを検証します。

```
# testparm
```

4. ドメイン管理者としてドメインに参加します。

- AD ドメインに参加するには、以下のコマンドを実行します。

```
# net ads join -U "DOMAIN\administrator"
```

- NT4 ドメインに参加するには、以下のコマンドを実行します。

```
# net rpc join -U "DOMAIN\administrator"
```

5. **/etc/nsswitch.conf** ファイルのデータベースエントリー **passwd** および **group** に **winbind** ソースを追加します。

```
passwd:     files winbind
group:      files winbind
```

6. **winbind** サービスを有効にして起動します。

```
# systemctl enable --now winbind
```

7. 必要に応じて、**authselect** ユーティリティーを使用して PAM を設定します。
詳細は、man ページの **authselect(8)** を参照してください。

8. AD 環境では、必要に応じて Kerberos クライアントを設定します。
詳細は、Kerberos クライアントのドキュメントを参照してください。

関連情報

- Samba のドメインへの参加
- Samba ID マッピングの理解および設定

3.20.2. net rpc rights コマンドの使用

Windows では、アカウントおよびグループに特権を割り当て、共有での ACL の設定やプリンタードライバーのアップロードなどの特別な操作を実行できます。Samba サーバーでは、**net rpc rights** コマンドを使用して権限を管理できます。

設定可能な権限のリスト表示
利用可能な特権とその所有者をすべて表示するには、**net rpc rights list** コマンドを使用します。以下に例を示します。

```
# net rpc rights list -U "DOMAIN\administrator"
Enter DOMAIN\administrator's password:
     SeMachineAccountPrivilege  Add machines to domain
      SeTakeOwnershipPrivilege  Take ownership of files or other objects
            SeBackupPrivilege  Back up files and directories
           SeRestorePrivilege  Restore files and directories
     SeRemoteShutdownPrivilege  Force shutdown from a remote system
      SePrintOperatorPrivilege  Manage printers
         SeAddUsersPrivilege  Add users and groups to the domain
       SeDiskOperatorPrivilege  Manage disk shares
         SeSecurityPrivilege  System security
```

特権の付与
アカウントまたはグループへの特権を付与するには、**net rpc rights grant** コマンドを使用します。

たとえば、**SePrintOperatorPrivilege** 権限を、**DOMAIN\printadmin** グループに付与します。

```
# net rpc rights grant "DOMAIN\printadmin" SePrintOperatorPrivilege -U
"DOMAIN\administrator"
Enter DOMAIN\administrator's password:
Successfully granted rights.
```

特権の取り消し
アカウントまたはグループから特権を取り消すには、**net rpc rights revoke** コマンドを使用します。

たとえば、**DOMAIN\printadmin** グループから **SePrintOperatorPrivilege** 権限を取り消すには、以下のコマンドを実行します。

```
# net rpc rights remoke "DOMAIN\printadmin" SePrintOperatorPrivilege -U
"DOMAIN\administrator"
Enter DOMAIN\administrator's password:
Successfully revoked rights.
```

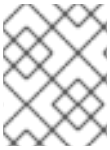
3.20.3. net rpc share コマンドの使用

net rpc share コマンドは、ローカルまたはリモートの Samba または Windows サーバーの共有のリスト表示、追加、および削除を行う機能を提供します。

共有のリスト表示
SMB サーバーの共有をリスト表示するには、**net rpc share list** コマンドを使用します。必要に応じて、**-S server_name** パラメーターをコマンドに渡して、リモートサーバーの共有をリスト表示します。以下に例を示します。

```
# net rpc share list -U "DOMAIN\administrator" -S server_name
Enter DOMAIN\administrator's password:
IPC$
```

```
share_1
share_2
...
```



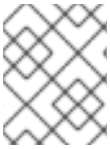
注記

/etc/samba/smb.conf ファイルのセクション内に **browseable = no** が設定されている Samba サーバーでホストされている共有は、出力には表示されません。

共有の追加
net rpc share add コマンドを使用すると、SMB サーバーに共有を追加できます。

たとえば、**C:\example** ディレクトリーを共有するリモートの Windows サーバーに、共有 **example** を追加するには、以下のコマンドを実行します。

```
# net rpc share add example="C:\example" -U "DOMAIN\administrator" -S server_name
```



注記

Windows のディレクトリー名を指定する際は、パスの末尾のバックスラッシュを省略する必要があります。

このコマンドを使用して Samba サーバーに共有を追加するには、以下を行います。

- **-U** パラメーターで指定したユーザーは、出力先サーバーで **SeDiskOperatorPrivilege** 特権が付与されている必要があります。
- 共有セクションを、**/etc/samba/smb.conf** ファイルに追加し、Samba を再読み込みするスクリプトを記述する必要があります。スクリプトは、**/etc/samba/smb.conf** の **[global]** セクションの **add share command** パラメーターで設定する必要があります。詳細は、man ページの **smb.conf(5)** の **add share コマンド** の説明を参照してください。

共有の削除
net rpc share delete コマンドを使用すると、SMB サーバーから共有を削除できます。

たとえば、example という名前の共有を、リモートの Windows サーバーから削除するには、以下のコマンドを実行します。

```
# net rpc share delete example -U "DOMAIN\administrator" -S server_name
```

このコマンドを使用して Samba サーバーから共有を削除するには、以下のコマンドを実行します。

- **-U** パラメーターで指定したユーザーは、**SeDiskOperatorPrivilege** 特権が付与されている必要があります。
- **/etc/samba/smb.conf** ファイルから共有のセクションを削除し、Samba を再読み込みするスクリプトを記述する必要があります。スクリプトは、**/etc/samba/smb.conf** の **[global]** セクションの **delete share command** パラメーターで設定する必要があります。詳細は、man ページの **smb.conf(5)** の **delete share コマンド** の説明を参照してください。

3.20.4. net user コマンドの使用

net user コマンドを使用すると、AD DC または NT4 PDC で以下の操作を実行できます。

- すべてのユーザーアカウントのリストを表示
- ユーザーの追加
- ユーザーの削除

注記

AD ドメイン用の **ads**、NT4 ドメイン用の **rpc** などの接続方法の指定は、ドメインユーザーアカウントをリスト表示する場合にのみ必要です。その他のユーザー関連のサブコマンドは、接続メソッドを自動検出できます。

-U user_name パラメーターをコマンドに渡して、要求されたアクションを実行できるユーザーを指定します。

ドメインユーザーアカウントのリスト表示
AD ドメイン内のユーザーをリスト表示するには、以下を実行します。

```
# net ads user -U "DOMAIN\administrator"
```

NT4 ドメインのユーザーをリスト表示するには、以下を実行します。

```
# net rpc user -U "DOMAIN\administrator"
```

ユーザーアカウントのドメインへの追加
Samba ドメインメンバーの場合は、**net user add** コマンドを使用して、ユーザーアカウントをドメインに追加できます。

たとえば、**user** アカウントをドメインに追加します。

1. 以下のアカウントを追加します。

```
# net user add user password -U "DOMAIN\administrator"
User user added
```

2. 必要に応じて、リモートプロシージャコール (RPC) シェルを使用して、AD DC または NT4 PDC でアカウントを有効にします。以下に例を示します。

```
# net rpc shell -U DOMAIN\administrator -S DC_or_PDC_name
Talking to domain DOMAIN (S-1-5-21-1424831554-512457234-5642315751)

net rpc> user edit disabled user: no
Set user's disabled flag from [yes] to [no]

net rpc> exit
```

ドメインからのユーザーアカウントの削除
Samba ドメインメンバーの場合は、**net user delete** コマンドを使用して、ドメインからユーザーアカウントを削除できます。

たとえば、ドメインから **user** アカウントを削除するには、以下のコマンドを実行します。

```
# net user delete user -U "DOMAIN\administrator"
User user deleted
```

-

3.20.5. rpcclient ユーティリティーの使用

rpcclient ユーティリティーを使用すると、ローカルまたはリモートの SMB サーバーでクライアント側の Microsoft Remote Procedure Call (MS-RPC) 機能を手動で実行できます。ただし、ほとんどの機能は、Samba が提供する個別のユーティリティーに統合されています。**rpcclient** は、MS-PRC 関数のテストにのみ使用します。

前提条件

- **samba-client** パッケージがインストールされている。

例
たとえば、**rpcclient** ユーティリティーを使用して以下を行うことができます。

- プリンターのスプールサブシステム (SPOOLSS) を管理します。

  例3.7 プリンターへのドライバーの割り当て

  ```
  # rpcclient server_name -U "DOMAIN\administrator" -c 'setdriver "printer_name" "driver_name"'
  Enter DOMAIN\administrators password:
  Successfully set printer_name to driver driver_name.
  ```

- SMB サーバーに関する情報を取得します。

  例3.8 すべてのファイル共有および共有プリンターのリスト表示

  ```
  # rpcclient server_name -U "DOMAIN\administrator" -c 'netshareenum'
  Enter DOMAIN\administrators password:
  netname: Example_Share
   remark:
   path:   C:\srv\samba\example_share\
   password:
  netname: Example_Printer
   remark:
   path:   C:\var\spool\samba\
   password:
  ```

- Security Account Manager Remote (SAMR) プロトコルを使用して操作を実行します。

  例3.9 SMB サーバー上のユーザーのリスト表示

  ```
  # rpcclient server_name -U "DOMAIN\administrator" -c 'enumdomusers'
  Enter DOMAIN\administrators password:
  user:[user1] rid:[0x3e8]
  user:[user2] rid:[0x3e9]
  ```

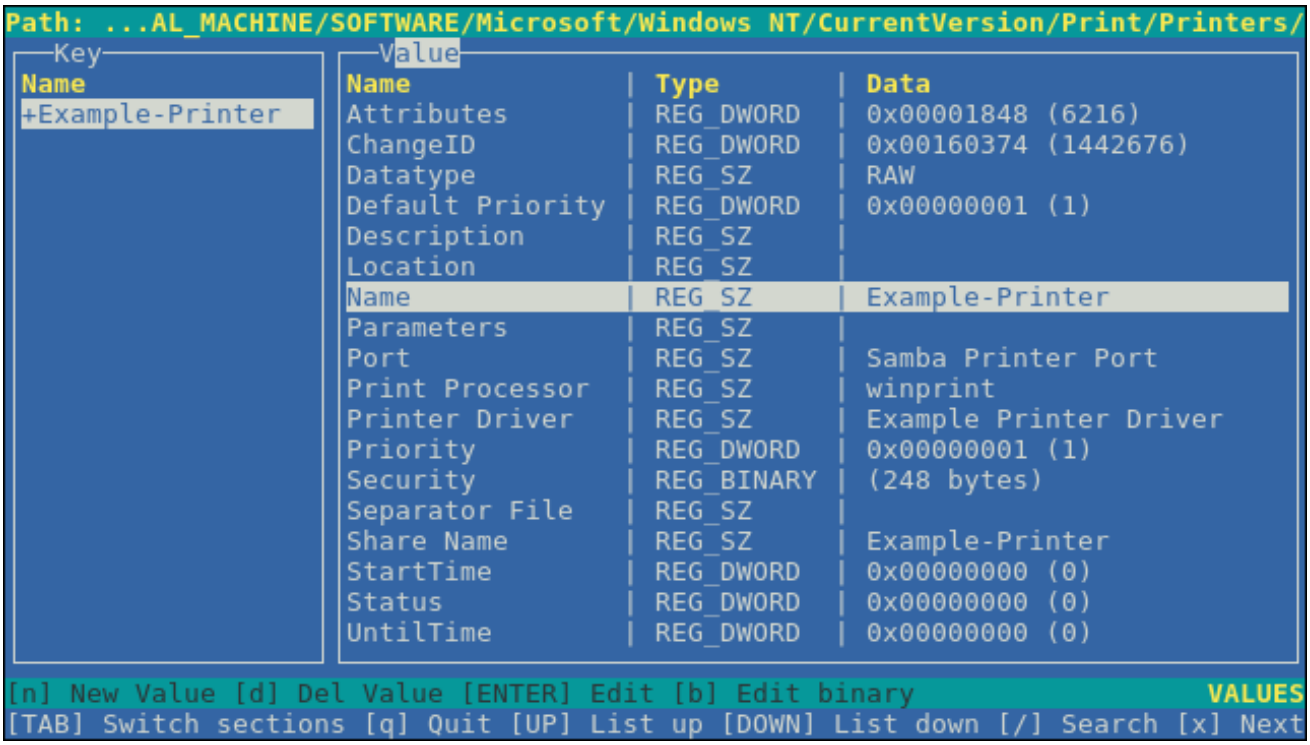
  スタンドアロンサーバーまたはドメインメンバーに対してコマンドを実行すると、ローカルデータベースのユーザーのリストが表示されます。ADDC または NT4 PDC に対してコマンドを実行すると、ドメインユーザーのリストが表示されます。

関連情報

- **rpcclient(1)** man ページ

3.20.6. samba-regedit アプリケーションの使用

プリンター設定などの特定の設定は、Samba サーバーのレジストリーに保存されます。ncurses ベースの **samba-regedit** アプリケーションを使用して、Samba サーバーのレジストリーを編集できます。



前提条件

- **samba-client** パッケージがインストールされている。

手順

アプリケーションを起動するには、次のコマンドを入力します。

```
# samba-regedit
```

次のキーを使用します。

- カーソルを上下に動かして、レジストリーツリーと値の間を移動します。
- **Enter** - キーを開くか、値を編集します。
- **Tab** - **Key** ペインと **Value** ペインを切り替えます。
- **Ctrl**+**C** - アプリケーションを閉じます。

3.20.7. smbcontrol ユーティリティーの使用

smbcontrol ユーティリティーを使用すると、**smbd**、**nmbd**、**winbindd**、またはこのすべてのサービスにコマンドメッセージを送信できます。この制御メッセージは、設定の再読み込みなどのサービスを指示します。

前提条件

- **samba-common-tools** パッケージがインストールされている。

手順

- **reload-config** メッセージタイプを **all** 宛に送信して、**smbd**、**nmbd**、**winbindd** サービスの設定をリロードします。

```
# smbcontrol all reload-config
```

関連情報

- **smbcontrol(1)** man ページ

3.20.8. smbpasswd ユーティリティーの使用

smbpasswd ユーティリティーは、ローカルの Samba データベースでユーザーアカウントおよびパスワードを管理します。

前提条件

- **samba-common-tools** パッケージがインストールされている。

手順

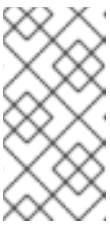
1. ユーザーとしてコマンドを実行すると、**smbpasswd** はコマンドを実行するユーザーの Samba パスワードを変更します。以下に例を示します。

```
[user@server ~]$ smbpasswd
New SMB password: password
Retype new SMB password: password
```

2. **root** で **smbpasswd** を実行すると、たとえば以下のようにユーティリティーを使用できます。

   - 新しいユーザーを作成します。

```
[root@server ~]# smbpasswd -a user_name
New SMB password: password
Retype new SMB password: password
Added user user_name.
```



注記

Samba データベースにユーザーを追加する前に、ローカルのオペレーティングシステムにアカウントを作成する必要があります。基本的なシステム設定の設定の コマンドラインでの新規ユーザーの追加 セクションを参照してください。

   - Samba ユーザーを有効にします。

```
[root@server ~]# smbpasswd -e user_name
Enabled user user_name.
```

- Samba ユーザーを無効にします。

```
[root@server ~]# smbpasswd -x user_name
Disabled user user_name
```

- ユーザーを削除します。

```
[root@server ~]# smbpasswd -x user_name
Deleted user user_name.
```

関連情報

- **smbpasswd(8)** man ページ

3.20.9. smbstatus ユーティリティーの使用

smbstatus ユーティリティーは以下について報告します。

- 各 **smbd** デーモンの PID ごとの接続を Samba サーバーに接続します。このレポートには、ユーザー名、プライマリーグループ、SMB プロトコルのバージョン、暗号、および署名の情報が含まれます。
- Samba 共有ごとの接続このレポートには、**smbd** デーモンの PID、接続しているマシンの IP、接続が確立された時点のタイムスタンプ、暗号、および署名情報が含まれます。
- ロックされたファイルのリスト。レポートエントリーには、日和見ロック (oplock) タイプなどの詳細が含まれます。

前提条件

- **samba** パッケージがインストールされている。
- **smbd** サービスが実行している。

手順

```
# smbstatus

Samba version 4.15.2
PID  Username              Group             Machine                        Protocol Version  Encryption
Signing
----------------------------------------------------------------------------------------------------------------
-
963  DOMAIN\administrator  DOMAIN\domain users  client-pc  (ipv4:192.0.2.1:57786)  SMB3_02
-           AES-128-CMAC

Service  pid  Machine    Connected at                Encryption  Signing:
---------------------------------------------------------------------------
example  969  192.0.2.1  Thu Nov  1 10:00:00 2018 CEST  -           AES-128-CMAC

Locked files:
Pid  Uid    DenyMode   Access    R/W     Oplock      SharePath          Name      Time
```

```
....----------------------------------------------------------------------------------------------------
969  10000  DENY_WRITE 0x120089  RDONLY  LEASE(RWH)  /srv/samba/example  file.txt  Thu
Nov  1 10:00:00 2018
```

関連情報

- **smbstatus(1)** man ページ

3.20.10. smbtar ユーティリティーの使用

smbtar ユーティリティーは、SMB 共有またはそのサブディレクトリーのコンテンツのバックアップを作成し、そのコンテンツを **tar** アーカイブに保存します。または、コンテンツをテープデバイスに書き込むこともできます。

前提条件

- **samba-client** パッケージがインストールされている。

手順

- 以下のコマンドを使用して、**//server/example/** 共有上の **demo** ディレクトリーのコンテンツをバックアップして、**/root/example.tar** アーカイブにコンテンツを保存するには、以下を実行します。

```
# smbtar -s server -x example -u user_name -p password -t /root/example.tar
```

関連情報

- **smbtar(1)** の man ページ

3.20.11. wbinfo ユーティリティーの使用

wbinfo ユーティリティーは、**winbindd** サービスが作成および使用する情報をクエリーし、返します。

前提条件

- **samba-winbind-clients** パッケージがインストールされている。

手順

たとえば、以下のように、**wbinfo** を使用できます。

- ドメインユーザーのリストを表示します。

```
# wbinfo -u
AD\administrator
AD\guest
...
```

- ドメイングループのリストを表示します。

```
# wbinfo -g
AD\domain computers
```

```
AD\domain admins
AD\domain users
...
```

- ユーザーの SID を表示します。

```
# wbinfo --name-to-sid="AD\administrator"
S-1-5-21-1762709870-351891212-3141221786-500 SID_USER (1)
```

- ドメインおよび信頼に関する情報を表示します。

```
# wbinfo --trusted-domains --verbose
Domain Name   DNS Domain          Trust Type  Transitive  In   Out
BUILTIN                           None        Yes         Yes  Yes
server                            None        Yes         Yes  Yes
DOMAIN1       domain1.example.com None        Yes         Yes  Yes
DOMAIN2       domain2.example.com External    No          Yes  Yes
```

関連情報

- **wbinfo(1)** man ページ

3.21. 関連情報

- **smb.conf(5)** man ページ
- **/usr/share/docs/samba-version/** ディレクトリーには、Samba プロジェクトが提供する一般的なドキュメント、サンプルスクリプト、および LDAP スキーマファイルが含まれています。
- Setting up Samba and the Clustered Trivial Database (CDTB) to share directories stored on an GlusterFS volume
- Red Hat Enterprise Linux での SMB 共有のマウント

第4章 BIND DNS サーバーのセットアップおよび設定

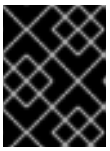
BIND は、Internet Engineering Task Force (IETF) の DNS 標準およびドラフト標準に完全に準拠した機能豊富な DNS サーバーです。たとえば、管理者は BIND を次のように頻繁に使用します。

- ローカルネットワークでの DNS サーバーのキャッシング
- ゾーンの権威 DNS サーバー
- ゾーンに高可用性を提供するセカンダリーサーバー

4.1. SELINUX を使用した BIND の保護または CHANGE-ROOT 環境での BIND の実行に関する考慮事項

BIND インストールを保護するには、次のことができます。

- change-root 環境なしで **named** サービスを実行します。この場合、**enforcing** モードの SELinux は、既知の BIND セキュリティー脆弱性の悪用を防ぎます。デフォルトでは、Red Hat Enterprise Linux は SELinux を **enforcing** モードで使用します。



重要

RHEL で SELinux を **enforcing** モードで使用して BIND を実行すると、change-root 環境で BIND を実行するよりも安全です。

- **name-chroot** サービスを change-root 環境で実行します。
管理者は、change-root 機能を使用して、プロセスとそのサブプロセスのルートディレクトリーが / ディレクトリーとは異なるものであることを定義できます。**named-chroot** サービスを開始すると、BIND はそのルートディレクトリーを **/var/named/chroot/** に切り替えます。その結果、サービスは **mount --bind** コマンドを使用して、**/etc/named-chroot.files** にリストされているファイルおよびディレクトリーを **/var/named/chroot/** で使用できるようにし、プロセスは **/var/named/chroot/** 以外のファイルにアクセスできません。

BIND を使用する場合:

- 通常モードでは、**named** サービスを使用します。
- change-root 環境では、**named-chroot** サービスを使用します。これには、**named-chroot** パッケージを追加でインストールする必要があります。

関連情報

- **named(8)** man ページの **Red Hat SELinux BIND security profile** セクション

4.2. BIND 管理者リファレンスマニュアル

bind パッケージに含まれる総合的な **BIND Administrator Reference Manual** には、次の内容が記載されています。

- 設定例
- 高度な機能に関するドキュメント
- 設定リファレンス

- セキュリティーに関する考慮事項

bind パッケージがインストールされているホストで **BIND Administrator Reference Manual** を表示するには、ブラウザーで **/usr/share/doc/bind/Bv9ARM.html** ファイルを開きます。

4.3. BIND をキャッシュ DNS サーバーとして設定する

デフォルトでは、BIND DNS サーバーは、成功したルックアップと失敗したルックアップを解決してキャッシュします。その後、サービスはキャッシュから同じレコードへの要求に応答します。これにより、DNS ルックアップの速度が大幅に向上します。

前提条件

- サーバーの IP アドレスは静的です。

手順

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

```
# yum install bind bind-utils
```

これらのパッケージは BIND 9.11 を提供します。BIND 9.16 が必要な場合は、**bind9.16** および **bind9.16-utils** パッケージをインストールしてください。

2. BIND を change-root 環境で実行する場合は、**bind-chroot** パッケージをインストールします。

```
# yum install bind-chroot
```

デフォルトである **enforcing** モードで SELinux を使用するホストで BIND を実行すると、より安全になることに注意してください。

3. **/etc/named.conf** ファイルを編集し、**options** ステートメントに次の変更を加えます。

   a. **listen-on** および **listen-on-v6** ステートメントを更新して、BIND がリッスンする IPv4 インターフェイスおよび IPv6 インターフェイスを指定します。

```
listen-on port 53 { 127.0.0.1; 192.0.2.1; };
listen-on-v6 port 53 { ::1; 2001:db8:1::1; };
```

   b. **allow-query** ステートメントを更新して、クライアントがこの DNS サーバーにクエリーを実行できる IP アドレスおよび範囲を設定します。

```
allow-query { localhost; 192.0.2.0/24; 2001:db8:1::/64; };
```

   c. **allow-recursion** ステートメントを追加して、BIND が再帰クエリーを受け入れる IP アドレスおよび範囲を定義します。

```
allow-recursion { localhost; 192.0.2.0/24; 2001:db8:1::/64; };
```

警告

サーバーのパブリック IP アドレスで再帰を許可しないでください。そうしないと、サーバーが大規模な DNS 増幅攻撃の一部になる可能性があります。

d. デフォルトでは、BIND は、ルートサーバーから権限のある DNS サーバーに再帰的にクエリーを実行することにより、クエリーを解決します。または、プロバイダーのサーバーなど、他の DNS サーバーにクエリーを転送するように BIND を設定することもできます。この場合、BIND がクエリーを転送する DNS サーバーの IP アドレスのリストを含む **forwarders** ステートメントを追加します。

```
forwarders { 198.51.100.1; 203.0.113.5; };
```

フォールバック動作として、フォワーダーサーバーが応答しないと、BIND はクエリーを再帰的に解決します。この動作を無効にするには、**forward only;** ステートメントを追加します。

4. **/etc/named.conf** ファイルの構文を確認します。

```
# named-checkconf
```

コマンドが出力を表示しない場合は、構文に間違いがありません。

5. 着信 DNS トラフィックを許可するように **firewalld** ルールを更新します。

```
# firewall-cmd --permanent --add-service=dns
# firewall-cmd --reload
```

6. BIND を開始して有効にします。

```
# systemctl enable --now named
```

change-root 環境で BIND を実行する場合は、**systemctl enable --now named-chroot** コマンドを使用して、サービスを有効にして開始します。

検証

1. 新しく設定した DNS サーバーを使用してドメインを解決します。

```
# dig @localhost www.example.org
...
www.example.org.    86400    IN    A    198.51.100.34

;; Query time: 917 msec
...
```

この例では、BIND が同じホストで実行し、**localhost** インターフェイスでクエリーに応答することを前提としています。

初めてレコードをクエリーした後、BIND はエントリーをそのキャッシュに追加します。

2. 前のクエリーを繰り返します。

```
# dig @localhost www.example.org
...
www.example.org.    85332    IN    A    198.51.100.34

;; Query time: 1 msec
...
```

エントリーがキャッシュされるため、エントリーの有効期限が切れるまで、同じレコードに対するそれ以降のリクエストは大幅に高速化されます。

次のステップ

- この DNS サーバーを使用するようにネットワーク内のクライアントを設定します。DHCP サーバーが DNS サーバー設定をクライアントに提供する場合は、それに応じて DHCP サーバーの設定を更新します。

関連情報

- SELinux を使用した BIND の保護または change-root 環境での BIND の実行に関する考慮事項
- **named.conf(5)** man ページ
- **/usr/share/doc/bind/sample/etc/named.conf**
- BIND 管理者リファレンスマニュアル

4.4. BIND DNS サーバーでのログの設定

bind パッケージによって提供されるデフォルトの **/etc/named.conf** ファイル内の設定は、**default_debug** チャネルを使用し、メッセージのログを **/var/named/data/named.run** ファイルに記録します。**default_debug** チャネルは、サーバーのデバッグレベルがゼロ以外の場合にのみエントリーをログに記録します。

さまざまなチャネルおよびカテゴリーを使用して、BIND を設定して、定義された重大度でさまざまなイベントを個別のファイルに書き込むことができます。

前提条件

- BIND は、たとえばキャッシングネームサーバーとしてすでに設定されています。
- **named** または **named-chroot** サービスが実行しています。

手順

1. **/etc/named.conf** ファイルを編集し、**category** および **channel** フレーズを **logging** ステートメントに追加します。次に例を示します。

```
logging {
    ...

    category notify { zone_transfer_log; };
```

```
    category xfer-in { zone_transfer_log; };
    category xfer-out { zone_transfer_log; };
    channel zone_transfer_log {
       file "/var/named/log/transfer.log" versions 10 size 50m;
       print-time yes;
       print-category yes;
       print-severity yes;
       severity info;
     };

     ...
};
```

この設定例では、BIND はゾーン転送に関連するメッセージのログを **/var/named/log/transfer.log** に記録します。BIND は最大 **10** バージョンのログファイルを作成し、最大サイズが **50** MB に達するとローテーションします。

category 句は、BIND がカテゴリーのメッセージを送信するチャネルを定義します。

channel 句は、バージョン数、最大ファイルサイズ、および BIND がチャネルにログ記録する必要がある重大度レベルを含むログメッセージの宛先を定義します。イベントのタイムスタンプ、カテゴリー、および重大度のログ記録を有効にするなどの追加設定はオプションですが、デバッグ目的で役立ちます。

2. ログディレクトリーが存在しない場合は作成し、このディレクトリーの **named** ユーザーに書き込み権限を付与します。

```
# mkdir /var/named/log/
# chown named:named /var/named/log/
# chmod 700 /var/named/log/
```

3. **/etc/named.conf** ファイルの構文を確認します。

```
# named-checkconf
```

コマンドが出力を表示しない場合は、構文に間違いがありません。

4. BIND を再起動します。

```
# systemctl restart named
```

change-root 環境で BIND を実行する場合は、**systemctl restart named-chroot** コマンドを使用してサービスを再起動します。

検証

- ログファイルの内容を表示します。

```
# cat /var/named/log/transfer.log
...
06-Jul-2022 15:08:51.261 xfer-out: info: client @0x7fecbc0b0700 192.0.2.2#36121/key
example-transfer-key (example.com): transfer of 'example.com/IN': AXFR started: TSIG
example-transfer-key (serial 2022070603)
06-Jul-2022 15:08:51.261 xfer-out: info: client @0x7fecbc0b0700 192.0.2.2#36121/key
example-transfer-key (example.com): transfer of 'example.com/IN': AXFR ended
```

関連情報

- **named.conf(5)** man ページ
- BIND 管理者リファレンスマニュアル

4.5. BIND ACL の作成

BIND の特定の機能へのアクセスを制御することで、サービス拒否 (DoS) などの不正アクセスや攻撃を防ぐことができます。BIND アクセス制御リスト (**acl**) ステートメントは、IP アドレスと範囲のリストです。各 ACL には、指定された IP アドレスと範囲を参照するために **allow-query** などのいくつかのステートメントで使用できるニックネームがあります。

警告

BIND は、ACL で最初に一致したエントリーのみを使用します。たとえば、ACL **{ 192.0.2/24; !192.0.2.1; }** を定義し、IP アドレス **192.0.2.1** でホストが接続すると、2 番目のエントリーでこのアドレスが除外されていてもアクセスが許可されます。

BIND には次の組み込み ACL があります。

- **none**: どのホストとも一致しません。
- **any**: すべてのホストに一致します。
- **localhost**: ループバックアドレス **127.0.0.1** と **::1**、および BIND を実行するサーバー上のすべてのインターフェイスの IP アドレスに一致します。
- **localnets**: ループバックアドレス **127.0.0.1** と **::1**、および BIND を実行するサーバーが直接接続しているすべてのサブネットに一致します。

前提条件

- BIND は、たとえばキャッシングネームサーバーとしてすでに設定されています。
- **named** または **named-chroot** サービスが実行しています。

手順

1. **/etc/named.conf** ファイルを編集して、次の変更を行います。
   a. **acl** ステートメントをファイルに追加します。たとえば、**127.0.0.1**、**192.0.2.0/24**、および **2001:db8:1::/64** に対して **internal-networks** という名前の ACL を作成するには、次のように入力します。

   ```
   acl internal-networks { 127.0.0.1; 192.0.2.0/24; 2001:db8:1::/64; };
   acl dmz-networks { 198.51.100.0/24; 2001:db8:2::/64; };
   ```

   b. ACL のニックネームをサポートするステートメントで使用します。たとえば、次のようになります。

```
allow-query { internal-networks; dmz-networks; };
allow-recursion { internal-networks; };
```

2. **/etc/named.conf** ファイルの構文を確認します。

```
# named-checkconf
```

コマンドが出力を表示しない場合は、構文に間違いがありません。

3. BIND をリロードします。

```
# systemctl reload named
```

change-root 環境で BIND を実行する場合は、**systemctl reload named-chroot** コマンドを使用してサービスをリロードします。

検証

- 設定された ACL を使用する機能をトリガーするアクションを実行します。たとえば、この手順の ACL は、定義された IP アドレスからの再帰クエリーのみを許可します。この場合は、ACL の定義に含まれていないホストで次のコマンドを入力して、外部ドメインの解決を試みます。

```
# dig +short @192.0.2.1 www.example.com
```

コマンドが出力を返さないと、BIND はアクセスを拒否し、ACL は機能しています。クライアントで詳細な出力を得るには、**+short** オプションを指定せずにコマンドを使用します。

```
# dig @192.0.2.1 www.example.com
...
;; WARNING: recursion requested but not available
...
```

関連情報

- The BIND Administrator Reference Manual の **Access control lists** セクション。

4.6. BIND DNS サーバーでのゾーンの設定

DNS ゾーンは、ドメインスペース内の特定のサブツリーのリソースレコードを含むデータベースです。たとえば、**example.com** ドメインを担当している場合は、BIND でそのゾーンを設定できます。その結果、クライアントは **www.example.com** をこのゾーンで設定された IP アドレスに解決できます。

4.6.1. ゾーンファイルの SOA レコード

SOA (Start of Authority) レコードは、DNS ゾーンで必要なレコードです。このレコードは、たとえば、複数の DNS サーバーがゾーンだけでなく DNS リゾルバーに対しても権限を持っている場合に重要です。

BIND の SOA レコードの構文は次のとおりです。

```
name class type mname rname serial refresh retry expire minimum
```

読みやすくするために、管理者は通常、ゾーンファイル内のレコードを、セミコロン (;) で始まるコメントを使用して複数の行に分割します。SOA レコードを分割する場合は、括弧でレコードをまとめることに注意してください。

```
@ IN SOA ns1.example.com. hostmaster.example.com. (
                2022070601 ; serial number
                1d         ; refresh period
                3h         ; retry period
                3d         ; expire time
                3h )       ; minimum TTL
```

重要

完全修飾ドメイン名 (FQDN) の末尾にあるドットに注意してください。FQDN は、ドットで区切られた複数のドメインラベルで設定されます。DNS ルートには空のラベルがあるため、FQDN はドットで終わります。したがって、BIND はゾーン名を末尾のドットなしで名前に追加します。末尾にドットがないホスト名 (例: **ns1.example.com**) は、**ns1.example.com.example.com.** に展開されます。これは、プライマリーネームサーバーの正しいアドレスではありません。

SOA レコードのフィールドは次のとおりです。

- **name**: ゾーンの名前 (つまり **origin**)。このフィールドを **@** に設定すると、BIND はそれを **/etc/named.conf** で定義されたゾーン名に展開します。
- **class**: SOA レコードでは、このフィールドを常に Internet (**IN**) に設定する必要があります。
- **type**: SOA レコードでは、このフィールドを常に **SOA** に設定する必要があります。
- **mname** (マスター名): このゾーンのプライマリーネームサーバーのホスト名。
- **rname** (責任者名): このゾーンの責任者の電子メールアドレス。形式が異なりますのでご注意ください。アットマーク (@) をドット (**.**) に置き換える必要があります。
- **serial**: このゾーンファイルのバージョン番号。セカンダリーネームサーバーは、プライマリーサーバーのシリアル番号の方が大きい場合にのみ、ゾーンのコピーを更新します。
  形式は任意の数値にすることができます。一般的に使用される形式は **<year><month><day><two-digit-number>** です。この形式を使用すると、理論的には、ゾーンファイルを 1 日に 100 回まで変更できます。
- **refresh**: ゾーンが更新された場合は、プライマリーサーバーをチェックする前にセカンダリーサーバーが待機する時間。
- **retry**: 試行が失敗した後、セカンダリーサーバーがプライマリーサーバーへのクエリーを再試行するまでの時間。
- **expire**: 以前の試行がすべて失敗した場合に、セカンダリーサーバーがプライマリーサーバーへのクエリーを停止した後の時間。
- **minimum**: RFC 2308 は、このフィールドの意味を負のキャッシュ時間に変更しました。準拠リゾルバーは、**NXDOMAIN** 名エラーをキャッシュする期間を決定するために使用します。

注記

refresh、**retry**、**expire**、および **maximum** フィールドの数値は、時間を秒単位で定義します。ただし、読みやすくするために、時間の接尾辞 (**m** は分、**h** は時間、**d** は日など) を使用してください。たとえば、**3h** は 3 時間を表します。

関連情報

- RFC 1035: ドメイン名 - 実装および仕様
- RFC 1034: ドメイン名 - 概念および機能
- RFC 2308: DNS クエリーのネガティブキャッシュ (DNS キャッシュ)

4.6.2. BIND プライマリーサーバーでの正引きゾーンの設定

正引きゾーンは、名前を IP アドレスやその他の情報にマップします。たとえば、ドメイン **example.com** を担当している場合は、BIND で転送ゾーンを設定して、**www.example.com** などの名前を解決できます。

前提条件

- BIND は、たとえばキャッシングネームサーバーとしてすでに設定されています。
- **named** または **named-chroot** サービスが実行しています。

手順

1. **/etc/named.conf** ファイルにゾーン定義を追加します。

```
zone "example.com" {
    type master;
    file "example.com.zone";
    allow-query { any; };
    allow-transfer { none; };
};
```

これらの設定により、次が定義されます。

- このサーバーは、**example.com** ゾーンのプライマリーサーバー (**type master**) です。
- **/var/named/example.com.zone** ファイルはゾーンファイルです。この例のように相対パスを設定すると、このパスは、**options** ステートメントの **directory** に設定したディレクトリーからの相対パスになります。
- どのホストもこのゾーンにクエリーを実行できます。または、IP 範囲または BIND アクセス制御リスト (ACL) のニックネームを指定して、アクセスを制限します。
- どのホストもゾーンを転送できません。ゾーン転送を許可するのは、セカンダリーサーバーをセットアップする際に限られ、セカンダリーサーバーの IP アドレスに対してのみ許可します。

2. **/etc/named.conf** ファイルの構文を確認します。

```
# named-checkconf
```

コマンドが出力を表示しない場合は、構文に間違いがありません。

3. たとえば、次の内容で **/var/named/example.com.zone** ファイルを作成します。

```
$TTL 8h
@ IN SOA ns1.example.com. hostmaster.example.com. (
                    2022070601 ; serial number
                    1d         ; refresh period
                    3h         ; retry period
                    3d         ; expire time
                    3h )       ; minimum TTL

             IN NS   ns1.example.com.
             IN MX   10 mail.example.com.

www            IN A    192.0.2.30
www            IN AAAA 2001:db8:1::30
ns1            IN A    192.0.2.1
ns1            IN AAAA 2001:db8:1::1
mail           IN A    192.0.2.20
mail           IN AAAA 2001:db8:1::20
```

このゾーンファイル:

- リソースレコードのデフォルトの有効期限 (TTL) 値を 8 時間に設定します。時間の **h** などの時間接尾辞がない場合、BIND は値を秒として解釈します。
- ゾーンに関する詳細を含む、必要な SOA リソースレコードが含まれています。
- このゾーンの権威 DNS サーバーとして **ns1.example.com** を設定します。ゾーンを機能させるには、少なくとも 1 つのネームサーバー (**NS**) レコードが必要です。ただし、RFC 1912 に準拠するには、少なくとも 2 つのネームサーバーが必要です。
- **example.com** ドメインのメールエクスチェンジャー (**MX**) として **mail.example.com** を設定します。ホスト名の前の数値は、レコードの優先度です。値が小さいエントリーほど優先度が高くなります。
- **www.example.com**、**mail.example.com**、および **ns1.example.com** の IPv4 アドレスおよび IPv6 アドレスを設定します。

4. **named** グループだけがそれを読み取ることができるように、ゾーンファイルに安全なアクセス許可を設定します。

```
# chown root:named /var/named/example.com.zone
# chmod 640 /var/named/example.com.zone
```

5. **/var/named/example.com.zone** ファイルの構文を確認します。

```
# named-checkzone example.com /var/named/example.com.zone
zone example.com/IN: loaded serial 2022070601
OK
```

6. BIND をリロードします。

```
# systemctl reload named
```

change-root 環境で BIND を実行する場合は、**systemctl reload named-chroot** コマンドを使用してサービスをリロードします。

検証

- **example.com** ゾーンからさまざまなレコードをクエリーし、出力がゾーンファイルで設定したレコードと一致することを確認します。

```
# dig +short @localhost AAAA www.example.com
2001:db8:1::30

# dig +short @localhost NS example.com
ns1.example.com.

# dig +short @localhost A ns1.example.com
192.0.2.1
```

この例では、BIND が同じホストで実行し、**localhost** インターフェイスでクエリーに応答することを前提としています。

関連情報

- ゾーンファイルの SOA レコード
- BIND ACL の作成
- BIND 管理者リファレンスマニュアル
- RFC 1912 - 一般的な DNS 操作および設定エラー

4.6.3. BIND プライマリーサーバーでの逆引きゾーンの設定

逆ゾーンは、IP アドレスを名前にマップします。たとえば、IP 範囲 **192.0.2.0/24** を担当している場合は、BIND で逆引きゾーンを設定して、この範囲の IP アドレスをホスト名に解決できます。

注記

クラスフルネットワーク全体の逆引きゾーンを作成する場合は、それに応じてゾーンに名前を付けます。たとえば、クラス C ネットワーク **192.0.2.0/24** の場合は、ゾーンの名前が **2.0.192.in-addr.arpa** です。**192.0.2.0/28** など、異なるネットワークサイズの逆引きゾーンを作成する場合は、ゾーンの名前が **28-2.0.192.in-addr.arpa** です。

前提条件

- BIND は、たとえばキャッシングネームサーバーとしてすでに設定されています。
- **named** または **named-chroot** サービスが実行しています。

手順

1. **/etc/named.conf** ファイルにゾーン定義を追加します。

```
zone "2.0.192.in-addr.arpa" {
    type master;
```

```
    file "2.0.192.in-addr.arpa.zone";
    allow-query { any; };
    allow-transfer { none; };
};
```

これらの設定により、次が定義されます。

- **2.0.192.in-addr.arpa** 逆引きゾーンのプライマリーサーバー (**type master**) としてのこのサーバー。
- **/var/named/2.0.192.in-addr.arpa.zone** ファイルはゾーンファイルです。この例のように相対パスを設定すると、このパスは、**options** ステートメントの **directory** に設定したディレクトリーからの相対パスになります。
- どのホストもこのゾーンにクエリーを実行できます。または、IP 範囲または BIND アクセス制御リスト (ACL) のニックネームを指定して、アクセスを制限します。
- どのホストもゾーンを転送できません。ゾーン転送を許可するのは、セカンダリーサーバーをセットアップする際に限られ、セカンダリーサーバーの IP アドレスに対してのみ許可します。

2. **/etc/named.conf** ファイルの構文を確認します。

```
# named-checkconf
```

コマンドが出力を表示しない場合は、構文に間違いがありません。

3. たとえば、次の内容で **/var/named/2.0.192.in-addr.arpa.zone** ファイルを作成します。

```
$TTL 8h
@ IN SOA ns1.example.com. hostmaster.example.com. (
                    2022070601 ; serial number
                    1d         ; refresh period
                    3h         ; retry period
                    3d         ; expire time
                    3h )       ; minimum TTL

               IN NS   ns1.example.com.

1              IN PTR  ns1.example.com.
30             IN PTR  www.example.com.
```

このゾーンファイル:

- リソースレコードのデフォルトの有効期限 (TTL) 値を 8 時間に設定します。時間の **h** などの時間接尾辞がない場合、BIND は値を秒として解釈します。
- ゾーンに関する詳細を含む、必要な SOA リソースレコードが含まれています。
- **ns1.example.com** をこの逆引きゾーンの権威 DNS サーバーとして設定します。ゾーンを機能させるには、少なくとも 1 つのネームサーバー (**NS**) レコードが必要です。ただし、RFC 1912 に準拠するには、少なくとも 2 つのネームサーバーが必要です。
- **192.0.2.1** および **192.0.2.30** アドレスのポインター (**PTR**) レコードを設定します。

4. **named** グループのみがそれを読み取ることができるように、ゾーンファイルに安全なアクセス許可を設定します。

```
# chown root:named /var/named/2.0.192.in-addr.arpa.zone
# chmod 640 /var/named/2.0.192.in-addr.arpa.zone
```

5. **/var/named/2.0.192.in-addr.arpa.zone** ファイルの構文を確認します。

```
# named-checkzone 2.0.192.in-addr.arpa /var/named/2.0.192.in-addr.arpa.zone
zone 2.0.192.in-addr.arpa/IN: loaded serial 2022070601
OK
```

6. BIND をリロードします。

```
# systemctl reload named
```

change-root 環境で BIND を実行する場合は、**systemctl reload named-chroot** コマンドを使用してサービスをリロードします。

検証

- 逆引きゾーンからさまざまなレコードをクエリーし、出力がゾーンファイルで設定したレコードと一致することを確認します。

```
# dig +short @localhost -x 192.0.2.1
ns1.example.com.

# dig +short @localhost -x 192.0.2.30
www.example.com.
```

この例では、BIND が同じホストで実行し、**localhost** インターフェイスでクエリーに応答することを前提としています。

関連情報

- ゾーンファイルの SOA レコード
- BIND ACL の作成
- BIND 管理者リファレンスマニュアル
- RFC 1912 - 一般的な DNS 操作および設定エラー

4.6.4. BIND ゾーンファイルの更新

サーバーの IP アドレスが変更された場合など、特定の状況では、ゾーンファイルを更新する必要があります。複数の DNS サーバーが 1 つのゾーンを担ってる場合は、この手順をプライマリーサーバーでのみ実行してください。ゾーンのコピーを保存する他の DNS サーバーは、ゾーン転送を通じて更新を受け取ります。

前提条件

- ゾーンが設定されました。

- **named** または **named-chroot** サービスが実行しています。

手順

1. オプション: **/etc/named.conf** ファイル内のゾーンファイルへのパスを特定します。

```
options {
    ...
    directory       "/var/named";
}

zone "example.com" {
    ...
    file "example.com.zone";
};
```

ゾーンの定義の **file** ステートメントで、ゾーンファイルへのパスを見つけます。相対パスは、**options** ステートメントの **directory** に設定されたディレクトリーからの相対パスです。

2. ゾーンファイルを編集します。

   a. 必要な変更を行います。

   b. SOA (Start of Authority) レコードのシリアル番号を増やします。

重要

シリアル番号が以前の値以下の場合、セカンダリーサーバーはゾーンのコピーを更新しません。

3. ゾーンファイルの構文を確認します。

```
# named-checkzone example.com /var/named/example.com.zone
zone example.com/IN: loaded serial 2022062802
OK
```

4. BIND をリロードします。

```
# systemctl reload named
```

change-root 環境で BIND を実行する場合は、**systemctl reload named-chroot** コマンドを使用してサービスをリロードします。

検証

- 追加、変更、または削除したレコードを照会します。たとえば、次のようになります。

```
# dig +short @localhost A ns2.example.com
192.0.2.2
```

この例では、BIND が同じホストで実行し、**localhost** インターフェイスでクエリーに応答することを前提としています。

関連情報

- ゾーンファイルの SOA レコード
- BIND プライマリーサーバーでの正引きゾーンの設定
- BIND プライマリーサーバーでの逆引きゾーンの設定

4.6.5. 自動鍵生成およびゾーン保守機能を使用した DNSSEC ゾーン署名

DNSSEC (Domain Name System Security Extensions) でゾーンに署名して、認証およびデータの整合性を確保できます。このようなゾーンには、追加のリソースレコードが含まれます。クライアントはそれらを使用して、ゾーン情報の信頼性を検証できます。

ゾーンの DNSSEC ポリシー機能を有効にすると、BIND は次のアクションを自動的に実行します。

- キーを作成します。
- ゾーンに署名します
- 鍵の再署名や定期的な交換など、ゾーンを維持します。

重要

外部 DNS サーバーがゾーンの信頼性を検証できるようにするには、ゾーンの公開キーを親ゾーンに追加する必要があります。これを行う方法の詳細については、ドメインプロバイダーまたはレジストリーにお問い合わせください。

この手順では、BIND に組み込まれている **default** の DNSSEC ポリシーを使用します。このポリシーは、単一の **ECDSAP256SHA** 鍵署名を使用します。または、独自のポリシーを作成して、カスタムキー、アルゴリズム、およびタイミングを使用します。

前提条件

- BIND 9.16 以降がインストールされている。この要件を満たすには、**bind** の代わりに **bind9.16** パッケージをインストールします。
- DNSSEC を有効にするゾーンが設定されている。
- **named** または **named-chroot** サービスが実行しています。
- サーバーは時刻をタイムサーバーと同期します。DNSSEC 検証では、正確なシステム時刻が重要です。

手順

1. **/etc/named.conf** ファイルを編集し、DNSSEC を有効にするゾーンに **dnssec-policy default;** を追加します。

```
zone "example.com" {
    ...
    dnssec-policy default;
};
```

2. BIND をリロードします。

```
# systemctl reload named
```

change-root 環境で BIND を実行する場合は、**systemctl reload named-chroot** コマンドを使用してサービスをリロードします。

3. BIND は公開鍵を **/var/named/K<zone_name>.+<algorithm>+<key_ID>.key** ファイルに保存します。このファイルを使用して、親ゾーンが必要とする形式でゾーンの公開鍵を表示します。

   - DS レコード形式:

     ```
     # dnssec-dsfromkey /var/named/Kexample.com.+013+61141.key
     example.com. IN DS 61141 13 2
     3E184188CF6D2521EDFDC3F07CFEE8D0195AACBD85E68BAE0620F638B4B1B027
     ```

   - DNSKEY 形式:

     ```
     # grep DNSKEY /var/named/Kexample.com.+013+61141.key
     example.com. 3600 IN DNSKEY 257 3 13
     sjzT3jNEp120aSO4mPEHHSkReHUf7AABNnT8hNRTzD5cKMQSjDJin2l3
     5CaKVcWO1pm+HltxUEt+X9dfp8OZkg==
     ```

4. ゾーンの公開鍵を親ゾーンに追加するように要求します。これを行う方法の詳細については、ドメインプロバイダーまたはレジストリーにお問い合わせください。

検証

1. DNSSEC 署名を有効にしたゾーンのレコードについて、独自の DNS サーバーにクエリーを実行します。

   ```
   # dig +dnssec +short @localhost A www.example.com
   192.0.2.30
   A 13 3 28800 20220718081258 20220705120353 61141 example.com.
   e7Cfh6GuOBMAWsgsHSVTPh+JJSOI/Y6zctzluqIU1JqEgOOAfL/Qz474
   M0sgi54m1Kmnr2ANBKJN9uvOs5eXYw==
   ```

   この例では、BIND が同じホストで実行し、**localhost** インターフェイスでクエリーに応答することを前提としています。

2. 公開鍵が親ゾーンに追加され、他のサーバーに伝播されたら、サーバーが署名付きゾーンへのクエリーで認証済みデータ (**ad**) フラグを設定することを確認します。

   ```
   #  dig @localhost example.com +dnssec
   ...
   ;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1
   ...
   ```

関連情報

- BIND プライマリーサーバーでの正引きゾーンの設定
- BIND プライマリーサーバーでの逆引きゾーンの設定

4.7. BIND DNS サーバー間のゾーン転送の設定

ゾーン転送により、ゾーンのコピーを持つすべての DNS サーバーが最新のデータを使用することが保証されます。

前提条件

- 将来のプライマリーサーバーでは、ゾーン転送を設定するゾーンが設定されている。
- 将来のセカンダリーサーバーでは、BIND はキャッシュネームサーバーなどとして設定されている。
- 両方のサーバーで、**named** サービスまたは **named-chroot** サービスが実行している。

手順

1. 既存のプライマリーサーバーで、以下を行います。
   a. 共有キーを作成し、**/etc/named.conf** ファイルに追加します。

```
# tsig-keygen example-transfer-key | tee -a /etc/named.conf
key "example-transfer-key" {
        algorithm hmac-sha256;
        secret "q7ANbnyliDMuvWgnKOxMLi313JGcTZB5ydMW5CyUGXQ=";
};
```

   このコマンドは、**tsig-keygen** コマンドの出力を表示し、自動的に **/etc/named.conf** に追加します。

   後でセカンダリーサーバーでもコマンドの出力が必要になります。

   b. **/etc/named.conf** ファイルのゾーン定義を編集します。
      i. **allow-transfer** ステートメントで、サーバーがゾーンを転送するために **example-transfer-key** ステートメントで指定されたキーを提供する必要があることを定義します。

```
zone "example.com" {
    ...
    allow-transfer { key example-transfer-key; };
};
```

      または、**allow-transfer** ステートメントで BIND アクセス制御リスト (ACL) ニックネームを使用します。

      ii. デフォルトでは、ゾーンが更新された後、BIND は、このゾーンにネームサーバー (**NS**) レコードを持つすべてのネームサーバーに通知します。セカンダリーサーバーの **NS** レコードをゾーンに追加する予定がない場合は、BIND がこのサーバーに通知するように設定できます。そのために、このセカンダリーサーバーの IP アドレスを含む **also-notify** ステートメントをゾーンに追加します。

```
zone "example.com" {
    ...
    also-notify { 192.0.2.2; 2001:db8:1::2; };
};
```

   c. **/etc/named.conf** ファイルの構文を確認します。

```
# named-checkconf
```

コマンドが出力を表示しない場合は、構文に間違いがありません。

d. BIND をリロードします。

```
# systemctl reload named
```

change-root 環境で BIND を実行する場合は、**systemctl reload named-chroot** コマンドを使用してサービスをリロードします。

2. 将来のセカンダリーサーバーで、以下を行います。

a. **/etc/named.conf** ファイルを次のように編集します。

i. プライマリーサーバーと同じキー定義を追加します。

```
key "example-transfer-key" {
        algorithm hmac-sha256;
        secret "q7ANbnyliDMuvWgnKOxMLi313JGcTZB5ydMW5CyUGXQ=";
};
```

ii. **/etc/named.conf** ファイルにゾーン定義を追加します。

```
zone "example.com" {
    type slave;
    file "slaves/example.com.zone";
    allow-query { any; };
    allow-transfer { none; };
    masters {
      192.0.2.1 key example-transfer-key;
      2001:db8:1::1 key example-transfer-key;
    };
};
```

これらの設定状態:

- このサーバーは、**example.com** ゾーンのセカンダリーサーバー (**type slave**) です。
- **/var/named/slaves/example.com.zone** ファイルはゾーンファイルです。この例のように相対パスを設定すると、このパスは、**options** ステートメントの **directory** に設定したディレクトリーからの相対パスになります。このサーバーがセカンダリーであるゾーンファイルをプライマリーサーバーから分離するには、それを **/var/named/slaves/** ディレクトリーなどに保存します。
- どのホストもこのゾーンにクエリーを実行できます。または、IP 範囲または ACL ニックネームを指定して、アクセスを制限します。
- このサーバーからゾーンを転送できるホストはありません。
- このゾーンのプライマリーサーバーの IP アドレスは **192.0.2.1** および **2001:db8:1::2** です。または、ACL ニックネームを指定できます。このセカンダリーサーバーは、**example-transfer-key** という名前のキーを使用して、プライマリーサーバーに対する認証を行います。

b. **/etc/named.conf** ファイルの構文を確認します。

```
# named-checkconf
```

c. BIND をリロードします。

```
# systemctl reload named
```

change-root 環境で BIND を実行する場合は、**systemctl reload named-chroot** コマンドを使用してサービスをリロードします。

3. オプション: プライマリーサーバーのゾーンファイルを変更し、新しいセカンダリーサーバーの **NS** レコードを追加します。

検証

セカンダリーサーバーで次の手順を実行します。

1. **named** サービスの **systemd** ジャーナルエントリーを表示します。

```
# journalctl -u named
...
Jul 06 15:08:51 ns2.example.com named[2024]: zone example.com/IN: Transfer started.
Jul 06 15:08:51 ns2.example.com named[2024]: transfer of 'example.com/IN' from
192.0.2.1#53: connected using 192.0.2.2#45803
Jul 06 15:08:51 ns2.example.com named[2024]: zone example.com/IN: transferred serial
2022070101
Jul 06 15:08:51 ns2.example.com named[2024]: transfer of 'example.com/IN' from
192.0.2.1#53: Transfer status: success
Jul 06 15:08:51 ns2.example.com named[2024]: transfer of 'example.com/IN' from
192.0.2.1#53: Transfer completed: 1 messages, 29 records, 2002 bytes, 0.003 secs (667333
bytes/sec)
```

change-root 環境で BIND を実行する場合は、**journalctl -u named-chroot** コマンドを使用してジャーナルエントリーを表示します。

2. BIND がゾーンファイルを作成したことを確認します。

```
# ls -l /var/named/slaves/
total 4
-rw-r--r--. 1 named named 2736 Jul  6 15:08 example.com.zone
```

デフォルトでは、セカンダリーサーバーはゾーンファイルを未修正のバイナリー形式で保存することに注意してください。

3. セカンダリーサーバーから転送されたゾーンのレコードをクエリーします。

```
# dig +short @192.0.2.2 AAAA www.example.com
2001:db8:1::30
```

この例では、この手順で設定したセカンダリーサーバーが IP アドレス **192.0.2.2** でリッスンしていると想定しています。

関連情報

- BIND プライマリーサーバーでの正引きゾーンの設定
- BIND プライマリーサーバーでの逆引きゾーンの設定
- BIND ACL の作成
- BIND ゾーンファイルの更新

4.8. DNS レコードを上書きするように BIND で応答ポリシーゾーンを設定する

管理者は、DNS のブロックとフィルタリングを使用して、DNS 応答を書き換えて、特定のドメインまたはホストへのアクセスをブロックできます。BIND では、応答ポリシーゾーン (RPZ) がこの機能を提供します。**NXDOMAIN** エラーを返す、クエリーに応答しないなど、ブロックされたエントリーに対してさまざまなアクションを設定できます。

環境内に複数の DNS サーバーがある場合は、この手順を使用してプライマリーサーバーで RPZ を設定し、後でゾーン転送を設定して、セカンダリーサーバーで RPZ を使用できるようにします。

前提条件

- BIND は、たとえばキャッシングネームサーバーとしてすでに設定されています。
- **named** または **named-chroot** サービスが実行しています。

手順

1. **/etc/named.conf** ファイルを編集し、次の変更を行います。
   a. **options** ステートメントに **response-policy** 定義を追加します。

```
options {
    ...

    response-policy {
        zone "rpz.local";
    };

    ...
}
```

   response-policy の **zone** ステートメントで RPZ のカスタム名を設定できます。ただし、次のステップのゾーン定義で同じ名前を使用する必要があります。

   b. 前の手順で設定した RPZ の **zone** 定義を追加します。

```
zone "rpz.local" {
    type master;
    file "rpz.local";
    allow-query { localhost; 192.0.2.0/24; 2001:db8:1::/64; };
    allow-transfer { none; };
};
```

   これらの設定状態:

- このサーバーは、**rpz.local** という名前の RPZ のプライマリーサーバー (**type master**) です。
- **/var/named/rpz.local** ファイルはゾーンファイルです。この例のように相対パスを設定すると、このパスは、**options** ステートメントの **directory** に設定したディレクトリーからの相対パスになります。
- **allow-query** で定義されたホストは、この RPZ をクエリーできます。または、IP 範囲または BIND アクセス制御リスト (ACL) のニックネームを指定して、アクセスを制限します。
- どのホストもゾーンを転送できません。ゾーン転送を許可するのは、セカンダリーサーバーをセットアップする際に限られ、セカンダリーサーバーの IP アドレスに対してのみ許可します。

2. **/etc/named.conf** ファイルの構文を確認します。

```
# named-checkconf
```

コマンドが出力を表示しない場合は、構文に間違いがありません。

3. たとえば、次の内容で **/var/named/rpz.local** ファイルを作成します。

```
$TTL 10m
@ IN SOA ns1.example.com. hostmaster.example.com. (
                    2022070601 ; serial number
                    1h         ; refresh period
                    1m         ; retry period
                    3d         ; expire time
                    1m )       ; minimum TTL

               IN NS    ns1.example.com.

example.org      IN CNAME .
*.example.org    IN CNAME .
example.net      IN CNAME rpz-drop.
*.example.net    IN CNAME rpz-drop.
```

このゾーンファイル:

- リソースレコードのデフォルトの有効期限 (TTL) 値を 10 分に設定します。時間の **h** などの時間接尾辞がない場合、BIND は値を秒として解釈します。
- ゾーンに関する詳細を含む、必要な SOA (Start of Authority) リソースレコードが含まれます。
- このゾーンの権威 DNS サーバーとして **ns1.example.com** を設定します。ゾーンを機能させるには、少なくとも 1 つのネームサーバー (**NS**) レコードが必要です。ただし、RFC 1912 に準拠するには、少なくとも 2 つのネームサーバーが必要です。
- このドメイン内の **example.org** およびホストへのクエリーに対して **NXDOMAIN** エラーを返します。
- このドメイン内の **example.net** およびホストにクエリーを破棄します。

アクションおよび例の完全なリストは、IETF draft: DNS Response Policy Zones (RPZ) を参照してください。

4. **/var/named/rpz.local** ファイルの構文を確認します。

```
# named-checkzone rpz.local /var/named/rpz.local
zone rpz.local/IN: loaded serial 2022070601
OK
```

5. BIND をリロードします。

```
# systemctl reload named
```

change-root 環境で BIND を実行する場合は、**systemctl reload named-chroot** コマンドを使用してサービスをリロードします。

検証

1. **NXDOMAIN** エラーを返すように RPZ で設定されている **example.org** のホストを解決しようとします。

```
# dig @localhost www.example.org
...
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 30286
...
```

この例では、BIND が同じホストで実行し、**localhost** インターフェイスでクエリーに応答することを前提としています。

2. RPZ でクエリーを破棄するように設定されている **example.net** ドメイン内のホストの解決を試みます。

```
# dig @localhost www.example.net
...
;; connection timed out; no servers could be reached
...
```

関連情報

- IETF draft: DNS Response Policy Zones (RPZ)

4.9. DNSTAP を使用して DNS クエリーを記録する

ネットワーク管理者は、ドメインネームシステム (DNS) の詳細を記録して、DNS トラフィックパターンの分析、DNS サーバーのパフォーマンスの監視、DNS 問題のトラブルシューティングを行うことができます。受信する名前クエリーの詳細を監視してログに記録する高度な方法が必要な場合は、**named** サービスから送信されたメッセージを記録する **dnstap** インターフェイスを使用します。DNS クエリーをキャプチャーおよび記録して、Web サイトまたは IP アドレスに関する情報を収集できます。

前提条件

- **bind-9.11.26-2** パッケージ以降のバージョンがインストールされている。

警告

BIND バージョンがすでにインストールされ、実行されている場合、新しいバージョンの **BIND** を追加すると、既存のバージョンが上書きされます。

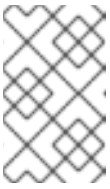
手順

1. **/etc/named.conf** ファイルの **options** ブロックを編集して、**dnstap** とターゲットファイルを有効にします。

```
options
{
# ...
dnstap { all; }; # Configure filter
dnstap-output file "/var/named/data/dnstap.bin";
# ...
};
# end of options
```

2. ログに記録する DNS トラフィックのタイプを指定するには、**/etc/named.conf** ファイルの **dnstap** ブロックに **dnstap** フィルターを追加します。次のフィルターを使用できます。

   - **auth**: 権威ゾーンの応答または回答。
   - **client**: 内部クライアントクエリーまたは回答。
   - **forwarder**: 転送クエリーまたは応答。
   - **resolver**: 反復的解決クエリーまたは応答。
   - **update**: 動的ゾーン更新要求。
   - **all**: 上記のオプションのいずれか。
   - **query** または **response**: **query** または **response** キーワードを指定しない場合、**dnstap** は両方を記録します。



注記

dnstap フィルターでは、**dnstap {}** ブロック内に **;** で区切った複数の定義を含めます。次の構文を使用してください。**dnstap { (all | auth | client | forwarder | resolver | update) [(query | response)]; ... };**

3. 変更を適用するために、**named** サービスを再起動します。

```
# systemctl restart named.service
```

4. アクティブなログの定期的なロールアウトを設定します。
   次の例では、**cron** スケジューラーは、ユーザーが編集したスクリプトの内容を 1 日に 1 回実行します。**roll** オプションに値 **3** 指定し、**dnstap** が最大 3 つのバックアップログファイルを作成

できるようにしています。この値 **3** は、**dnstap-output** 変数の **version** パラメーターをオーバーライドし、バックアップログファイルの数を 3 に制限します。また、バイナリーログファイルは別のディレクトリーに移動されて名前が変更されます。3 つのバックアップログファイルがすでに存在する場合でも、ファイルの接尾辞が **.2** に達することはありません。サイズ制限に基づくバイナリーログの自動ローリングで十分な場合は、このステップを省略できます。

Example:

```
sudoedit /etc/cron.daily/dnstap

#!/bin/sh
rndc dnstap -roll 3
mv /var/named/data/dnstap.bin.1 /var/log/named/dnstap/dnstap-$(date -I).bin

# use dnstap-read to analyze saved logs
sudo chmod a+x /etc/cron.daily/dnstap
```

5. **dnstap-read** ユーティリティーを使用して、人間が判読できる形式でログを処理および分析します。
次の例では、**dnstap-read** ユーティリティーは出力を **YAML** ファイル形式で出力します。

Example:

```
dnstap-read -y [file-name]
```

第5章 NFS サーバーのデプロイ

ネットワークファイルシステム (NFS) プロトコルを使用すると、リモートユーザーはネットワーク経由で共有ディレクトリーをマウントし、ローカルにマウントされたディレクトリーと同じように使用できます。また、リソースを、ネットワークの集中化サーバーに統合できるようになります。

5.1. NFSV4 のマイナーバージョンの主な機能

NFSv4 の各マイナーバージョンでは、パフォーマンスとセキュリティーの向上を目的とした機能強化が導入されます。この強化を利用して NFSv4 の可能性を最大限に活用すれば、ネットワーク全体で効率的かつ信頼性の高いファイル共有を実現できます。

NFSv4.2 の主な機能

サーバー側コピー

サーバー側コピーは、ネットワーク経由でデータを転送せずにサーバー上のファイルをコピーする NFS サーバーの機能です。

スパースファイル

ファイルに 1 つ以上の空きスペース、つまりギャップを持たせることができます。ギャップとは、ゼロのみで構成される未割り当てまたは未初期化データブロックです。これにより、アプリケーションがスパースファイル内のホールの位置を計画できるようになります。

領域の予約

クライアントが、データを書き込む前にストレージサーバー上の領域を予約または確保できます。これにより、サーバーの領域不足が防止されます。

ラベル付き NFS

データアクセス権を強制し、NFS ファイルシステム上の個々のファイルに対して、クライアントとサーバーとの間の SELinux ラベルを有効にします。

レイアウトの機能強化

Parallel NFS (pNFS) サーバーがより優れたパフォーマンス統計情報を収集できるようにする機能を提供します。

NFSv4.1 の主な機能

pNFS のクライアント側サポート

クラスター化されたサーバーへの高速 I/O のサポートにより、複数のマシンへのデータ保存、データへの直接アクセス、メタデータの更新の同期が可能になります。

セッション

セッションは、クライアントに属する接続に関連するサーバーの状態を維持します。この種類のセッションは、各リモートプロシージャーコール (RPC) 操作の接続の確立と終了に関連するオーバーヘッドを削減し、パフォーマンスと効率を向上させます。

NFSv4.0 の主な機能

RPC とセキュリティー

RPCSEC_GSS フレームワークにより、RPC のセキュリティーが強化されます。NFSv4 プロトコルで、インバンドセキュリティーネゴシエーション用の新しい操作が導入されました。これにより、クライアントがファイルシステムリソースにセキュアにアクセスするためのサーバーポリシーをクエリーできるようになります。

プロシージャーと操作の構造

NFS 4.0 で、**COMPOUND** プロシージャーが導入されました。これにより、クライアントが複数の操作を 1 つの要求にマージして RPC を削減できるようになりました。

ファイルシステムモデル

NFS 4.0 は、階層型ファイルシステムモデルを保持し、ファイルをバイトストリームとして扱い、国際化のために名前を UTF-8 でエンコードします。

- **ファイルハンドルの種類**
  揮発性のファイルハンドルにより、サーバーがファイルシステムの変更に適応できます。また、クライアントが、永続的なファイルハンドルを必要とせずに、必要に応じて適応できます。

- **属性タイプ**
  ファイル属性構造には、必須属性、推奨属性、および名前付き属性が含まれています。各属性は異なる目的を果たします。NFSv3 から派生した必須属性は、ファイルタイプを区別するために必要です。一方、ACL などの推奨属性は、アクセス制御を強化します。

- **マルチサーバー名前空間**
  名前空間は、複数のサーバー全体を対象に、属性に基づいてファイルシステム転送を簡素化します。また、参照、冗長性、シームレスなサーバー移行をサポートします。

OPEN および CLOSE 操作

これらの操作により、ファイルの検索、作成、セマンティック共有を 1 カ所で組み合わせて、ファイルアクセス管理を効率化できます。

ファイルロック

ファイルロックがプロトコルに含まれているため、RPC コールバックが不要になります。ファイルロックの状態は、リースベースのモデルに基づいてサーバーによって管理されます。リースの更新に失敗すると、サーバーによって状態が解放されることがあります。

クライアントのキャッシュと委譲

キャッシュは以前のバージョンと似ています。属性とディレクトリーのキャッシュのタイムアウトが、クライアントによって決定されます。NFS 4.0 の委譲により、サーバーがクライアントに特定の役割を割り当てることができます。これにより、特定のファイル共有セマンティクスが確保され、サーバーとの直接のやり取りなしでローカルファイル操作が可能になります。

5.2. AUTH_SYS 認証方式

AUTH_SYS 方式 (**AUTH_UNIX** とも呼ばれます) は、クライアント認証メカニズムです。**AUTH_SYS** を使用すると、クライアントがファイルにアクセスするときに、ユーザーのアイデンティティーと権限を確認するために、ユーザーのユーザー ID (UID) とグループ ID (GID) をサーバーに送信します。AUTH_SYS は、クライアントが提供する情報に依存するため、誤って設定された場合に不正アクセスを受ける可能性があり、セキュリティーが低いと考えられています。

マッピングメカニズムにより、UID と GID の割り当てがシステム間で異なる場合でも、NFS クライアントが適切な権限でサーバー上のファイルにアクセスできます。UID と GID は、次のメカニズムによって NFS クライアントとサーバーの間でマッピングされます。

直接マッピング

UID と GID は、NFS サーバーとクライアントによってローカルシステムとリモートシステム間で直接マッピングされます。これを行うには、NFS ファイル共有に参加しているすべてのシステム間で一貫した UID と GID の割り当てが必要です。たとえば、クライアント上の UID 1000 のユーザーは、サーバー上の UID 1000 のユーザーがアクセスできる共有上のファイルにのみアクセスできます。

管理者は、NFS 環境での ID 管理を簡素化するために、多くの場合、LDAP やネットワーク情報サービス (NIS) などの集中型サービスを利用して、複数のシステムにわたる UID と GID のマッピングを管理します。

ユーザー ID とグループ ID のマッピング

NFS サーバーおよびクライアントは、**idmapd** サービスを使用して、異なるシステム間で UID と GID を変換し、一貫した ID 識別と権限の割り当てを実現できます。

5.3. AUTH_GSS 認証方式

Kerberos は、セキュアでないネットワーク上でクライアントとサーバーのセキュアな認証を可能にするネットワーク認証プロトコルです。対称鍵暗号を使用し、ユーザーとサービスを認証するために、信頼できる Key Distribution Center (KDC) を必要とします。

AUTH_SYS とは異なり、**RPCSEC_GSS** Kerberos メカニズムでは、ファイルにアクセスしているユーザーを正しく表すために、サーバーがクライアントに依存することがありません。代わりに、暗号化を使用してサーバーに対してユーザーを認証します。これにより、悪意のあるクライアントがユーザーの Kerberos 認証情報を持たないユーザーになりすますことを防ぎます。

/etc/exports ファイルの **sec** オプションで、共有が提供する Kerberos セキュリティー方式を 1 つ以上定義します。クライアントはこれらの方法のいずれかを使用して共有をマウントできます。**sec** オプションは次の値をサポートします。

- **sys**: 暗号化保護なし (デフォルト)
- **krb5**: 認証のみ
- **krb5i**: 認証と整合性保護
- **krb5p**: 認証、整合性チェック、およびトラフィック暗号化

方式が提供する暗号化機能が多いほど、パフォーマンスが低下することに注意してください。

5.4. エクスポートされたファイルシステムのファイル権限

エクスポートされたファイルシステムのファイル権限によって、NFS 経由でファイルとディレクトリーにアクセスするクライアントのアクセス権が決まります。

NFS ファイルシステムがリモートホストによってマウントされると、各共有ファイルに対する保護がファイルシステムの権限だけになります。同じユーザー ID (UID) の値を共有する 2 つのユーザーが、異なるクライアントシステムに同じ NFS ファイルシステムをマウントした場合、そのユーザーはお互いのファイルを変更できます。

NFS は、クライアント上の **root** ユーザーをサーバー上の **root** ユーザーと同等のものとして扱います。ただし、NFS サーバーは、NFS 共有にアクセスするときに、デフォルトで **root** を **nobody** アカウントにマップします。この動作は **root_squash** オプションにより制御します。

関連情報

- **exports(5)** man ページ

5.5. NFS サーバーに必要なサービス

Red Hat Enterprise Linux (RHEL) は、NFS ファイル共有を提供するのに、カーネルモジュールとユーザー空間プロセスの組み合わせを使用します。

表5.1 NFS サーバーに必要なサービス

サービス名	NFS バージョン	説明
nfsd	3、4	共有 NFS ファイルシステムに対する要求を処理する NFS カーネルモジュール。
rpcbind	3	このプロセスは、ローカルのリモートプロシージャーコール (RPC) サービスからのポート予約を受け入れ、それを使用可能にするかアドバタイズして、対応するリモート RPC サービスがポート予約にアクセスできるようにします。**rpcbind** サービスは、要求に応答し、指定された RPC サービスへの接続を設定します。
rpc.mountd	3、4	このサービスは NFSv3 クライアントからの **MOUNT** 要求を処理します。NFSv4 サーバーはこのサービスの内部機能を使用します。 要求されている NFS 共有が現在 NFS サーバーによりエクスポートされているか、またその共有へのクライアントのアクセスが許可されているかを確認します。
rpc.nfsd	3、4	このプロセスは、サーバーが定義する明示的な NFS バージョンとプロトコルをアドバタイズします。NFS クライアントが接続するたびにサーバースレッドを提供するなど、NFS クライアントの動的な要求に対応するために、カーネルと連携して動作します。 **nfs-server** サービスがこのプロセスを起動します。
lockd	3	このカーネルモジュールは、Network Lock Manager (NLM) プロトコルを実装します。これにより、クライアントがサーバー上のファイルをロックできるようになります。RHEL は、NFS サーバーの実行時にこのモジュールを自動的にロードします。
rpc.rquotad	3、4	このサービスは、リモートユーザーのユーザークォータ情報を提供します。
rpc.idmapd	4	このプロセスは、NFSv4 名 (`user@domain` 形式の文字列) とローカルのユーザー ID およびグループ ID をマッピングする NFSv4 クライアントおよびサーバーのアップコールを提供します。
gssproxy	3、4	このサービスは、**rpc.nfsd** に代わって **krb5** 認証を処理します。
nfsdcld	4	このサービスは、NFSv4 クライアント追跡デーモンを提供します。このデーモンは、ネットワークパーティションとサーバーの再起動中に他のクライアントが競合するロックを取得したときに、サーバーがロックの回収を許可するのを防止します。

サービス名	NFS バージョン	説明
rpc.statd	3	このサービスは、ローカルホストが再起動したときに他の NFSv3 クライアントに通知し、リモート NFSv3 ホストが再起動したときにカーネルに通知します。

関連情報

- **rpcbind(8)**、**rpc.mountd(8)**、**rpc.nfsd(8)**、**rpc.statd(8)**、**rpc.rquotad(8)**、**rpc.idmapd(8)**、**nfsdcld(8)** man ページ

5.6. /ETC/EXPORTS 設定ファイル

/etc/exports ファイルは、サーバーがエクスポートするディレクトリーを制御します。各行に、エクスポートポイント、ディレクトリーのマウントが許可されているクライアントの空白区切りのリスト、および各クライアントのオプションが含まれています。

```
<directory> <host_or_network_1>(<options_1>) <host_or_network_n>(<options_n>)...
```

以下は **/etc/exports** のエントリーの各部分です。

<export>
エクスポートするディレクトリー。

<host_or_network>
エクスポートを共有するホストまたはネットワーク。たとえば、ホスト名、IP アドレス、または IP ネットワークを指定できます。

<options>
ホストまたはネットワークのオプション。

クライアントとオプションの間にスペースを追加すると、動作が変わります。たとえば、次の行はそれぞれ意味が異なります。

```
/projects client.example.com(rw)
/projects client.example.com (rw)
```

最初の行では、サーバーは **client.example.com** にのみ、**/projects** ディレクトリーを読み取り/書き込みモードでマウントすることを許可します。他のホストは共有をマウントできません。一方、2 番目の行では、**client.example.com** と **(rw)** の間にスペースがあるため、サーバーはディレクトリーを読み取り専用モード (デフォルト設定) で **client.example.com** にエクスポートします。他のすべてのホストは、読み取り/書き込みモードで共有をマウントできます。

NFS サーバーは、エクスポートされた各ディレクトリーに対して次のデフォルト設定を使用します。

表5.2 /etc/exports のエントリーのデフォルトオプション

デフォルト設定	説明
ro	ディレクトリーを読み取り専用モードでエクスポートします。

デフォルト設定	説明
sync	NFS サーバーは、以前の要求で発生した変更がディスクに書き込まれるまで、要求に応答しません。
wdelay	別の書き込み要求が保留中であると疑われる場合、サーバーはディスクへの書き込みを遅延します。
root_squash	クライアントの **root** ユーザーがエクスポートされたディレクトリーに対して **root** 権限を持つことを防ぎます。**root_squash** を有効にすると、NFS サーバーは **root** からのアクセスをユーザー **nobody** にマッピングします。

5.7. NFSV4 専用サーバーの設定

ネットワーク内に NFSv3 クライアントが存在しない場合は、NFSv4 またはその特定のマイナープロトコルバージョンのみをサポートするように NFS サーバーを設定できます。サーバー上で NFSv4 のみを使用すると、ネットワークに開放されるポートの数が減ります。

手順

1. **nfs-utils** パッケージをインストールします。

```
# dnf install nfs-utils
```

2. **/etc/nfs.conf** ファイルを編集し、次の変更を加えます。

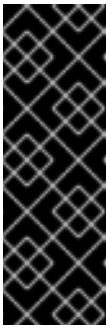
   a. NFSv3 を無効にするには、**[nfsd]** セクションの **vers3** パラメーターを無効にします。

```
[nfsd]
vers3=n
```

   b. オプション: 特定の NFSv4 マイナーバージョンのみが必要な場合は、すべての **vers4.<minor_version>** パラメーターのコメントを解除し、各パラメーターを適切に設定します。次に例を示します。

```
[nfsd]
vers3=n
# vers4=y
vers4.0=n
vers4.1=n
vers4.2=y
```

   この設定では、サーバーは NFS バージョン 4.2 のみを提供します。



重要

特定の NFSv4 マイナーバージョンのみが必要な場合は、そのマイナーバージョンのパラメーターのみを設定してください。予期しないマイナーバージョンのアクティブ化や非アクティブ化を回避するために、**vers4** パラメーターのコメントは解除しないでください。**vers4** パラメーターは、デフォルトですべての NFSv4 マイナーバージョンを有効または無効にします。ただし、**vers4** を他の **vers** パラメーターと組み合わせて設定すると、この動作は変わります。

3. NFSv3 関連のすべてのサービスを無効にします。

```
# systemctl mask --now rpc-statd.service rpcbind.service rpcbind.socket
```

4. オプション: 共有するディレクトリーを作成します。以下に例を示します。

```
# mkdir -p /nfs/projects/
```

既存のディレクトリーを共有する場合は、このステップをスキップしてください。

5. **/nfs/projects/** ディレクトリーに必要な権限を設定します。

```
# chmod 2770 /nfs/projects/
# chgrp users /nfs/projects/
```

これらのコマンドは、**/nfs/projects/** ディレクトリーの **users** グループの書き込み権限を設定し、このディレクトリーに作成される新しいエントリーに対して同じグループを自動的に設定します。

6. 共有する各ディレクトリーについて、**/etc/exports** ファイルにエクスポートポイントを追加します。

```
/nfs/projects/     192.0.2.0/24(rw) 2001:db8::/32(rw)
```

このエントリーは、**/nfs/projects/** ディレクトリーを共有し、**192.0.2.0/24** および **2001:db8::/32** サブネット内のクライアントに読み取りおよび書き込みアクセスを許可します。

7. **firewalld** で適切なポートを開きます。

```
# firewall-cmd --permanent --add-service nfs
# firewall-cmd --reload
```

8. NFS サーバーを有効にして起動します。

```
# systemctl enable --now nfs-server
```

検証

- サーバー上で、設定した NFS バージョンのみがサーバーから提供されていることを確認します。

```
# cat /proc/fs/nfsd/versions
-3 +4 -4.0 -4.1 +4.2
```

- クライアントで次の手順を実行します。

  1. **nfs-utils** パッケージをインストールします。

     ```
     # dnf install nfs-utils
     ```

  2. エクスポートされた NFS 共有をマウントします。

     ```
     # mount server.example.com:/nfs/projects/ /mnt/
     ```

  3. **users** グループのメンバーであるユーザーとして、**/mnt/** にファイルを作成します。

     ```
     # touch /mnt/file
     ```

  4. ファイルが作成されたことを確認するためにディレクトリーの内容をリスト表示します。

     ```
     # ls -l /mnt/
     total 0
     -rw-r--r--. 1 demo users 0 Jan 16 14:18 file
     ```

5.8. オプションの NFSV4 サポートを備えた NFSV3 サーバーの設定

NFSv3 クライアントを現在も使用しているネットワークでは、NFSv3 プロトコルを使用して共有を提供するようにサーバーを設定します。ネットワーク内に新しいクライアントもある場合は、さらに NFSv4 を有効にできます。デフォルトでは、Red Hat Enterprise Linux の NFS クライアントは、サーバーが提供する最新の NFS バージョンを使用します。

手順

1. **nfs-utils** パッケージをインストールします。

   ```
   # dnf install nfs-utils
   ```

2. オプション: デフォルトでは、NFSv3 と NFSv4 が有効になっています。NFSv4 が必要ない場合、または特定のマイナーバージョンのみが必要な場合は、すべての **vers4.<minor_version>** パラメーターのコメントを解除し、各パラメーターを適切に設定します。

   ```
   [nfsd]
   # vers3=y
   # vers4=y
   vers4.0=n
   vers4.1=n
   vers4.2=y
   ```

   この設定では、サーバーは NFS バージョン 3 と 4.2 のみを提供します。

重要

特定の NFSv4 マイナーバージョンのみが必要な場合は、そのマイナーバージョンのパラメーターのみを設定してください。予期しないマイナーバージョンのアクティブ化や非アクティブ化を回避するために、**vers4** パラメーターのコメントは解除しないでください。**vers4** パラメーターは、デフォルトですべての NFSv4 マイナーバージョンを有効または無効にします。ただし、**vers4** を他の **vers** パラメーターと組み合わせて設定すると、この動作は変わります。

3. デフォルトでは、NFSv3 の RPC サービスはランダムなポートを使用します。ファイアウォール設定を有効にするには、**/etc/nfs.conf** ファイルで固定ポート番号を設定します。

   a. **[lockd]** セクションで、**nlockmgr** RPC サービスの固定ポート番号を設定します。以下に例を示します。

   ```
   [lockd]
   port=5555
   ```

   この設定により、サービスが UDP プロトコルと TCP プロトコルの両方にこのポート番号を自動的に使用するようになります。

   b. **[statd]** セクションで、**rpc.statd** サービスの固定ポート番号を設定します。以下に例を示します。

   ```
   [statd]
   port=6666
   ```

   この設定により、サービスが UDP プロトコルと TCP プロトコルの両方にこのポート番号を自動的に使用するようになります。

4. オプション: 共有するディレクトリーを作成します。以下に例を示します。

   ```
   # mkdir -p /nfs/projects/
   ```

   既存のディレクトリーを共有する場合は、このステップをスキップしてください。

5. **/nfs/projects/** ディレクトリーに必要な権限を設定します。

   ```
   # chmod 2770 /nfs/projects/
   # chgrp users /nfs/projects/
   ```

   これらのコマンドは、**/nfs/projects/** ディレクトリーの **users** グループの書き込み権限を設定し、このディレクトリーに作成される新しいエントリーに対して同じグループを自動的に設定します。

6. 共有する各ディレクトリーについて、**/etc/exports** ファイルにエクスポートポイントを追加します。

   ```
   /nfs/projects/     192.0.2.0/24(rw) 2001:db8::/32(rw)
   ```

   このエントリーは、**/nfs/projects/** ディレクトリーを共有し、**192.0.2.0/24** および **2001:db8::/32** サブネット内のクライアントに読み取りおよび書き込みアクセスを許可します。

7. **firewalld** で適切なポートを開きます。

```
# firewall-cmd --permanent --add-service={nfs,rpc-bind,mountd}
# firewall-cmd --permanent --add-port={5555/tcp,5555/udp,6666/tcp,6666/udp}
# firewall-cmd --reload
```

8. NFS サーバーを有効にして起動します。

```
# systemctl enable --now rpc-statd nfs-server
```

検証

- サーバー上で、設定した NFS バージョンのみがサーバーから提供されていることを確認します。

```
# cat /proc/fs/nfsd/versions
+3 +4 -4.0 -4.1 +4.2
```

- クライアントで次の手順を実行します。

1. **nfs-utils** パッケージをインストールします。

```
# dnf install nfs-utils
```

2. エクスポートされた NFS 共有をマウントします。

```
# mount -o vers=<version> server.example.com:/nfs/projects/ /mnt/
```

3. 指定した NFS バージョンを使用して共有がマウントされたことを確認します。

```
# mount | grep "/mnt"
server.example.com:/nfs/projects/ on /mnt type nfs (rw,relatime,vers=3,...
```

4. **users** グループのメンバーであるユーザーとして、**/mnt/** にファイルを作成します。

```
# touch /mnt/file
```

5. ファイルが作成されたことを確認するためにディレクトリーの内容をリスト表示します。

```
# ls -l /mnt/
total 0
-rw-r--r--. 1 demo users 0 Jan 16 14:18 file
```

5.9. NFS サーバーでクォータサポートを有効にする

ユーザーまたはグループが保存できるデータの量を制限する場合は、ファイルシステムにクォータを設定できます。クォータは、NFS サーバー上の **rpc-rquotad** サービスにより、NFS クライアント上のユーザーにも適用されます。

前提条件

- NFS サーバーが実行および設定されている。
- ext または XFS ファイルシステムにクォータが設定されている。

手順

1. エクスポートするディレクトリーでクォータが有効になっていることを確認します。

   - ext ファイルシステムの場合は、次のように入力します。

     ```
     # quotaon -p /nfs/projects/
     group quota on /nfs/projects (/dev/sdb1) is on
     user quota on /nfs/projects (/dev/sdb1) is on
     project quota on /nfs/projects (/dev/sdb1) is off
     ```

   - XFS ファイルシステムの場合は、次のように入力します。

     ```
     # findmnt /nfs/projects
     TARGET     SOURCE FSTYPE OPTIONS
     /nfs/projects /dev/sdb1 xfs
     rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,usrquota,grpquota
     ```

2. **quota-rpc** パッケージをインストールします。

   ```
   # dnf install quota-rpc
   ```

3. オプション: デフォルトでは、クォータ RPC サービスはポート 875 で実行されます。別のポートでサービスを実行する場合は、**/etc/sysconfig/rpc-rquotad** ファイルの **RPCRQUOTADOPTS** 変数に **-p <port_number>** を追加します。

   ```
   RPCRQUOTADOPTS="-p __<port_number>__"
   ```

4. オプション: デフォルトでは、リモートホストはクォータの読み取りのみが可能です。クライアントにクォータの設定を許可するには、**/etc/sysconfig/rpc-rquotad** ファイルの **RPCRQUOTADOPTS** 変数に **-S** オプションを追加します。

   ```
   RPCRQUOTADOPTS="-S"
   ```

5. **firewalld** でポートを開きます。

   ```
   # firewall-cmd --permanent --add-port=875/udp
   # firewall-cmd --reload
   ```

6. **rpc-rquotad** サービスを有効にして起動します。

   ```
   # systemctl enable --now rpc-rquotad
   ```

検証

1. クライアントで以下を実行します。

   a. エクスポートされた共有をマウントします。

      ```
      # mount server.example.com:/nfs/projects/ /mnt/
      ```

   b. クォータを表示します。コマンドは、エクスポートされたディレクトリーのファイルシステムによって異なります。以下に例を示します。

- マウントした全 ext ファイルシステムの特定ユーザーのクォータを表示するには、次のように入力します。

```
# quota -u <user_name>
Disk quotas for user demo (uid 1000):
     Filesystem  space  quota  limit  grace  files  quota  limit  grace
server.example.com:/nfs/projects
               0K      100M     200M              0        0        0
```

- XFS ファイルシステムのユーザーおよびグループのクォータを表示するには、次のように入力します。

```
# xfs_quota -x -c "report -h" /mnt/
User quota on /nfs/projects (/dev/vdb1)
                Blocks
User ID     Used     Soft     Hard     Warn/Grace
---------- ---------------------------------
root        0        0        0        00 [------]
demo        0        100M     200M     00 [------]
```

関連情報

- **quota(1)** man ページ
- **xfs_quota(8)** man ページ

5.10. NFS サーバーで NFS OVER RDMA を有効にする

Remote Direct Memory Access (RDMA) は、クライアントシステムがストレージサーバーのメモリーから自身のメモリーにデータを直接転送できるようにするプロトコルです。これにより、ストレージのスループットが向上し、サーバーとクライアント間のデータ転送の遅延が減少し、両側の CPU 負荷が軽減されます。NFS サーバーとクライアントの両方が RDMA 経由で接続されている場合、クライアントは NFS over RDMA (NFSoRDMA) を使用してエクスポートされたディレクトリーをマウントできます。

前提条件

- NFS サービスが実行および設定されている。
- InfiniBand または RDMA over Converged Ethernet (RoCE) デバイスがサーバーにインストールされている。
- サーバーに IP over InfiniBand (IPoIB) が設定され、InfiniBand デバイスに IP アドレスが割り当てられている。

手順

1. **rdma-core** パッケージをインストールします。

```
# dnf install rdma-core
```

2. パッケージがすでにインストールされている場合は、**/etc/rdma/modules/rdma.conf** ファイル内の **xprtrdma** および **svcrdma** モジュールのコメントが解除されていることを確認します。

```
# NFS over RDMA client support
```

```
xprtrdma
# NFS over RDMA server support
svcrdma
```

3. オプション: デフォルトでは、NFS over RDMA はポート 20049 を使用します。別のポートを使用する場合は、**/etc/nfs.conf** ファイルの **[nfsd]** セクションで **rdma-port** 設定を指定します。

```
rdma-port=<port>
```

4. **firewalld** で NFSoRDMA ポートを開きます。

```
# firewall-cmd --permanent --add-port={20049/tcp,20049/udp}
# firewall-cmd --reload
```

20049 以外のポートを設定する場合は、ポート番号を変更します。

5. **nfs-server** サービスを再起動します。

```
# systemctl restart nfs-server
```

検証

1. InfiniBand ハードウェアを搭載したクライアントで、次の手順を実行します。
   a. 以下のパッケージをインストールします。

```
# dnf install nfs-utils rdma-core
```

   b. エクスポートされた NFS 共有を RDMA 経由でマウントします。

```
# mount -o rdma server.example.com:/nfs/projects/ /mnt/
```

デフォルト (20049) 以外のポート番号を設定する場合は、コマンドに **port=<port_number>** を渡します。

```
# mount -o rdma,port=<port_number> server.example.com:/nfs/projects/ /mnt/
```

   c. **rdma** オプションを使用して共有がマウントされたことを確認します。

```
# mount | grep "/mnt"
server.example.com:/nfs/projects/ on /mnt type nfs (...,proto=rdma,...)
```

関連情報

- InfiniBand ネットワークおよび RDMA ネットワークの設定

5.11. RED HAT IDENTITY MANAGEMENT ドメインで KERBEROS を使用する NFS サーバーを設定する

Red Hat Identity Management (IdM) を使用すると、NFS サーバーを IdM ドメインに参加させることができます。これにより、ユーザーとグループを一元管理し、認証、整合性保護、トラフィック暗号化に Kerberos を使用できるようになります。

前提条件

- NFS サーバーが Red Hat Identity Management (IdM) ドメインに 登録 されている。
- NFS サーバーが実行および設定されている。

手順

1. IdM 管理者として Kerberos チケットを取得します。

```
# kinit admin
```

2. **nfs/<FQDN>** サービスプリンシパルを作成します。

```
# ipa service-add nfs/nfs_server.idm.example.com
```

3. IdM から **nfs** サービスプリンシパルを取得し、**/etc/krb5.keytab** ファイルに保存します。

```
# ipa-getkeytab -s idm_server.idm.example.com -p nfs/nfs_server.idm.example.com -k
/etc/krb5.keytab
```

4. オプション: **/etc/krb5.keytab** ファイル内のプリンシパルを表示します。

```
# klist -k /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
---- --------------------------------------------------------------------------
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
   7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
```

デフォルトでは、ホストを IdM ドメインに参加させると、IdM クライアントがホストプリンシパルを **/etc/krb5.keytab** ファイルに追加します。ホストプリンシパルがない場合は、**ipa-getkeytab -s idm_server.idm.example.com -p host/nfs_server.idm.example.com -k /etc/krb5.keytab** コマンドを使用して追加します。

5. **ipa-client-automount** ユーティリティーを使用して、IdM ID のマッピングを設定します。

```
#  ipa-client-automount
Searching for IPA server...
IPA server: DNS discovery
Location: default
Continue to configure the system with these values? [no]: yes
Configured /etc/idmapd.conf
Restarting sssd, waiting for it to become available.
Started autofs
```

6. **/etc/exports** ファイルを更新し、クライアントオプションに Kerberos セキュリティー方式を追加します。以下に例を示します。

```
/nfs/projects/       192.0.2.0/24(rw,sec=krb5i)
```

クライアントが複数のセキュリティー方式を選択できるようにするには、それらをコロンで区切って指定します。

```
/nfs/projects/       192.0.2.0/24(rw,sec=krb5:krb5i:krb5p)
```

7. エクスポートされたファイルシステムを再ロードします。

```
# exportfs -r
```

第6章 SQUID キャッシングプロキシーサーバーの設定

Squid は、コンテンツをキャッシュして帯域幅を削減し、Web ページをより迅速に読み込むプロキシーサーバーです。本章では、HTTP、HTTPS、FTP のプロトコルのプロキシーとして Squid を設定する方法と、アクセスの認証および制限を説明します。

6.1. 認証なしで SQUID をキャッシングプロキシーとして設定

認証なしで Squid をキャッシュプロキシーとして設定できます。以下の手順では、IP 範囲に基づいてプロキシーへのアクセスを制限します。

前提条件

- **/etc/squid/squid.conf** ファイルが、**squid** パッケージにより提供されている。このファイルを編集した場合は、ファイルを削除して、パッケージを再インストールしている。

手順

1. **squid** パッケージをインストールします。

```
# yum install squid
```

2. **/etc/squid/squid.conf** ファイルを編集します。

   a. **localnet** アクセス制御リスト (ACL) を、プロキシーを使用できる IP 範囲と一致するように変更します。

```
acl localnet src 192.0.2.0/24
acl localnet 2001:db8:1::/64
```

   デフォルトでは、**/etc/squid/squid.conf** ファイルには **localnet** ACL で指定されたすべての IP 範囲のプロキシーを使用できるようにする **http_access allow localnet** ルールが含まれます。**http_access allow localnet** ルールの前に、**localnet** の ACL をすべて指定する必要があります。

重要

環境に一致しない既存の **acl localnet** エントリーをすべて削除します。

   b. 以下の ACL はデフォルト設定にあり、HTTPS プロトコルを使用するポートとして **443** を定義します。

```
acl SSL_ports port 443
```

   ユーザーが他のポートでも HTTPS プロトコルを使用できるようにするには、ポートごとに ACL を追加します。

```
acl SSL_ports port port_number
```

   c. Squid が接続を確立できるポートに設定する **acl Safe_ports** ルールの一覧を更新します。たとえば、プロキシーを使用するクライアントがポート 21 (FTP)、80 (HTTP)、443 (HTTPS) のリソースにのみアクセスできるようにするには、その設定の以下の **acl Safe_ports** ステートメントのみを保持します。

```
acl Safe_ports port 21
acl Safe_ports port 80
acl Safe_ports port 443
```

デフォルトでは、設定には **http_access deny !Safe_ports** ルールが含まれ、**Safe_ports** ACL で定義されていないポートへのアクセス拒否を定義します。

d. **cache_dir** パラメーターにキャッシュの種類、キャッシュディレクトリーへのパス、キャッシュサイズ、さらにキャッシュの種類ごとの設定を設定します。

```
cache_dir ufs /var/spool/squid 10000 16 256
```

この設定により、以下が可能になります。

- Squid は、**ufs** キャッシュタイプを使用します。
- Squid は、キャッシュを **/var/spool/squid/** ディレクトリーに保存します。
- キャッシュのサイズが **10000** MB まで大きくなります。
- Squid は、**16** 個の レベル1サブディレクトリーを **/var/spool/squid/** ディレクトリーに作成します。
- Squid は、レベル1の各ディレクトリーに **256** 個のサブディレクトリーを作成します。**cache_dir** ディレクティブを設定しないと、Squid はキャッシュをメモリーに保存します。

3. **cache_dir** パラメーターに **/var/spool/squid/** 以外のキャッシュディレクトリーを設定する場合は、以下を行います。

a. キャッシュディレクトリーを作成します。

```
# mkdir -p path_to_cache_directory
```

b. キャッシュディレクトリーの権限を設定します。

```
# chown squid:squid path_to_cache_directory
```

c. SELinux を **enforcing** モードで実行する場合は、**squid_cache_t** コンテキストをキャッシュディレクトリーに設定します。

```
# semanage fcontext -a -t squid_cache_t "path_to_cache_directory(/.*)?"
# restorecon -Rv path_to_cache_directory
```

semanage ユーティリティーがシステムで利用できない場合は、**policycoreutils-python-utils** パッケージをインストールします。

4. ファイアウォールで **3128** ポートを開きます。

```
# firewall-cmd --permanent --add-port=3128/tcp
# firewall-cmd --reload
```

5. **squid** サービスを有効にして開始します。

```
# systemctl enable --now squid
```

検証手順

プロキシーが正しく機能することを確認するには、**curl** ユーティリティーを使用して Web ページをダウンロードします。

```
# curl -O -L "https://www.redhat.com/index.html" -x "proxy.example.com:3128"
```

curl でエラーが表示されず、**index.html** ファイルが現在のディレクトリーにダウンロードされている場合は、プロキシーが動作します。

6.2. LDAP 認証を使用したキャッシングプロキシーとしての SQUID の設定

Squid を、LDAP を使用してユーザーを認証するキャッシングプロキシーとして設定できます。この手順では、認証されたユーザーのみがプロキシーを使用できるように設定します。

前提条件

- **/etc/squid/squid.conf** ファイルが、**squid** パッケージにより提供されている。このファイルを編集した場合は、ファイルを削除して、パッケージを再インストールしている。
- **uid=proxy_user,cn=users,cn=accounts,dc=example,dc=com** などのサービスユーザーが LDAP ディレクトリーに存在します。Squid はこのアカウントを使用して認証ユーザーを検索します。認証ユーザーが存在する場合、Squid はこのユーザーをディレクトリーにバインドして、認証を確認します。

手順

1. **squid** パッケージをインストールします。

```
# yum install squid
```

2. **/etc/squid/squid.conf** ファイルを編集します。

   a. **basic_ldap_auth** ヘルパーユーティリティーを設定するには、**/etc/squid/squid.conf** に以下の設定エントリーを追加します。

```
auth_param basic program /usr/lib64/squid/basic_ldap_auth -b
"cn=users,cn=accounts,dc=example,dc=com" -D
"uid=proxy_user,cn=users,cn=accounts,dc=example,dc=com" -W
/etc/squid/ldap_password -f "(&(objectClass=person)(uid=%s))" -ZZ -H
ldap://ldap_server.example.com:389
```

   以下では、上記の **basic_ldap_auth** ヘルパーユーティリティーに渡されるパラメーターを説明します。

   - **-b base_DN** は LDAP 検索ベースを設定します。
   - **-D proxy_service_user_DN** は、Squid が、ディレクトリー内の認証ユーザーを検索する際に使用するアカウントの識別名 (DN) を設定します。

- **-W path_to_password_file** は、プロキシーサービスユーザーのパスワードが含まれるファイルへのパスを設定します。パスワードファイルを使用すると、オペレーティングシステムのプロセスリストにパスワードが表示されなくなります。

- **-f LDAP_filter** は 、DAP 検索フィルターを指定します。Squid は、**%s** 変数を、認証ユーザーにより提供されるユーザー名に置き換えます。
  上記の例の **(&(objectClass=person)(uid=%s))** フィルターは、ユーザー名が **uid** 属性に設定された値と一致する必要があり、ディレクトリーエントリーに **person** オブジェクトクラスが含まれることを定義します。

- **-ZZ** は、**STARTTLS** コマンドを使用して、LDAP プロトコルで TLS 暗号化接続を強制します。以下の状況で **-ZZ** を省略します。

  - LDAP サーバーは、暗号化された接続にを対応しません。

  - URL に指定されたポートは、LDAPS プロトコルを使用します。

- -H LDAP_URL パラメーターは、プロトコル、ホスト名、IP アドレス、および LDAP サーバーのポートを URL 形式で指定します。

b. 以下の ACL およびルールを追加して、Squid で、認証されたユーザーのみがプロキシーを使用できるように設定します。

```
acl ldap-auth proxy_auth REQUIRED
http_access allow ldap-auth
```

重要

http_access deny all ルールの前にこの設定を指定します。

c. 次のルールを削除して、**localnet** ACL で指定された IP 範囲のプロキシー認証の回避を無効にします。

```
http_access allow localnet
```

d. 以下の ACL はデフォルト設定にあり、HTTPS プロトコルを使用するポートとして **443** を定義します。

```
acl SSL_ports port 443
```

ユーザーが他のポートでも HTTPS プロトコルを使用できるようにするには、ポートごとに ACL を追加します。

```
acl SSL_ports port port_number
```

e. Squid が接続を確立できるポートに設定する **acl Safe_ports** ルールの一覧を更新します。たとえば、プロキシーを使用するクライアントがポート 21 (FTP)、80 (HTTP)、443 (HTTPS) のリソースにのみアクセスできるようにするには、その設定の以下の **acl Safe_ports** ステートメントのみを保持します。

```
acl Safe_ports port 21
acl Safe_ports port 80
acl Safe_ports port 443
```

デフォルトでは、設定には **http_access deny !Safe_ports** ルールが含まれ、**Safe_ports** ACL で定義されていないポートへのアクセス拒否を定義します。

f. **cache_dir** パラメーターにキャッシュの種類、キャッシュディレクトリーへのパス、キャッシュサイズ、さらにキャッシュの種類ごとの設定を設定します。

```
cache_dir ufs /var/spool/squid 10000 16 256
```

この設定により、以下が可能になります。

- Squid は、**ufs** キャッシュタイプを使用します。
- Squid は、キャッシュを **/var/spool/squid/** ディレクトリーに保存します。
- キャッシュのサイズが **10000** MB まで大きくなります。
- Squid は、**16** 個の レベル1サブディレクトリーを **/var/spool/squid/** ディレクトリーに作成します。
- Squid は、レベル1の各ディレクトリーに **256** 個のサブディレクトリーを作成します。**cache_dir** ディレクティブを設定しないと、Squid はキャッシュをメモリーに保存します。

3. **cache_dir** パラメーターに **/var/spool/squid/** 以外のキャッシュディレクトリーを設定する場合は、以下を行います。

a. キャッシュディレクトリーを作成します。

```
# mkdir -p path_to_cache_directory
```

b. キャッシュディレクトリーの権限を設定します。

```
# chown squid:squid path_to_cache_directory
```

c. SELinux を **enforcing** モードで実行する場合は、**squid_cache_t** コンテキストをキャッシュディレクトリーに設定します。

```
# semanage fcontext -a -t squid_cache_t "path_to_cache_directory(/.*)?"
# restorecon -Rv path_to_cache_directory
```

semanage ユーティリティーがシステムで利用できない場合は、**policycoreutils-python-utils** パッケージをインストールします。

4. LDAP サービスユーザーのパスワードを **/etc/squid/ldap_password** ファイルに保存し、ファイルに適切なパーミッションを設定します。

```
# echo "password" > /etc/squid/ldap_password
# chown root:squid /etc/squid/ldap_password
# chmod 640 /etc/squid/ldap_password
```

5. ファイアウォールで **3128** ポートを開きます。

```
# firewall-cmd --permanent --add-port=3128/tcp
# firewall-cmd --reload
```

6. **squid** サービスを有効にして開始します。

```
# systemctl enable --now squid
```

検証手順

プロキシーが正しく機能することを確認するには、**curl** ユーティリティーを使用して Web ページをダウンロードします。

```
# curl -O -L "https://www.redhat.com/index.html" -x
"user_name:password@proxy.example.com:3128"
```

curl がエラーを表示せず、**index.html** ファイルが現在のディレクトリーにダウンロードされている場合は、プロキシーが動作します。

トラブルシューティングの手順

ヘルパーユーティリティーが正しく機能していることを確認するには、以下の手順を行います。

1. **auth_param** パラメーターで使用したのと同じ設定で、ヘルパーユーティリティーを手動で起動します。

```
# /usr/lib64/squid/basic_ldap_auth -b "cn=users,cn=accounts,dc=example,dc=com" -D
"uid=proxy_user,cn=users,cn=accounts,dc=example,dc=com" -W
/etc/squid/ldap_password -f "(&(objectClass=person)(uid=%s))" -ZZ -H
ldap://ldap_server.example.com:389
```

2. 有効なユーザー名とパスワードを入力し、**Enter** を押します。

```
user_name password
```

ヘルパーユーティリティーが **OK** を返すと、認証に成功しました。

6.3. KERBEROS 認証を使用したキャッシングプロキシーとしての SQUID の設定

Squid を、Kerberos を使用して Active Directory (AD) に対してユーザーを認証するキャッシングプロキシーとして設定できます。この手順では、認証されたユーザーのみがプロキシーを使用できるように設定します。

前提条件

- **/etc/squid/squid.conf** ファイルが、**squid** パッケージにより提供されている。このファイルを編集した場合は、ファイルを削除して、パッケージを再インストールしている。

手順

1. 以下のパッケージをインストールします。

```
# yum install squid krb5-workstation
```

2. AD ドメイン管理者として認証します。

```
# kinit administrator@AD.EXAMPLE.COM
```

3. Squid 用のキータブを作成し、これを **/etc/squid/HTTP.keytab** ファイルに保存します。

```
# export KRB5_KTNAME=FILE:/etc/squid/HTTP.keytab
# net ads keytab CREATE -U administrator
```

4. **HTTP** サービスプリンシパルをキータブに追加します。

```
# net ads keytab ADD HTTP -U administrator
```

5. キータブファイルの所有者を **squid** ユーザーに設定します。

```
# chown squid /etc/squid/HTTP.keytab
```

6. 必要に応じて、キータブファイルに、プロキシーサーバーの完全修飾ドメイン名 (FQDN) の **HTTP** サービスプリンシパルが含まれていることを確認します。

```
# klist -k /etc/squid/HTTP.keytab
Keytab name: FILE:/etc/squid/HTTP.keytab
KVNO Principal
---- --------------------------------------------------
...
   2 HTTP/proxy.ad.example.com@AD.EXAMPLE.COM
...
```

7. **/etc/squid/squid.conf** ファイルを編集します。

   a. **negotiate_kerberos_auth** ヘルパーユーティリティーを設定するには、**/etc/squid/squid.conf** 部に以下の設定エントリーを追加します。

```
auth_param negotiate program /usr/lib64/squid/negotiate_kerberos_auth -k
/etc/squid/HTTP.keytab -s HTTP/proxy.ad.example.com@AD.EXAMPLE.COM
```

   以下は、上記の例で **negotiate_kerberos_auth** ヘルパーユーティリティーに渡されるパラメーターを説明します。

   - **-k file** は、キータブファイルへのパスを設定します。squid ユーザーには、このファイルに対する読み取り権限があることに注意してください。
   - **-s HTTP/host_name@kerberos_realm** は、Squid が使用する Kerberos プリンシパルを設定します。
     必要に応じて、以下のパラメーターのいずれかまたは両方をヘルパーユーティリティーに渡すことによりロギングを有効にできます。
   - **-i** は、認証ユーザーなどの情報メッセージをログに記録します。
   - **-d** は、デバッグロギングを有効にします。
     Squid は、ヘルパーユーティリティーから、**/var/log/squid/cache.log** ファイルにデバッグ情報のログに記録します。

   b. 以下の ACL およびルールを追加して、Squid で、認証されたユーザーのみがプロキシーを使用できるように設定します。

```
acl kerb-auth proxy_auth REQUIRED
http_access allow kerb-auth
```

重要

http_access deny all ルールの前にこの設定を指定します。

c. 次のルールを削除して、**localnet** ACL で指定された IP 範囲のプロキシー認証の回避を無効にします。

```
http_access allow localnet
```

d. 以下の ACL はデフォルト設定にあり、HTTPS プロトコルを使用するポートとして **443** を定義します。

```
acl SSL_ports port 443
```

ユーザーが他のポートでも HTTPS プロトコルを使用できるようにするには、ポートごとに ACL を追加します。

```
acl SSL_ports port port_number
```

e. Squid が接続を確立できるポートに設定する **acl Safe_ports** ルールの一覧を更新します。たとえば、プロキシーを使用するクライアントがポート 21 (FTP)、80 (HTTP)、443 (HTTPS) のリソースにのみアクセスできるようにするには、その設定の以下の **acl Safe_ports** ステートメントのみを保持します。

```
acl Safe_ports port 21
acl Safe_ports port 80
acl Safe_ports port 443
```

デフォルトでは、設定には **http_access deny !Safe_ports** ルールが含まれ、**Safe_ports** ACL で定義されていないポートへのアクセス拒否を定義します。

f. **cache_dir** パラメーターにキャッシュの種類、キャッシュディレクトリーへのパス、キャッシュサイズ、さらにキャッシュの種類ごとの設定を設定します。

```
cache_dir ufs /var/spool/squid 10000 16 256
```

この設定により、以下が可能になります。

- Squid は、**ufs** キャッシュタイプを使用します。
- Squid は、キャッシュを **/var/spool/squid/** ディレクトリーに保存します。
- キャッシュのサイズが **10000** MB まで大きくなります。
- Squid は、**16** 個の レベル 1 サブディレクトリーを **/var/spool/squid/** ディレクトリーに作成します。
- Squid は、レベル 1 の各ディレクトリーに **256** 個のサブディレクトリーを作成します。**cache_dir** ディレクティブを設定しないと、Squid はキャッシュをメモリーに保存します。

8. **cache_dir** パラメーターに **/var/spool/squid/** 以外のキャッシュディレクトリーを設定する場合は、以下を行います。

   a. キャッシュディレクトリーを作成します。

   ```
   # mkdir -p path_to_cache_directory
   ```

   b. キャッシュディレクトリーの権限を設定します。

   ```
   # chown squid:squid path_to_cache_directory
   ```

   c. SELinux を **enforcing** モードで実行する場合は、**squid_cache_t** コンテキストをキャッシュディレクトリーに設定します。

   ```
   # semanage fcontext -a -t squid_cache_t "path_to_cache_directory(/.*)?"
   # restorecon -Rv path_to_cache_directory
   ```

   semanage ユーティリティーがシステムで利用できない場合は、**policycoreutils-python-utils** パッケージをインストールします。

9. ファイアウォールで **3128** ポートを開きます。

   ```
   # firewall-cmd --permanent --add-port=3128/tcp
   # firewall-cmd --reload
   ```

10. **squid** サービスを有効にして開始します。

    ```
    # systemctl enable --now squid
    ```

検証手順

プロキシーが正しく機能することを確認するには、**curl** ユーティリティーを使用して Web ページをダウンロードします。

```
# curl -O -L "https://www.redhat.com/index.html" --proxy-negotiate -u : -x
"proxy.ad.example.com:3128"
```

curl がエラーを表示せず、**index.html** ファイルが現在のディレクトリーに存在すると、プロキシーが機能します。

トラブルシューティングの手順

Kerberos 認証を手動でテストするには、以下を行います。

1. AD アカウントの Kerberos チケットを取得します。

   ```
   # kinit user@AD.EXAMPLE.COM
   ```

2. 必要に応じて、キーを表示します。

   ```
   # klist
   ```

3. **negotiate_kerberos_auth_test** ユーティリティーを使用して認証をテストします。

```
# /usr/lib64/squid/negotiate_kerberos_auth_test proxy.ad.example.com
```

ヘルパーユーティリティーがトークンを返す場合は、認証に成功しました。

```
Token: YIIFtAYGKwYBBQUCoIIFqDC...
```

6.4. SQUID でのドメイン拒否リストの設定

多くの場合、管理者は特定のドメインへのアクセスをブロックする必要があります。本セクションでは、Squid でドメインの拒否リストを設定する方法を説明します。

前提条件

- Squid が設定され、ユーザーはプロキシーを使用できます。

手順

1. **/etc/squid/squid.conf** ファイルを編集し、以下の設定を追加します。

```
acl domain_deny_list dstdomain "/etc/squid/domain_deny_list.txt"
http_access deny all domain_deny_list
```

重要

ユーザーまたはクライアントへのアクセスを許可する最初の **http_access allow** ステートメントの前にこれらのエントリーを追加します。

2. **/etc/squid/domain_deny_list.txt** ファイルを作成し、ブロックするドメインを追加します。たとえば、サブドメインを含む **example.com** へのアクセスをブロックして、**example.net** をブロックするには、以下を追加します。

```
.example.com
example.net
```

重要

squid 設定の **/etc/squid/domain_deny_list.txt** ファイルを参照している場合は、このファイルは空にすることはできません。このファイルが空の場合、Squid は起動できません。

3. **squid** サービスを再開します。

```
# systemctl restart squid
```

6.5. SQUID サービスが特定のポートまたは IP アドレスをリッスンするように設定

デフォルトでは、Squid プロキシーサービスは、すべてのネットワークインターフェイスの **3128** ポートでリッスンします。ポートを変更し、Squid が特定の IP アドレスをリッスンするように設定できます。

前提条件

- **squid** パッケージがインストールされている。

手順

1. **/etc/squid/squid.conf** ファイルを編集します。
   - Squid サービスがリッスンするポートを設定するには、**http_port** パラメーターにポート番号を設定します。たとえば、ポートを **8080** に設定するには、以下を設定します。

     ```
     http_port 8080
     ```

   - Squid サービスがリッスンする IP アドレスを設定するには、**http_port** パラメーターに IP アドレスとポート番号を設定します。たとえば、Squid が、**3128** ポートの IP アドレス **192.0.2.1** でのみリッスンするように設定するには、以下を設定します。

     ```
     http_port 192.0.2.1:3128
     ```

     複数の **http_port** パラメーターを設定ファイルに追加して、Squid が複数のポートおよび IP アドレスでリッスンするように設定します。

     ```
     http_port 192.0.2.1:3128
     http_port 192.0.2.1:8080
     ```

2. Squid が別のポートをデフォルト (**3128**) として使用するように設定する場合は、以下のようになります。
   a. ファイアウォールのポートを開きます。

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

   b. enforcing モードで SELinux を実行した場合は、ポートを **squid_port_t** ポートタイプ定義に割り当てます。

      ```
      # semanage port -a -t squid_port_t -p tcp port_number
      ```

      semanage ユーティリティーがシステムで利用できない場合は、**policycoreutils-python-utils** パッケージをインストールします。

3. **squid** サービスを再開します。

   ```
   # systemctl restart squid
   ```

6.6. 関連情報

- 設定パラメーター **usr/share/doc/squid-<version>/squid.conf.documented**

第7章 データベースサーバー

7.1. データベースサーバーの概要

データベースサーバーは、データベース管理システム (DBMS) の機能を提供するサービスです。DBMS は、データベース管理のためのユーティリティーを提供し、エンドユーザー、アプリケーション、およびデータベースと対話します。

Red Hat Enterprise Linux 8 は、以下のデータベース管理システムを提供します。

- **MariaDB 10.3**
- **MariaDB 10.5** - RHEL 8.4 以降で利用できます。
- **MariaDB 10.11** - RHEL 9.4 以降で利用可能
- **MySQL 8.0**
- **PostgreSQL 10**
- **PostgreSQL 9.6**
- **PostgreSQL 12** - RHEL 8.1.1 以降で利用できます。
- **PostgreSQL 13** - RHEL 8.4 以降で利用できます。
- **PostgreSQL 15** - RHEL 8.8 以降で利用できます。
- **PostgreSQL 16** - RHEL 9.4 以降で利用可能

7.2. MARIADB の使用

MariaDB サーバーは、**MySQL** テクノロジーに基づくオープンソースの高速で堅牢なデータベースサーバーです。**MariaDB** は、データを構造化情報に変換して、データにアクセスする SQL インターフェイスを提供するリレーショナルデータベースです。これには、複数のストレージエンジンとプラグインに加え、地理情報システム (GIS) と JavaScript Object Notation (JSON) 機能も含まれています。

RHEL システムに **MariaDB** をインストールして設定する方法、**MariaDB** データをバックアップする方法、**MariaDB** の以前のバージョンから移行する方法、および **MariaDB Galera クラスター** を使用してデータベースを複製する方法について説明します。

7.2.1. MariaDB のインストール

RHEL 8 では、**MariaDB** サーバーは、それぞれ個別のストリームにより提供される以下のバージョンで利用できます。

- **MariaDB 10.3**
- **MariaDB 10.5** - RHEL 8.4 以降で利用できます。
- **MariaDB 10.11** - RHEL 9.4 以降で利用可能

注記

設計上、同じモジュールの複数のバージョン (ストリーム) を並行してインストールすることはできません。したがって、**mariadb** モジュールから利用可能なストリームのいずれかを選択する必要があります。コンテナー内では、別々のバージョンの MariaDB データベースサーバーを使用できます。コンテナー内で複数の MariaDB バージョンを実行する を参照してください。

RPM パッケージが競合しているため、RHEL 8 では MariaDB および MySQL データベースサーバーを同時にインストールすることはできません。コンテナー内では、MariaDB および MySQL データベースサーバーを並行して使用できます。コンテナー内で複数の MySQL および MariaDB バージョンを実行する を参照してください。

MariaDB をインストールするには、以下の手順を行います。

手順

1. **mariadb** モジュールからストリーム (バージョン) を選択し、**server** プロファイルを指定して MariaDB サーバーパッケージをインストールします。以下に例を示します。

```
# yum module install mariadb:10.3/server
```

2. **mariadb** サービスを起動します。

```
# systemctl start mariadb.service
```

3. **mariadb** サービスが、システムの起動時に起動するようにします。

```
# systemctl enable mariadb.service
```

4. **MariaDB 10.3 に推奨**: MariaDB をインストールする際のセキュリティーを向上させるには、次のコマンドを実行します。

```
$ mysql_secure_installation
```

このコマンドは、完全にインタラクティブなスクリプトを起動して、プロセスの各ステップのプロンプトを表示します。このスクリプトを使用すると、次の方法でセキュリティーを改善できます。

- root アカウントのパスワードの設定
- 匿名ユーザーの削除
- リモート root ログインの拒否 (ローカルホスト外)

注記

mysql_secure_installation スクリプトは、MariaDB 10.5 以降では価値がなくなりました。セキュリティーの強化は、MariaDB 10.5 以降のデフォルトの動作の一部です。

重要

RHEL 8 内で以前の **mariadb** ストリームからアップグレードする場合は、新しいストリームへの切り替え と MariaDB 10.3 から MariaDB 10.5 へのアップグレード または MariaDB 10.5 から MariaDB 10.11 へのアップグレード の両方の手順に従ってください。

7.2.1.1. コンテナー内で複数の MariaDB バージョンを実行する

同じホスト上で別々のバージョンの MariaDB を実行するには、コンテナー内で実行してください。同じモジュールの複数のバージョン (ストリーム) を並行してインストールすることはできないためです。

前提条件

- **container-tools** モジュールがインストールされている。

手順

1. Red Hat カスタマーポータルアカウントを使用して、**registry.redhat.io** レジストリーに認証します。

```
# podman login registry.redhat.io
```

すでにコンテナーレジストリーにログインしている場合は、このステップをスキップしてください。

2. コンテナー内で MariaDB 10.11 を実行します。

```
$ podman run -d --name <container_name> -e
MYSQL_ROOT_PASSWORD=<mariadb_root_password> -p <host_port_1>:3306
rhel8/mariadb-103
```

このコンテナーイメージを使用する方法の詳細は、Red Hat Ecosystem Catalog を参照してください。

3. コンテナー内で MariaDB 10.5 を実行します。

```
$ podman run -d --name <container_name> -e
MYSQL_ROOT_PASSWORD=<mariadb_root_password> -p <host_port_2>:3306
rhel8/mariadb-105
```

このコンテナーイメージを使用する方法の詳細は、Red Hat Ecosystem Catalog を参照してください。

4. コンテナー内で MariaDB 10.11 を実行します。

```
$ podman run -d --name <container_name> -e
MYSQL_ROOT_PASSWORD=<mariadb_root_password> -p <host_port_3>:3306
rhel8/mariadb-1011
```

このコンテナーイメージを使用する方法の詳細は、Red Hat Ecosystem Catalog を参照してください。

注記

2 つのデータベースサーバーのコンテナー名とホストポートが異なっている必要があります。

5. クライアントがネットワーク上のデータベースサーバーにアクセスできるように、ファイアウォールでホストポートを開きます。

```
# firewall-cmd --permanent --add-port=
{<host_port_1>/tcp,<host_port_2>/tcp,<host_port_3>/tcp...}
# firewall-cmd --reload
```

検証手順

1. 実行中のコンテナーに関する情報を表示します。

```
$ podman ps
```

2. データベースサーバーに接続し、root としてログインします。

```
# mysql -u root -p -h localhost -P <host_port> --protocol tcp
```

関連情報

- コンテナーの構築、実行、および管理
- Red Hat Ecosystem Catalog でコンテナーを参照する

7.2.2. MariaDB の設定

MariaDB サーバーをネットワーク用に設定するには、以下の手順に従います。

手順

1. **/etc/my.cnf.d/mariadb-server.cnf** ファイルの **[mysqld]** セクションを編集します。以下の設定ディレクティブを設定できます。
   - **bind-address**: サーバーがリッスンするアドレスです。設定可能なオプションは以下のとおりです。
     - ホスト名
     - IPv4 アドレス
     - IPv6 アドレス
   - **skip-networking**: サーバーが TCP/IP 接続をリッスンするかどうかを制御します。以下の値が使用できます。
     - 0 - すべてのクライアントをリッスンする
     - 1 - ローカルクライアントのみをリッスンする
   - **port**: MariaDB が TCP/IP 接続をリッスンするポート。

2. **mariadb** サービスを再起動します。

```
# systemctl restart mariadb.service
```

7.2.3. MariaDB サーバーでの TLS 暗号化の設定

デフォルトでは、MariaDB は暗号化されていない接続を使用します。安全な接続のために、MariaDB サーバーで TLS サポートを有効にし、暗号化された接続を確立するようにクライアントを設定します。

7.2.3.1. MariaDB サーバーに CA 証明書、サーバー証明書、および秘密鍵を配置する

MariaDB サーバーで TLS 暗号化を有効にする前に、認証局 (CA) 証明書、サーバー証明書、および秘密鍵を MariaDB サーバーに保存します。

前提条件

- Privacy Enhanced Mail(PEM) 形式の以下のファイルがサーバーにコピーされています。
  - サーバーの秘密鍵: **server.example.com.key.pem**
  - サーバー証明書: **server.example.com.crt.pem**
  - 認証局 (CA) 証明書: **ca.crt.pem**

  秘密鍵および証明書署名要求 (CSR) の作成や、CA からの証明書要求に関する詳細は、CA のドキュメントを参照してください。

手順

1. CA およびサーバー証明書を **/etc/pki/tls/certs/** ディレクトリーに保存します。

```
# mv <path>/server.example.com.crt.pem /etc/pki/tls/certs/
# mv <path>/ca.crt.pem /etc/pki/tls/certs/
```

2. MariaDB サーバーがファイルを読み込めるように、CA およびサーバー証明書にパーミッションを設定します。

```
# chmod 644 /etc/pki/tls/certs/server.example.com.crt.pem /etc/pki/tls/certs/ca.crt.pem
```

   証明書は、セキュアな接続が確立される前は通信の一部であるため、任意のクライアントは認証なしで証明書を取得できます。そのため、CA およびサーバーの証明書ファイルに厳密なパーミッションを設定する必要はありません。

3. サーバーの秘密鍵を **/etc/pki/tls/private/** ディレクトリーに保存します。

```
# mv <path>/server.example.com.key.pem /etc/pki/tls/private/
```

4. サーバーの秘密鍵にセキュアなパーミッションを設定します。

```
# chmod 640 /etc/pki/tls/private/server.example.com.key.pem
# chgrp mysql /etc/pki/tls/private/server.example.com.key.pem
```

承認されていないユーザーが秘密鍵にアクセスできる場合は、MariaDB サーバーへの接続は安全ではなくなります。

5. SELinux コンテキストを復元します。

```
# restorecon -Rv /etc/pki/tls/
```

7.2.3.2. MariaDB サーバーでの TLS の設定

セキュリティーを向上させるには、MariaDB サーバーで TLS サポートを有効にします。その結果、クライアントは TLS 暗号化を使用してサーバーでデータを送信できます。

前提条件

- MariaDB サーバーをインストールしている。
- **mariadb** サービスが実行している。
- Privacy Enhanced Mail(PEM) 形式の以下のファイルがサーバー上にあり、**mysql** ユーザーが読み取りできます。
  - サーバーの秘密鍵: **/etc/pki/tls/private/server.example.com.key.pem**
  - サーバー証明書: **/etc/pki/tls/certs/server.example.com.crt.pem**
  - 認証局 (CA) 証明書 **/etc/pki/tls/certs/ca.crt.pem**
- サーバー証明書のサブジェクト識別名 (DN) またはサブジェクトの別名 (SAN) フィールドは、サーバーのホスト名と一致します。

手順

1. **/etc/my.cnf.d/mariadb-server-tls.cnf** ファイルを作成します。
   a. 以下の内容を追加して、秘密鍵、サーバー、および CA 証明書へのパスを設定します。

```
[mariadb]
ssl_key = /etc/pki/tls/private/server.example.com.key.pem
ssl_cert = /etc/pki/tls/certs/server.example.com.crt.pem
ssl_ca = /etc/pki/tls/certs/ca.crt.pem
```

   b. 証明書失効リスト (CRL) がある場合は、それを使用するように MariaDB サーバーを設定します。

```
ssl_crl = /etc/pki/tls/certs/example.crl.pem
```

   c. 必要に応じて、MariaDB 10.5.2 以降を実行する場合は、暗号化せずに接続を拒否できます。この機能を有効にするには、以下を追加します。

```
require_secure_transport = on
```

   d. 必要に応じて、MariaDB 10.4.6 以降を実行する場合は、サーバーが対応する TLS バージョンを設定できます。たとえば、TLS 1.2 および TLS 1.3 をサポートするには、以下を追加します。

```
tls_version = TLSv1.2,TLSv1.3
```

デフォルトでは、サーバーは TLS 1.1、TLS 1.2、および TLS 1.3 をサポートします。

2. **mariadb** サービスを再起動します。

```
# systemctl restart mariadb
```

検証

トラブルシューティングを簡素化するには、ローカルクライアントが TLS 暗号化を使用するように設定する前に、MariaDB サーバーで以下の手順を実行します。

1. MariaDB で TLS 暗号化が有効になっていることを確認します。

```
# mysql -u root -p -e "SHOW GLOBAL VARIABLES LIKE 'have_ssl';"
+---------------+-----------------+
| Variable_name | Value           |
+---------------+-----------------+
| have_ssl      | YES             |
+---------------+-----------------+
```

have_ssl 変数が **yes** に設定されている場合、TLS 暗号化が有効になります。

2. MariaDB サービスが特定の TLS バージョンのみをサポートするように設定している場合は、**tls_version** 変数を表示します。

```
# mysql -u root -p -e "SHOW GLOBAL VARIABLES LIKE 'tls_version';"
+---------------+-----------------+
| Variable_name | Value           |
+---------------+-----------------+
| tls_version   | TLSv1.2,TLSv1.3 |
+---------------+-----------------+
```

関連情報

- MariaDB サーバーに CA 証明書、サーバー証明書、および秘密鍵を配置する
- MariaDB 10.3 から MariaDB 10.5 へのアップグレード
- MariaDB 10.5 から MariaDB 10.11 へのアップグレード

7.2.3.3. 特定のユーザーアカウントに TLS で暗号化された接続を要求する

機密データにアクセスできるユーザーは、ネットワーク上で暗号化されていないデータ送信を回避するために、常に TLS で暗号化された接続を使用する必要があります。

すべての接続にセキュアなトランスポートが必要なサーバーで設定できない場合は (**require_secure_transport = on**)、TLS 暗号化を必要とするように個別のユーザーアカウントを設定します。

前提条件

- MariaDB サーバーで TLS サポートが有効になっている。

- セキュアなトランスポートを必要とするように設定するユーザーが存在する。
- クライアントは、サーバーの証明書を発行した CA 証明書を信頼している。

手順

1. 管理ユーザーとしてMariaDB サーバーに接続します。

```
# mysql -u root -p -h server.example.com
```

管理ユーザーがリモートでサーバーにアクセスする権限を持たない場合は、MariaDB サーバーでコマンドを実行し、**localhost** に接続します。

2. **REQUIRE SSL** 句を使用して、ユーザーが TLS 暗号化接続を使用して接続する必要があるよう強制します。

```
MariaDB [(none)]> ALTER USER 'example'@'%' REQUIRE SSL;
```

検証

1. TLS 暗号化を使用して、**example** ユーザーとしてサーバーに接続します。

```
# mysql -u example -p -h server.example.com --ssl
...
MariaDB [(none)]>
```

エラーが表示されず、インタラクティブな MariaDB コンソールにアクセスできる場合は、TLS との接続は成功します。

2. TLS を無効にして、**example** ユーザーとして接続を試みます。

```
# mysql -u example -p -h server.example.com --skip-ssl
ERROR 1045 (28000): Access denied for user 'example'@'server.example.com' (using password: YES)
```

このユーザーに TLS が必要だが無効になっているため、サーバーはログインの試行を拒否しました (**--skip-ssl**)。

関連情報

- MariaDB サーバーでの TLS 暗号化の設定

7.2.4. MariaDB クライアントでの TLS 暗号化のグローバルな有効化

MariaDB サーバーが TLS 暗号化に対応している場合は、安全な接続のみを確立し、サーバー証明書を検証するようにクライアントを設定します。この手順では、サーバー上のすべてのユーザーで TLS サポートを有効にする方法を説明します。

7.2.4.1. デフォルトで TLS 暗号化を使用するように MariaDB クライアントを設定する

RHEL では、MariaDB クライアントが TLS 暗号化を使用するようにグローバルに設定でき、サーバー証明書の Common Name (CN) が、ユーザーが接続するホスト名と一致することを検証します。これにより、man-in-the-middle 攻撃 (中間者攻撃) を防ぎます。

前提条件

- MariaDB サーバーで TLS サポートが有効になっている。
- サーバー証明書を発行した認証局 (CA) が RHEL で信頼されていない場合は、CA 証明書がクライアントにコピーされています。

手順

1. RHEL が、サーバー証明書を発行した CA を信頼しない場合は、以下を行います。
   a. 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 <path>/ca.crt.pem /etc/pki/ca-trust/source/anchors/
      ```

   b. すべてのユーザーが CA 証明書ファイルを読み取りできるようにするパーミッションを設定します。

      ```
      # chmod 644 /etc/pki/ca-trust/source/anchors/ca.crt.pem
      ```

   c. CA 信頼データベースを再構築します。

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

2. 以下の内容で **/etc/my.cnf.d/mariadb-client-tls.cnf** ファイルを作成します。

   ```
   [client-mariadb]
   ssl
   ssl-verify-server-cert
   ```

   これらの設定は、MariaDB クライアントが TLS 暗号化 (**ssl**) を使用し、クライアントがホスト名をサーバー証明書 (**ssl-verify-server-cert**) の CN と比較することを定義します。

検証

- ホスト名を使用してサーバーに接続し、サーバーの状態を表示します。

  ```
  # mysql -u root -p -h server.example.com -e status
  ...
  SSL:        Cipher in use is TLS_AES_256_GCM_SHA384
  ```

  SSL エントリーに **Cipher in use is…** が含まれている場合、接続は暗号化されています。

  このコマンドで使用するユーザーには、リモートで認証するパーミッションがあることに注意してください。

  接続するホスト名がサーバーの TLS 証明書のホスト名と一致しない場合、**ssl-verify-server-cert** パラメーターにより接続が失敗します。たとえば、**localhost** に接続する場合は、以下のようになります。

  ```
  # mysql -u root -p -h localhost -e status
  ERROR 2026 (HY000): SSL connection error: Validation of SSL server certificate failed
  ```

関連情報

- **mysql(1)** man ページの **--ssl*** パラメーターの説明。

7.2.5. MariaDB データのバックアップ

Red Hat EnterpriseLinux 8 で **MariaDB** データベースからデータをバックアップする主な方法は 2 つあります。

- 論理バックアップ
- 物理バックアップ

論理バックアップ は、データの復元に必要な SQL ステートメントで設定されます。この種類のバックアップは、情報およびレコードをプレーンテキストファイルにエクスポートします。

物理バックアップに対する論理バックアップの主な利点は、移植性と柔軟性です。データは、物理バックアップではできない他のハードウェア設定である **MariaDB** バージョンまたはデータベース管理システム (DBMS) で復元できます。

mariadb.service が稼働している場合は、論理バックアップを実行できることに注意してください。論理バックアップには、ログと設定ファイルが含まれません。

物理バックアップ は、コンテンツを格納するファイルおよびディレクトリーのコピーで設定されます。

物理バックアップは、論理バックアップと比較して、以下の利点があります。

- 出力が少なくなる。
- バックアップのサイズが小さくなる。
- バックアップおよび復元が速くなる。
- バックアップには、ログファイルと設定ファイルが含まれる。

mariadb.service が実行していない場合や、データベースのすべてのテーブルがロックされていて、バックアップ中に変更しないようにする場合は、物理バックアップを実行する必要があります。

以下のいずれかの **MariaDB** バックアップ方法で、**MariaDB** データベースのデータのバックアップを使用できます。

- **mysqldump** を使用した論理バックアップ
- **Mariabackup** ユーティリティーを使用した物理的なオンラインバックアップ
- ファイルシステムのバックアップ
- バックアップソリューションとしてレプリケーションを使用

7.2.5.1. mysqldump を使用した論理バックアップの実行

mysqldump クライアントはバックアップユーティリティーで、バックアップ目的でデータベースまたはデータベースの集合をダンプしたり、別のデータベースサーバーに転送したりできます。通常、**mysqldump** の出力は、サーバーテーブル構造を再作成する、それにデータを取り込む、またはその両方の SQL ステートメントで設定されます。**mysqldump** は、XML および (CSV などの) コンマ区切りテキスト形式など、他の形式でファイルを生成することもできます。

mysqldump バックアップを実行するには、以下のいずれかのオプションを使用できます。

- 選択したデータベースを 1 つまたは複数バックアップ
- すべてのデータベースをバックアップする。
- あるデータベースのテーブルのサブセットのバックアップを作成する。

手順

- 単一のデータベースをダンプするには、以下を実行します。

```
# mysqldump [options] --databases db_name > backup-file.sql
```

- 複数のデータベースを一度にダンプするには、次のコマンドを実行します。

```
# mysqldump [options] --databases db_name1 [db_name2 ...] > backup-file.sql
```

- すべてのデータベースをダンプするには、以下を実行します。

```
# mysqldump [options] --all-databases > backup-file.sql
```

- 1 つ以上のダンプされたフルデータベースをサーバーにロードし直すには、以下を実行します。

```
# mysql < backup-file.sql
```

- データベースをリモート MariaDB サーバーにロードするには、以下を実行します。

```
# mysql --host=remote_host < backup-file.sql
```

- あるデータベースでテーブルのサブセットをダンプするには、**mysqldump** コマンドの末尾に、選択したテーブルのリストを追加します。

```
# mysqldump [options] db_name [tbl_name ...] > backup-file.sql
```

- 1 つのデータベースからダンプされたテーブルのサブセットをロードするには、以下を実行します。

```
# mysql db_name < backup-file.sql
```

注記

この時点で、db_name データベースが存在している必要があります。

- mysqldump がサポートするオプションのリストを表示するには、以下を実行します。

```
$ mysqldump --help
```

関連情報

- mysqldump を使用した論理バックアップの詳細は、MariaDB Documentation を参照してください。

7.2.5.2. Mariabackup ユーティリティーを使用した物理的なオンラインバックアップの実行

Mariabackup は、Percona XtraBackup テクノロジーをベースとしたユーティリティーです。これにより、InnoDB、Aria、および MyISAM テーブルの物理的なオンラインバックアップを実行できます。このユーティリティーは、AppStream リポジトリーから **mariadb-backup** パッケージで提供されます。

Mariabackup は、MariaDB サーバーの完全バックアップ機能に対応します。これには、暗号化されたデータおよび圧縮データが含まれます。

前提条件

- **mariadb-backup** パッケージがシステムにインストールされている。

```
# yum install mariadb-backup
```

- Mariabackup には、バックアップを実行するユーザーの認証情報を指定する必要があります。認証情報はコマンドラインまたは設定ファイルで指定できます。
- Mariabackup のユーザーは、**RELOAD**、**LOCK TABLES**、および **REPLICATION CLIENT** の権限が必要です。

Mariabackup を使用してデータベースのバックアップを作成するには、以下の手順を行います。

手順

- コマンドラインで認証情報を提供する間にバックアップを作成するには、以下を実行します。

```
$ mariabackup --backup --target-dir <backup_directory> --user <backup_user> --
password <backup_passwd>
```

target-dir オプションは、バックアップファイルを格納するディレクトリーを定義します。完全バックアップを実行する場合は、ターゲットディレクトリーが空であるか、存在しない必要があります。

ユーザー オプションおよび **パスワード** オプションにより、ユーザー名とパスワードを設定できます。

- 設定ファイルに認証情報を設定してバックアップを作成するには、次のコマンドを実行します。

1. **/etc/my.cnf.d/** ディレクトリーに設定ファイルを作成します (例: **/etc/my.cnf.d/mariabackup.cnf**)。
2. 以下の行を新規ファイルの **[xtrabackup]** セクションまたは **[mysqld]** セクションに追加します。

```
[xtrabackup]
user=myuser
password=mypassword
```

3. バックアップを実行します。

```
$ mariabackup --backup --target-dir <backup_directory>
```

関連情報

- Mariabackupによる完全バックアップと復元

7.2.5.3. Mariabackup ユーティリティーを使用したデータの復元

バックアップが完了したら、**mariabackup** コマンドに以下のいずれかのオプションを使用して、バックアップからデータを復元できます。

- **--copy-back** を使用すると、元のバックアップファイルを保持できます。
- **--move-back** は、バックアップファイルをデータディレクトリーに移動し、元のバックアップファイルを削除します。

Mariabackup ユーティリティーを使用してデータを復元するには、以下の手順に従います。

前提条件

- **mariadb** サービスが実行されていないことを確認します。

```
# systemctl stop mariadb.service
```

- データディレクトリーが空であることを確認します。
- Mariabackup のユーザーは、**RELOAD**、**LOCK TABLES**、および **REPLICATION CLIENT** の権限が必要です。

手順

1. **mariabackup** コマンドを実行します。
   - データを復元し、元のバックアップファイルを保持するには、**--copy-back** オプションを使用します。

     ```
     $ mariabackup --copy-back --target-dir=/var/mariadb/backup/
     ```

   - データを復元し、元のバックアップファイルを削除するには、**--move-back** オプションを使用します。

     ```
     $ mariabackup --move-back --target-dir=/var/mariadb/backup/
     ```

2. ファイルの権限を修正します。
   データベースを復元するとき、Mariabackup は、バックアップのファイルおよびディレクトリーの権限を保持します。ただし、Mariabackup は、ユーザーおよびグループがデータベースを復元する際にファイルをディスクに書き込みます。バックアップの復元後、MariaDB サーバーのユーザーおよびグループ (通常は共に **mysql**) が一致するように、データディレクトリーの所有者の調整が必要になる場合があります。

   たとえば、ファイルの所有権を **mysql** ユーザーおよびグループに再帰的に変更するには、次のコマンドを実行します。

   ```
   # chown -R mysql:mysql /var/lib/mysql/
   ```

3. **mariadb** サービスを起動します。

```
# systemctl start mariadb.service
```

関連情報

- Mariabackupによる完全バックアップと復元

7.2.5.4. ファイルシステムのバックアップの実行

MariaDB データファイルのファイルシステムバックアップを作成するには、MariaDB データディレクトリーの内容をバックアップ場所にコピーします。

現在の設定またはログファイルのバックアップも作成するには、以下の手順の中から任意の手順を選択します。

手順

1. **mariadb** サービスを停止します。

```
# systemctl stop mariadb.service
```

2. データファイルを必要な場所にコピーします。

```
# cp -r /var/lib/mysql /backup-location
```

3. 必要に応じて、設定ファイルを必要な場所にコピーします。

```
# cp -r /etc/my.cnf /etc/my.cnf.d /backup-location/configuration
```

4. 必要に応じて、ログファイルを必要な場所にコピーします。

```
# cp /var/log/mariadb/* /backup-location/logs
```

5. **mariadb** サービスを起動します。

```
# systemctl start mariadb.service
```

6. バックアップされたデータをバックアップ場所から **/var/lib/mysql**ディレクトリーに読み込む際は、**mysql:mysql** が **/var/lib/mysql** 内のすべてのデータの所有者であることを確認してください。

```
# chown -R mysql:mysql /var/lib/mysql
```

7.2.5.5. バックアップソリューションとしてレプリケーションを使用

レプリケーションは、ソースサーバー用の代替バックアップソリューションです。ソースサーバーの複製となるレプリカサーバーを作成すると、ソースに影響を与えずにレプリカでバックアップを実行できます。ソースは、レプリカをシャットダウンする間に依然として実行でき、レプリカからデータのバックアップを作成できます。

警告

レプリケーション自体は、バックアップソリューションとしては十分ではありません。レプリケーションは、ハードウェア障害からソースサーバーを保護しますが、データ損失に対する保護は保証していません。この方法とともに、レプリカでその他のバックアップソリューションを使用することが推奨されます。

関連情報

- Galera で MariaDB を複製する
- バックアップソリューションとしてレプリケーションを使用

7.2.6. MariaDB 10.3 への移行

RHEL 7 には、MySQL データベースファミリーからのサーバーのデフォルトの実装として、**MariaDB 5.5** が同梱されています。新しいバージョンの **MariaDB** データベースサーバーは、RHEL 7 の Software Collections として利用できます。RHEL 8 では、**MariaDB 10.3**、**MariaDB 10.5**、**MariaDB 10.11**、および **MySQL 8.0** が提供されます。

ここでは、**MariaDB** の RHEL 7 または Red Hat Software Collections バージョンから **MariaDB 10.3** への移行を説明します。

RHEL 8 内で **MariaDB 10.3** から **MariaDB 10.5** へ移行する場合は、MariaDB 10.3 から MariaDB 10.5 へのアップグレード を参照してください。

RHEL 8 内で MariaDB 10.3 から MariaDB 10.5 へ移行する場合は、MariaDB 10.3 から MariaDB 10.5 へのアップグレード を参照してください。

7.2.6.1. RHEL 7 と RHEL 8 のバージョンにおける MariaDB の主な相違点

MariaDB 5.5 と **MariaDB 10.3** の間の最も重要な変更点は次のとおりです。

- 10.1 以降、同期マルチソースクラスターの**MariaDB Galera クラスター** は、**MariaDB** の標準部分です。
- ARCHIVE ストレージエンジンはデフォルトで有効ではなくなるため、プラグインを明示的に有効にする必要があります。
- BLACKHOLE ストレージエンジンはデフォルトで有効ではなくなるため、プラグインを明示的に有効にする必要があります。
- InnoDB は、**MariaDB 10.1** 以前のバージョンで使用されていた XtraDB ではなく、デフォルトのストレージエンジンとして使用されます。
  詳細は、Why does MariaDB 10.2 use InnoDB instead of XtraDB? を参照してください。
- 新しい **mariadb-connector-c** パッケージは、MySQL と MariaDB に共通のクライアントライブラリーを提供します。このライブラリーは、データベースサーバー **MySQL** および **MariaDB** のすべてのバージョンで使用できます。その結果、Red Hat Enterprise Linux 8 に同梱される MySQL サーバーまたは **MariaDB** サーバーのいずれかに構築されるアプリケーションの 1 つに接続できます。

MariaDB 5.5 から MariaDB 10.3 に移行するには、複数の 設定変更を 実行する必要があります。

7.2.6.2. 設定変更

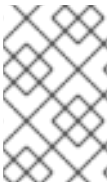
MariaDB 5.5 から MariaDB 10.3 への推奨される移行パスは、最初に MariaDB 10.0 にアップグレードしてから、1 バージョンずつ順次アップグレードすることです。

一度に 1 つのマイナーバージョンをアップグレードする主な利点は、変更に対するデータと設定の両方を含む、データベースの適合に優れている点です。アップグレードは、RHEL 8 (MariaDB 10.3) で利用可能なものと同じメジャーバージョンで終了します。これにより、設定変更やその他の問題が大幅に低下します。

MariaDB 5.5 から MariaDB 10.0 への移行時の設定の変更の詳細は、Red Hat Software Collections の MariaDB 10.0 への移行 を参照してください。

以下の後続バージョンの MariaDB への以降と、必要な設定変更は、以下のドキュメントで説明します。

- Red Hat Software Collections ドキュメントの Migrating to MariaDB 10.1
- Red Hat Software Collections ドキュメントの Migrating to MariaDB 10.2
- Red Hat Software Collections ドキュメントの Migrating to MariaDB 10.3



注記

MariaDB 5.5 から MariaDB 10.3 へ直接移行することもできますが、上記の移行ドキュメントに記載されている違いにより必要とされる設定変更をすべて実行する必要があります。

7.2.6.3. mysql_upgrade ユーティリティーを使用したインプレースアップグレード

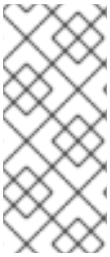
データベースファイルを RHEL 8 に移行するには、RHEL 7 の MariaDB ユーザーは、**mysql_upgrade** ユーティリティーを使用してインプレースアップグレードを実行する必要があります。**mysql_upgrade** ユーティリティーは、**mariadb-server-utils** サブパッケージにより提供され、**mariadb-server** パッケージの依存関係としてインストールされます。

インプレースアップグレードを実行するには、RHEL 8 システムの **/var/lib/mysql/** データディレクトリーにバイナリーデータファイルをコピーして、**mysql_upgrade** ユーティリティーを使用する必要があります。

この方法を使用すると、以下からデータを移行できます。

- Red Hat Enterprise Linux 7 バージョンの **MariaDB 5.5**
- Red Hat Software Collections のバージョンは、以下の通りです。
  - **MariaDB 5.5** (サポート対象外になりました)
  - **MariaDB 10.0** (サポート対象外になりました)
  - **MariaDB 10.1** (サポート対象外になりました)
  - **MariaDB 10.2** (サポート対象外になりました)
  - **MariaDB 10.2** (サポート対象外になりました)

MariaDB 10.3 へのアップグレードは、バージョンごとに連続して行うことが推奨されます。移行は、Release Notes for Red Hat Software Collections で該当する章を参照してください。



注記

RHEL 7 バージョンの MariaDB からアップグレードする場合は、ソースデータは **/var/lib/mysql/** ディレクトリーに保存されます。Red Hat Software Collections バージョンの MariaDB の場合、ソースデータディレクトリーは **/var/opt/rh/<collection_name>/lib/mysql/** です (**/opt/rh/mariadb55/root/var/lib/mysql/** データディレクトリーを使用する **mariadb55** を除く)。

mysql_upgrade ユーティリティーを使用してアップグレード を実行するには、以下の手順を行います。

前提条件

- アップグレードを実行する前に、MariaDB データベースに保存されている全データのバックアップを作成します。

手順

1. RHEL 8 システムに **mariadb-server** パッケージがインストールされていることを確認します。

```
# yum install mariadb-server
```

2. データのコピー時に、**mariadb** サービスがソースおよびターゲットのシステムで稼働していないことを確認します。

```
# systemctl stop mariadb.service
```

3. ソースの場所から RHEL 8 ターゲットシステムの **/var/lib/mysql/** ディレクトリーにデータをコピーします。

4. ターゲットシステムでコピーされたファイルに適切なパーミッションと SELinux コンテキストを設定します。

```
# restorecon -vr /var/lib/mysql
```

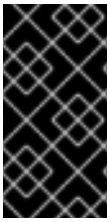
5. ターゲットシステムで、MariaDB サーバーを起動します。

```
# systemctl start mariadb.service
```

6. **mysql_upgrade** コマンドを実行して、内部テーブルをチェックし、修復します。

```
$ mysql_upgrade
```

7. アップグレードが完了したら、**/etc/my.cnf.d/** ディレクトリー内のすべての設定ファイルに MariaDB 10.3 に対して有効なオプションのみが含まれていることを確認します。



重要

インプレースアップグレードには、特定のリスクと既知の問題があります。たとえば、一部のクエリーが動作しなかったり、アップグレード前とは異なる順序で実行される場合があります。これらのリスクと問題、およびインプレースアップグレードに関する全般的な情報は、MariaDB 10.3 Release Notes を参照してください。

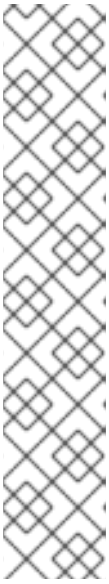
7.2.7. MariaDB 10.3 から MariaDB 10.5 へのアップグレード

ここでは、RHEL 8 における MariaDB 10.3 から MariaDB 10.5 への移行について説明します。

7.2.7.1. MariaDB 10.3 と MariaDB 10.5 の主な相違点

MariaDB 10.3 と MariaDB 10.5 の間の重要な変更点には以下が含まれます。

- MariaDB がデフォルトで **unix_socket** 認証プラグインを使用するようになりました。このプラグインにより、ユーザーがローカルの UNIX ソケットファイルを介して MariaDB に接続するときにオペレーティングシステムの認証情報を使用できるようになります。
- **MariaDB** に、**mariadb-*** という名前のバイナリーと、**mariadb-*** バイナリーを参照する **mysql*** シンボリックリンクが追加されました。たとえば、**mysqladmin**、**mysqlaccess**、および **mysqlshow** のシンボリックリンクは、**mariadb-admin**、**mariadb-access**、および **mariadb-show** のバイナリーをそれぞれポイントします。
- 各ユーザーロールに合わせて、**SUPER** 特権が複数の特権に分割されました。その結果、一部のステートメントで必要な特権が変更されました。
- 並列のレプリケーションでは、**slave_parallel_mode** はデフォルトで **optimistic** に設定されるようになりました。
- InnoDB ストレージエンジンで、変数のデフォルトが変更されました (**innodb_adaptive_hash_index** は **OFF** へ、**innodb_checksum_algorithm** は **full_crc32** へ変更)。
- MariaDB は、以前使用された **readline** ライブラリーではなく、MariaDB コマンド履歴 (**.mysql_history** ファイル) を管理する基盤となるソフトウェアの **libedit** 実装を使用するようになりました。この変更は、**.mysql_history** ファイルを直接使用しているユーザーに影響します。**.mysql_history** は MariaDB または MySQL アプリケーションによって管理されるファイルであるため、ユーザーは直接ファイルでは機能しないことに注意してください。人間が判読可能な外観は偶然です。



注記

セキュリティーを強化するために、履歴ファイルの維持を考慮することができます。コマンド履歴の記録を無効にするには、以下を実行します。

1. 存在する場合は、**.mysql_history** ファイルを削除します。
2. 以下のいずれかの方法を使用します。
   - **MYSQL_HISTFILE** 変数を **/dev/null** に設定し、これをシェルの起動ファイルに追加します。
   - **.mysql_history** ファイルを **/dev/null** へのシンボリックリンクに変更します。

```
$ ln -s /dev/null $HOME/.mysql_history
```

MariaDB Galera **クラスター** がバージョン 4 にアップグレードされ、以下の主な変更点が加えられました。

- Galera は、サイズ制限なしのトランザクションの複製をサポートする、新しいストリーミングレプリケーション機能を追加します。ストリーミングレプリケーションの実行時に、クラスターは小さなフラグメントでトランザクションを複製します。
- Galera が Global Transaction ID (GTID) に完全に対応するようになりました。
- **/etc/my.cnf.d/galera.cnf** ファイルの **wsrep_on** オプションのデフォルト値が **1** から **0** に変更され、エンドユーザーが必要な追加オプションを設定せずに **wsrep** レプリケーションを開始できないようにします。

MariaDB 10.5 の PAM プラグインが次のように変更されました。

- MariaDB 10.5 で、Pluggable Authentication Modules (PAM) プラグインの新しいバージョンが追加されました。PAM プラグインバージョン 2.0 は、個別の **setuid root** ヘルパーバイナリーを使用して PAM 認証を実行します。これにより、MariaDB が追加の PAM モジュールを使用できるようになります。
- ヘルパーバイナリーは、**mysql** グループのユーザーによってのみ実行できます。デフォルトでは、グループには **mysql** ユーザーのみが含まれます。Red Hat では、このヘルパーユーティリティーを介してスロットルまたはログの記録をせずにパスワード推測攻撃を防ぐために、管理者が **mysql** グループにユーザーをさらに追加しないことを推奨しています。
- MariaDB 10.5 で、Pluggable Authentication Modules (PAM) プラグインとその関連ファイルが新しいパッケージ **mariadb-pam** に移動しました。したがって、**MariaDB** に PAM 認証を使用しないシステムに、新しい **setuid root** バイナリーが導入されることはありません。
- **mariadb-pam** パッケージには、両方の PAM プラグインバージョンが含まれています。バージョン 2.0 はデフォルトで、バージョン 1.0 は **auth_pam_v1** 共有オブジェクトライブラリーとして利用できます。
- MariaDB サーバーでは、**mariadb-pam** パッケージはデフォルトでインストールされません。MariaDB 10.5 で PAM 認証プラグインを利用できるようにするには、**mariadb-pam** パッケージを手動でインストールします。

7.2.7.2. RHEL 8 バージョンの MariaDB 10.3 から MariaDB 10.5 へのアップグレード

この手順では、**yum** および **mariadb-upgrade** ユーティリティーを使用して、**mariadb:10.3** モジュールストリームから **mariadb:10.5** モジュールストリームにアップグレードする方法を説明します。

mariadb-upgrade ユーティリティーは、**mariadb-server-utils** サブパッケージにより提供され、**mariadb-server** パッケージの依存関係としてインストールされます。

前提条件

- アップグレードを実行する前に、MariaDB データベースに保存されている全データのバックアップを作成します。

手順

1. MariaDB サーバーを停止します。

```
# systemctl stop mariadb.service
```

2. 以下のコマンドを実行して、後続のストリームに切り替えるためのシステムの準備が整っているかどうかを判断します。

```
# yum distro-sync
```

このコマンドは、Nothing to do.Complete! のメッセージで終了する必要があります。詳細は、後続のストリームへの切り替え を参照してください。

3. システムで **mariadb** モジュールをリセットします。

```
# yum module reset mariadb
```

4. 新しい **mariadb:10.5** モジュールストリームを有効にします。

```
# yum module enable mariadb:10.5
```

5. インストール済みパッケージを同期し、ストリーム間の変更を実行します。

```
# yum distro-sync
```

これにより、インストールされている MariaDB パッケージがすべて更新されます。

6. **/etc/my.cnf.d/** にあるオプションファイルに MariaDB 10.5 に対して有効なオプションのみが含まれるように、設定を調整します。詳細は、MariaDB 10.4 および MariaDB 10.5 のアップストリームドキュメントを参照してください。

7. MariaDB サーバーを起動します。

- スタンドアロンを実行しているデータベースをアップグレードする場合:

```
# systemctl start mariadb.service
```

- Galera クラスターノードをアップグレードする場合:

```
# galera_new_cluster
```

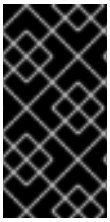
mariadb サービスが自動的に起動します。

8. **mariadb-upgrade** ユーティリティーを実行して、内部テーブルをチェックし、修復します。
   - スタンドアロンを実行しているデータベースをアップグレードする場合:

     ```
     # mariadb-upgrade
     ```

   - Galera クラスターノードをアップグレードする場合:

     ```
     # mariadb-upgrade --skip-write-binlog
     ```



重要

インプレースアップグレードには、特定のリスクと既知の問題があります。たとえば、一部のクエリーが動作しなかったり、アップグレード前とは異なる順序で実行される場合があります。これらのリスクと問題、およびインプレースアップグレードに関する全般的な情報は、MariaDB 10.5 Release Notes を参照してください。

7.2.8. MariaDB 10.5 から MariaDB 10.11 へのアップグレード

ここでは、RHEL 9 における MariaDB 10.5 から MariaDB 10.11 への移行について説明します。

7.2.8.1. MariaDB 10.5 と MariaDB 10.11 の主な相違点

MariaDB 10.5 と MariaDB 10.11 の間の重要な変更点は次のとおりです。

- 新しい **sys_schema** 機能。これは、データベースの使用状況に関する情報を提供するビュー、関数、およびプロシージャーのコレクションです。
- **CREATE TABLE**、**ALTER TABLE**、**RENAME TABLE**、**DROP TABLE**、**DROP DATABASE**、および関連するデータ定義言語 (DDL) ステートメントがアトミックになりました。ステートメントは完全に完結している必要があります。そうでない場合、変更が元に戻されます。**DROP TABLE** を使用して複数のテーブルを削除する場合、テーブルの全リストではなく、個々のドロップのみがアトミックであることに注意してください。
- 新しい **GRANT ... TO PUBLIC** 権限が利用可能になりました。
- **SUPER** 権限と **READ ONLY ADMIN** 権限が分離されました。
- 新しい **UUID** データベースデータ型に、ユニバーサルー意識別子を格納できるようになりました。
- MariaDB が Secure Socket Layer (SSL) プロトコルバージョン 3 をサポートするようになりました。
- MariaDB サーバーの起動に正しく設定された SSL が必要になりました。以前は、SSL の設定が誤っている場合、MariaDB は SSL を暗黙的に無効にし、セキュアでない接続を使用していました。
- MariaDB が **natural_sort_key()** 関数により自然なソート順序をサポートするようになりました。
- 新しい **SFORMAT** 関数を任意のテキスト書式設定に使用できるようになりました。
- **utf8** 文字セット (および関連する照合順序) が、デフォルトで **utf8mb3** のエイリアスになりました。

- MariaDB は、Unicode Collation Algorithm (UCA) 14 の照合順序をサポートしています。
- MariaDB の **systemd** ソケットのアクティベーションファイルが **/usr/share/** ディレクトリーで利用できるようになりました。アップストリームとは異なり、これらのファイルは RHEL のデフォルト設定の一部ではないことに注意してください。
- エラーメッセージに、**MySQL** ではなく **MariaDB** 文字列が含まれるようになりました。
- エラーメッセージが中国語で利用できるようになりました。
- デフォルトの logrotate ファイルが大幅に変更されました。MariaDB 10.11 に移行する前に設定を確認してください。
- MariaDB および MySQL クライアントの場合、コマンドラインで指定した接続プロパティー (例: **--port=3306**) によって、クライアントとサーバー間の通信のプロトコルタイプ (**tcp**、**socket**、**pipe**、**memory** など) が強制されるようになりました。たとえば、以前は MariaDB クライアントが UNIX ソケットを介して接続した場合、指定したポートが無視されていました。

7.2.8.2. RHEL 9 バージョンの MariaDB 10.5 から MariaDB 10.11 へのアップグレード

この手順では、**yum** および **mariadb-upgrade** ユーティリティーを使用して、**mariadb:10.3** モジュールストリームから **mariadb:10.5** モジュールストリームにアップグレードする方法を説明します。

mariadb-upgrade ユーティリティーは、**mariadb-server-utils** サブパッケージにより提供され、**mariadb-server** パッケージの依存関係としてインストールされます。

前提条件

- アップグレードを実行する前に、MariaDB データベースに保存されている全データのバックアップを作成します。

手順

1. MariaDB サーバーを停止します。

```
# systemctl stop mariadb.service
```

2. 以下のコマンドを実行して、後続のストリームに切り替えるためのシステムの準備が整っているかどうかを判断します。

```
# yum distro-sync
```

このコマンドは、Nothing to do.Complete!のメッセージで終了する必要があります。詳細は、後続のストリームへの切り替え を参照してください。

3. システムで **mariadb** モジュールをリセットします。

```
# yum module reset mariadb
```

4. 新しい **mariadb:10.5** モジュールストリームを有効にします。

```
# yum module enable mariadb:10.11
```

5. インストール済みパッケージを同期し、ストリーム間の変更を実行します。

```
# yum distro-sync
```

これにより、インストールされている MariaDB パッケージがすべて更新されます。

6. **/etc/my.cnf.d/** にあるオプションファイルに MariaDB 10.11 に対して有効なオプションのみが含まれるように、設定を調整します。詳細は、MariaDB 10.6 および MariaDB 10.11 のアップストリームドキュメントを参照してください。

7. MariaDB サーバーを起動します。

- スタンドアロンを実行しているデータベースをアップグレードする場合:

```
# systemctl start mariadb.service
```

- Galera クラスターノードをアップグレードする場合:

```
# galera_new_cluster
```

mariadb サービスが自動的に起動します。

8. mariadb-upgrade ユーティリティーを実行して、内部テーブルをチェックし、修復します。

- スタンドアロンを実行しているデータベースをアップグレードする場合:

```
# mariadb-upgrade
```

- Galera クラスターノードをアップグレードする場合:

```
# mariadb-upgrade --skip-write-binlog
```

重要

インプレースアップグレードには、特定のリスクと既知の問題があります。たとえば、一部のクエリーが動作しなかったり、アップグレード前とは異なる順序で実行される場合があります。これらのリスクと問題、およびインプレースアップグレードに関する全般的な情報は、MariaDB 10.11 Release Notes を参照してください。

7.2.9. Galera で MariaDB を複製する

本セクションでは、Red Hat Enterprise Linux 8 の Galera ソリューションを使用して、MariaDB データベースを複製する方法を説明します。

7.2.9.1. MariaDB Galera クラスターの概要

Galera レプリケーションは、複数の MariaDB サーバーで設定される同期マルチソース **MariaDB Galera クラスター** の作成に基づいています。レプリカが通常読み取り専用である従来のプライマリー/レプリカ設定とは異なり、MariaDB Galera クラスターのノードはすべて書き込み可能にすることができます。

Galera レプリケーションと MariaDB データベースとの間のインターフェイスは、書き込みセットレプリケーション API (**wsrep API**) で定義されます。

MariaDB Galera クラスター の主な機能は以下のとおりです。

- 同期のレプリケーション
- アクティブ/アクティブのマルチソーストポロジー
- クラスターノードへの読み取りおよび書き込み
- 自動メンバーシップ制御、失敗したノードのクラスターからの削除
- 自動ノードの参加
- 行レベルの並列レプリケーション
- ダイレクトクライアント接続: ユーザーはクラスターノードにログインし、レプリケーションの実行中にノードを直接操作できます。

同期レプリケーションとは、サーバーがトランザクションに関連付けられた書き込みセットをクラスター内のすべてのノードにブロードキャストすることで、コミット時にトランザクションをレプリケートすることを意味します。クライアント (ユーザーアプリケーション) はデータベース管理システム (DBMS) に直接接続し、ネイティブの **MariaDB** と同様の動作が発生します。

同期レプリケーションは、クラスター内の 1 つのノードで発生した変更が、クラスター内の他のノードで同時に発生することを保証します。

そのため、同期レプリケーションには、非同期のレプリケーションと比べて次のような利点があります。

- 特定のクラスターノード間の変更の伝播に遅延がない
- すべてのクラスターノードには常に一貫性がある
- いずれかのクラスターノードがクラッシュしても、最新の変更は失われない
- すべてのクラスターノードのトランザクションが並列に実行する
- クラスター全体にわたる因果関係

関連情報

- About Galera replication
- What is MariaDB Galera Cluster
- Getting started with MariaDB Galera Cluster

7.2.9.2. MariaDB Galera クラスターを構築するためのコンポーネント

MariaDB Galera クラスター を構築するには、システムに以下のパッケージをインストールする必要があります。

- **mariadb-server-galera**: **MariaDB Galera クラスター** のサポートファイルとスクリプトが含まれます。
- **mariadb-server**: MariaDB アップストリームがパッチを適用し、書き込みセットレプリケーション API (**wsrep API**) を組み込みます。この API は、**Galera** レプリケーションと **MariaDB** との間のインターフェイスを提供します。

- **galera**: MariaDB アップストリームがパッチを適用し、MariaDB の完全サポートを追加します。**galera** パッケージには、以下の内容が含まれます。
  - Galera Replication Library は、レプリケーション機能全体を提供します。
  - Galera Arbitrator ユーティリティーは、スプリットブレインのシナリオで投票に参加するクラスターメンバーとして使用できます。ただし、Galera Arbitrator は実際のレプリケーションには参加できません。
  - Galera Arbitrator ユーティリティーのデプロイに使用される **Galera Systemd service** および **Galera wrapper script**。RHEL 8 の MariaDB 10.3 および MariaDB 10.5 には、Red Hat バージョンの **garbd** systemd サービスと、**/usr/lib/systemd/system/garbd.service** ファイルおよび **/usr/sbin/garbd-wrapper** ファイルにそれぞれある galera パッケージのラッパースクリプトが含まれています。RHEL 8.6 以降、RHEL とともに配布される **MariaDB は、/usr/share/doc/galera/garb-systemd** および **/usr/share/doc/galera/garbd.service** にあるこれらのファイルのアップストリームバージョンも提供します。

関連情報

- Galera Replication Library
- Galera Arbitrator
- mysql-wsrep プロジェクト

7.2.9.3. MariaDB Galera クラスターのデプロイメント

前提条件

- **mariadb** モジュールからストリーム (バージョン) を選択し、**galera** プロファイルを指定して、MariaDB Galera **クラスター** パッケージをインストールします。以下に例を示します。

```
# yum module install mariadb:10.3/galera
```

  これにより、以下のパッケージがインストールされます。
  - **mariadb-server-galera**
  - **mariadb-server**
  - **galera**
    mariadb-server-galera パッケージが **mariadb-server** パッケージおよび **galera** パッケージを依存関係としてプルします。

    MariaDB Galera Cluster をビルドするためにインストールする必要があるパッケージについては、MariaDB クラスターをビルドするためのコンポーネント を参照してください。
- MariaDB サーバーのレプリケーション設定は、システムを初めてクラスターに追加する前に更新する必要があります。
  デフォルト設定は、**/etc/my.cnf.d/galera.cnf** ファイルで配布されます。

  MariaDB Galera **クラスター** をデプロイする前に、以下の文字列で開始するように、すべてのノードの **/etc/my.cnf.d/galera.cnf** ファイルに **wsrep_cluster_address** オプションを設定します。

```
gcomm://
```

- 初期ノードでは、**wsrep_cluster_address** を空のリストとして設定できます。

```
wsrep_cluster_address="gcomm://"
```

- その他のすべてのノードに **wsrep_cluster_address** を設定して、実行中のクラスターに属するノードへのアドレスを追加します。以下に例を示します。

```
wsrep_cluster_address="gcomm://10.0.0.10"
```

Galera Cluster アドレスの設定方法は、Galera Cluster Address を参照してください。

手順

1. ノードで以下のラッパーを実行して、新規クラスターの最初のノードをブートストラップします。

```
# galera_new_cluster
```

このラッパーにより、MariaDB サーバーデーモン (**mysqld**) に **--wsrep-new-cluster** オプションが指定されて実行されるようになります。このオプションは、接続する既存クラスターがないという情報を提供します。したがって、ノードは新規 UUID を作成し、新しいクラスターを特定します。

注記

mariadb サービスは、複数の MariaDB サーバープロセスと対話する systemd メソッドをサポートします。したがって、複数の MariaDB サーバーを実行している場合は、インスタンス名を接尾辞として指定して、特定のインスタンスをブートストラップできます。

```
# galera_new_cluster mariadb@node1
```

2. 各ノードで次のコマンドを実行して、その他のノードをクラスターに接続します。

```
# systemctl start mariadb
```

その結果、ノードはクラスターに接続し、それ自体をクラスターの状態と同期します。

関連情報

- Getting started with MariaDB Galera Cluster .

7.2.9.4. 新規ノードの MariaDB Galera クラスターへの追加

新規ノードを MariaDB Galera クラスター に追加するには、以下の手順に従います。

この手順に従って、既存のノードを再接続することもできます。

手順

- 特定のノードで、**/etc/my.cnf.d/galera.cnf** 設定ファイルの **[mariadb]** セクション内にある **wsrep_cluster_address** オプションで、1 つ以上の既存クラスターメンバーにアドレスを指定します。

```
[mariadb]
wsrep_cluster_address="gcomm://192.168.0.1"
```

新規ノードを既存クラスターノードのいずれかに接続すると、クラスター内のすべてのノードを表示できるようになります。

ただし、**wsrep_cluster_address** のクラスターの全ノードを表示することが推奨されます。

したがって、1 つ以上のクラスターノードがダウンしても、その他のクラスターノードに接続することでノードがクラスターに参加できます。すべてのメンバーがメンバーシップに同意すると、クラスターの状態が変更します。新規ノードの状態がクラスターの状態と異なる場合、新しいノードは Incremental State Transfer (IST) または State Snapshot Transfer (SST) のいずれかを要求し、他のノードとの一貫性を確保します。

関連情報

- Getting started with MariaDB Galera Cluster
- Introduction to State Snapshot Transfers

7.2.9.5. MariaDB Galera クラスターの再起動

すべてのノードを同時にシャットダウンすると、クラスターが終了し、実行中のクラスターは存在しなくなります。ただし、クラスターのデータは引き続き存在します。

クラスターを再起動するには、MariaDB Galera クラスターの設定 の説明に従って、最初のノードをブートストラップします。

警告

クラスターがブートストラップされず、最初のノードの **mysqld** が **systemctl start mariadb** コマンドでのみ起動した場合、ノードは **/etc/my.cnf.d/galera.cnf** ファイルの **wsrep_cluster_address** オプションに記載されている少なくとも 1 つのノードに接続しようとします。ノードが現在実行していない場合は、再起動に失敗します。

関連情報

- Getting started with MariaDB Galera Cluster .

7.2.10. MariaDB クライアントアプリケーションの開発

Red Hat では、**MariaDB** クライアントライブラリーに対して **MariaDB** クライアントアプリケーションを開発することを推奨します。

MariaDB クライアントライブラリーに対してアプリケーションをビルドするために必要な開発ファイルとプログラムは、**mariadb-connector-c-devel** パッケージで提供されます。

直接ライブラリー名を使用する代わりに、**mariadb-connector-c-devel** パッケージで配布されている **mariadb_config** プログラムを使用します。このプログラムにより、正しいビルドフラグが確実に返されるようになります。

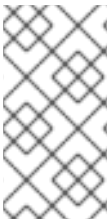
7.3. MYSQL の使用

MySQL サーバーは、オープンソースの高速で堅牢なデータベースサーバーです。**MySQL** は、データを構造化情報に変換して、データにアクセスする SQL インターフェイスを提供するリレーショナルデータベースです。これには、複数のストレージエンジンとプラグインに加え、地理情報システム (GIS) と JavaScript Object Notation (JSON) 機能も含まれています。

RHEL システムに **MySQL** をインストールして設定する方法、**MySQL** データをバックアップする方法、**MySQL** の以前のバージョンから移行する方法、および **MySQL** を複製する方法について説明します。

7.3.1. MySQL のインストール

RHEL 8 では、**MySQL 8.0** サーバーは **mysql:8.0** モジュールストリームとして利用できます。



注記

RPM パッケージが競合しているため、RHEL 8 では **MySQL** および **MariaDB** データベースサーバーを同時にインストールすることはできません。コンテナー内では、**MySQL** および **MariaDB** データベースサーバーを並行して使用できます。コンテナー内で複数の MySQL および MariaDB バージョンを実行する を参照してください。

MySQL をインストールするには、以下の手順に従います。

手順

1. **mysql** モジュールから **8.0** ストリーム (バージョン) を選択し、**server** プロファイルを指定して、**MySQL** サーバーパッケージをインストールします。

```
# yum module install mysql:8.0/server
```

2. **mysqld** サービスを開始します。

```
# systemctl start mysqld.service
```

3. **mysqld** サービスを有効にして、起動時に起動するようにします。

```
# systemctl enable mysqld.service
```

4. **推奨手順: MySQL** のインストール時にセキュリティーを強化するには、次のコマンドを実行します。

```
$ mysql_secure_installation
```

このコマンドは、完全にインタラクティブなスクリプトを起動して、プロセスの各ステップのプロンプトを表示します。このスクリプトを使用すると、次の方法でセキュリティーを改善できます。

- root アカウントのパスワードの設定
- 匿名ユーザーの削除
- リモート root ログインの拒否 (ローカルホスト外)

7.3.1.1. コンテナー内で複数の MySQL および MariaDB バージョンを実行する

MySQL と MariaDB の両方を同じホストで実行するには、コンテナー内で実行します。これは、RPM パッケージが競合し、これらのデータベースサーバーを並行してインストールできないためです。

この手順では、例として MySQL 8.0 と MariaDB 10.5 を記載していますが、Red Hat Ecosystem Catalog で利用可能な任意の MySQL または MariaDB コンテナーバージョンを使用できます。

前提条件

- **container-tools** モジュールがインストールされている。

手順

1. Red Hat カスタマーポータルアカウントを使用して、**registry.redhat.io** レジストリーに認証します。

```
# podman login registry.redhat.io
```

すでにコンテナーレジストリーにログインしている場合は、このステップをスキップしてください。

2. コンテナー内で MySQL 8.0 を実行します。

```
$ podman run -d --name <container_name> -e
MYSQL_ROOT_PASSWORD=<mysql_root_password> -p <host_port_1>:3306
rhel8/mysql-80
```

このコンテナーイメージを使用する方法の詳細は、Red Hat Ecosystem Catalog を参照してください。

3. コンテナー内で MariaDB 10.5 を実行します。

```
$ podman run -d --name <container_name> -e
MYSQL_ROOT_PASSWORD=<mariadb_root_password> -p <host_port_2>:3306
rhel8/mariadb-105
```

このコンテナーイメージを使用する方法の詳細は、Red Hat Ecosystem Catalog を参照してください。

4. コンテナー内で MariaDB 10.11 を実行します。

```
$ podman run -d --name <container_name> -e
MYSQL_ROOT_PASSWORD=<mariadb_root_password> -p <host_port_3>:3306
rhel8/mariadb-1011
```

このコンテナーイメージを使用する方法の詳細は、Red Hat Ecosystem Catalog を参照してください。

注記

2 つのデータベースサーバーのコンテナー名とホストポートが異なっている必要があります。

5. クライアントがネットワーク上のデータベースサーバーにアクセスできるように、ファイアウォールでホストポートを開きます。

```
# firewall-cmd --permanent --add-port={<host_port>/tcp,<host_port>/tcp,...}
# firewall-cmd --reload
```

検証手順

1. 実行中のコンテナーに関する情報を表示します。

```
$ podman ps
```

2. データベースサーバーに接続し、root としてログインします。

```
# mysql -u root -p -h localhost -P <host_port> --protocol tcp
```

関連情報

- コンテナーの構築、実行、および管理
- Red Hat Ecosystem Catalog でコンテナーを参照する

7.3.2. MySQL の設定

MySQL サーバーをネットワーク用に設定するには、以下の手順に従います。

手順

1. **/etc/my.cnf.d/mysql-server.cnf** ファイルの **[mysqld]** セクションを編集します。以下の設定ディレクティブを設定できます。
   - **bind-address**: サーバーがリッスンするアドレスです。設定可能なオプションは以下のとおりです。
     - ホスト名
     - IPv4 アドレス
     - IPv6 アドレス
   - **skip-networking**: サーバーが TCP/IP 接続をリッスンするかどうかを制御します。以下の値が使用できます。
     - 0 - すべてのクライアントをリッスンする
     - 1 - ローカルクライアントのみをリッスンする

- **port**: MySQL が TCP/IP 接続をリッスンするポート。

2. **mysqld** サービスを再起動します。

```
# systemctl restart mysqld.service
```

7.3.3. MySQL サーバーでの TLS 暗号化の設定

デフォルトでは、MySQL は暗号化されていない接続を使用します。安全な接続のために、MySQL サーバーで TLS サポートを有効にし、暗号化された接続を確立するようにクライアントを設定します。

7.3.3.1. MySQL サーバーに CA 証明書、サーバー証明書、および秘密鍵を配置する

MySQL サーバーで TLS 暗号化を有効にする前に、認証局 (CA) 証明書、サーバー証明書、および秘密鍵を MySQL サーバーに保存します。

前提条件

- Privacy Enhanced Mail(PEM) 形式の以下のファイルがサーバーにコピーされています。
  - サーバーの秘密鍵: **server.example.com.key.pem**
  - サーバー証明書: **server.example.com.crt.pem**
  - 認証局 (CA) 証明書: **ca.crt.pem**

  秘密鍵および証明書署名要求 (CSR) の作成や、CA からの証明書要求に関する詳細は、CA のドキュメントを参照してください。

手順

1. CA およびサーバー証明書を **/etc/pki/tls/certs/** ディレクトリーに保存します。

```
# mv <path>/server.example.com.crt.pem /etc/pki/tls/certs/
# mv <path>/ca.crt.pem /etc/pki/tls/certs/
```

2. MySQL サーバーがファイルを読み込めるように、CA およびサーバー証明書にパーミッションを設定します。

```
# chmod 644 /etc/pki/tls/certs/server.example.com.crt.pem /etc/pki/tls/certs/ca.crt.pem
```

   証明書は、セキュアな接続が確立される前は通信の一部であるため、任意のクライアントは認証なしで証明書を取得できます。そのため、CA およびサーバーの証明書ファイルに厳密なパーミッションを設定する必要はありません。

3. サーバーの秘密鍵を **/etc/pki/tls/private/** ディレクトリーに保存します。

```
# mv <path>/server.example.com.key.pem /etc/pki/tls/private/
```

4. サーバーの秘密鍵にセキュアなパーミッションを設定します。

```
# chmod 640 /etc/pki/tls/private/server.example.com.key.pem
# chgrp mysql /etc/pki/tls/private/server.example.com.key.pem
```

承認されていないユーザーが秘密鍵にアクセスできる場合は、MySQL サーバーへの接続は安全ではなくなります。

5. SELinux コンテキストを復元します。

```
# restorecon -Rv /etc/pki/tls/
```

7.3.3.2. MySQL サーバーでの TLS の設定

セキュリティーを強化するには、MySQL サーバーで TLS サポートを有効にします。その結果、クライアントは TLS 暗号化を使用してサーバーでデータを送信できます。

前提条件

- MySQL サーバーをインストールしている。
- **mysqld** サービスが実行されている。
- Privacy Enhanced Mail(PEM) 形式の以下のファイルがサーバー上にあり、**mysql** ユーザーが読み取りできます。
  - サーバーの秘密鍵: **/etc/pki/tls/private/server.example.com.key.pem**
  - サーバー証明書: **/etc/pki/tls/certs/server.example.com.crt.pem**
  - 認証局 (CA) 証明書 **/etc/pki/tls/certs/ca.crt.pem**
- サーバー証明書のサブジェクト識別名 (DN) またはサブジェクトの別名 (SAN) フィールドは、サーバーのホスト名と一致します。

手順

1. **/etc/my.cnf.d/mysql-server-tls.cnf** ファイルを作成します。
   a. 以下の内容を追加して、秘密鍵、サーバー、および CA 証明書へのパスを設定します。

```
[mysqld]
ssl_key = /etc/pki/tls/private/server.example.com.key.pem
ssl_cert = /etc/pki/tls/certs/server.example.com.crt.pem
ssl_ca = /etc/pki/tls/certs/ca.crt.pem
```

   b. 証明書失効リスト (CRL) がある場合は、それを使用するように MySQL サーバーを設定します。

```
ssl_crl = /etc/pki/tls/certs/example.crl.pem
```

   c. オプション: 暗号化なしの接続試行を拒否します。この機能を有効にするには、以下を追加します。

```
require_secure_transport = on
```

   d. オプション: サーバーがサポートする必要がある TLS バージョンを設定します。たとえば、TLS 1.2 および TLS 1.3 をサポートするには、以下を追加します。

```
tls_version = TLSv1.2,TLSv1.3
```

デフォルトでは、サーバーは TLS 1.1、TLS 1.2、および TLS 1.3 をサポートします。

2. **mysqld** サービスを再起動します。

```
# systemctl restart mysqld
```

検証

トラブルシューティングを簡素化するには、ローカルクライアントが TLS 暗号化を使用するように設定する前に、**MySQL** サーバーで以下の手順を実行します。

1. **MySQL** で TLS 暗号化が有効になっていることを確認します。

```
# mysql -u root -p -h <MySQL_server_hostname> -e "SHOW session status LIKE 'Ssl_cipher';"
+---------------+-----------------------+
| Variable_name | Value                 |
+---------------+-----------------------+
| Ssl_cipher    | TLS_AES_256_GCM_SHA384 |
+---------------+-----------------------+
```

2. **MySQL** サーバーが特定の TLS バージョンのみをサポートするように設定している場合は、**tls_version** 変数を表示します。

```
# mysql -u root -p -e "SHOW GLOBAL VARIABLES LIKE 'tls_version';"
+---------------+-----------------+
| Variable_name | Value           |
+---------------+-----------------+
| tls_version   | TLSv1.2,TLSv1.3 |
+---------------+-----------------+
```

3. サーバーが正しい CA 証明書、サーバー証明書、および秘密鍵ファイルを使用していることを確認します。

```
# mysql -u root -e "SHOW GLOBAL VARIABLES WHERE Variable_name REGEXP '^ssl_ca|^ssl_cert|^ssl_key';"
+----------------+-----------------------------------------------+
| Variable_name  | Value                                         |
+----------------+-----------------------------------------------+
| ssl_ca         | /etc/pki/tls/certs/ca.crt.pem                 |
| ssl_capath     |                                               |
| ssl_cert       | /etc/pki/tls/certs/server.example.com.crt.pem |
| ssl_key        | /etc/pki/tls/private/server.example.com.key.pem |
+----------------+-----------------------------------------------+
```

関連情報

- MySQL サーバーに CA 証明書、サーバー証明書、および秘密鍵を配置する

7.3.3.3. 特定のユーザーアカウントに TLS で暗号化された接続を要求する

機密データにアクセスできるユーザーは、ネットワーク上で暗号化されていないデータ送信を回避するために、常に TLS で暗号化された接続を使用する必要があります。

すべての接続にセキュアなトランスポートが必要なサーバーで設定できない場合は (**require_secure_transport = on**)、TLS 暗号化を必要とするように個別のユーザーアカウントを設定します。

前提条件

- MySQL サーバーで TLS サポートが有効になっている。
- セキュアなトランスポートを必要とするように設定するユーザーが存在する。
- CA 証明書がクライアントに保存されている。

手順

1. 管理ユーザーとして MySQL サーバーに接続します。

```
# mysql -u root -p -h server.example.com
```

管理ユーザーがリモートでサーバーにアクセスする権限を持たない場合は、MySQL サーバーでコマンドを実行し、**localhost** に接続します。

2. **REQUIRE SSL** 句を使用して、ユーザーが TLS 暗号化接続を使用して接続する必要があるよう強制します。

```
MySQL [(none)]> ALTER USER 'example'@'%' REQUIRE SSL;
```

検証

1. TLS 暗号化を使用して、**example** ユーザーとしてサーバーに接続します。

```
# mysql -u example -p -h server.example.com
...
MySQL [(none)]>
```

エラーが表示されず、インタラクティブな MySQL コンソールにアクセスできる場合は、TLS との接続は成功します。

デフォルトでは、サーバーが TLS 暗号化を提供している場合、クライアントは自動的にその TLS 暗号化を使用します。したがって、**--ssl-ca=ca.crt.pem** および **--ssl-mode=VERIFY_IDENTITY** オプションは必須ではありません。ただし、これらのオプションを使用するとクライアントはサーバーの ID を検証するため、セキュリティーが向上します。

2. TLS を無効にして、**example** ユーザーとして接続を試みます。

```
# mysql -u example -p -h server.example.com --ssl-mode=DISABLED
ERROR 1045 (28000): Access denied for user 'example'@'server.example.com' (using password: YES)
```

このユーザーには TLS が必要なのにもかかわらず無効になっているため、サーバーはログインの試行を拒否しました (**--ssl-mode=DISABLED**)。

関連情報

- MySQL サーバーでの TLS の設定

7.3.4. MySQL クライアントで CA 証明書の検証を使用して TLS 暗号化をグローバルで有効にする

MySQL サーバーが TLS 暗号化に対応している場合は、安全な接続のみを確立し、サーバー証明書を検証するようにクライアントを設定します。この手順では、サーバー上のすべてのユーザーで TLS サポートを有効にする方法を説明します。

7.3.4.1. デフォルトで TLS 暗号化を使用するように MySQL クライアントを設定する

RHEL では、MySQL クライアントが TLS 暗号化を使用するようにグローバルに設定でき、サーバー証明書の Common Name (CN) が、ユーザーが接続するホスト名と一致することを検証します。これにより、man-in-the-middle 攻撃 (中間者攻撃) を防ぎます。

前提条件

- MySQL サーバーで TLS サポートが有効になっている。
- CA 証明書は、クライアントの **/etc/pki/tls/certs/ca.crt.pem** ファイルに保存されます。

手順

- 以下の内容で **/etc/my.cnf.d/mysql-client-tls.cnf** ファイルを作成します。

```
[client]
ssl-mode=VERIFY_IDENTITY
ssl-ca=/etc/pki/tls/certs/ca.crt.pem
```

これらの設定は、MySQL クライアントが TLS 暗号化を使用すること、およびクライアントがホスト名をサーバー証明書の CN と比較すること (**ssl-mode=VERIFY_IDENTITY**) を定義します。さらに、CA 証明書 (**ssl-ca**) へのパスも指定します。

検証

- ホスト名を使用してサーバーに接続し、サーバーの状態を表示します。

```
# mysql -u root -p -h server.example.com -e status
...
SSL:        Cipher in use is TLS_AES_256_GCM_SHA384
```

SSL エントリーに **Cipher in use is...** が含まれている場合、接続は暗号化されています。

このコマンドで使用するユーザーには、リモートで認証するパーミッションがあることに注意してください。

接続するホスト名がサーバーの TLS 証明書のホスト名と一致しない場合、**ssl-mode=VERIFY_IDENTITY** パラメーターにより接続が失敗します。たとえば、**localhost** に接続する場合は、以下のようになります。

```
# mysql -u root -p -h localhost -e status
ERROR 2026 (HY000): SSL connection error: error:0A000086:SSL routines::certificate verify failed
```

関連情報

- **mysql(1)** man ページの **--ssl*** パラメーターの説明。

7.3.5. MySQL データのバックアップ

Red Hat Enterprise Linux 8 で**MySQL** データベースからデータをバックアップする主な方法は 2 つあります。

- 論理バックアップ
- 物理バックアップ

論理バックアップ は、データの復元に必要な SQL ステートメントで設定されます。この種類のバックアップは、情報およびレコードをプレーンテキストファイルにエクスポートします。

物理バックアップに対する論理バックアップの主な利点は、移植性と柔軟性です。データは、物理バックアップではできない他のハードウェア設定である **MySQL** バージョンまたはデータベース管理システム (DBMS) で復元できます。

mysqld.service が実行されている場合は、論理バックアップを実行できることに注意してください。論理バックアップには、ログと設定ファイルが含まれません。

物理バックアップ は、コンテンツを格納するファイルおよびディレクトリーのコピーで設定されます。

物理バックアップは、論理バックアップと比較して、以下の利点があります。

- 出力が少なくなる。
- バックアップのサイズが小さくなる。
- バックアップおよび復元が速くなる。
- バックアップには、ログファイルと設定ファイルが含まれる。

mysqld.service が実行されていない場合、またはバックアップ中の変更を防ぐためにデータベース内のすべてのテーブルがロックされている場合は、物理バックアップを実行する必要があることに注意してください。

以下の **MySQL** バックアップアプローチのいずれかを使用して、**MySQL** データベースからデータをバックアップできます。

- **mysqldump** を使用した論理バックアップ
- ファイルシステムのバックアップ
- バックアップソリューションとしてレプリケーションを使用

7.3.5.1. mysqldump を使用した論理バックアップの実行

mysqldump クライアントはバックアップユーティリティーで、バックアップ目的でデータベースまたはデータベースの集合をダンプしたり、別のデータベースサーバーに転送したりできます。通常、**mysqldump** の出力は、サーバーテーブル構造を再作成する、それにデータを取り込む、またはその両方の SQL ステートメントで設定されます。**mysqldump** は、XML および (CSV などの) コンマ区切りテキスト形式など、他の形式でファイルを生成することもできます。

mysqldump バックアップを実行するには、以下のいずれかのオプションを使用できます。

- 選択したデータベースを1つまたは複数バックアップ
- すべてのデータベースをバックアップする。
- あるデータベースのテーブルのサブセットのバックアップを作成する。

手順

- 単一のデータベースをダンプするには、以下を実行します。

```
# mysqldump [options] --databases db_name > backup-file.sql
```

- 複数のデータベースを一度にダンプするには、次のコマンドを実行します。

```
# mysqldump [options] --databases db_name1 [db_name2 ...] > backup-file.sql
```

- すべてのデータベースをダンプするには、以下を実行します。

```
# mysqldump [options] --all-databases > backup-file.sql
```

- 1つ以上のダンプされたフルデータベースをサーバーにロードし直すには、以下を実行します。

```
# mysql < backup-file.sql
```

- データベースをリモート MySQL サーバーにロードするには、以下を実行します。

```
# mysql --host=remote_host < backup-file.sql
```

- あるデータベースでリテラルなテーブルのサブセットをダンプするには、**mysqldump** コマンドの末尾に、選択したテーブルのリストを追加します。

```
# mysqldump [options] db_name [tbl_name ...] > backup-file.sql
```

- 1つのデータベースからダンプされたリテラルなテーブルのサブセットをロードするには、次のコマンドを実行します。

```
# mysql db_name < backup-file.sql
```

注記

この時点で、db_name データベースが存在している必要があります。

- mysqldump がサポートするオプションのリストを表示するには、以下を実行します。

```
$ mysqldump --help
```

関連情報

- mysqldump を使用した論理バックアップ

7.3.5.2. ファイルシステムのバックアップの実行

MySQL データファイルのファイルシステムバックアップを作成するには、MySQL データディレクトリーの内容をバックアップ場所にコピーします。

現在の設定またはログファイルのバックアップも作成するには、以下の手順の中から任意の手順を選択します。

手順

1. **mysqld** サービスを停止します。

```
# systemctl stop mysqld.service
```

2. データファイルを必要な場所にコピーします。

```
# cp -r /var/lib/mysql /backup-location
```

3. 必要に応じて、設定ファイルを必要な場所にコピーします。

```
# cp -r /etc/my.cnf /etc/my.cnf.d /backup-location/configuration
```

4. 必要に応じて、ログファイルを必要な場所にコピーします。

```
# cp /var/log/mysql/* /backup-location/logs
```

5. **mysqld** サービスを開始します。

```
# systemctl start mysqld.service
```

6. バックアップされたデータをバックアップ場所から **/var/lib/mysql**ディレクトリーに読み込む際は、**mysql:mysql** が **/var/lib/mysql** 内のすべてのデータの所有者であることを確認してください。

```
# chown -R mysql:mysql /var/lib/mysql
```

7.3.5.3. バックアップソリューションとしてレプリケーションを使用

レプリケーションは、ソースサーバー用の代替バックアップソリューションです。ソースサーバーの複製となるレプリカサーバーを作成すると、ソースに影響を与えずにレプリカでバックアップを実行できます。ソースは、レプリカをシャットダウンする間に依然として実行でき、レプリカからデータのバックアップを作成できます。

MySQLデータベースを複製する方法の手順については、MySQL の複製 を参照してください。

警告

レプリケーション自体は、バックアップソリューションとしては十分ではありません。レプリケーションは、ハードウェア障害からソースサーバーを保護しますが、データ損失に対する保護は保証していません。この方法とともに、レプリカでその他のバックアップソリューションを使用することが推奨されます。

関連情報

- MySQL replication documentation

7.3.6. MySQL 8.0 の RHEL 8 バージョンへの移行

RHEL 7 には、**MySQL** データベースファミリーからのサーバーのデフォルトの実装として、**MariaDB 5.5** が同梱されています。RHEL 7 用の Red Hat Software Collections オファリングは、 **MySQL 8.0** と **MariaDB** のいくつかのバージョンを提供します。RHEL 8 は、 **MySQL 8.0**、**MariaDB 10.3**、および **MariaDB 10.5** を提供します。

この手順では、**mysql_upgrade** ユーティリティーを使用した **MySQL 8.0** の Red Hat Software Collections バージョンから **MySQL 8.0** の RHEL8 バージョンへの移行について説明します。**mysql_upgrade** ユーティリティーは、**mysql-server** パッケージによって提供されます。

注記

MySQLの Red Hat Software Collections バージョンでは、ソースデータディレクトリーは **/var/opt/rh/rh-mysql80/lib/mysql/** です。RHEL 8 では、 **MySQL** データは **/var/lib/mysql/** ディレクトリーに保存されます。

前提条件

- アップグレードを実行する前に、**MySQL** データベースに保存されているすべてのデータをバックアップすること。

手順

1. **mysql-server** パッケージが RHEL 8 システムにインストールされていることを確認します。

```
# yum install mysql-server
```

2. データのコピー時に、**mysqld** サービスがソースシステムとターゲットシステムのどちらでも実行されていないことを確認してください。

```
# systemctl stop mysqld.service
```

3. RHEL 7 ソースシステムの **/var/opt/rh/rh-mysql80/lib/mysql/** ディレクトリーから RHEL 8 ターゲットシステムの **/var/lib/mysql/** ディレクトリーにデータをコピーします。

4. ターゲットシステムでコピーされたファイルに適切なパーミッションと SELinux コンテキストを設定します。

```
# restorecon -vr /var/lib/mysql
```

5. **mysql:mysql** が、**/var/lib/mysql** ディレクトリー内のすべてのデータの所有者であることを確認してください。

```
# chown -R mysql:mysql /var/lib/mysql
```

6. ターゲットシステムで MySQL サーバーを起動します。

```
# systemctl start mysqld.service
```

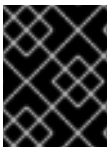
注意: MySQL の以前のバージョンでは、内部テーブルをチェックおよび修復するために **mysql_upgrade** コマンドが必要でした。これは、サーバーの起動時に自動的に実行されるようになりました。

7.3.7. MySQL の複製

MySQL には、基本的なものから高度なものまで、レプリケーション用のさまざまな設定オプションが用意されています。このセクションでは、グローバルトランザクション識別子 (GTID) を使用して、新しくインストールした MySQL サーバーに MySQL でレプリケートするトランザクションベースの方法について説明します。GTID を使用すると、トランザクションの識別と整合性の検証が簡素化されます。

MySQL でレプリケーションを設定するには、以下を行う必要があります。

- ソースサーバーを設定する
- レプリカサーバーを設定する
- ソースサーバーにレプリケーションユーザーを作成する
- レプリカサーバーをソースサーバーに接続する



重要

レプリケーションに既存の MySQL サーバーを使用する場合は、最初にデータを同期する必要があります。詳細は、アップストリームのドキュメント を参照してください。

7.3.7.1. MySQL ソースサーバーの設定

MySQL ソースサーバーがデータベースサーバーで行われたすべての変更を適切に実行および複製するために必要な設定オプションを設定できます。

前提条件

- ソースサーバーがインストールされている。

手順

1. **[mysqld]** セクションの **/etc/my.cnf.d/mysql-server.cnf** ファイルに以下のオプションを含めます。

   - **bind-address=source_ip_adress**
     このオプションは、レプリカからソースへの接続に必要です。

- **server-id=id**
  id は一意である必要があります。

- **log_bin=path_to_source_server_log**
  このオプションは、MySQL ソースサーバーのバイナリーログファイルへのパスを定義します。例: **log_bin=/var/log/mysql/mysql-bin.log**

- **gtid_mode=ON**
  このオプションは、サーバー上でグローバルトランザクション識別子 (GTID) を有効にします。

- **enforce-gtid-consistency=ON**
  サーバーは、GTID を使用して安全にログに記録できるステートメントのみの実行を許可することにより、GTID の整合性を強化します。

- **オプション: binlog_do_db=db_name**
  選択したデータベースのみを複製する場合は、このオプションを使用します。選択した複数のデータベースを複製するには、各データベースを個別に指定します。

  ```
  binlog_do_db=db_name1
  binlog_do_db=db_name2
  binlog_do_db=db_name3
  ```

- **オプション: binlog_ignore_db=db_name**
  このオプションを使用して、特定のデータベースをレプリケーションから除外します。

2. **mysqld** サービスを再起動します。

   ```
   # systemctl restart mysqld.service
   ```

7.3.7.2. MySQL レプリカサーバーの設定

レプリケーションを成功させるために MySQL レプリカサーバーに必要な設定オプションを設定できます。

前提条件

- レプリカサーバーがインストールされている。

手順

1. **[mysqld]** セクションの **/etc/my.cnf.d/mysql-server.cnf** ファイルに以下のオプションを含めます。

   - **server-id=id**
     id は一意である必要があります。

   - **relay-log=path_to_replica_server_log**
     リレーログは、レプリケーション中に MySQL レプリカサーバーによって作成されたログファイルのセットです。

   - **log_bin=path_to_replica_sever_log**
     このオプションは、MySQL レプリカサーバーのバイナリーログファイルへのパスを定義します。例: **log_bin=/var/log/mysql/mysql-bin.log**

このオプションはレプリカでは必須ではありませんが、強く推奨します。

- **gtid_mode=ON**
  このオプションは、サーバー上でグローバルトランザクション識別子 (GTID) を有効にします。

- **enforce-gtid-consistency=ON**
  サーバーは、GTID を使用して安全にログに記録できるステートメントのみの実行を許可することにより、GTID の整合性を強化します。

- **log-replica-updates=ON**
  このオプションにより、ソースサーバーから受信した更新がレプリカのバイナリーログに記録されます。

- **skip-replica-start=ON**
  このオプションは、レプリカサーバーの起動時に、レプリカサーバーがレプリケーションスレッドを開始しないようにします。

- **オプション: binlog_do_db=db_name**
  特定のデータベースのみを複製する場合は、このオプションを使用します。複数のデータベースを複製するには、各データベースを個別に指定します。

  ```
  binlog_do_db=db_name1
  binlog_do_db=db_name2
  binlog_do_db=db_name3
  ```

- **オプション: binlog_ignore_db=db_name**
  このオプションを使用して、特定のデータベースをレプリケーションから除外します。

2. **mysqld** サービスを再起動します。

```
# systemctl restart mysqld.service
```

7.3.7.3. MySQL ソースサーバーでのレプリケーションユーザーの作成

レプリケーションユーザーを作成し、このユーザーにレプリケーショントラフィックに必要なパーミッションを付与する必要があります。この手順は、適切なパーミッションを持つレプリケーションユーザーを作成する方法を示しています。これらの手順は、ソースサーバーでのみ実行してください。

前提条件

- ソースサーバーは、MySQL ソースサーバーの設定 で説明されているように、インストールおよび設定されている。

手順

1. レプリケーションユーザーを作成します。

```
mysql> CREATE USER 'replication_user'@'replica_server_ip' IDENTIFIED WITH
mysql_native_password BY 'password';
```

2. ユーザーにレプリケーション権限を付与します。

```
mysql> GRANT REPLICATION SLAVE ON *.* TO
'replication_user'@'replica_server_ip';
```

3. MySQL データベースの付与テーブルを再読み込みします。

```
mysql> FLUSH PRIVILEGES;
```

4. ソースサーバーを読み取り専用状態に設定します。

```
mysql> SET @@GLOBAL.read_only = ON;
```

7.3.7.4. レプリカサーバーをソースサーバーに接続する

MySQL レプリカサーバーでは、認証情報とソースサーバーのアドレスを設定する必要があります。次の手順を使用して、レプリカサーバーを実装します。

前提条件

- ソースサーバーは、MySQL ソースサーバーの設定 で説明されているように、インストールおよび設定されている。
- レプリカサーバーは、MySQL レプリカサーバーの設定 で説明されているように、インストールおよび設定されている。
- レプリケーションユーザーを作成している。MySQL ソースサーバーでのレプリケーションユーザーの作成 を参照してください。

手順

1. レプリカサーバーを読み取り専用状態に設定します。

```
mysql> SET @@GLOBAL.read_only = ON;
```

2. レプリケーションソースを設定します。

```
mysql> CHANGE REPLICATION SOURCE TO
   -> SOURCE_HOST='source_ip_address',
   -> SOURCE_USER='replication_user',
   -> SOURCE_PASSWORD='password',
   -> SOURCE_AUTO_POSITION=1;
```

3. MySQL レプリカサーバーでレプリカスレッドを開始します。

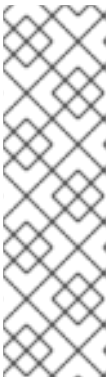
```
mysql> START REPLICA;
```

4. ソースサーバーとレプリカサーバーの両方で、読み取り専用状態の設定を解除します。

```
mysql> SET @@GLOBAL.read_only = OFF;
```

5. **オプション:** デバッグの目的で、レプリカサーバーのステータスを検査します。

```
mysql> SHOW REPLICA STATUS\G;
```



注記

レプリカサーバーの起動または接続に失敗した場合は、**SHOW MASTER STATUS** コマンドの出力に表示されるバイナリーログファイルの位置に続く特定の数のイベントをスキップできます。たとえば、定義された位置から最初のイベントをスキップします。

```
mysql> SET GLOBAL SQL_SLAVE_SKIP_COUNTER=1;
```

レプリカサーバーを再起動してみてください。

6. **オプション:** レプリカサーバーでレプリカスレッドを停止します。

```
mysql> STOP REPLICA;
```

7.3.7.5. 検証手順

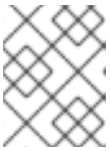
1. ソースサーバーにサンプルデータベースを作成します。

```
mysql> CREATE DATABASE test_db_name;
```

2. **test_db_name** データベースが、レプリカサーバーで複製されていることを確認します。
3. ソースサーバーまたはレプリカサーバーのいずれかで以下のコマンドを実行して、MySQL サーバーのバイナリーログファイルに関するステータス情報を表示します。

```
mysql> SHOW MASTER STATUS;
```

ソースで実行されたトランザクションの GTID のセットを示す **Executed_Gtid_Set** 列は、空であってはなりません。



注記

レプリカサーバーで **SHOW SLAVE STATUS** を使用すると、同じ GTID のセットが **Executed_Gtid_Set** 行に表示されます。

7.3.7.6. 関連情報

- MySQL Replication documentation
- How To Set Up Replication in MySQL
- Replication with Global Transaction Identifiers

7.3.8. MySQL クライアントアプリケーションの開発

Red Hat では、**MariaDB** クライアントライブラリーに対して **MySQL** クライアントアプリケーションを開発することを推奨します。クライアントとサーバー間の通信プロトコルは、**MariaDB** と **MySQL** の間で互換性があります。**MariaDB** クライアントライブラリーは、**MySQL** 実装に固有の限られた数の機能を除き、ほとんどの一般的な **MySQL** シナリオで機能します。

MariaDB クライアントライブラリーに対してアプリケーションをビルドするために必要な開発ファイルとプログラムは、**mariadb-connector-c-devel** パッケージで提供されます。

直接ライブラリー名を使用する代わりに、**mariadb-connector-c-devel** パッケージで配布されている **mariadb_config** プログラムを使用します。このプログラムにより、正しいビルドフラグが確実に返されるようになります。

7.4. POSTGRESQL の使用

PostgreSQL サーバーは、SQL 言語をベースにした、オープンソースの堅牢かつ拡張性に優れたデータベースサーバーです。**PostgreSQL** サーバーは、オブジェクトリレーショナルデータベースシステムを提供します。これにより、広範なデータセットと多数の同時ユーザーを管理できます。このような理由から、**PostgreSQL** サーバーは、大量のデータを管理するためにクラスターで使用できます。

PostgreSQL サーバーには、データの整合性の確保、耐障害性のある環境やアプリケーションの構築を行うための機能が含まれます。**PostgreSQL** サーバーを使用すると、データベースを再コンパイルすることなく、独自のデータ型、カスタム関数、またはさまざまなプログラミング言語のコードでデータベースを拡張できます。

RHEL システムに **PostgreSQL** をインストールして設定する方法、**PostgreSQL** データをバックアップする方法、および **PostgreSQL** の以前のバージョンから移行する方法について説明します。

7.4.1. PostgreSQL のインストール

RHEL 8 では、**PostgreSQL** サーバーは複数のバージョンで利用でき、各バージョンは個別のストリームで提供されます。

- **PostgreSQL 10** - デフォルトのストリーム
- **PostgreSQL 9.6**
- **PostgreSQL 12** - RHEL 8.1.1 以降で利用できます。
- **PostgreSQL 13** - RHEL 8.4 以降で利用できます。
- **PostgreSQL 15** - RHEL 8.8 以降で利用できます。
- **PostgreSQL 16** - RHEL 9.4 以降で利用可能

注記

設計上、同じモジュールの複数のバージョン (ストリーム) を並行してインストールすることはできません。したがって、**postgresql** モジュールから利用可能なストリームのいずれかを選択する必要があります。コンテナー内では、別々のバージョンの **PostgreSQL** データベースサーバーを使用できます。コンテナー内で複数の PostgreSQL バージョンを実行する を参照してください。

PostgreSQL をインストールするには、以下の手順に従います。

手順

1. **postgresql** モジュールからストリーム (バージョン) を選択し、サーバープロファイルを指定して **PostgreSQL** サーバーパッケージをインストールします。以下に例を示します。

```
# yum module install postgresql:16/server
```

postgres のスーパーユーザーが自動的に作成されます。

2. データベースクラスターを初期化します。

```
# postgresql-setup --initdb
```

Red Hat は、デフォルトの **/var/lib/pgsql/data** ディレクトリーにデータを保存することを推奨します。

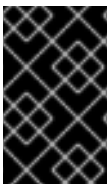
3. **postgresql** サービスを開始します。

```
# systemctl start postgresql.service
```

4. **postgresql** サービスが、システムの起動時に起動するようにします。

```
# systemctl enable postgresql.service
```

モジュールストリームの使用方法は、ユーザー空間コンポーネントのインストール、管理、および削除を参照してください。



重要

RHEL 9 内の以前の **postgresql** ストリームからアップグレードする場合は、後続のストリームへの切り替え と RHEL 9 バージョンの PostgreSQL への移行 の両方の手順に従ってください。

7.4.1.1. コンテナー内で複数の PostgreSQL バージョンを実行する

同じホスト上で別々のバージョンの PostgreSQL を実行するには、コンテナー内で実行してください。同じモジュールの複数のバージョン (ストリーム) を並行してインストールすることはできないためです。

この手順では、例として PostgreSQL 13 と PostgreSQL 15 を記載していますが、Red Hat Ecosystem Catalog で利用可能な任意の PostgreSQL コンテナーバージョンを使用できます。

前提条件

- **container-tools** モジュールがインストールされている。

手順

1. Red Hat カスタマーポータルアカウントを使用して、**registry.redhat.io** レジストリーに認証します。

```
# podman login registry.redhat.io
```

すでにコンテナーレジストリーにログインしている場合は、このステップをスキップしてください。

2. コンテナー内で PostgreSQL 13 を実行します。

```
$ podman run -d --name <container_name> -e POSTGRESQL_USER=<user_name> -e
POSTGRESQL_PASSWORD=<password> -e
POSTGRESQL_DATABASE=<database_name> -p <host_port_1>:5432
rhel8/postgresql-13
```

このコンテナーイメージを使用する方法の詳細は、Red Hat Ecosystem Catalog を参照してください。

3. コンテナー内で PostgreSQL 15 を実行します。

```
$ podman run -d --name <container_name> -e POSTGRESQL_USER=<user_name> -e
POSTGRESQL_PASSWORD=<password> -e
POSTGRESQL_DATABASE=<database_name> -p <host_port_2>:5432
rhel8/postgresql-15
```

このコンテナーイメージを使用する方法の詳細は、Red Hat Ecosystem Catalog を参照してください。

4. コンテナー内で PostgreSQL 16 を実行します。

```
$ podman run -d --name <container_name> -e POSTGRESQL_USER=<user_name> -e
POSTGRESQL_PASSWORD=<password> -e
POSTGRESQL_DATABASE=<database_name> -p <host_port_3>:5432
rhel8/postgresql-16
```

このコンテナーイメージを使用する方法の詳細は、Red Hat Ecosystem Catalog を参照してください。

注記

2 つのデータベースサーバーのコンテナー名とホストポートが異なっている必要があります。

5. クライアントがネットワーク上のデータベースサーバーにアクセスできるように、ファイアウォールでホストポートを開きます。

```
# firewall-cmd --permanent --add-port={<host_port_1>/tcp,<host_port_2>/tcp,...}
# firewall-cmd --reload
```

検証手順

1. 実行中のコンテナーに関する情報を表示します。

```
$ podman ps
```

2. データベースサーバーに接続し、root としてログインします。

```
# psql -u postgres -p -h localhost -P <host_port> --protocol tcp
```

関連情報

- コンテナーの構築、実行、および管理
- Red Hat Ecosystem Catalog でコンテナーを参照する

7.4.2. PostgreSQL ユーザーの作成

PostgreSQL ユーザーは以下のタイプのものです。

- **postgres** UNIX システムユーザー: **PostgreSQL** サーバーおよびクライアントアプリケーション (**pg_dump** など) を実行する場合にのみ使用してください。データベース作成およびユーザー管理などの、**PostgreSQL** 管理上の対話的な作業には、**postgres** システムユーザーを使用しないでください。

- データベースのスーパーユーザー: デフォルトの **postgres PostgreSQL** スーパーユーザーは、**postgres** システムユーザーとは関係ありません。**pg_hba.conf** ファイルの **postgres** のスーパーユーザーのアクセスを制限することができます。制限しない場合には、その他のパーミッションの制限はありません。他のデータベースのスーパーユーザーを作成することもできます。

- 特定のデータベースアクセスパーミッションを持つロール:

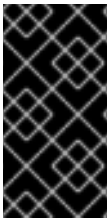
  - データベースユーザー: デフォルトでログインするパーミッションがある。

  - ユーザーのグループ: グループ全体のパーミッションを管理できるようにします。

ロールはデータベースオブジェクト (テーブルや関数など) を所有することができ、SQL コマンドを使用してオブジェクト権限を他のロールに割り当てることができます。

標準のデータベース管理権限には
SELECT、**INSERT**、**UPDATE**、**DELETE**、**TRUNCATE**、**REFERENCES**、**TRIGGER**、**CREATE**、**CONNECT**、**TEMPORARY**、**EXECUTE**、および **USAGE** が含まれます。

ロール属性は、**LOGIN**、**SUPERUSER**、**CREATEDB**、および **CREATEROLE** などの特別な権限です。



重要

Red Hat は、スーパーユーザーではないロールとしてほとんどのタスクを実行することを推奨します。一般的な方法として、**CREATEDB** および **CREATEROLE** の権限を持つロールを作成し、このロールをデータベースおよびロールのすべてのルーチン管理に使用します。

前提条件

- **PostgreSQL** サーバーがインストールされている
- データベースクラスターが初期化されている

手順

- ユーザーを作成するには、ユーザーのパスワードを設定し、ユーザーに **CREATEROLE** および **CREATEDB** の権限を割り当てます。

  ```
  postgres=# CREATE USER mydbuser WITH PASSWORD 'mypasswd' CREATEROLE CREATEDB;
  ```

  mydbuser をユーザー名に、**mypasswd** をユーザーのパスワードに置き換えます。

関連情報

- PostgreSQL データベースのロール
- PostgreSQL 権限

- PostgreSQL の設定

例7.1 PostgreSQL データベースの初期化、作成、接続

この例では、PostgreSQL データベースを初期化方法、日常的なデータベース管理権限を持つデータベースユーザーの作成方法、および管理権限を持つデータベースユーザーを介して任意のシステムアカウントからアクセスできるデータベースの作成方法を示します。

1. PosgreSQL サーバーをインストールします。

```
# yum module install postgresql:13/server
```

2. データベースクラスターを初期化します。

```
# postgresql-setup --initdb
* Initializing database in '/var/lib/pgsql/data'
* Initialized, logs are in /var/lib/pgsql/initdb_postgresql.log
```

3. パスワードハッシュアルゴリズムを **scram-sha-256** に設定します。

   a. **/var/lib/pgsql/data/postgresql.conf** ファイルで、次の行を変更します。

   ```
   #password_encryption = md5              # md5 or scram-sha-256
   ```

   更新後は次のようになります。

   ```
   password_encryption = scram-sha-256
   ```

   b. **/var/lib/pgsql/data/pg_hba.conf** ファイルで、IPv4 ローカル接続用に次の行を変更します。

   ```
   host    all             all             127.0.0.1/32            ident
   ```

   更新後は次のようになります。

   ```
   host    all             all             127.0.0.1/32            scram-sha-256
   ```

4. postgresql サービスを起動します。

```
# systemctl start postgresql.service
```

5. **postgres** という名前のシステムユーザーとしてログインします。

```
# su - postgres
```

6. **PostgreSQL** インタラクティブターミナルを起動します。

```
$ psql
psql (13.7)
Type "help" for help.

postgres=#
```

7. オプション: 現在のデータベース接続に関する情報を取得します。

```
postgres=# \conninfo
You are connected to database "postgres" as user "postgres" via socket in
"/var/run/postgresql" at port "5432".
```

8. **mydbuser** という名前のユーザーを作成し、**mydbuser** のパスワードを設定して、**CREATEROLE** および **CREATEDB** の権限を **mydbuser** に割り当てます。

```
postgres=# CREATE USER mydbuser WITH PASSWORD 'mypasswd' CREATEROLE
CREATEDB;
CREATE ROLE
```

これで、**mydbuser** ユーザーは、日常的なデータベース管理操作 (データベースの作成とユーザーインデックスの管理) を実行できるようになりました。

9. **\q** メタコマンドを使用して、インタラクティブターミナルからログアウトします。

```
postgres=# \q
```

10. **postgres** ユーザーセッションからログアウトします。

```
$ logout
```

11. **mydbuser** として PostgreSQL ターミナルにログインし、ホスト名を指定して、初期化中に作成されたデフォルトの **postgres** データベースに接続します。

```
# psql -U mydbuser -h 127.0.0.1 -d postgres
Password for user mydbuser:
Type the password.
psql (13.7)
Type "help" for help.

postgres=>
```

12. **mydatabase** という名前のデータベースを作成します。

```
postgres=> CREATE DATABASE mydatabase;
CREATE DATABASE
postgres=>
```

13. セッションからログアウトします。

```
postgres=# \q
```

14. **mydbuser** として mydatabase に接続します。

```
# psql -U mydbuser -h 127.0.0.1 -d mydatabase
Password for user mydbuser:
psql (13.7)
Type "help" for help.
mydatabase=>
```

15. オプション: 現在のデータベース接続に関する情報を取得します。

```
mydatabase=> \conninfo
You are connected to database "mydatabase" as user "mydbuser" on host "127.0.0.1" at port "5432".
```

7.4.3. PostgreSQL の設定

PostgreSQL データベースでは、データおよび設定ファイルはすべて、データベースクラスターと呼ばれる 1 つのディレクトリーに保存されます。Red Hat は、設定ファイルを含むすべてのデータをデフォルトの **/var/lib/pgsql/data/** ディレクトリーに保存することを推奨しています。

PostgreSQL 設定は、以下のファイルで設定されます。

- **postgresql.conf**: データベースのクラスターパラメーターの設定に使用されます。
- **postgresql.auto.conf**: **postgresql.conf** と同様の基本的な PostgreSQL 設定を保持します。ただし、このファイルはサーバーの制御下にあります。これは、**ALTER SYSTEM** クエリーにより編集され、手動で編集することはできません。
- **pg_ident.conf**: 外部認証メカニズムから PostgreSQL ユーザー ID へのユーザー ID のマッピングに使用されます。
- **pg_hba.conf**: PostgreSQL データベースのクライアント認証の設定に使用されます。

PostgreSQL 設定を変更するには、以下の手順に従います。

手順

1. 各設定ファイル (例: **/var/lib/pgsql/data/postgresql.conf**) を編集します。
2. **postgresql** サービスを再起動して、変更を有効にします。

```
# systemctl restart postgresql.service
```

例7.2 PostgreSQL データベースクラスターパラメーターの設定

以下の例では、**/var/lib/pgsql/data/postgresql.conf** ファイルのデータベースクラスターパラメーターの基本設定を示しています。

```
# This is a comment
log_connections = yes
log_destination = 'syslog'
search_path = '"$user", public'
shared_buffers = 128MB
password_encryption = scram-sha-256
```

例7.3 PostgreSQL でのクライアント認証の設定

以下の例では、**/var/lib/pgsql/data/pg_hba.conf** ファイルでクライアント認証を設定する方法を説明します。

```
# TYPE    DATABASE       USER        ADDRESS             METHOD
local     all            all                             trust
host      postgres       all         192.168.93.0/24     ident
host      all            all         .example.com        scram-sha-256
```

7.4.4. PostgreSQL サーバーにおける TLS 暗号化の設定

デフォルトでは、**PostgreSQL** は暗号化されていない接続を使用します。よりセキュアな接続のために、**PostgreSQL** サーバーで Transport Layer Security (TLS) サポートを有効にし、暗号化された接続を確立するようにクライアントを設定できます。

前提条件

- **PostgreSQL** サーバーがインストールされている
- データベースクラスターが初期化されている

手順

1. OpenSSL ライブラリーをインストールします。

```
# yum install openssl
```

2. TLS 証明書とキーを生成します。

```
# openssl req -new -x509 -days 365 -nodes -text -out server.crt \
  -keyout server.key -subj "/CN=dbhost.yourdomain.com"
```

dbhost.yourdomain.com を データベースのホストとドメイン名に置き換えます。

3. 署名済み証明書と秘密鍵をデータベースサーバー上の必要なロケーションにコピーします。

```
# cp server.{key,crt} /var/lib/pgsql/data/.
```

4. 署名付き証明書と秘密鍵の所有者とグループの所有権を **postgres** ユーザーに変更します。

```
# chown postgres:postgres /var/lib/pgsql/data/server.{key,crt}
```

5. 所有者だけが読み取れるように、秘密鍵の権限を制限します。

```
# chmod 0400 /var/lib/pgsql/data/server.key
```

6. **/var/lib/pgsql/data/postgresql.conf** ファイルの次の行を変更して、パスワードハッシュアルゴリズムを **scram-sha-256** に設定します。

```
#password_encryption = md5             # md5 or scram-sha-256
```

更新後は次のようになります。

```
password_encryption = scram-sha-256
```

7. **/var/lib/pgsql/data/postgresql.conf** ファイルの次の行を変更して、SSL/TLS を使用するように PostgreSQL を設定します。

```
#ssl = off
```

更新後は次のようになります。

```
ssl=on
```

8. **/var/lib/pgsql/data/pg_hba.conf** ファイルの IPv4 ローカル接続で次の行を変更して、TLS を使用するクライアントからの接続のみを受け入れるように、すべてのデータベースへのアクセスを制限します。

```
host  all  all  127.0.0.1/32  ident
```

更新後は次のようになります。

```
hostssl  all  all  127.0.0.1/32  scram-sha-256
```

または、次の行を新たに追加して、単一のデータベースとユーザーのアクセスを制限できます。

```
hostssl mydatabase mydbuser 127.0.0.1/32  scram-sha-256
```

mydatabase をデータベース名に、**mydbuser** をユーザー名に置き換えます。

9. **postgresql** サービスを再起動して、変更を有効にします。

```
# systemctl restart postgresql.service
```

検証

- 接続が暗号化されていることを手動で確認するには、以下を行います。

1. **mydbuser** ユーザーとして **PostgreSQL** データベースに接続し、ホスト名とデータベース名を指定します。

```
$ psql -U mydbuser -h 127.0.0.1 -d mydatabase
Password for user mydbuser:
```

mydatabase をデータベース名に、**mydbuser** をユーザー名に置き換えます。

2. 現在のデータベース接続に関する情報を取得します。

```
mydbuser=> \conninfo
You are connected to database "mydatabase" as user "mydbuser" on host "127.0.0.1" at port "5432".
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
```

- **PostgreSQL** への接続が暗号化されているかどうかを検証する簡単なアプリケーションを作成できます。この例は、**libpq-devel** パッケージで提供される **libpq** クライアントライブラリーを使用する C で記述されたアプリケーションを示しています。

```
#include <stdio.h>
#include <stdlib.h>
#include <libpq-fe.h>

int main(int argc, char* argv[])
{
//Create connection
PGconn* connection = PQconnectdb("hostaddr=127.0.0.1 password=mypassword
port=5432 dbname=mydatabase user=mydbuser");

if (PQstatus(connection) ==CONNECTION_BAD)
    {
    printf("Connection error\n");
    PQfinish(connection);
    return -1; //Execution of the program will stop here
    }
    printf("Connection ok\n");
    //Verify TLS
    if (PQsslInUse(connection)){
     printf("TLS in use\n");
     printf("%s\n", PQsslAttribute(connection,"protocol"));
    }
    //End connection
    PQfinish(connection);
    printf("Disconnected\n");
    return 0;
}
```

mypassword をパスワードに、**mydatabase** をデータベース名に、**myduser** をユーザー名に置き換えます。

注記

-lpq オプションを使用して、コンパイルのために **pq** ライブラリーをロードする必要があります。たとえば、GCC コンパイラーを使用してアプリケーションをコンパイルするには、次のようにします。

```
$ gcc source_file.c -lpq -o myapplication
```

この **source_file.c** には上記のサンプルコードが含まれており、**myapplication** はセキュアな **PostgreSQL** 接続を検証するためのアプリケーションの名前です。

例7.4 TLS 暗号化を使用した PostgreSQL データベースの初期化、作成、接続

この例では、PostgreSQL データベースの初期化方法、データベースユーザーとデータベースの作成方法、セキュアな接続を使用したデータベースへの接続方法を示します。

1. PosgreSQL サーバーをインストールします。

```
# yum module install postgresql:13/server
```

2. データベースクラスターを初期化します。

```
# postgresql-setup --initdb
* Initializing database in '/var/lib/pgsql/data'
* Initialized, logs are in /var/lib/pgsql/initdb_postgresql.log
```

3. OpenSSL ライブラリーをインストールします。

```
# yum install openssl
```

4. TLS 証明書とキーを生成します。

```
# openssl req -new -x509 -days 365 -nodes -text -out server.crt \
  -keyout server.key -subj "/CN=dbhost.yourdomain.com"
```

dbhost.yourdomain.com を データベースのホストとドメイン名に置き換えます。

5. 署名済み証明書と秘密鍵をデータベースサーバー上の必要なロケーションにコピーします。

```
# cp server.{key,crt} /var/lib/pgsql/data/.
```

6. 署名付き証明書と秘密鍵の所有者とグループの所有権を **postgres** ユーザーに変更します。

```
# chown postgres:postgres /var/lib/pgsql/data/server.{key,crt}
```

7. 所有者だけが読み取れるように、秘密鍵の権限を制限します。

```
# chmod 0400 /var/lib/pgsql/data/server.key
```

8. パスワードハッシュアルゴリズムを **scram-sha-256** に設定します。**/var/lib/pgsql/data/postgresql.conf** ファイルで、次の行を変更します。

```
#password_encryption = md5              # md5 or scram-sha-256
```

更新後は次のようになります。

```
password_encryption = scram-sha-256
```

9. SSL/TLS を使用するように PostgreSQL を設定します。**/var/lib/pgsql/data/postgresql.conf** ファイルで、次の行を変更します。

```
#ssl = off
```

更新後は次のようになります。

```
ssl=on
```

10. **postgresql** サービスを開始します。

```
# systemctl start postgresql.service
```

11. **postgres** という名前のシステムユーザーとしてログインします。

```
# su - postgres
```

12. **postgres** ユーザーとして PostgreSQL インタラクティブターミナルを起動します。

```
$ psql -U postgres
psql (13.7)
Type "help" for help.

postgres=#
```

13. **mydbuser** という名前のユーザーを作成し、**mydbuser** のパスワードを設定します。

```
postgres=# CREATE USER mydbuser WITH PASSWORD 'mypasswd';
CREATE ROLE
postgres=#
```

14. **mydatabase** という名前のデータベースを作成します。

```
postgres=# CREATE DATABASE mydatabase;
CREATE DATABASE
postgres=#
```

15. すべての権限を **mydbuser** ユーザーに付与します。

```
postgres=# GRANT ALL PRIVILEGES ON DATABASE mydatabase TO mydbuser;
GRANT
postgres=#
```

16. インタラクティブターミナルからログアウトします。

```
postgres=# \q
```

17. **postgres** ユーザーセッションからログアウトします。

```
$ logout
```

18. **/var/lib/pgsql/data/pg_hba.conf** ファイルの IPv4 ローカル接続で次の行を変更して、TLS を使用するクライアントからの接続のみを受け入れるように、すべてのデータベースへのアクセスを制限します。

```
host  all  all  127.0.0.1/32  ident
```

更新後は次のようになります。

```
hostssl  all  all  127.0.0.1/32  scram-sha-256
```

19. **postgresql** サービスを再起動して、変更を有効にします。

```
# systemctl restart postgresql.service
```

20. **mydbuser** ユーザーとして PostgreSQL データベースに接続し、ホスト名とデータベース名を指定します。

```
$ psql -U mydbuser -h 127.0.0.1 -d mydatabase
Password for user mydbuser:
psql (13.7)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256,
compression: off)
Type "help" for help.

mydatabase=>
```

7.4.5. PostgreSQL データのバックアップ

PostgreSQL データをバックアップするには、以下のいずれかの方法を使用します。

SQL ダンプ

Backing up with SQL dump を参照してください。

ファイルシステムレベルのバックアップ

File system level backup を参照してください。

継続的アーカイブ

継続的アーカイブ を参照してください。

7.4.5.1. SQL ダンプを使用した PostgreSQL データのバックアップ

SQL ダンプのメソッドは、SQL コマンドを使用したダンプファイルの生成に基づいています。ダンプがデータベースサーバーにアップロードされると、ダンプ時と同じ状態でデータベースが再作成されます。

SQL ダンプは、以下の PostgreSQL クライアントアプリケーションによって保証されます。

- pg_dump は、ロールまたはテーブル空間に関するクラスター全体の情報なしに単一のデータベースをダンプします。
- pg_dumpall は、指定のクラスターに各データベースをダンプし、ロールやテーブル空間定義などのクラスター全体のデータを保持します。

デフォルトでは、pg_dump コマンドおよび pg_dumpall コマンドは、結果を標準出力に書き込みます。ダンプをファイルに保存するには、出力を SQL ファイルにリダイレクトします。作成される SQL ファイルは、テキスト形式またはその他の形式のいずれかになります。これにより並列処理が可能になり、オブジェクトの復元をより詳細に制御できます。

データベースにアクセスできる任意のリモートホストから、SQL ダンプを実行できます。

7.4.5.1.1. SQL ダンプの長所と短所

SQL ダンプには、他の PostgreSQL バックアップ方法と比較して、以下の長所があります。

- SQL ダンプは、サーバーのバージョン固有ではない唯一の PostgreSQL バックアップメソッドです。pg_dump ユーティリティーの出力は、PostgreSQL の後続のバージョンに再読み込みできます。これは、ファイルシステムレベルのバックアップ、または継続的なアーカイブにはできません。
- SQL ダンプは、32 ビットサーバーから 64 ビットサーバーへなど、異なるアーキテクチャーにデータベースを転送する際に有効な唯一の方法です。

- SQL ダンプは、内部的に一貫性のあるダンプを提供します。ダンプは、**pg_dump** の実行開始時のデータベースのスナップショットを表します。
- **pg_dump** ユーティリティーは、実行中のデータベースの他の操作をブロックしません。

SQL ダンプの短所は、ファイルシステムレベルのバックアップと比較して時間がかかることです。

7.4.5.1.2. pg_dump を使用した SQL ダンプの実行

クラスター全体の情報なしに単一のデータベースをダンプするには、**pg_dump** ユーティリティーを使用します。

前提条件

- ダンプするすべてのテーブルへの読み取りアクセスが必要です。データベース全体をダンプするには、**postgres** のスーパーユーザーまたはデータベースの管理者権限を持つユーザーとして、コマンドを実行する必要があります。

手順

- クラスター全体の情報なしでデータベースをダンプします。

```
$ pg_dump dbname > dumpfile
```

pg_dump が接続するデータベースサーバーを指定するには、以下のコマンドラインオプションを使用します。

- ホストを定義する **-h** オプション
  デフォルトのホストは、ローカルホストか、**PGHOST** 環境変数で指定されているものです。
- ポートを定義する **-p** オプション
  デフォルトのポートは、**PGPORT** 環境変数またはコンパイル済みデフォルトで示されます。

7.4.5.1.3. pg_dumpall を使用した SQL ダンプの実行

特定のデータベースクラスターで各データベースをダンプし、クラスター全体のデータを保持するには、**pg_dumpall** ユーティリティーを使用します。

前提条件

- **postgres** スーパーユーザーまたはデータベースの管理者権限を持つユーザーとして、コマンドを実行する必要があります。

手順

- データベースクラスターのすべてのデータベースをダンプし、クラスター全体のデータを保存します。

```
$ pg_dumpall > dumpfile
```

pg_dumpall が接続するデータベースサーバーを指定するには、以下のコマンドラインオプションを使用します。

- ホストを定義する **-h** オプション

デフォルトのホストは、ローカルホストか、**PGHOST** 環境変数で指定されているものです。

- ポートを定義する **-p** オプション
  デフォルトのポートは、**PGPORT** 環境変数またはコンパイル済みデフォルトで示されます。

- デフォルトのデータベースを定義する **-l** オプション
  このオプションにより、初期化時に自動的に作成された **postgres** データベースとは異なるデフォルトのデータベースを選択できます。

7.4.5.1.4. pg_dump を使用したダンプされたデータベースの復元

pg_dump ユーティリティーを使用してダンプした SQL ダンプからデータベースを復元するには、以下の手順に従います。

前提条件

- **postgres** スーパーユーザーまたはデータベースの管理者権限を持つユーザーとして、コマンドを実行する必要があります。

手順

1. 新しいデータベースを作成します。

```
$ createdb dbname
```

2. ダンプされたデータベースのオブジェクトを所有するか、オブジェクトに対する権限が許可されたユーザーがすべて存在していることを検証してください。このようなユーザーが存在しない場合、復元は元の所有権と権限でオブジェクトの再作成に失敗します。

3. psql ユーティリティーを実行して、pg_dump ユーティリティーが作成したテキストファイルのダンプを復元します。

```
$ psql dbname < dumpfile
```

ここでの **dumpfile** は、**pg_dump** コマンドの出力になります。非テキストファイルのダンプを復元するには、代わりに **pg_restore** ユーティリティーを使用します。

```
$ pg_restore non-plain-text-file
```

7.4.5.1.5. pg_dumpall を使用したダンプされたデータベースの復元

pg_dumpall ユーティリティーを使用してダンプしたデータベースクラスターからデータを復元するには、以下の手順に従います。

前提条件

- **postgres** スーパーユーザーまたはデータベースの管理者権限を持つユーザーとして、コマンドを実行する必要があります。

手順

1. ダンプされたデータベースのオブジェクトを所有するか、オブジェクトに対する権限が許可されたユーザーがすべて、すでに存在していることを検証してください。このようなユーザーが存在しない場合、復元は元の所有権と権限でオブジェクトの再作成に失敗します。

2. **psql** ユーティリティーを実行して、**pg_dumpall** ユーティリティーにより作成されたテキストファイルのダンプを復元します。

```
$ psql < dumpfile
```

ここでの **dumpfile** は、**pg_dumpall** コマンドの出力になります。

7.4.5.1.6. 別のサーバーでのデータベースの SQL ダンプの実行

pg_dump および **psql** はパイプに対する読み書きが可能であるため、あるサーバーから別のサーバーにデータベースを直接ダンプできます。

手順

- データベースを、サーバーから別のサーバーにダンプするには、以下のコマンドを実行します。

```
$ pg_dump -h host1 dbname | psql -h host2 dbname
```

7.4.5.1.7. 復元中の SQL エラーの処理

デフォルトでは、SQL エラーが発生した場合、**psql** は実行を継続します。これにより、データベースの復元は一部のみとなります。

デフォルトの動作を変更するには、ダンプを復元する際に以下のいずれかの方法を使用します。

前提条件

- **postgres** スーパーユーザーまたはデータベースの管理者権限を持つユーザーとして、コマンドを実行する必要があります。

手順

- **ON_ERROR_STOP** 変数を設定して SQL エラーが発生した場合は、終了ステータスが 3 で **psql** を終了します。

```
$ psql --set ON_ERROR_STOP=on dbname < dumpfile
```

- ダンプ全体が単一のトランザクションとして復元されるように指定して、復元が完全に完了するかキャンセルされるようにします。
  - **psql** ユーティリティーを使用してテキストファイルのダンプを復元する場合:

```
$ psql -1
```

  - **pg_restore** ユーティリティーを使用してテキストファイル以外のダンプを復元する場合:

```
$ pg_restore -e
```

この方法を使用する場合は、多少のエラーでも、すでに何時間も実行している復元操作をキャンセルできます。

7.4.5.1.8. 関連情報

- PostgreSQL Documentation - SQL dump

7.4.5.2. ファイルシステムレベルのバックアップを使用した PostgreSQL データのバックアップ

ファイルシステムレベルのバックアップを実行するには、**PostgreSQL** データベースファイルを別の場所に作成します。たとえば、以下のいずれかの方法を使用できます。

- **tar** ユーティリティーを使用してアーカイブファイルを作成します。
- **rsync** ユーティリティーを使用して、ファイルを別の場所にコピーします。
- データディレクトリーの一貫したスナップショットを作成します。

7.4.5.2.1. ファイルシステムのバックアップを作成する利点と制限

ファイルシステムレベルのバックアップは、他の **PostgreSQL** バックアップ方法と比較して、以下の長所があります。

- 通常、ファイルシステムレベルのバックアップは、SQL ダンプよりも高速です。

ファイルシステムレベルのバックアップは、他の **PostgreSQL** バックアップ方法と比較して、以下の制限があります。

- このバックアップメソッドは、RHEL 7 から RHEL 8 にアップグレードし、アップグレードしたシステムにデータを移行する場合には適していません。ファイルシステムレベルのバックアップは、アーキテクチャーと RHEL メジャーバージョンに固有のものです。アップグレードに成功しなかった場合は、RHEL 7 システムでデータを復元できますが、RHEL 8 システムではデータを復元できません。
- データをバックアップおよび復元する前に、データベースサーバーをシャットダウンする必要があります。
- 特定のファイルまたはテーブルを個々にバックアップまたは復元することはできません。ファイルシステムのバックアップは、データベースクラスター全体を完全にバックアップおよび復元する場合にのみ機能します。

7.4.5.2.2. ファイルシステムレベルのバックアップの実行

ファイルシステムレベルのバックアップを実行するには、次の手順を使用します。

手順

1. データベースクラスターの場所を選択し、このクラスターを初期化します。

```
# postgresql-setup --initdb
```

2. postgresql サービスを停止します。

```
# systemctl stop postgresql.service
```

3. 任意のメソッドを使用してファイルシステムのバックアップを作成します (例: **tar** アーカイブ)。

```
$ tar -cf backup.tar /var/lib/pgsql/data
```

4. postgresql サービスを起動します。

```
# systemctl start postgresql.service
```

関連情報

- PostgreSQL Documentation - file system level backup

7.4.5.3. 継続的にアーカイブして PostgreSQL データのバックアップを作成

7.4.5.3.1. 継続的なアーカイブの概要

PostgreSQL は、データベースのデータファイルに対するすべての変更を、クラスターのデータディレクトリーの **pg_wal/** サブディレクトリーで利用可能なログ先行書き込み (WAL) ファイルに記録します。このログは、主にクラッシュからの復元を目的としています。クラッシュ後、最後のチェックポイント以降に作成されたログエントリーを使用して、データベースの整合性まで復元できます。

オンラインバックアップとも呼ばれる継続的なアーカイブメソッドは、WAL ファイルを、稼働中のサーバーまたはファイルシステムレベルのバックアップで実行されるベースバックアップの形式でデータベースクラスターのコピーと組み合わせます。

データベース復元が必要な場合は、データベースクラスターのコピーからデータベースを復元してから、バックアップを作成した WAL ファイルからログを再生して、システムを現在の状態にすることができます。

継続的なアーカイブメソッドでは、少なくとも最後のベースバックアップの開始時間までさかのぼって、アーカイブされたすべての WAL ファイルの連続したシーケンスを保持する必要があります。そのため、基本バックアップの理想的な頻度は、次の条件により異なります。

- アーカイブされた WAL ファイルで利用可能なストレージボリューム。
- 復元が必要な場合の、データ復元の最大許容期間。最後のバックアップからの期間が長い場合、システムはより多くの WAL セグメントを再生するため、回復に時間がかかります。

注記

pg_dump および **pg_dumpall** SQL ダンプは、継続的にアーカイブするバックアップソリューションの一部として使用することができません。SQL ダンプは論理バックアップを生成しますが、WAL 再生で使用する上で十分な情報は含まれていません。

継続的なアーカイブメソッドを使用してデータベースのバックアップと復元を実行するには、以下の手順に従います。

1. WAL ファイルのアーカイブ手順をセットアップおよびテストします。WAL アーカイブ を参照してください。
2. ベースバックアップを実行します。ベースバックアップ を参照してください。

データを復元するには、連続アーカイブを使用したデータベースの復元 の手順に従います。

7.4.5.3.2. 継続的なアーカイブの長所と短所

継続的なアーカイブは、PostgreSQL のその他のバックアップ方法と比較して、以下の利点があります。

- 継続的なバックアップメソッドでは、バックアップ内の内部不整合がログ再生により修正されるため、整合性が完全に取れないベースバックアップを使用することができます。したがって、実行中の PostgreSQL サーバーでベースバックアップを実行できます。
- ファイルシステムのスナップショットは必要ありません。**tar** または同様のアーカイブユーティリティーで十分です。
- 継続的にバックアップを行うには、継続的に WAL ファイルをアーカイブします。これは、ログ再生用の WAL ファイルの順序が無限に長くなる可能性があるためです。これは、特に大規模なデータベースで有用です。
- 継続的バックアップは、特定の時点への復旧 (ポイントインタイムリカバリー) をサポートします。WAL エントリーを最後まで再生する必要はありません。再生はいつでも停止でき、ベースバックアップを作成してから、データベースをいつでもその状態に復元できます。
- 一連の WAL ファイルが同じベースのバックアップファイルで読み込まれた別のマシンが継続的に利用可能である場合は、任意の時点で、データベースで現在に一番近いコピーで、他のマシンを復元できます。

継続的なアーカイブには、その他の PostgreSQL バックアップ方法と比較して、以下の短所があります。

- 継続バックアップ方法は、サブセットではなく、データベースクラスター全体の復元のみをサポートします。
- 継続的にバックアップするには、大きなアーカイブストレージが必要です。

7.4.5.3.3. WAL アーカイブの設定

稼働中の PostgreSQL サーバーでは、ログ先行書き込み (WAL) レコードのシーケンスを生成します。サーバーは、このシーケンスを WAL セグメントファイルに物理的に分割します。このファイルには、WAL シーケンスの位置を反映する数値名が与えられます。WAL のアーカイブを使用しない場合は、セグメントファイルは再利用され、より大きなセグメント番号に名前が変更されます。

WAL データをアーカイブする場合、各セグメントファイルの内容がキャプチャーされ、新しい場所に保存されてから、セグメントファイルが再利用されます。別のマシン上の NFS マウントディレクトリー、テープドライブ、または CD など、コンテンツの保存場所には複数のオプションがあります。

WAL レコードには、設定ファイルへの変更が含まれていないことに注意してください。

WAL のアーカイブを有効にするには、以下の手順に従います。

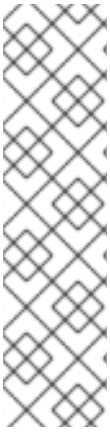
手順

1. **/var/lib/pgsql/data/postgresql.conf** ファイルで以下を行います。
   a. **wal_level** 設定パラメーターを **replica** 以降に設定します。
   b. **archive_mode** パラメーターを **on** に設定します。
   c. **archive_command** 設定パラメーターでシェルコマンドを指定します。**cp** コマンド、別のコマンド、またはシェルスクリプトを使用できます。
2. **postgresql** サービスを再起動して、変更を適用します。

```
# systemctl restart postgresql.service
```

3. アーカイブコマンドをテストし、既存のファイルが上書きされないこと、失敗した場合にゼロ以外の終了ステータスが返されることを確認します。

4. データを保護するには、セグメントファイルがグループまたはワールド読み取りアクセスを持たないディレクトリーにアーカイブされていることを確認してください。



注記

archive コマンドは、完了した WAL セグメントでのみ実行されます。WAL トラフィックをほとんど生成しないサーバーでは、トランザクションの完了とアーカイブストレージへの安全な記録の間にかなりの遅延が生じる可能性があります。アーカイブされていないデータの古さを制限するには、以下を行います。

- **archive_timeout** パラメーターを設定して、サーバーが特定の頻度で新しい WAL セグメントファイルに切り替えるように強制します。
- **pg_switch_wal** パラメーターを使用して、セグメント切り替えを強制し、トランザクションが終了後すぐにアーカイブされるようにします。

例7.5 WAL セグメントをアーカイブするためのシェルコマンド

この例は、**archive_command** 設定パラメーターに設定できる簡単なシェルコマンドを示しています。

以下のコマンドは、完了したセグメントファイルを必要な場所にコピーします。

```
archive_command = 'test ! -f /mnt/server/archivedir/%f && cp %p /mnt/server/archivedir/%f'
```

%p パラメーターは、アーカイブするファイルの相対パスに置き換えられ、**%f** パラメーターはファイル名に置き換えられます。

このコマンドは、アーカイブ可能な WAL セグメントを **/mnt/server/archivedir/** ディレクトリーにコピーします。**%p** パラメーターおよび **%f** パラメーターを置き換えると、実行されたコマンドは以下のようになります。

```
test ! -f /mnt/server/archivedir/000000010000000A900000065 && cp
pg_wal/000000010000000A900000065 /mnt/server/archivedir/000000010000000A900000065
```

アーカイブされる新規ファイルごとに同様のコマンドが生成されます。

関連情報

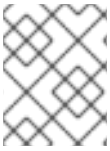
- PostgreSQL 16 ドキュメント

7.4.5.3.4. ベースバックアップの作成

ベースバックアップは、複数の方法で作成できます。ベースバックアップを実行する最も簡単な方法は、実行中の PostgreSQL サーバーで pg_basebackup ユーティリティーを使用することです。

ベースバックアップブロセスは、WAL アーカイブ領域に保存され、ベースバックアップに必要な最初の WAL セグメントファイルにちなんで名付けられたバックアップ履歴ファイルを作成します。

バックアップ履歴ファイルは、開始時間および終了時間、およびバックアップの WAL セグメントが含まれる小さなテキストファイルです。ラベル文字列を使用して関連するダンプファイルを特定した場合は、バックアップ履歴ファイルを使用して復元するダンプファイルを判断できます。



注記

データを確実に復元できるようにするために、複数のバックアップセットを維持することを検討してください。

ベースバックアップを実行するには、以下の手順を行います。

前提条件

- **postgres** スーパーユーザー、データベースの管理者権限のあるユーザー、または少なくとも **REPLICATION** パーミッションを持つ別のユーザーとして、コマンドを実行する必要があります。
- ベースバックアップ中およびベースバックアップ後に生成されたすべての WAL セグメントファイルを保持する必要があります。

手順

1. **pg_basebackup** ユーティリティーを使用してベースバックアップを実行します。
   - 個々のファイル (プレーン形式) としてベースバックアップを作成するには、以下を実行します。

     ```
     $ pg_basebackup -D backup_directory -Fp
     ```

     backup_directory は、任意のバックアップの場所に置き換えます。

     テーブル空間を使用し、サーバーと同じホストでベースバックアップを実行する場合は、**--tablespace-mapping** オプションも使用する必要があります。そうしないと、バックアップを同じ場所に書き込もうとすると、バックアップが失敗します。

   - **tar** アーカイブ (**tar** および圧縮形式) としてベースバックアップを作成するには、以下を実行します。

     ```
     $ pg_basebackup -D backup_directory -Ft -z
     ```

     backup_directory は、任意のバックアップの場所に置き換えます。

     このようなデータを復元するには、ファイルを正しい場所に手動で抽出する必要があります。

2. ベースバックアッププロセスが完了すると、バックアップ履歴ファイルで指定されている、データベースクラスターのコピーとバックアップ中に使用された WAL セグメントファイルを安全にアーカイブします。
3. ベースバックアップで使用されている WAL セグメントファイルよりも数値が小さい WAL セグメントを削除します。これらはベースバックアップよりも古く、復元には必要ないためです。

pg_basebackup が接続するデータベースサーバーを指定するには、以下のコマンドラインオプションを使用します。

- ホストを定義する **-h** オプション
  デフォルトのホストは、ローカルホストまたは **PGHOST** 環境変数により指定されたホストです。

- ポートを定義する **-p** オプション
  デフォルトのポートは、**PGPORT** 環境変数またはコンパイル済みデフォルトで示されます。

関連情報

- PostgreSQL ドキュメント - ベースバックアップ
- PostgreSQL ドキュメント - pg_basebackup ユーティリティー

7.4.5.3.5. 継続的なアーカイブバックアップを使用したデータベースの復元

継続バックアップを使用してデータベースを復元するには、以下の手順を行います。

手順

1. サーバーを停止します。

```
# systemctl stop postgresql.service
```

2. 必要なデータを一時的な場所にコピーします。
   必要に応じて、クラスターデータのディレクトリー全体と、すべてのテーブル空間をコピーします。既存データベースのコピーを 2 つ保持するには、システムに十分な空き領域が必要になることに注意してください。

   十分な容量がない場合は、クラスターの **pg_wal** ディレクトリーの内容を保存します。これには、システムがダウンする前にアーカイブされなかったログが含まれます。

3. クラスターデータディレクトリー、および使用しているテーブル空間のルートディレクトリー下の既存ファイルおよびサブディレクトリーをすべて削除します。

4. ベースバックアップからデータベースファイルを復元します。
   以下の点を確認してください。

   - ファイルは、正しい所有権 (**root** ではなくデータベースシステムのユーザー) で復元されます。

   - ファイルは、正しい権限で復元されます。

   - **pg_tblspc/** サブディレクトリーのシンボリックリンクが正しく復元されます。

5. **pg_wal/** サブディレクトリーにあるファイルをすべて削除します。
   このファイルは、ベースバックアップから作成されるため、非推奨になりました。**pg_wal/** をアーカイブしていない場合は、適切な権限で再作成します。

6. 手順 2 で保存したアーカイブされていない WAL セグメントファイルを **pg_wal/** にコピーします。

7. クラスターデータディレクトリーの **recovery.conf** リカバリーコマンドファイルを作成し、**restore_command** 設定パラメーターにシェルコマンドを指定します。**cp** コマンド、別のコマンド、またはシェルスクリプトを使用できます。以下に例を示します。

```
restore_command = 'cp /mnt/server/archivedir/%f "%p"'
```

8. サーバーを起動します。

```
# systemctl start postgresql.service
```

サーバーは復元モードに入り、引き続き必要なアーカイブファイル (WAL) を読み込みます。

外部エラーにより復元が終了した場合は、サーバーを再起動して復元を続行します。復元プロセスが完了すると、サーバーは **recovery.conf** の名前を **recovery. done** に変更します。これにより、サーバーが通常のデータベース操作を開始した後に、誤ってリカバリーモードに戻るのを防ぐことができます。

9. データベースのコンテンツを確認して、データベースが必要な状態に復元されたことを検証します。
データベースが必要な状態に復元されていない場合は、手順 1 に戻ります。データベースが必要な状態に復元された場合は、**pg_hba.conf** ファイルでクライアント認証設定を復元して接続できるようにします。

継続バックアップを使用した復元の詳細は、PostgreSQL ドキュメント を参照してください。

7.4.5.3.6. 関連情報

- 継続的アーカイブ方法

7.4.6. RHEL 8 バージョンの PostgreSQL への移行

Red Hat Enterprise Linux 7 には、**PostgreSQL 9.2** が、**PostgreSQL** サーバーのデフォルトバージョンとして含まれます。また、RHEL 7 の Software Collections として、複数の **PostgreSQL** のバージョンが提供されます。

Red Hat Enterprise Linux 8 は、**PostgreSQL 10** (デフォルトの **postgresql** ストリーム)、**PostgreSQL 9.6**、**PostgreSQL 12**、**PostgreSQL 13**、および **PostgreSQL 15** を提供します。

Red Hat Enterprise Linux 上の **PostgreSQL** のユーザーは、データベースファイルの移行パスを使用できます。

- pg_upgrade ユーティリティーを使用した高速アップグレード
- ダンプおよび復元のアップグレード

高速アップグレードメソッドは、ダンプおよび復元のプロセスよりも速くなります。ただし、場合によっては高速アップグレードが機能せず、ダンプおよび復元プロセスしか使用できない場合があります。そのようなケースには、以下が含まれます。

- アーキテクチャー間のアップグレード
- **plpython** または **plpython2** 拡張を使用するシステム。RHEL 8 AppStream リポジトリーには **postgresql-plpython3** パッケージのみが含まれ、**postgresql-plpython2** パッケージは含まれません。

- 高速アップグレードは、**PostgreSQL**の Red Hat Software Collections バージョンからの移行ではサポートされません。

新しいバージョンの **PostgreSQL** に移行するための前提条件として、すべての **PostgreSQL** データベースをバックアップします。

データベースをダンプし、SQL ファイルのバックアップを実行することは、ダンプおよび復元プロセスで必要であり、高速アップグレードメソッドとして推奨されます。

新しいバージョンの **PostgreSQL** に移行する前に、移行する **PostgreSQL** バージョンと、移行元と移行先のバージョンの間にあるすべて **PostgreSQL** バージョンの アップストリームの互換性ノート を参照してください。

7.4.6.1. PostgreSQL 15 と PostgreSQL 16 間の主な違い

PostgreSQL 16 では、次の主な変更点が導入されました。

postmasters バイナリーが利用できなくなりました

PostgreSQL が **postmaster** バイナリーとともに配布されなくなりました。提供されている **systemd** ユニットファイル (**systemctl start postgres** コマンド) を使用して **postgresql** サーバーを起動するユーザーは、この変更の影響を受けません。以前に **postmaster** バイナリーを介して **postgresql** サーバーを直接起動していた場合は、今後は代わりに **postgres** バイナリーを使用する必要があります。

ドキュメントがパッケージに同梱されなくなりました

PostgreSQL のパッケージで PDF 形式のドキュメントが提供されなくなりました。代わりに オンラインドキュメント を使用してください。

7.4.6.2. PostgreSQL 13 と PostgreSQL 15 間の主な違い

PostgreSQL 15 では、以下の後方互換性のない変更が導入されました。

パブリックスキーマのデフォルトパーミッション

パブリックスキーマのデフォルトパーミッションは、**PostgreSQL 15** で変更されています。新規に作成されたユーザーは、**GRANT ALL ON SCHEMA public TO myuser;** コマンドを使用して、権限を明示的に付与する必要があります。

次の例は **PostgreSQL 13** 以前で動作します。

```
postgres=# CREATE USER mydbuser;
postgres=# \c postgres mydbuser
postgres=$ CREATE TABLE mytable (id int);
```

次の例は **PostgreSQL 15** 以降で動作します。

```
postgres=# CREATE USER mydbuser;
postgres=# GRANT ALL ON SCHEMA public TO mydbuser;
postgres=# \c postgres mydbuser
postgres=$ CREATE TABLE mytable (id int);
```

注記

pg_hba.conf ファイルで **mydbuser** のアクセス権が適切に設定されていることを確認してください。詳細は、PostgreSQL ユーザーの作成 を参照してください。

PQsendQuery() がパイプラインモードでサポートされなくなる

PostgreSQL 15 以降、**libpq PQsendQuery()** 関数はパイプラインモードでサポートされなくなりました。影響を受けるアプリケーションを変更して、代わりに **PQsendQueryParams()** 関数を使用します。

7.4.6.3. pg_upgrade ユーティリティーを使用した高速アップグレード

高速アップグレードでは、バイナリーデータファイルを **/var/lib/pgsql/data/** ディレクトリーにコピーし、**pg_upgrade** ユーティリティーを使用する必要があります。

この方法を使用すると、データを移行できます。

- RHEL 7 システムバージョンの **PostgreSQL 9.2** から、RHEL 8 バージョンの **PostgreSQL 10** へ
- RHEL 8 バージョンの **PostgreSQL 12** から RHEL バージョンの **PostgreSQL 13** へ
- RHEL 8 バージョンの **PostgreSQL 12** から RHEL バージョンの **PostgreSQL 13** へ
- RHEL 8 または 9 バージョンの **PostgreSQL 15** から RHEL バージョンの **PostgreSQL 16** へ
- RHEL 8 または 9 バージョンの **PostgreSQL 15** から RHEL バージョンの **PostgreSQL 16** へ

RHEL 8 内の以前の **postgresql** ストリームからアップグレードする場合は 後続のストリームへの切り替え の説明に従い、**PostgreSQL** データを移行します。

RHEL 内の **PostgreSQL** バージョンの組み合わせと、Red Hat Software Collections バージョンの **PostgreSQL** から RHEL への移行には、 アップグレードのダンプおよび復元 を使用します。

以下の手順では、高速アップグレードメソッドを使用して、RHEL 7 システムバージョンの **PostgreSQL 9.2** から、RHEL 8 バージョンの **PostgreSQL** へ移行する方法を説明します。

前提条件

- アップグレードを実行する前に、**PostgreSQL** データベースに保存されているすべてのデータのバックアップを作成します。デフォルトでは、すべてのデータは、RHEL 7 および RHEL 8 システムの両方の **/var/lib/pgsql/data/** ディレクトリーに保存されます。

手順

1. RHEL 8 システムで、移行するストリーム (バージョン) を有効にします。

   ```
   # yum module enable postgresql:stream
   ```

   stream は、選択した **PostgreSQL** サーバーのバージョンに置き換えます。

   PostgreSQL 10 を提供するデフォルトストリームを使用する場合は、この手順を省略できます。

2. RHEL 8 システムで、**postgresql-server** パッケージおよび **postgresql-upgrade** パッケージをインストールします。

   ```
   # yum install postgresql-server postgresql-upgrade
   ```

   必要に応じて、RHEL 7 で **PostgreSQL** サーバーモジュールを使用している場合は、その 2 つ

のバージョンを RHEL 8 システムにインストールし、PostgreSQL 9.2 (**postgresql-upgrade** パッケージでインストール) および対象バージョンの PostgreSQL (**postgresql-server** パッケージでインストール) の両方に対してコンパイルします。サードパーティーの PostgreSQL サーバーモジュールをコンパイルする必要がある場合は、**postgresql-devel** パッケージと **postgresql-upgrade-devel** パッケージの両方に対してビルドしてください。

3. 以下の項目を確認します。

   - 基本設定 - RHEL 8 システムで、サーバーがデフォルトの **/var/lib/pgsql/data** ディレクトリーを使用し、データベースが正しく初期化され、有効になっているかどうかを確認します。さらに、データファイルは、**/usr/lib/systemd/system/postgresql.service** ファイルに記載されているパスと同じパスに保存する必要があります。

   - PostgreSQL サーバー - システムは複数の PostgreSQL サーバーを実行できます。これらのすべてのサーバーのデータディレクトリーが独立して処理されていることを確認してください。

   - PostgreSQL サーバーモジュール - RHEL 7 で使用されていた PostgreSQL サーバーモジュールも、RHEL 8 システムにインストールされていることを確認してください。プラグインは、/usr/lib64/pgsql/ ディレクトリー (32 ビットシステムの /usr/lib/pgsql/ ディレクトリー) にインストールされる点に注意してください。

4. データのコピー時に、**postgresql** サービスがソースおよびターゲットのシステムで稼働していないことを確認します。

   ```
   # systemctl stop postgresql.service
   ```

5. データベースファイルをソースの場所から RHEL 8 システムの **/var/lib/pgsql/data/** ディレクトリーにコピーします。

6. PostgreSQL ユーザーで以下のコマンドを実行して、アップグレードプロセスを実行します。

   ```
   # postgresql-setup --upgrade
   ```

   これでバックグラウンドで **pg_upgrade** プロセスが開始します。

   障害が発生すると、**postgresql-setup** は通知のエラーメッセージを提供します。

7. **/var/lib/pgsql/data-old** から新規クラスターに、以前の設定をコピーします。
   高速アップグレードは、新しいデータスタックで以前の設定を再利用せず、設定がゼロから生成されることに注意してください。古い設定と新しい設定を手動で組み合わせたい場合は、データディレクトリーの *.conf ファイルを使用します。

8. 新しい PostgreSQL サーバーを起動します。

   ```
   # systemctl start postgresql.service
   ```

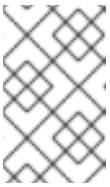
9. 新しいデータベースクラスターを分析します。

   - PostgreSQL 13 以前の場合:

     ```
     su postgres -c '~/analyze_new_cluster.sh'
     ```

   - PostgreSQL 15 以降の場合:

```
su postgres -c 'vacuumdb --all --analyze-in-stages'
```



注記

場合によって、**ALTER COLLATION name REFRESH VERSION** を使用する必要があります。詳細は、アップストリームのドキュメント を参照してください。

10. システムの起動時に、新しい **PostgreSQL** サーバーを自動的に起動させる場合は、次のコマンドを実行します。

```
# systemctl enable postgresql.service
```

7.4.6.4. ダンプおよび復元のアップグレード

ダンプおよび復元のアップグレードを使用する場合は、すべてのデータベースのコンテンツを SQL ファイルのダンプファイルにダンプする必要があります。

ダンプおよび復元のアップグレードは高速アップグレード方法よりも低速であり、生成された SQL ファイルで手動修正が必要になる場合があります。

この方法を使用すると、以下からデータを移行できます。

- Red Hat Enterprise Linux 7 システムバージョンの **PostgreSQL 9.2**
- 以前の Red Hat Enterprise Linux 8 バージョンの **PostgreSQL**
- Red Hat Software Collections の **PostgreSQL** の以前のバージョンまたは同等バージョン:
  - **PostgreSQL 9.2** (サポート対象外になりました)
  - **PostgreSQL 9.4** (サポート対象外になりました)
  - **PostgreSQL 9.6** (サポート対象外になりました)
  - **PostgreSQL 10**
  - **PostgreSQL 12**
  - **PostgreSQL 13**

RHEL 7 および RHEL 8 システムでは、**PostgreSQL** データは、デフォルトで **/var/lib/pgsql/data/** ディレクトリーに保存されます。Red Hat Software Collections バージョンの **PostgreSQL**の場合、デフォルトのデータディレクトリーは **/var/opt/rh/collection_name/lib/pgsql/data/** です (**/opt/rh/postgresql92/root/var/lib/pgsql/data/** ディレクトリーを使用する **postgresql92** を除く)。

RHEL 8 内の以前の **postgresql** ストリームからアップグレードする場合は 後続のストリームへの切り替え の説明に従い、**PostgreSQL** データを移行します。

ダンプおよび復元のアップグレードを実行するには、ユーザーを **root** に変更します。

以下の手順では、RHEL 7 システムバージョンの **PostgreSQL 9.2** から、RHEL 8 バージョンの **PostgreSQL** への移行方法を説明します。

手順

1. RHEL 7 システムで **PostgreSQL 9.2** サーバーを起動します。

```
# systemctl start postgresql.service
```

2. RHEL 7 システムで、すべてのデータベースのコンテンツを **pgdump_file.sql** ファイルにダンプします。

```
su - postgres -c "pg_dumpall > ~/pgdump_file.sql"
```

3. データベースが正しくダンプされたことを確認します。

```
su - postgres -c 'less "$HOME/pgdump_file.sql"'
```

これにより、ダンプされた sql ファイルのパスが **/var/lib/pgsql/pgdump_file.sql** に表示されます。

4. RHEL 8 システムで、移行するストリーム (バージョン) を有効にします。

```
# yum module enable postgresql:stream
```

stream は、選択した **PostgreSQL** サーバーのバージョンに置き換えます。

PostgreSQL 10 を提供するデフォルトストリームを使用する場合は、この手順を省略できます。

5. RHEL 8 システムで、**postgresql-server** パッケージをインストールします。

```
# yum install postgresql-server
```

必要に応じて、RHEL 7 で **PostgreSQL** サーバーモジュールを使用している場合は、RHEL 8 システムにもインストールしてください。サードパーティーの **PostgreSQL** サーバーモジュールをコンパイルする必要がある場合は、**postgresql-devel** パッケージに対してビルドします。

6. RHEL 8 システムで、新しい **PostgreSQL** サーバーのデータディレクトリーを初期化します。

```
# postgresql-setup --initdb
```

7. RHEL 8 システムで、**pgdump_file.sql** を **PostgreSQL** ホームディレクトリーにコピーし、ファイルが正しくコピーされたことを確認します。

```
su - postgres -c 'test -e "$HOME/pgdump_file.sql" && echo exists'
```

8. RHEL 7 システムから設定ファイルをコピーします。

```
su - postgres -c 'ls -1 $PGDATA/*.conf'
```

コピーされる設定ファイルは、以下のとおりです。

- **/var/lib/pgsql/data/pg_hba.conf**
- **/var/lib/pgsql/data/pg_ident.conf**
- **/var/lib/pgsql/data/postgresql.conf**

9. RHEL 8 システムで、新しい **PostgreSQL** サーバーを起動します。

```
# systemctl start postgresql.service
```

10. RHEL 8 システムで、ダンプされた sql ファイルからデータをインポートします。

```
su - postgres -c 'psql -f ~/pgdump_file.sql postgres'
```

注記

Red Hat Software Collections バージョンの **PostgreSQL** からアップグレードする場合は、**scl enable collection_name** が含まれるようにコマンドを調整します。たとえば、**rh-postgresql96** Software Collection からデータをダンプする場合は、以下のコマンドを使用します。

```
su - postgres -c 'scl enable rh-postgresql96 "pg_dumpall > ~/pgdump_file.sql"'
```

第8章 POSTFIX SMTP サーバーのデプロイと設定

システム管理者は、Postfix などのメールトランスポートエージェント (MTA) を使用して、電子メールインフラストラクチャーを設定し、SMTP プロトコルを使用してホスト間で電子メールメッセージを転送できます。Postfix は、メールのルーティングと配信を行うサーバー側アプリケーションです。Postfix を使用して、ローカルメールサーバーの設定、null クライアントメールリレーの作成、複数のドメインの宛先としての Postfix サーバーの使用、検索用ファイルに代わる LDAP ディレクトリーの選択を行うことができます。

Postfix の主な機能:

- 一般的なメール関連の脅威から保護するためのセキュリティー機能
- 仮想ドメインおよびエイリアスのサポートを含むカスタマイズオプション

8.1. 主な POSTFIX 設定ファイルの概要

postfix パッケージは、**/etc/postfix/** ディレクトリーに複数の設定ファイルを提供します。

電子メールインフラストラクチャーを設定するには、次の設定ファイルを使用します。

- **main.cf** – Postfix のグローバル設定が含まれています。
- **master.cf** – メール配信を実現するために、さまざまなプロセスとの Postfix の対話を指定します。
- **access** – Postfix に接続できるホストなどのアクセスルールを指定します。
- **transport** – 電子メールアドレスをリレーホストにマッピングします。
- **aliases** – ユーザー ID エイリアスを説明するメールプロトコルで必要な設定可能な一覧が含まれます。このファイルは、**/etc/** ディレクトリーにあることに留意してください。

8.2. POSTFIX SMTP サーバーのインストールおよび設定

電子メールメッセージを受信、保存、配信するように Postfix SMTP サーバーを設定できます。システムのインストール時にメールサーバーパッケージが選択されていない場合、Postfix はデフォルトで利用できません。Postfix をインストールするには、以下の手順を実行します。

前提条件

- root アクセスがある。
- システムを登録する。
- Sendmail を無効にして削除するには、以下を実行します。

```
# yum remove sendmail
```

手順

1. Postfix をインストールします。

```
# yum install postfix
```

2. Postfix を設定するには、**/etc/postfix/main.cf** ファイルを編集し、以下の変更を加えます。

   a. デフォルトでは、Postfix は **loopback** インターフェイスでのみメールを受信します。特定のインターフェイスをリッスンするように Postfix を設定するには、**inet_interfaces** パラメーターをこれらのインターフェイスの IP アドレスに更新します。

   ```
   inet_interfaces = 127.0.0.1/32, [::1]/128, 192.0.2.1, [2001:db8:1::1]
   ```

   すべてのインターフェイスをリッスンするように Postfix を設定するには、以下を設定します。

   ```
   inet_interfaces = all
   ```

   b. **gethostname()** 関数によって返される完全修飾ドメイン名 (FQDN) とは異なるホスト名を Postfix が使用するようにしたい場合は、**myhostname** パラメーターを追加します。

   ```
   myhostname = <smtp.example.com>
   ```

   たとえば、Postfix はこのホスト名を、処理するメールのヘッダーに追加します。

   c. ドメイン名が **myhostname** パラメーターのものと異なる場合は、**mydomain** パラメーターを追加します。

   ```
   mydomain = <example.com>
   ```

   d. **myorigin** パラメーターを追加し、**mydomain** の値に設定します。

   ```
   myorigin = $mydomain
   ```

   この設定では、Postfix はホスト名ではなく、ローカルで投稿されたメールの発信元としてドメイン名を使用します。

   e. **mynetworks** パラメーターを追加し、メールの送信が許可される信頼できるネットワークの IP 範囲を定義します。

   ```
   mynetworks = 127.0.0.1/32, [::1]/128, 192.0.2.1/24, [2001:db8:1::1]/64
   ```

   インターネットなどの信頼できないネットワークからのクライアントがこのサーバー経由でメールを送信できるようにする必要がある場合は、後の手順でリレー制限を設定する必要があります。

3. **main.cf** ファイルの Postfix 設定が正しいか確認します。

   ```
   $ postfix check
   ```

4. **postfix** サービスが起動時に開始できるように有効化し、開始します。

   ```
   # systemctl enable --now postfix
   ```

5. smtp トラフィックがファイアウォールを通過することを許可し、ファイアウォールルールをリロードします。

```
# firewall-cmd --permanent --add-service smtp

# firewall-cmd --reload
```

検証

1. **postfix** サービスが実行していることを確認します。

```
# systemctl status postfix
```

- オプション: 出力が停止し、待機中、またはサービスが実行されていない場合は、**postfix** サービスを再起動します。

```
# systemctl restart postfix
```

- オプション: **/etc/postfix/** ディレクトリーの設定ファイル内のオプションを変更した後、**postfix** サービスをリロードして、これらの変更を適用します。

```
# systemctl reload postfix
```

2. システム上のローカルユーザー間の電子メール通信を確認します。

```
# echo "This is a test message" | mail -s <SUBJECT> <user@mydomain.com>
```

3. クライアント (**server1**) からメールサーバー (**server2**) へ、ドメイン外のメールアドレスにメールを送信して、メールサーバーがオープンリレーではないことを確認します。

a. 次のように、server1 の **/etc/postfix/main.cf** ファイルを編集します。

```
relayhost = <ip_address_of_server2>
```

b. 次のように、server2 の **/etc/postfix/main.cf** ファイルを編集します。

```
mynetworks = <ip_address_of_server2>
```

c. **server1** で、以下のメールを送信します。

```
# echo "This is an open relay test message" | mail -s <SUBJECT>
<user@example.com>
```

d. **/var/log/maillog** ファイルを確認します。

```
554 Relay access denied - the server is not going to relay.
250 OK or similar - the server is going to relay.
```

トラブルシューティング

- エラーが発生した場合は、**/var/log/maillog** を確認してください。

関連情報

- **/etc/postfix/main.cf** 設定ファイル

- **/usr/share/doc/postfix/README_FILES** ディレクトリー
- firewalld の使用および設定

8.3. POSTFIX サーバーの TLS 設定のカスタマイズ

電子メールトラフィックを暗号化してよりセキュアにするために、自己署名証明書の代わりに、信頼できる認証局 (CA) からの証明書を使用し、Transport Layer Security (TLS) セキュリティー設定をカスタマイズするように Postfix を設定できます。RHEL 8 では、TLS 暗号化プロトコルが Postfix サーバーでデフォルトで有効になっています。基本的な Postfix TLS 設定には、受信 SMTP 用の自己署名証明書と、発信 SMTP の日和見 TLS が含まれています。

前提条件

- root アクセスがある。
- サーバーに **postfix** パッケージがインストールされている。
- 信頼できる認証局 (CA) によって署名された証明書と秘密鍵を持っている。
- 以下のファイルを Postfix サーバーにコピーしている。
  - サーバー証明書: **/etc/pki/tls/certs/postfix.pem**
  - 秘密鍵y: **/etc/pki/tls/private/postfix.key**

手順

1. 以下の行を **/etc/postfix/main.cf** ファイルに追加して、Postfix が実行されているサーバー上の証明書と秘密鍵ファイルへのパスを設定します。

```
smtpd_tls_cert_file = /etc/pki/tls/certs/postfix.pem
smtpd_tls_key_file = /etc/pki/tls/private/postfix.key
```

2. **/etc/postfix/main.cf** ファイルを編集して、受信した SMTP 接続を認証されたユーザーのみに制限します。

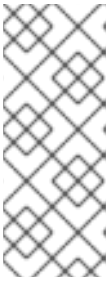
```
smtpd_tls_auth_only = yes
```

3. **postfix** サービスをリロードして変更を適用します。

```
# systemctl reload postfix
```

検証

- TLS 暗号化を使用してメールを送信するようにクライアントを設定します。



注記

Postfix クライアント TLS アクティビティーに関する追加情報を取得するには、**/etc/postfix/main.cf** の次の行を変更して、ログレベルを **0** から **1** に増やします。

```
smtp_tls_loglevel = 1
```

8.4. すべての電子メールをメールリレーに転送するように POSTFIX を設定する

すべての電子メールをメールリレーに転送する場合は、Postfix サーバーを Null クライアントとして設定できます。この設定では、Postfix はメールを別のメールサーバーに転送するだけで、メールの受信はできません。

前提条件

- root アクセスがある。
- サーバーに **postfix** パッケージがインストールされている。
- メールを転送するリレーホストの IP アドレスまたはホスト名がある。

手順

1. Postfix がローカルの電子メール配信を受け入れ、それが Null クライアントになるのを防ぐには、**/etc/postfix/main.cf** ファイルを編集し、以下の変更を加えます。

   a. **mydestination** パラメーターを空の値に等しくなるように設定して、すべてのメールを転送するように Postfix を設定します。

   ```
   mydestination =
   ```

   この設定では、Postfix サーバーはメールの宛先ではなく、null クライアントとして機能します。

   b. Null クライアントからメールを受信するメールリレーサーバーを指定します。

   ```
   relayhost = <[ip_address_or_hostname]>
   ```

   リレーホストはメール配信を行います。**<ip_address_or_hostname>** を角括弧で囲みます。

   c. メールを配信するために、ループバックインターフェイスでのみリッスンするように Postfix メールサーバーを設定します。

   ```
   inet_interfaces = loopback-only
   ```

   d. Postfix がすべての送信メールの送信者ドメインをリレーメールサーバーの企業ドメインに書き換えるには、以下を設定します。

   ```
   myorigin = <relay.example.com>
   ```

e. ローカルメール配信を無効にするには、設定ファイルの最後に次のディレクティブを追加します。

```
local_transport = error: local delivery disabled
```

f. **mynetworks** パラメーターを追加して、Postfix が 127.0.0.0/8 IPv4 ネットワークと [::1]/128 IPv6 ネットワークから送信されたローカルシステムからの電子メールをメールリレーサーバーに転送するようにします。

```
mynetworks = 127.0.0.0/8, [::1]/128
```

2. **main.cf** ファイルの Postfix 設定が正しいか確認します。

```
$ postfix check
```

3. **postfix** サービスを再起動して変更を適用します。

```
# systemctl restart postfix
```

検証

- 電子メール通信がメールリレーに転送されていることを確認します。

```
# echo "This is a test message" | mail -s <SUBJECT> <user@example.com>
```

トラブルシューティング

- エラーが発生した場合は、**/var/log/maillog** を確認してください。

関連情報

- **/etc/postfix/main.cf** 設定ファイル

8.5. POSTFIX を複数のドメインの宛先として設定する

Postfix を、複数のドメインのメールを受信できるメールサーバーとして設定できます。この設定では、Postfix は、指定されたドメイン内のアドレスに送信された電子メールの最終宛先として機能します。以下を設定できます。

- 同じ電子メール宛先を指す複数の電子メールアドレスを設定する。
- 複数のドメインの受信メールを同じ Postfix サーバーにルーティングする。

前提条件

- root アクセスがある。
- Postfix サーバーを設定している。

手順

1. **/etc/postfix/virtual** 仮想エイリアスファイルで、各ドメインのメールアドレスを指定します。各電子メールアドレスを新しい行に追加します。

```
<info@example.com>  <user22@example.net>
<sales@example.com>  <user11@example.org>
```

この例では、Postfix は info@example.com に送信されたすべての電子メールを user22@example.net にリダイレクトし、sales@example.com に送信された電子メールを user11@example.org にリダイレクトします。

2. 仮想エイリアスマップのハッシュファイルを作成します。

```
# postmap /etc/postfix/virtual
```

このコマンドは、**/etc/postfix/virtual.db** ファイルを作成します。**/etc/postfix/virtual** ファイルを更新した後に、このコマンドを常に再実行する必要があります。

3. Postfix **/etc/postfix/main.cf** 設定ファイルで、**virtual_alias_maps** パラメーターを追加して、ハッシュファイルを指すようにします。

```
virtual_alias_maps = hash:/etc/postfix/virtual
```

4. **postfix** サービスをリロードして変更を適用します。

```
# systemctl reload postfix
```

検証

- 仮想メールアドレスの 1 つに電子メールを送信して、設定をテストします。

トラブルシューティング

- エラーが発生した場合は、**/var/log/maillog** を確認してください。

8.6. LDAP ディレクトリーの検索テーブルとしての使用

Lightweight Directory Access Protocol (LDAP) サーバーを使用してアカウント、ドメイン、またはエイリアスを保存する場合は、LDAP サーバーを検索テーブルとして使用するように Postfix を設定できます。検索用ファイルの代わりに LDAP を使用すると、中央データベースを使用できます。

前提条件

- root アクセスがある。
- サーバーに **postfix** パッケージがインストールされている。
- 必要なスキーマおよびユーザークレデンシャルを持つ LDAP サーバーがある。
- Postfix を実行しているサーバーに **postfix-ldap** プラグインがインストールされている。

手順

1. 以下の内容で **/etc/postfix/ldap-aliases.cf** ファイルを作成して、LDAP 検索パラメーターを設定します。

a. LDAP サーバーのホスト名を指定します。

```
server_host = <ldap.example.com>
```

b. LDAP 検索のベースドメイン名を指定します。

```
search_base = dc=<example>,dc=<com>
```

c. オプション: 要件に応じて LDAP 検索フィルターと属性をカスタマイズします。ディレクトリーを検索するフィルターのデフォルトは **query_filter = mailacceptinggeneralid=%s** です。

2. 以下の内容を追加して、LDAP ソースを **/etc/postfix/main.cf** 設定ファイルの検索テーブルとして有効にします。

```
virtual_alias_maps = ldap:/etc/postfix/ldap-aliases.cf
```

3. **postmap** コマンドを実行して LDAP 設定を確認します。これは、構文エラーまたは接続の問題をチェックします。

```
# postmap -q @<example.com> ldap:/etc/postfix/ldap-aliases.cf
```

4. **postfix** サービスをリロードして変更を適用します。

```
# systemctl reload postfix
```

検証

- テストメールを送信して、LDAP 検索が正しく機能していることを確認します。**/var/log/maillog** のメールログでエラーがないか確認します。

関連情報

- **/usr/share/doc/postfix/README_FILES/LDAP_README** ファイル
- **/usr/share/doc/postfix/README_FILES/DATABASE_README** ファイル

8.7. 認証されたユーザーのリレーを行う送信メールサーバーとしての POSTFIX の設定

認証されたユーザーのメールをリレーするように Postfix を設定できます。このシナリオでは、SMTP 認証、TLS 暗号化、および送信者アドレス制限を備えた送信メールサーバーとして Postfix を設定することで、ユーザーが自分自身を認証し、自分の電子メールアドレスを使用して SMTP サーバー経由でメールを送信できるようにします。

前提条件

- root アクセスがある。
- Postfix サーバーを設定している。

手順

1. Postfix を送信メールサーバーとして設定するには、**/etc/postfix/main.cf** ファイルを編集し、以下を追加します。

   a. SMTP 認証を有効にします。

   ```
   smtpd_sasl_auth_enable = yes
   broken_sasl_auth_clients = yes
   ```

   b. TLS を使用しないアクセスを無効にします。

   ```
   smtpd_tls_auth_only = yes
   ```

   c. 認証されたユーザーに対してのみメールリレーを許可します。

   ```
   smtpd_relay_restrictions = permit_mynetworks permit_sasl_authenticated
   defer_unauth_destination
   ```

   d. オプション: ユーザーが自分の電子メールアドレスを送信者としてのみ使用するように制限します。

   ```
   smtpd_sender_restrictions = reject_sender_login_mismatch
   ```

2. **postfix** サービスをリロードして変更を適用します。

   ```
   # systemctl reload postfix
   ```

検証

- TLS および SASL をサポートする SMTP クライアントで認証します。テストメールを送信して、SMTP 認証が正しく機能していることを確認します。

8.8. 同じホストで実行している POSTFIX から DOVECOT への電子メールの配信

UNIX ソケット経由で LMTP を使用して、受信メールを同じホスト上の Dovecot に配信するように Postfix を設定できます。このソケットは、ローカルマシンの Postfix と Dovecot との間の直接通信を有効にします。

前提条件

- root アクセスがある。
- Postfix サーバーを設定している。
- Dovecot サーバーを設定している。Dovecot IMAP および POP3 サーバーの設定と管理 を参照してください。
- Dovecot サーバーに LMTP ソケットを設定している。LMTP ソケットと LMTPS リスナーの設定 を参照してください。

手順

1. **/etc/postfix/main.cf** ファイルの Dovecot にメールを配信するために LMTP プロトコルと UNIX ドメインソケットを使用するように Postfix を設定します。
   - 仮想メールボックスを使用する場合は、次のコンテンツを追加します。

     ```
     virtual_transport = lmtp:unix:/var/run/dovecot/lmtp
     ```

   - 仮想以外のメールボックスを使用する場合は、次のコンテンツを追加します。

     ```
     mailbox_transport = lmtp:unix:/var/run/dovecot/lmtp
     ```

2. **postfix** をリロードして変更を適用します。

   ```
   # systemctl reload postfix
   ```

検証

- テストメールを送信して、LMTP ソケットが正常に動作することを確認します。**/var/log/maillog** のメールログでエラーがないか確認します。

8.9. POSTFIX から別のホストで実行されている DOVECOT への電子メールの配信

ネットワーク経由で Postfix メールサーバーと Dovecot 配信エージェントの間にセキュアな接続を確立できます。これを行うには、メールサーバー間でのメール配信にネットワークソケットを使用するように LMTP サービスを設定します。デフォルトでは、LMTP プロトコルは暗号化されていません。ただし、TLS 暗号化を設定した場合、Dovecot は LMTP サービスに同じ設定を自動的に使用します。続いて、SMTP サーバーは、LMTP 経由で **STARTTLS** コマンドを使用してそれに接続できます。

前提条件

- root アクセスがある。
- Postfix サーバーを設定している。
- Dovecot サーバーを設定している。Dovecot IMAP および POP3 サーバーの設定と管理 を参照してください。
- Dovecot サーバーに LMTP サービスを設定している。LMTP ソケットと LMTPS リスナーの設定 を参照してください。

手順

1. 以下の内容を追加して、**/etc/postfix/main.cf** ファイルで Dovecot にメールを配信するために LMTP プロトコルと INET ドメインソケットを使用するように Postfix を設定します。

   ```
   mailbox_transport = lmtp:inet:<dovecot_host>:<port>
   ```

   <dovecot_host> を Dovecot サーバーの IP アドレスまたはホスト名に置き換え、 **<port>** を LMTP サービスのポート番号に置き換えます。

2. **postfix** サービスをリロードして変更を適用します。

```
# systemctl reload postfix
```

検証

- リモート Dovecot サーバーがホストするアドレスにテストメールを送信し、Dovecot ログをチェックして、メールが正常に配信されたことを確認します。

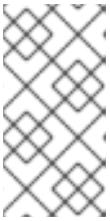
8.10. POSTFIX サービスを保護する

Postfix は、SMTP (Simple Mail Transfer Protocol) を使用して他の MTA 間で電子メッセージを配信したり、クライアントや配信エージェントに電子メールを送信したりするメール転送エージェント (MTA) です。MTA は相互間のトラフィックを暗号化できますが、デフォルトではそうしない場合があります。設定をより安全な値に変更することで、さまざまな攻撃に対するリスクを軽減することもできます。

8.10.1. Postfix ネットワーク関連のセキュリティーリスクの軽減

攻撃者がネットワーク経由でシステムに侵入するリスクを軽減するには、次のタスクをできるだけ多く実行してください。

- ネットワークファイルシステム (NFS) 共有ボリュームで **/var/spool/postfix/** メールスプールディレクトリーを共有しないでください。NFSv2 と NFSv3 は、ユーザー ID とグループ ID に対する制御を維持しません。したがって、2 人以上のユーザーが同じ UID を持っていると、互いのメールを受信して読むことができ、セキュリティー上のリスクが生じます。



注記

SECRPC_GSS カーネルモジュールは UID ベースの認証を使用しないため、この規則は Kerberos を使用する NFSv4 には適用されません。ただし、セキュリティーリスクを軽減するために、メールスプールディレクトリーを NFS 共有ボリュームに配置しないでください。

- Postfix サーバーの悪用の可能性を減らすために、メールユーザーは電子メールプログラムを使用して Postfix サーバーにアクセスする必要があります。メールサーバーでシェルアカウントを許可せず、**/etc/passwd** ファイル内のすべてのユーザーシェルを **/sbin/nologin** に設定します (**root** ユーザーは例外の可能性があります)。

- Postfix をネットワーク攻撃から保護するために、デフォルトではローカルループバックアドレスのみをリッスンするように設定されています。これは、**/etc/postfix/main.cf** ファイルの **inet_interfaces = localhost** 行を表示することで確認できます。これにより、Postfix はネットワークからではなく、ローカルシステムからのメールメッセージ (**cron** ジョブのレポートなど) のみを受け入れるようになります。これはデフォルトの設定で、Postfix をネットワーク攻撃から保護します。localhost の制限を取り除き、Postfix がすべてのインターフェイスでリッスンできるようにするには、**/etc/postfix/main.cf** で **inet_interfaces** パラメーターを **all** に設定します。

8.10.2. DoS 攻撃を制限するための Postfix 設定オプション

攻撃者は、トラフィックでサーバーをあふれさせたり、クラッシュを引き起こす情報を送信したりして、サービス拒否 (DoS) 攻撃を引き起こす可能性があります。**/etc/postfix/main.cf** ファイルで制限を設定することにより、このような攻撃のリスクを軽減するようにシステムを設定できます。既存のディレクティブの値を変更するか、**<directive> = <value>**形式のカスタム値で新しいディレクティブを追加できます。

DoS 攻撃を制限するには、次のディレクティブリストを使用します。

smtpd_client_connection_rate_limit

このディレクティブは、時間単位ごとにクライアントがこのサービスに対して行うことができる接続試行の最大数を制限します。デフォルト値は **0** です。これは、クライアントが時間単位で Postfix が受け入れることができる数と同じ数の接続を行うことができることを意味します。デフォルトでは、ディレクティブは信頼できるネットワークのクライアントを除外します。

anvil_rate_time_unit

このディレクティブは、レート制限を計算する時間単位です。デフォルト値は **60** 秒です。

smtpd_client_event_limit_exceptions

このディレクティブは、接続およびレート制限コマンドからクライアントを除外します。デフォルトでは、ディレクティブは信頼できるネットワークのクライアントを除外します。

smtpd_client_message_rate_limit

このディレクティブは、単位時間当たりのクライアントからリクエストへのメッセージ配信の最大数を定義します (Postfix が実際にそれらのメッセージを受け入れるかどうかに関係なく)。

default_process_limit

このディレクティブは、特定のサービスを提供する Postfix 子プロセスのデフォルトの最大数を定義します。**master.cf** ファイル内の特定のサービスについては、このルールを無視できます。デフォルトでは、値は **100** です。

queue_minfree

このディレクティブは、キューファイルシステムでメールを受信するために必要な空き容量の最小量を定義します。このディレクティブは現在、Postfix SMTP サーバーがメールを受け入れるかどうかを決定するために使用されています。デフォルトでは、Postfix SMTP サーバーは、空き容量が **message_size_limit** の 1.5 倍未満の場合に、**MAIL FROM** コマンドを拒否します。空き容量の最小値をこれよりも高く指定するには、**message_size_limit** の 1.5 倍以上の **queue_minfree** 値を指定します。デフォルトの **queue_minfree** 値は **0** です。

header_size_limit

このディレクティブは、メッセージヘッダーを格納するためのメモリーの最大量をバイト単位で定義します。ヘッダーが大きい場合、余分なヘッダーは破棄されます。デフォルトでは、値は **102400** バイトです。

message_size_limit

このディレクティブは、エンベロープ情報を含むメッセージの最大サイズをバイト単位で定義します。デフォルトでは、値は **10240000** バイトです。

8.10.3. Postfix が SASL を使用する設定

Postfix は Simple Authentication and Security Layer (SASL) ベースの SMTP 認証 (AUTH) をサポートしています。SMTP AUTH は Simple Mail Transfer Protocol の拡張です。現在、Postfix SMTP サーバーは次の方法で SASL 実装をサポートしています:

Dovecot SASL

Postfix SMTP サーバーは、UNIX ドメインソケットまたは TCP ソケットのいずれかを使用して、Dovecot SASL 実装と通信できます。Postfix と Dovecot アプリケーションが別のマシンで実行している場合は、この方法を使用します。

Cyrus SASL

有効にすると、SMTP クライアントは、サーバーとクライアントの両方でサポートおよび受け入れられる認証方法を使用して、SMTP サーバーで認証する必要があります。

前提条件

- **dovecot** パッケージがシステムにインストールされている

手順

1. Dovecot をセットアップします。

   a. **/etc/dovecot/conf.d/10-master.conf** ファイルに次の行を含めます。

   ```
   service auth {
     unix_listener /var/spool/postfix/private/auth {
       mode = 0660
       user = postfix
       group = postfix
     }
   }
   ```

   前の例では、Postfix と Dovecot の間の通信に UNIX ドメインソケットを使用しています。また、**/var/spool/postfix/** ディレクトリーにあるメールキュー、および **postfix** ユーザーとグループの下で実行しているアプリケーションを含む Postfix SMTP サーバーのデフォルト設定を想定しています。

   b. オプション: TCP 経由で Postfix 認証リクエストをリッスンするように Dovecot をセットアップします。

   ```
   service auth {
     inet_listener {
         port = port-number
     }
   }
   ```

   c. **/etc/dovecot/conf.d/10-auth.conf** ファイルの **auth_mechanisms** パラメーターを編集して、電子メールクライアントが Dovecot での認証に使用する方法を指定します。

   ```
   auth_mechanisms = plain login
   ```

   auth_mechanisms パラメーターは、さまざまなプレーンテキストおよび非プレーンテキストの認証方法をサポートしています。

2. **/etc/postfix/main.cf** ファイルを変更して Postfix をセットアップします。

   a. Postfix SMTP サーバーで SMTP 認証を有効にします。

   ```
   smtpd_sasl_auth_enable = yes
   ```

   b. SMTP 認証用の Dovecot SASL 実装の使用を有効にします。

   ```
   smtpd_sasl_type = dovecot
   ```

   c. Postfix キューディレクトリーに相対的な認証パスを指定します。相対パスを使用すると、Postfix サーバーが **chroot** で実行しているかどうかに関係なく、設定が確実に機能することに注意してください。

   ```
   smtpd_sasl_path = private/auth
   ```

この手順では、Postfix と Dovecot の間の通信に UNIX ドメインソケットを使用します。

通信に TCP ソケットを使用する場合に、別のマシンで Dovecot を探すように Postfix を設定するには、次のような設定値を使用します。

```
smtpd_sasl_path = inet: ip-address : port-number
```

前の例で、**ip-address** を Dovecot マシンの IP アドレスに置き換え、 **port-number** を Dovecot の **/etc/dovecot/conf.d/10-master.conf** ファイルで指定されたポート番号に置き換えます。

d. Postfix SMTP サーバーがクライアントに提供する SASL メカニズムを指定します。暗号化されたセッションと暗号化されていないセッションに異なるメカニズムを指定できることに注意してください。

```
smtpd_sasl_security_options = noanonymous, noplaintext
smtpd_sasl_tls_security_options = noanonymous
```

前のディレクティブは、暗号化されていないセッションでは匿名認証が許可されず、暗号化されていないユーザー名またはパスワードを送信するメカニズムが許可されていないことを指定しています。暗号化セッション (TLS を使用) の場合、非匿名認証メカニズムのみが許可されます。

関連情報

- Postfix SMTP server policy - SASL mechanism properties
- Postfix and Dovecot SASL
- Postfix SMTP サーバーで SASL 認証を設定する

第9章 DOVECOT IMAP および POP3 サーバーの設定と管理

Dovecot は、セキュリティーを重視する高パフォーマンスのメール配信エージェント (MDA) です。IMAP または POP3 互換の電子メールクライアントを使用して Dovecot サーバーに接続し、電子メールを読んだりダウンロードしたりできます。

Dovecot の主な機能:

- セキュリティーを重視する設計と実装
- 大規模環境でのパフォーマンスを向上させるために、高可用性を実現する双方向レプリケーションをサポート
- 高パフォーマンスの **dbox** メールボックス形式だけでなく、互換性の理由から **mbox** と **Maildir** もサポート
- 破損したインデックスファイルの修正などの自己修復機能
- IMAP 標準への準拠
- IMAP および POP3 クライアントのバグを回避するための回避策をサポート

9.1. PAM 認証を使用した DOVECOT サーバーのセットアップ

Dovecot は、ユーザーデータベースとして Name Service Switch (NSS) インターフェイスをサポートし、認証バックエンドとして Pluggable Authentication Module (PAM) フレームワークをサポートします。この設定により、Dovecot は、NSS を介してサーバー上でローカルに利用可能なユーザーにサービスを提供できます。

アカウントが次の場合に PAM 認証を使用します。

- **/etc/passwd** ファイルでローカルに定義されている。
- リモートデータベースに保存されているが、System Security Services Daemon (SSSD) またはその他の NSS プラグインを介してローカルで利用できる。

9.1.1. Dovecot のインストール

dovecot パッケージは以下を提供します。

- **dovecot** サービスとそれを管理するユーティリティー
- Dovecot がオンデマンドで開始するサービス (認証など)
- サーバーサイドメールフィルタリングなどのプラグイン
- **/etc/dovecot/** ディレクトリーの設定ファイル
- **/usr/share/doc/dovecot/** ディレクトリーのドキュメント

手順

- **dovecot** パッケージをインストールします。

```
# yum install dovecot
```

注記

Dovecot がすでにインストールされていて、クリーンな設定ファイルが必要な場合は、**/etc/dovecot/** ディレクトリーを名前変更するか削除してください。その後、パッケージを再インストールします。設定ファイルを削除しないと、**yum uninstall dovecot** コマンドは **/etc/dovecot/** 内の設定ファイルをリセットしません。

次のステップ

- Dovecot サーバーでの TLS 暗号化の設定。

9.1.2. Dovecot サーバーでの TLS 暗号化の設定

Dovecot はセキュアなデフォルト設定を提供します。たとえば、TLS はデフォルトで有効になっており、認証情報と暗号化されたデータをネットワーク経由で送信します。Dovecot サーバーで TLS を設定するには、証明書と秘密鍵ファイルへのパスを設定するだけです。さらに、Diffie-Hellman パラメーターを生成して使用し、Perfect Forward Secrecy (PFS) を提供することで、TLS 接続のセキュリティーを強化できます。

前提条件

- Dovecot がインストールされています。
- 次のファイルが、サーバー上のリストされた場所にコピーされました。
  - サーバー証明書: **/etc/pki/dovecot/certs/server.example.com.crt**
  - 秘密鍵: **/etc/pki/dovecot/private/server.example.com.key**
  - 認証局 (CA) 証明書: **/etc/pki/dovecot/certs/ca.crt**
- サーバー証明書の **Subject DN** フィールドのホスト名は、サーバーの完全修飾ドメイン名 (FQDN) と一致します。

手順

1. 秘密鍵ファイルにセキュアな権限を設定します。

```
# chown root:root /etc/pki/dovecot/private/server.example.com.key
# chmod 600 /etc/pki/dovecot/private/server.example.com.key
```

2. Diffie-Hellman パラメーターを使用してファイルを生成します。

```
# openssl dhparam -out /etc/dovecot/dh.pem 4096
```

サーバーのハードウェアとエントロピーによっては、4096 ビットの Diffie-Hellman パラメーターを生成するのに数分かかる場合があります。

3. **/etc/dovecot/conf.d/10-ssl.conf** ファイルで証明書と秘密鍵ファイルへのパスを設定します。
   a. **ssl_cert** および **ssl_key** パラメーターを更新し、サーバーの証明書と秘密鍵へのパスを使用するように設定します。

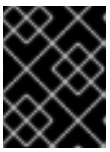
```
ssl_cert = </etc/pki/dovecot/certs/server.example.com.crt
ssl_key = </etc/pki/dovecot/private/server.example.com.key
```

b. **ssl_ca** パラメーターをコメント解除し、CA 証明書へのパスを使用するように設定します。

```
ssl_ca = </etc/pki/dovecot/certs/ca.crt
```

c. **ssl_dh** パラメーターをコメント解除し、Diffie-Hellman パラメーターファイルへのパスを使用するように設定します。

```
ssl_dh = </etc/dovecot/dh.pem
```



重要

Dovecot がファイルからパラメーターの値を確実に読み取るようにするには、パスの先頭に **<** 文字を付ける必要があります。

次のステップ

- 仮想ユーザーを使用するための Dovecot の準備

関連情報

- **/usr/share/doc/dovecot/wiki/SSL.DovecotConfiguration.txt**

9.1.3. 仮想ユーザーを使用するための Dovecot の準備

デフォルトでは、Dovecot はサービスを使用するユーザーとして、ファイルシステム上で多くのアクションを実行します。ただし、1 人のローカルユーザーを使用してこれらのアクションを実行するように Dovecot バックエンドを設定すると、複数の利点があります。

- Dovecot は、ユーザーの ID (UID) を使用する代わりに、特定のローカルユーザーとしてファイルシステムアクションを実行します。
- ユーザーは、サーバー上でローカルに利用できる必要はありません。
- すべてのメールボックスとユーザー固有のファイルを 1 つのルートディレクトリーに保存できます。
- ユーザーは UID とグループ ID (GID) を必要としないため、管理作業が軽減されます。
- サーバー上のファイルシステムにアクセスできるユーザーは、これらのファイルにアクセスできないため、メールボックスやインデックスを危険にさらす可能性はありません。
- レプリケーションのセットアップはより簡単です。

前提条件

- Dovecot がインストールされています。

手順

1. **vmail** ユーザーを作成します。

```
# useradd --home-dir /var/mail/ --shell /usr/sbin/nologin vmail
```

Dovecot は後でこのユーザーを使用してメールボックスを管理します。セキュリティー上の理由から、この目的で **dovecot** または **dovenull** システムユーザーを使用しないでください。

2. **/var/mail/** 以外のパスを使用する場合は、それに SELinux コンテキスト **mail_spool_t** を設定します。例:

```
# semanage fcontext -a -t mail_spool_t "<path>(/.*)?"
# restorecon -Rv <path>
```

3. **/var/mail/** への書き込み権限を **vmail** ユーザーにのみ付与します。

```
# chown vmail:vmail /var/mail/
# chmod 700 /var/mail/
```

4. **/etc/dovecot/conf.d/10-mail.conf** ファイルの **mail_location** パラメーターをコメント解除し、メールボックスの形式と場所を設定します。

```
mail_location = sdbox:/var/mail/%n/
```

この設定の場合:

- Dovecot は、**single** モードで高パフォーマンスの **dbox** メールボックス形式を使用します。このモードでは、サービスは、**maildir** 形式と同様に、各メールを個別のファイルに保存します。
- Dovecot はパス内の **%n** 変数をユーザー名に解決します。これは、各ユーザーがメールボックス用に個別のディレクトリーを持つようにするために必要です。

次のステップ

- PAM を Dovecot 認証バックエンドとして使用する。

関連情報

- **/usr/share/doc/dovecot/wiki/VirtualUsers.txt**
- **/usr/share/doc/dovecot/wiki/MailLocation.txt**
- **/usr/share/doc/dovecot/wiki/MailboxFormat.dbox.txt**
- **/usr/share/doc/dovecot/wiki/Variables.txt**

9.1.4. PAM を Dovecot 認証バックエンドとして使用する

デフォルトでは、Dovecot は Name Service Switch (NSS) インターフェイスをユーザーデータベースとして使用し、Pluggable Authentication Module (PAM) フレームワークを認証バックエンドとして使用します。

設定をカスタマイズして Dovecot を環境に適応させ、仮想ユーザー機能を使用して管理を簡素化します。

前提条件

- Dovecot がインストールされています。
- 仮想ユーザー機能が設定されています。

手順

1. **/etc/dovecot/conf.d/10-mail.conf** ファイルの **first_valid_uid** パラメーターを更新して、Dovecot に対して認証できる最小のユーザー ID (UID) を定義します。

```
first_valid_uid = 1000
```

デフォルトでは、**1000** 以上の UID を持つユーザーが認証を受けることができます。必要に応じて、**last_valid_uid** パラメーターを設定して、Dovecot がログインを許可する最大の UID を定義することもできます。

2. **/etc/dovecot/conf.d/auth-system.conf.ext** ファイルで、次のように **override_fields** パラメーターを **userdb** セクションに追加します。

```
userdb {
  driver = passwd
  override_fields = uid=vmail gid=vmail home=/var/mail/%n/
}
```

固定値のため、Dovecot は **/etc/passwd** ファイルからこれらの設定をクエリーしません。そのため、**/etc/passwd** に定義されたホームディレクトリーが存在する必要はありません。

次のステップ

- Dovecot 設定を完了します。

関連情報

- **/usr/share/doc/dovecot/wiki/PasswordDatabase.PAM.txt**
- **/usr/share/doc/dovecot/wiki/VirtualUsers.Home.txt**

9.1.5. Dovecot 設定の完了

Dovecot をインストールして設定したら、**firewalld** サービスで必要なポートを開き、サービスを有効にして開始します。その後、サーバーをテストできます。

前提条件

- 以下は Dovecot で設定されています。
  - TLS 暗号化
  - 認証バックエンド
- クライアントは認証局 (CA) 証明書を信頼します。

手順

1. IMAP または POP3 サービスのみをユーザーに提供する場合は、**/etc/dovecot/dovecot.conf** ファイルの **protocol** パラメーターをコメント解除し、必要なプロトコルに設定します。たとえば、POP3 を必要としない場合は、次のように設定します。

```
protocols = imap lmtp
```

   デフォルトでは、**imap**、**pop3**、および **lmtp** プロトコルが有効になっています。

2. ローカルファイアウォールでポートを開きます。たとえば、IMAPS、IMAP、POP3S、および POP3 プロトコルのポートを開くには、次のように入力します。

```
# firewall-cmd --permanent --add-service=imaps --add-service=imap --add-
service=pop3s --add-service=pop3
# firewall-cmd --reload
```

3. **dovecot** サービスを有効にして開始します。

```
# systemctl enable --now dovecot
```

検証

1. Dovecot に接続して電子メールを読むには、Mozilla Thunderbird などのメールクライアントを使用します。メールクライアントの設定は、使用するプロトコルによって異なります。

   表9.1 Dovecot サーバーへの接続設定

プロトコル	ポート	接続セキュリティー	認証方法
IMAP	143	STARTTLS	PLAIN[a]
IMAPS	993	SSL/TLS	PLAIN[a]
POP3	110	STARTTLS	PLAIN[a]
POP3S	995	SSL/TLS	PLAIN[a]

[a] クライアントは、TLS 接続を介して暗号化されたデータを送信します。したがって、認証情報は開示されません。

   デフォルトでは、Dovecot は TLS を使用しない接続ではプレーンテキスト認証を受け入れないため、この表には暗号化されていない接続の設定がリストされていないことに注意してください。

2. デフォルト以外の値を含む設定を表示します。

```
# doveconf -n
```

関連情報

- **firewall-cmd(1)** man ページ

9.2. LDAP 認証を使用した DOVECOT サーバーのセットアップ

インフラストラクチャーが LDAP サーバーを使用してアカウントを保存している場合、それに対して Dovecot ユーザーを認証できます。この場合、アカウントをディレクトリーで集中管理するため、ユーザーは Dovecot サーバー上のファイルシステムにローカルでアクセスする必要はありません。

複数の Dovecot サーバーをレプリケーションでセットアップして、メールボックスを高可用性にする予定がある場合にも、集中管理されたアカウントは利点があります。

9.2.1. Dovecot のインストール

dovecot パッケージは以下を提供します。

- **dovecot** サービスとそれを管理するユーティリティー
- Dovecot がオンデマンドで開始するサービス (認証など)
- サーバーサイドメールフィルタリングなどのプラグイン
- **/etc/dovecot/** ディレクトリーの設定ファイル
- **/usr/share/doc/dovecot/** ディレクトリーのドキュメント

手順

- **dovecot** パッケージをインストールします。

```
# yum install dovecot
```

注記

Dovecot がすでにインストールされていて、クリーンな設定ファイルが必要な場合は、**/etc/dovecot/** ディレクトリーを名前変更するか削除してください。その後、パッケージを再インストールします。設定ファイルを削除しないと、**yum uninstall dovecot** コマンドは **/etc/dovecot/** 内の設定ファイルをリセットしません。

次のステップ

- Dovecot サーバーでの TLS 暗号化の設定。

9.2.2. Dovecot サーバーでの TLS 暗号化の設定

Dovecot はセキュアなデフォルト設定を提供します。たとえば、TLS はデフォルトで有効になっており、認証情報と暗号化されたデータをネットワーク経由で送信します。Dovecot サーバーで TLS を設定するには、証明書と秘密鍵ファイルへのパスを設定するだけです。さらに、Diffie-Hellman パラメーターを生成して使用し、Perfect Forward Secrecy (PFS) を提供することで、TLS 接続のセキュリティーを強化できます。

前提条件

- Dovecot がインストールされています。
- 次のファイルが、サーバー上のリストされた場所にコピーされました。

- サーバー証明書: **/etc/pki/dovecot/certs/server.example.com.crt**
- 秘密鍵: **/etc/pki/dovecot/private/server.example.com.key**
- 認証局 (CA) 証明書: **/etc/pki/dovecot/certs/ca.crt**

- サーバー証明書の **Subject DN** フィールドのホスト名は、サーバーの完全修飾ドメイン名 (FQDN) と一致します。

手順

1. 秘密鍵ファイルにセキュアな権限を設定します。

```
# chown root:root /etc/pki/dovecot/private/server.example.com.key
# chmod 600 /etc/pki/dovecot/private/server.example.com.key
```

2. Diffie-Hellman パラメーターを使用してファイルを生成します。

```
# openssl dhparam -out /etc/dovecot/dh.pem 4096
```

サーバーのハードウェアとエントロピーによっては、4096 ビットの Diffie-Hellman パラメーターを生成するのに数分かかる場合があります。

3. **/etc/dovecot/conf.d/10-ssl.conf** ファイルで証明書と秘密鍵ファイルへのパスを設定します。

a. **ssl_cert** および **ssl_key** パラメーターを更新し、サーバーの証明書と秘密鍵へのパスを使用するように設定します。

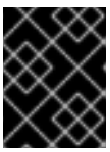
```
ssl_cert = </etc/pki/dovecot/certs/server.example.com.crt
ssl_key = </etc/pki/dovecot/private/server.example.com.key
```

b. **ssl_ca** パラメーターをコメント解除し、CA 証明書へのパスを使用するように設定します。

```
ssl_ca = </etc/pki/dovecot/certs/ca.crt
```

c. **ssl_dh** パラメーターをコメント解除し、Diffie-Hellman パラメーターファイルへのパスを使用するように設定します。

```
ssl_dh = </etc/dovecot/dh.pem
```



重要

Dovecot がファイルからパラメーターの値を確実に読み取るようにするには、パスの先頭に **<** 文字を付ける必要があります。

次のステップ

- 仮想ユーザーを使用するための Dovecot の準備

関連情報

- **/usr/share/doc/dovecot/wiki/SSL.DovecotConfiguration.txt**

9.2.3. 仮想ユーザーを使用するための Dovecot の準備

デフォルトでは、Dovecot はサービスを使用するユーザーとして、ファイルシステム上で多くのアクションを実行します。ただし、1 人のローカルユーザーを使用してこれらのアクションを実行するように Dovecot バックエンドを設定すると、複数の利点があります。

- Dovecot は、ユーザーの ID (UID) を使用する代わりに、特定のローカルユーザーとしてファイルシステムアクションを実行します。
- ユーザーは、サーバー上でローカルに利用できる必要はありません。
- すべてのメールボックスとユーザー固有のファイルを 1 つのルートディレクトリーに保存できます。
- ユーザーは UID とグループ ID (GID) を必要としないため、管理作業が軽減されます。
- サーバー上のファイルシステムにアクセスできるユーザーは、これらのファイルにアクセスできないため、メールボックスやインデックスを危険にさらす可能性はありません。
- レプリケーションのセットアップはより簡単です。

前提条件

- Dovecot がインストールされています。

手順

1. **vmail** ユーザーを作成します。

```
# useradd --home-dir /var/mail/ --shell /usr/sbin/nologin vmail
```

   Dovecot は後でこのユーザーを使用してメールボックスを管理します。セキュリティー上の理由から、この目的で **dovecot** または **dovenull** システムユーザーを使用しないでください。

2. **/var/mail/** 以外のパスを使用する場合は、それに SELinux コンテキスト **mail_spool_t** を設定します。例:

```
# semanage fcontext -a -t mail_spool_t "<path>(/.*)?"
# restorecon -Rv <path>
```

3. **/var/mail/** への書き込み権限を **vmail** ユーザーにのみ付与します。

```
# chown vmail:vmail /var/mail/
# chmod 700 /var/mail/
```

4. **/etc/dovecot/conf.d/10-mail.conf** ファイルの **mail_location** パラメーターをコメント解除し、メールボックスの形式と場所を設定します。

```
mail_location = sdbox:/var/mail/%n/
```

   この設定の場合:

   - Dovecot は、**single** モードで高パフォーマンスの **dbox** メールボックス形式を使用します。このモードでは、サービスは、**maildir** 形式と同様に、各メールを個別のファイルに保存します。

- Dovecot はパス内の **%n** 変数をユーザー名に解決します。これは、各ユーザーがメールボックス用に個別のディレクトリーを持つようにするために必要です。

次のステップ

- LDAP を Dovecot 認証バックエンドとして使用する。

関連情報

- **/usr/share/doc/dovecot/wiki/VirtualUsers.txt**
- **/usr/share/doc/dovecot/wiki/MailLocation.txt**
- **/usr/share/doc/dovecot/wiki/MailboxFormat.dbox.txt**
- **/usr/share/doc/dovecot/wiki/Variables.txt**

9.2.4. LDAP を Dovecot 認証バックエンドとして使用する

通常、LDAP ディレクトリー内のユーザーは、ディレクトリーサービスに対して自分自身を認証できます。Dovecot は、これを使用して、ユーザーが IMAP または POP3 サービスにログインする場合、ユーザーを認証できます。この認証方法には、次のとおり、多くの利点があります。

- 管理者は、ディレクトリーでユーザーを集中管理できます。
- LDAP アカウントには、特別な属性は必要ありません。LDAP サーバーから認証を受けることができれば十分です。したがって、この方法は、LDAP サーバーで使用されるパスワード保存方式とは無関係です。
- ユーザーは、Name Service Switch (NSS) インターフェイスと Pluggable Authentication Module (PAM) フレームワークを介して、サーバー上でローカルに利用できる必要はありません。

前提条件

- Dovecot がインストールされています。
- 仮想ユーザー機能が設定されています。
- LDAP サーバーへの接続は、TLS 暗号化をサポートします。
- Dovecot サーバー上の RHEL は、LDAP サーバーの認証局 (CA) 証明書を信頼します。
- ユーザーが LDAP ディレクトリーの異なるツリーに保存されている場合、ディレクトリーを検索するための Dovecot 専用の LDAP アカウントが存在します。このアカウントには、他のユーザーの識別名 (DN) を検索する権限が必要です。

手順

1. **/etc/dovecot/conf.d/10-auth.conf** ファイルで認証バックエンドを設定します。
   a. 不要な **auth-*.conf.ext** 認証バックエンド設定ファイルの **include** ステートメントをコメントアウトします。次に例を示します。

```
#!include auth-system.conf.ext
```

b. 次の行をコメント解除して、LDAP 認証を有効にします。

```
!include auth-ldap.conf.ext
```

2. **/etc/dovecot/conf.d/auth-ldap.conf.ext** ファイルを編集し、次のように **override_fields** パラメーターを **userdb** セクションに追加します。

```
userdb {
  driver = ldap
  args = /etc/dovecot/dovecot-ldap.conf.ext
  override_fields = uid=vmail gid=vmail home=/var/mail/%n/
}
```

固定値のため、Dovecot は LDAP サーバーからこれらの設定をクエリーしません。したがって、これらの属性も存在する必要はありません。

3. 次の設定で **/etc/dovecot/dovecot-ldap.conf.ext** ファイルを作成します。

a. LDAP 構造に応じて、次のいずれかを設定します。

- ユーザーが LDAP ディレクトリーの異なるツリーに保存されている場合は、動的 DN 検索を設定します。

```
dn = cn=dovecot_LDAP,dc=example,dc=com
dnpass = password
pass_filter = (&(objectClass=posixAccount)(uid=%n))
```

Dovecot は、指定された DN、パスワード、およびフィルターを使用して、ディレクトリー内の認証ユーザーの DN を検索します。この検索では、Dovecot はフィルター内の **%n** をユーザー名に置き換えます。LDAP 検索で返される結果は 1 つだけであることに注意してください。

- すべてのユーザーが特定のエントリーに保存されている場合は、DN テンプレートを設定します。

```
auth_bind_userdn = cn=%n,ou=People,dc=example,dc=com
```

b. LDAP サーバーへの認証バインドを有効にして、Dovecot ユーザーを確認します。

```
auth_bind = yes
```

c. URL を LDAP サーバーに設定します。

```
uris = ldaps://LDAP-srv.example.com
```

セキュリティー上の理由から、LDAP プロトコル上で LDAPS または **STARTTLS** コマンドを使用した暗号化された接続のみを使用してください。後者の場合は、さらに **tls = yes** を設定に追加します。

証明書の検証を機能させるには、LDAP サーバーのホスト名が TLS 証明書で使用されているホスト名と一致する必要があります。

d. LDAP サーバーの TLS 証明書の検証を有効にします。

```
tls_require_cert = hard
```

e. ベース DN には、ユーザーの検索を開始する DN を設定します。

```
base = ou=People,dc=example,dc=com
```

f. 検索範囲を設定します。

```
scope = onelevel
```

Dovecot は、指定されたベース DN のみを **onelevel** スコープで検索し、サブツリーも **subtree** スコープで検索します。

4. **/etc/dovecot/dovecot-ldap.conf.ext** ファイルにセキュアな権限を設定します。

```
# chown root:root /etc/dovecot/dovecot-ldap.conf.ext
# chmod 600 /etc/dovecot/dovecot-ldap.conf.ext
```

次のステップ

- Dovecot 設定を完了します。

関連情報

- **/usr/share/doc/dovecot/example-config/dovecot-ldap.conf.ext**
- **/usr/share/doc/dovecot/wiki/UserDatabase.Static.txt**
- **/usr/share/doc/dovecot/wiki/AuthDatabase.LDAP.txt**
- **/usr/share/doc/dovecot/wiki/AuthDatabase.LDAP.AuthBinds.txt**
- **/usr/share/doc/dovecot/wiki/AuthDatabase.LDAP.PasswordLookups.txt**

9.2.5. Dovecot 設定の完了

Dovecot をインストールして設定したら、**firewalld** サービスで必要なポートを開き、サービスを有効にして開始します。その後、サーバーをテストできます。

前提条件

- 以下は Dovecot で設定されています。
  - TLS 暗号化
  - 認証バックエンド
- クライアントは認証局 (CA) 証明書を信頼します。

手順

1. IMAP または POP3 サービスのみをユーザーに提供する場合は、**/etc/dovecot/dovecot.conf** ファイルの **protocol** パラメーターをコメント解除し、必要なプロトコルに設定します。たとえば、POP3 を必要としない場合は、次のように設定します。

```
protocols = imap lmtp
```

デフォルトでは、**imap**、**pop3**、および **lmtp** プロトコルが有効になっています。

2. ローカルファイアウォールでポートを開きます。たとえば、IMAPS、IMAP、POP3S、および POP3 プロトコルのポートを開くには、次のように入力します。

```
# firewall-cmd --permanent --add-service=imaps --add-service=imap --add-
service=pop3s --add-service=pop3
# firewall-cmd --reload
```

3. **dovecot** サービスを有効にして開始します。

```
# systemctl enable --now dovecot
```

検証

1. Dovecot に接続して電子メールを読むには、Mozilla Thunderbird などのメールクライアントを使用します。メールクライアントの設定は、使用するプロトコルによって異なります。

表9.2 Dovecot サーバーへの接続設定

プロトコル	ポート	接続セキュリティー	認証方法
IMAP	143	STARTTLS	PLAIN[a]
IMAPS	993	SSL/TLS	PLAIN[a]
POP3	110	STARTTLS	PLAIN[a]
POP3S	995	SSL/TLS	PLAIN[a]

[a] クライアントは、TLS 接続を介して暗号化されたデータを送信します。したがって、認証情報は開示されません。

デフォルトでは、Dovecot は TLS を使用しない接続ではプレーンテキスト認証を受け入れないため、この表には暗号化されていない接続の設定がリストされていないことに注意してください。

2. デフォルト以外の値を含む設定を表示します。

```
# doveconf -n
```

関連情報

- **firewall-cmd(1)** man ページ

9.3. MARIADB SQL 認証を使用した DOVECOT サーバーのセットアップ

ユーザーとパスワードを MariaDB SQL サーバーに保存する場合、それをユーザーデータベースと認証バックエンドとして使用するように、Dovecot を設定できます。この設定では、アカウントをデータベースで集中管理するため、ユーザーは Dovecot サーバー上のファイルシステムにローカルアクセスできません。

複数の Dovecot サーバーをレプリケーションでセットアップして、メールボックスを高可用性にする予定がある場合にも、集中管理されたアカウントは利点があります。

9.3.1. Dovecot のインストール

dovecot パッケージは以下を提供します。

- **dovecot** サービスとそれを管理するユーティリティー
- Dovecot がオンデマンドで開始するサービス (認証など)
- サーバーサイドメールフィルタリングなどのプラグイン
- **/etc/dovecot/** ディレクトリーの設定ファイル
- **/usr/share/doc/dovecot/** ディレクトリーのドキュメント

手順

- **dovecot** パッケージをインストールします。

```
# yum install dovecot
```

注記

Dovecot がすでにインストールされていて、クリーンな設定ファイルが必要な場合は、**/etc/dovecot/** ディレクトリーを名前変更するか削除してください。その後、パッケージを再インストールします。設定ファイルを削除しないと、**yum uninstall dovecot** コマンドは **/etc/dovecot/** 内の設定ファイルをリセットしません。

次のステップ

- Dovecot サーバーでの TLS 暗号化の設定。

9.3.2. Dovecot サーバーでの TLS 暗号化の設定

Dovecot はセキュアなデフォルト設定を提供します。たとえば、TLS はデフォルトで有効になっており、認証情報と暗号化されたデータをネットワーク経由で送信します。Dovecot サーバーで TLS を設定するには、証明書と秘密鍵ファイルへのパスを設定するだけです。さらに、Diffie-Hellman パラメーターを生成して使用し、Perfect Forward Secrecy (PFS) を提供することで、TLS 接続のセキュリティーを強化できます。

前提条件

- Dovecot がインストールされています。
- 次のファイルが、サーバー上のリストされた場所にコピーされました。
  - サーバー証明書: **/etc/pki/dovecot/certs/server.example.com.crt**

- 秘密鍵: **/etc/pki/dovecot/private/server.example.com.key**
- 認証局 (CA) 証明書: **/etc/pki/dovecot/certs/ca.crt**

- サーバー証明書の **Subject DN** フィールドのホスト名は、サーバーの完全修飾ドメイン名 (FQDN) と一致します。

手順

1. 秘密鍵ファイルにセキュアな権限を設定します。

```
# chown root:root /etc/pki/dovecot/private/server.example.com.key
# chmod 600 /etc/pki/dovecot/private/server.example.com.key
```

2. Diffie-Hellman パラメーターを使用してファイルを生成します。

```
# openssl dhparam -out /etc/dovecot/dh.pem 4096
```

サーバーのハードウェアとエントロピーによっては、4096 ビットの Diffie-Hellman パラメーターを生成するのに数分かかる場合があります。

3. **/etc/dovecot/conf.d/10-ssl.conf** ファイルで証明書と秘密鍵ファイルへのパスを設定します。

a. **ssl_cert** および **ssl_key** パラメーターを更新し、サーバーの証明書と秘密鍵へのパスを使用するように設定します。

```
ssl_cert = </etc/pki/dovecot/certs/server.example.com.crt
ssl_key = </etc/pki/dovecot/private/server.example.com.key
```

b. **ssl_ca** パラメーターをコメント解除し、CA 証明書へのパスを使用するように設定します。

```
ssl_ca = </etc/pki/dovecot/certs/ca.crt
```

c. **ssl_dh** パラメーターをコメント解除し、Diffie-Hellman パラメーターファイルへのパスを使用するように設定します。

```
ssl_dh = </etc/dovecot/dh.pem
```

重要

Dovecot がファイルからパラメーターの値を確実に読み取るようにするには、パスの先頭に **<** 文字を付ける必要があります。

次のステップ

- 仮想ユーザーを使用するための Dovecot の準備

関連情報

- **/usr/share/doc/dovecot/wiki/SSL.DovecotConfiguration.txt**

9.3.3. 仮想ユーザーを使用するための Dovecot の準備

デフォルトでは、Dovecot はサービスを使用するユーザーとして、ファイルシステム上で多くのアクションを実行します。ただし、1 人のローカルユーザーを使用してこれらのアクションを実行するように Dovecot バックエンドを設定すると、複数の利点があります。

- Dovecot は、ユーザーの ID (UID) を使用する代わりに、特定のローカルユーザーとしてファイルシステムアクションを実行します。
- ユーザーは、サーバー上でローカルに利用できる必要はありません。
- すべてのメールボックスとユーザー固有のファイルを 1 つのルートディレクトリーに保存できます。
- ユーザーは UID とグループ ID (GID) を必要としないため、管理作業が軽減されます。
- サーバー上のファイルシステムにアクセスできるユーザーは、これらのファイルにアクセスできないため、メールボックスやインデックスを危険にさらす可能性はありません。
- レプリケーションのセットアップはより簡単です。

前提条件

- Dovecot がインストールされています。

手順

1. **vmail** ユーザーを作成します。

```
# useradd --home-dir /var/mail/ --shell /usr/sbin/nologin vmail
```

   Dovecot は後でこのユーザーを使用してメールボックスを管理します。セキュリティー上の理由から、この目的で **dovecot** または **dovenull** システムユーザーを使用しないでください。

2. **/var/mail/** 以外のパスを使用する場合は、それに SELinux コンテキスト **mail_spool_t** を設定します。例:

```
# semanage fcontext -a -t mail_spool_t "<path>(/.*)?"
# restorecon -Rv <path>
```

3. **/var/mail/** への書き込み権限を **vmail** ユーザーにのみ付与します。

```
# chown vmail:vmail /var/mail/
# chmod 700 /var/mail/
```

4. **/etc/dovecot/conf.d/10-mail.conf** ファイルの **mail_location** パラメーターをコメント解除し、メールボックスの形式と場所を設定します。

```
mail_location = sdbox:/var/mail/%n/
```

   この設定の場合:

   - Dovecot は、**single** モードで高パフォーマンスの **dbox** メールボックス形式を使用します。このモードでは、サービスは、**maildir** 形式と同様に、各メールを個別のファイルに保存します。

- Dovecot はパス内の **%n** 変数をユーザー名に解決します。これは、各ユーザーがメールボックス用に個別のディレクトリーを持つようにするために必要です。

次のステップ

- Dovecot 認証バックエンドとして MariaDB SQL データベースを使用する

関連情報

- **/usr/share/doc/dovecot/wiki/VirtualUsers.txt**
- **/usr/share/doc/dovecot/wiki/MailLocation.txt**
- **/usr/share/doc/dovecot/wiki/MailboxFormat.dbox.txt**
- **/usr/share/doc/dovecot/wiki/Variables.txt**

9.3.4. Dovecot 認証バックエンドとして MariaDB SQL データベースを使用する

Dovecot は、MariaDB データベースからアカウントとパスワードを読み取り、これを使用して、ユーザーが IMAP または POP3 サービスにログインする場合、ユーザーを認証できます。この認証方法の利点は次のとおりです。

- 管理者は、データベースでユーザーを集中管理できます。
- ユーザーはサーバー上でローカルにアクセスできません。

前提条件

- Dovecot がインストールされています。
- 仮想ユーザー機能が設定されています。
- MariaDB サーバーへの接続では、TLS 暗号化がサポートされます。
- **dovecotDB** データベースは MariaDB に存在し、 **users** テーブルには、少なくとも **username** および **password** 列が含まれています。
- **password** 列には、Dovecot がサポートするスキームで暗号化されたパスワードが含まれています。
- パスワードは、同じスキームを使用するか、**{pw-storage-scheme}** 接頭辞を使用します。
- MariaDB ユーザー **dovecot** は、**dovecotDB** データベースの **users** テーブルに対する読み取り権限を持っています。
- MariaDB サーバーの TLS 証明書を発行した認証局 (CA) の証明書は、Dovecot サーバーの **/etc/pki/tls/certs/ca.crt** ファイルに保存されます。

手順

1. **dovecot-mysql** パッケージをインストールします。

```
# yum install dovecot-mysql
```

2. **/etc/dovecot/conf.d/10-auth.conf** ファイルで認証バックエンドを設定します。

a. 不要な **auth-*.conf.ext** 認証バックエンド設定ファイルの **include** ステートメントをコメントアウトします。次に例を示します。

```
#!include auth-system.conf.ext
```

b. 次の行をコメント解除して、SQL 認証を有効にします。

```
!include auth-sql.conf.ext
```

3. **/etc/dovecot/conf.d/auth-sql.conf.ext** ファイルを編集し、**override_fields** パラメーターを **userdb** セクションに次のように追加します。

```
userdb {
  driver = sql
  args = /etc/dovecot/dovecot-sql.conf.ext
  override_fields = uid=vmail gid=vmail home=/var/mail/%n/
}
```

固定値のため、Dovecot はこれらの設定を SQL サーバーからクエリーしません。

4. 次の設定で **/etc/dovecot/dovecot-sql.conf.ext** ファイルを作成します。

```
driver = mysql
connect = host=mariadb_srv.example.com dbname=dovecotDB user=dovecot
password=dovecotPW ssl_ca=/etc/pki/tls/certs/ca.crt
default_pass_scheme = SHA512-CRYPT
user_query = SELECT username FROM users WHERE username='%u';
password_query = SELECT username AS user, password FROM users WHERE
username='%u';
iterate_query = SELECT username FROM users;
```

データベースサーバーに対して TLS 暗号化を使用するには、**ssl_ca** オプションに MariaDB サーバー証明書を発行した CA の証明書のパスを設定します。証明書の検証を機能させるには、MariaDB サーバーのホスト名が TLS 証明書で使用されているホスト名と一致する必要があります。

データベースのパスワード値に **{pw-storage-scheme}** 接頭辞が含まれている場合は、**default_pass_scheme** 設定を省略できます。

ファイル内のクエリーは、次のように設定する必要があります。

- **user_query** パラメーターの場合、クエリーは Dovecot ユーザーのユーザー名を返す必要があります。また、クエリーは 1 つの結果のみを返す必要があります。
- **password_query** パラメーターの場合、クエリーはユーザー名とパスワードを返す必要があり、Dovecot は **user** および **password** 変数でこれらの値を使用する必要があります。したがって、データベースが異なる列名を使用している場合は、**AS** SQL コマンドを使用して、結果の列の名前を変更してください。
- **iterate_query** パラメーターの場合、クエリーはすべてのユーザーのリストを返す必要があります。

5. **/etc/dovecot/dovecot-sql.conf.ext** ファイルにセキュアな権限を設定します。

```
# chown root:root /etc/dovecot/dovecot-sql.conf.ext
# chmod 600 /etc/dovecot/dovecot-sql.conf.ext
```

次のステップ

- Dovecot 設定を完了します。

関連情報

- **/usr/share/doc/dovecot/example-config/dovecot-sql.conf.ext**
- **/usr/share/doc/dovecot/wiki/Authentication.PasswordSchemes.txt**

9.3.5. Dovecot 設定の完了

Dovecot をインストールして設定したら、**firewalld** サービスで必要なポートを開き、サービスを有効にして開始します。その後、サーバーをテストできます。

前提条件

- 以下は Dovecot で設定されています。
  - TLS 暗号化
  - 認証バックエンド
- クライアントは認証局 (CA) 証明書を信頼します。

手順

1. IMAP または POP3 サービスのみをユーザーに提供する場合は、**/etc/dovecot/dovecot.conf** ファイルの **protocol** パラメーターをコメント解除し、必要なプロトコルに設定します。たとえば、POP3 を必要としない場合は、次のように設定します。

   ```
   protocols = imap lmtp
   ```

   デフォルトでは、**imap**、**pop3**、および **lmtp** プロトコルが有効になっています。

2. ローカルファイアウォールでポートを開きます。たとえば、IMAPS、IMAP、POP3S、および POP3 プロトコルのポートを開くには、次のように入力します。

   ```
   # firewall-cmd --permanent --add-service=imaps --add-service=imap --add-
   service=pop3s --add-service=pop3
   # firewall-cmd --reload
   ```

3. **dovecot** サービスを有効にして開始します。

   ```
   # systemctl enable --now dovecot
   ```

検証

1. Dovecot に接続して電子メールを読むには、Mozilla Thunderbird などのメールクライアントを使用します。メールクライアントの設定は、使用するプロトコルによって異なります。

表9.3 Dovecot サーバーへの接続設定

プロトコル	ポート	接続セキュリティー	認証方法
IMAP	143	STARTTLS	PLAIN[a]
IMAPS	993	SSL/TLS	PLAIN[a]
POP3	110	STARTTLS	PLAIN[a]
POP3S	995	SSL/TLS	PLAIN[a]

[a] クライアントは、TLS 接続を介して暗号化されたデータを送信します。したがって、認証情報は開示されません。

デフォルトでは、Dovecot は TLS を使用しない接続ではプレーンテキスト認証を受け入れないため、この表には暗号化されていない接続の設定がリストされていないことに注意してください。

2. デフォルト以外の値を含む設定を表示します。

```
# doveconf -n
```

関連情報

- **firewall-cmd(1)** man ページ

9.4. 2 つの DOVECOT サーバー間のレプリケーションの設定

双方向のレプリケーションを使用すると、Dovecot サーバーを高可用性にすることができ、IMAP および POP3 クライアントは両方のサーバーのメールボックスにアクセスできます。Dovecot は、各メールボックスのインデックスログの変更を追跡し、競合を安全な方法で解決します。

両方の複製パートナーでこの手順を実行します。

注記

レプリケーションは、サーバーペア間でのみ機能します。したがって、大規模なクラスターでは、複数の独立したバックエンドペアが必要になります。

前提条件

- 両方のサーバーが同じ認証バックエンドを使用します。できれば、LDAP または SQL を使用して、アカウントを集中管理してください。
- Dovecot ユーザーデータベース設定は、ユーザーリストをサポートします。これを確認するには、**doveadm user '*'** コマンドを使用します。
- Dovecot は、ユーザー ID (UID) ではなく、**vmail** ユーザーとしてファイルシステム上のメールボックスにアクセスします。

手順

1. **/etc/dovecot/conf.d/10-replication.conf** ファイルを作成し、その中で次の手順を実行します。

   a. **notify** および **replication** プラグインを有効にします。

   ```
   mail_plugins = $mail_plugins notify replication
   ```

   b. **service replicator** セクションを追加します。

   ```
   service replicator {
    process_min_avail = 1

    unix_listener replicator-doveadm {
     mode = 0600
     user = vmail
    }
   }
   ```

   これらの設定により、**dovecot** サービスの開始時に、Dovecot は 1 つ以上のレプリケータープロセスを開始します。さらに、このセクションは **replicator-doveadm** ソケットの設定を定義します。

   c. **service aggregator** セクションを追加して、**replication-notify-fifo** パイプと **replication-notify** ソケットを設定します。

   ```
   service aggregator {
    fifo_listener replication-notify-fifo {
     user = vmail
    }
    unix_listener replication-notify {
     user = vmail
    }
   }
   ```

   d. **service doveadm** セクションを追加して、レプリケーションサービスのポートを定義します。

   ```
   service doveadm {
    inet_listener {
     port = 12345
    }
   }
   ```

   e. **doveadm** レプリケーションサービスのパスワードを設定します。

   ```
   doveadm_password = replication_password
   ```

   パスワードは、両方のサーバーで同じにする必要があります。

   f. レプリケーションパートナーを設定します。

   ```
   plugin {
    mail_replica = tcp:server2.example.com:12345
   }
   ```

-

g. オプション: 並列 **dsync** プロセスの最大数を定義します。

```
replication_max_conns = 20
```

replication_max_conns のデフォルト値は **10** です。

2. **/etc/dovecot/conf.d/10-replication.conf** ファイルにセキュアな権限を設定します。

```
# chown root:root /etc/dovecot/conf.d/10-replication.conf
# chmod 600 /etc/dovecot/conf.d/10-replication.conf
```

3. Dovecot が **doveadm** レプリケーションポートを開くことができるように、SELinux ブール値 **nis_enabled** を有効にします。

```
setsebool -P nis_enabled on
```

4. レプリケーションパートナーのみがレプリケーションポートにアクセスできるように、**firewalld** ルールを設定します。次に例を示します。

```
# firewall-cmd --permanent --zone=public --add-rich-rule="rule family="ipv4" source
address="192.0.2.1/32" port protocol="tcp" port="12345" accept"
# firewall-cmd --permanent --zone=public --add-rich-rule="rule family="ipv6" source
address="2001:db8:2::1/128" port protocol="tcp" port="12345" accept"
# firewall-cmd --reload
```

IPv4 アドレスのサブネットマスク **/32** と IPv6 アドレスのサブネットマスク **/128** は、指定されたアドレスへのアクセスを制限します。

5. この手順は、他のレプリケーションパートナーでも実行します。

6. Dovecot をリロードします。

```
# systemctl reload dovecot
```

検証

1. 1 つのサーバーのメールボックスでアクションを実行し、Dovecot が変更を他のサーバーにレプリケートしたかどうかを確認します。

2. レプリケーターステータスを表示します。

```
# doveadm replicator status
Queued 'sync' requests       0
Queued 'high' requests       0
Queued 'low' requests        0
Queued 'failed' requests     0
Queued 'full resync' requests 30
Waiting 'failed' requests    0
Total number of known users  75
```

3. 特定のユーザーのレプリケーターステータスを表示します。

```
# doveadm replicator status example_user
username        priority  fast sync  full sync  success sync  failed
example_user    none      02:05:28   04:19:07   02:05:28      -
```

関連情報

- **dsync(1)** man ページ
- **/usr/share/doc/dovecot/wiki/Replication.txt**

9.5. ユーザーを IMAP メールボックスに自動的に登録する

通常、IMAP サーバー管理者は、Dovecot が **Sent** や **Trash** などの特定のメールボックスを自動的に作成し、ユーザーをそれらに登録することを望んでいます。これは設定ファイルに設定できます。

さらに、**特殊用途のメールボックス** を定義できます。多くの場合、IMAP クライアントは、メールの送信など、特別な目的のためにメールボックスを定義することをサポートしています。ユーザーが正しいメールボックスを手動で選択して設定する必要がないようにするために、IMAP サーバーは IMAP **LIST** コマンドで **special-use** 属性を送信できます。その後、クライアントはこの属性を使用して、送信済みメールのメールボックスなどを識別および設定できます。

前提条件

- Dovecot が設定されている。

手順

1. **/etc/dovecot/conf.d/15-mailboxes.conf** ファイルの **inbox** namespace セクションを更新します。
   a. **auto = subscribe** 設定を、ユーザーが利用できるようにする必要がある各特殊用途のメールボックスに追加します。次に例を示します。

```
namespace inbox {
  ...
  mailbox Drafts {
    special_use = \Drafts
    auto = subscribe
  }

  mailbox Junk {
    special_use = \Junk
    auto = subscribe
  }

  mailbox Trash {
    special_use = \Trash
    auto = subscribe
  }

  mailbox Sent {
    special_use = \Sent
    auto = subscribe
```

```
  }
  ...
}
```

メールクライアントがより特殊用途のメールボックスをサポートしている場合は、同様のエントリーを追加できます。**special_use** パラメーターは、Dovecot が **special-use** 属性でクライアントに送信する値を定義します。

b. オプション: 特別な目的のない他のメールボックスを定義する場合は、ユーザーの受信トレイにそれらの **mailbox** セクションを追加します。次に例を示します。

```
namespace inbox {
  ...
  mailbox "Important Emails" {
    auto = <value>
  }
  ...
}
```

auto パラメーターは、次のいずれかの値に設定できます。

- **subscribe**: メールボックスを自動的に作成し、ユーザーを登録します。
- **create**: ユーザーを登録せずに、メールボックスを自動的に作成します。
- **no** (デフォルト): Dovecot はメールボックスを作成することも、ユーザーを登録することもしません。

2. Dovecot をリロードします。

```
# systemctl reload dovecot
```

検証

- IMAP クライアントを使用してメールボックスにアクセスします。
auto = subscribe が設定されたメールボックスは、自動的に表示されます。クライアントが特殊用途のメールボックスと定義された目的をサポートしている場合、クライアントはそれらを自動的に使用します。

関連情報

- RFC 6154: 特殊用途メールボックスの IMAP LIST 拡張
- **/usr/share/doc/dovecot/wiki/MailboxSettings.txt**

9.6. LMTP ソケットと LMTPS リスナーの設定

Postfix などの SMTP サーバーは、Local Mail Transfer Protocol (LMTP) を使用して電子メールを Dovecot に配信します。SMTP サーバーが実行されている場合:

- Dovecot と同じホストで、LMTP ソケットを使用します。
- 別のホストで、LMTP サービスを使用する

デフォルトでは、LMTP プロトコルは暗号化されていません。ただし、TLS 暗号化を設定した場合、Dovecot は LMTP サービスに同じ設定を自動的に使用します。その後、SMTP サーバーは、LMTPS プロトコルまたは LMTP 上の **STARTTLS** コマンドを使用して接続できます。

前提条件

- Dovecot がインストールされています。
- LMTP サービスを設定する場合、Dovecot で TLS 暗号化が設定されます。

手順

1. LMTP プロトコルが有効になっていることを確認します。

```
# doveconf -a | egrep "^protocols"
protocols = imap pop3 lmtp
```

出力に **lmtp** が含まれている場合、プロトコルは有効になっています。

2. **lmtp** プロトコルが無効になっている場合は、**/etc/dovecot/dovecot.conf** ファイルを編集し、**protocols** パラメーターの値に **lmtp** を追加します。

```
protocols = ... lmtp
```

3. LMTP ソケットまたはサービスが必要かどうかに応じて、**/etc/dovecot/conf.d/10-master.conf** ファイルの **service lmtp** セクションで次の変更を行います。

- LMTP ソケット: デフォルトでは、Dovecot は自動的に **/var/run/dovecot/lmtp** ソケットを作成します。
  オプション: 所有権と権限をカスタマイズします。

```
service lmtp {
  ...
  unix_listener lmtp {
    mode = 0600
    user = postfix
    group = postfix
  }
  ...
}
```

- LMTP サービス: **inet_listener** サブセクションを追加します。

```
service lmtp {
  ...
  inet_listener lmtp {
    port = 24
  }
  ...
}
```

4. SMTP サーバーのみが LMTP ポートにアクセスできるように、**firewalld** ルールを設定します。次に例を示します。

```
# firewall-cmd --permanent --zone=public --add-rich-rule="rule family="ipv4" source
address="192.0.2.1/32" port protocol="tcp" port="24" accept"
# firewall-cmd --permanent --zone=public --add-rich-rule="rule family="ipv6" source
address="2001:db8:2::1/128" port protocol="tcp" port="24" accept"
# firewall-cmd --reload
```

IPv4 アドレスのサブネットマスク **/32** と IPv6 アドレスのサブネットマスク **/128** は、指定されたアドレスへのアクセスを制限します。

5. Dovecot をリロードします。

```
# systemctl reload dovecot
```

検証

1. LMTP ソケットを設定した場合は、Dovecot がソケットを作成したことと、権限が正しいことを確認します。

```
# ls -l /var/run/dovecot/lmtp
srw-------. 1 postfix postfix 0 Nov 22 17:17 /var/run/dovecot/lmtp
```

2. LMTP ソケットまたはサービスを使用して、Dovecot に電子メールを送信するように、SMTP サーバーを設定します。
LMTP サービスを使用する場合は、SMTP サーバーが LMTPS プロトコルを使用するか、**STARTTLS** コマンドを送信して暗号化された接続を使用するようにしてください。

関連情報

- **/usr/share/doc/dovecot/wiki/LMTP.txt**

9.7. DOVECOT で IMAP または POP3 サービスを無効にする

デフォルトでは、Dovecot は IMAP および POP3 サービスを提供します。そのうちの 1 つだけが必要な場合は、もう 1 つを無効にして、攻撃サーフェスを減らすことができます。

前提条件

- Dovecot がインストールされています。

手順

1. **/etc/dovecot/dovecot.conf** ファイルの **protocols** パラメーターをコメント解除し、必要なプロトコルを使用するように設定します。たとえば、POP3 を必要としない場合は、次のように設定します。

```
protocols = imap lmtp
```

デフォルトでは、**imap**、**pop3**、および **lmtp** プロトコルが有効になっています。

2. Dovecot をリロードします。

```
# systemctl reload dovecot
```

3. ローカルファイアウォールで不要になったポートを閉じます。たとえば、POP3S および POP3 プロトコルのポートを閉じるには、次のように入力します。

```
# firewall-cmd --remove-service=pop3s --remove-service=pop3
# firewall-cmd --reload
```

検証

- **dovecot** プロセスによって開かれた **LISTEN** モードのすべてのポートを表示します。

```
# ss -tulp | grep dovecot
tcp  LISTEN 0  100  0.0.0.0:993  0.0.0.0:*  users:(("dovecot",pid=1405,fd=44))
tcp  LISTEN 0  100  0.0.0.0:143  0.0.0.0:*  users:(("dovecot",pid=1405,fd=42))
tcp  LISTEN 0  100     [::]:993     [::]:*  users:(("dovecot",pid=1405,fd=45))
tcp  LISTEN 0  100     [::]:143     [::]:*  users:(("dovecot",pid=1405,fd=43))
```

この例では、Dovecot は TCP ポート **993** (IMAPS) と **143** (IMAP) のみをリッスンします。

ソケットを使用する代わりにポートをリッスンするようにサービスを設定した場合、Dovecot は LMTP プロトコルのポートのみを開くことに注意してください。

関連情報

- **firewall-cmd(1)** man ページ

9.8. DOVECOT IMAP サーバーで SIEVE を使用してサーバーサイドメールフィルタリングを有効にする

ManageSieve プロトコルを使用して、Sieve スクリプトをサーバーにアップロードできます。Sieve スクリプトは、受信メールに対してサーバーが検証して実行するルールとアクションを定義します。たとえば、ユーザーは Sieve を使用して特定の送信者からの電子メールを転送でき、管理者はグローバルフィルターを作成して、スパムフィルターによってフラグが付けられたメールを別の IMAP フォルダーに移動できます。

ManageSieve プラグインは、Sieve スクリプトと ManageSieve プロトコルのサポートを Dovecot IMAP サーバーに追加します。

警告

TLS 接続を介した ManageSieve プロトコルの使用をサポートするクライアントのみを使用してください。このプロトコルの TLS を無効にすると、クライアントはネットワーク経由で認証情報をプレーンテキストで送信します。

前提条件

- Dovecot が設定され、IMAP メールボックスを提供します。
- TLS 暗号化は Dovecot で設定されます。

- メールクライアントは、TLS 接続を介して ManageSieve プロトコルをサポートします。

手順

1. **dovecot-pigeonhole** パッケージをインストールします。

```
# yum install dovecot-pigeonhole
```

2. **/etc/dovecot/conf.d/20-managesieve.conf** の次の行をコメント解除して、**sieve** プロトコルを有効にします。

```
protocols = $protocols sieve
```

この設定により、すでに有効になっている他のプロトコルに加えて、Sieve が有効になります。

3. **firewalld** で ManageSieve ポートを開きます。

```
# firewall-cmd --permanent --add-service=managesieve
# firewall-cmd --reload
```

4. Dovecot をリロードします。

```
# systemctl reload dovecot
```

検証

1. クライアントを使用し、Sieve スクリプトをアップロードします。次の接続設定を使用します。
   - ポート: 4190
   - 接続セキュリティー: SSL/TLS
   - 認証方法: PLAIN
2. Sieve スクリプトをアップロードしたユーザーに電子メールを送信します。電子メールがスクリプトのルールと一致する場合は、サーバーが定義されたアクションを実行することを確認します。

関連情報

- **/usr/share/doc/dovecot/wiki/Pigeonhole.Sieve.Plugins.IMAPSieve.txt**
- **/usr/share/doc/dovecot/wiki/Pigeonhole.Sieve.Troubleshooting.txt**
- **firewall-cmd(1)** man ページ

9.9. DOVECOT が設定ファイルを処理する方法

dovecot パッケージは、メインの設定ファイル **/etc/dovecot/dovecot.conf**、および **/etc/dovecot/conf.d/** ディレクトリー内の複数の設定ファイルを提供します。Dovecot は、サービスの開始時にファイルを組み合わせて設定を構築します。

複数の設定ファイルの主な利点は、設定をグループ化し、読みやすくすることです。単一の設定ファイルを使用する場合は、代わりに **/etc/dovecot/dovecot.conf** ですべての設定を維持し、そのファイルからすべての **include** および **include_try** ステートメントを削除できます。

関連情報

- **/usr/share/doc/dovecot/wiki/ConfigFile.txt**
- **/usr/share/doc/dovecot/wiki/Variables.txt**

第10章 印刷の設定

Red Hat Enterprise Linux では、Common UNIX Printing System (CUPS) によって印刷を管理します。ユーザーはホスト上の CUPS でプリンターを設定して印刷します。さらに、CUPS でプリンターを共有し、ホストをプリントサーバーとして使用できます。

CUPS は次のプリンターへの印刷をサポートしています。

- AirPrint™ および IPP Everywhere™ プリンター
- 従来の PostScript Printer description (PPD) ベースのドライバーを備えたネットワークおよびローカル USB プリンター

10.1. CUPS のインストールと設定

CUPS を使用してローカルホストから印刷できます。このホストを使用してネットワーク内でプリンターを共有し、プリントサーバーとして機能させることもできます。

手順

1. **cups** パッケージをインストールします。

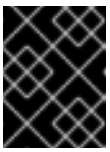
```
# yum install cups
```

2. CUPS をプリントサーバーとして設定する場合は、**/etc/cups/cupsd.conf** ファイルを編集し、次の変更を加えます。

   a. CUPS をリモートで設定する場合、またはこのホストをプリントサーバーとして使用する場合は、CUPS がリッスンする IP アドレスとポートを設定します。

```
Listen 192.0.2.1:631
Listen [2001:db8:1::1]:631
```

   デフォルトでは、CUPS は **localhost** インターフェイス (**127.0.0.1** および **::1**) でのみリッスンします。IPv6 アドレスは角括弧で囲んで指定します。



重要

信頼できないネットワーク (インターネットなど) からのアクセスを許可するインターフェイスをリッスンするように CUPS を設定しないでください。

   b. **<Location />** ディレクティブでそれぞれの IP 範囲を許可することで、サービスにアクセスできる IP 範囲を設定します。

```
<Location />
  Allow from 192.0.2.0/24
  Allow from [2001:db8:1::1]/32
  Order allow,deny
</Location>
```

   c. **<Location /admin>** ディレクティブで、CUPS 管理サービスにアクセスできる IP アドレスと範囲を設定します。

```
<Location /admin>
  Allow from 192.0.2.15/32
  Allow from [2001:db8:1::22]/128
  Order allow,deny
</Location>
```

この設定では、IP アドレス **192.0.2.15** および **2001:db8:1::22** を持つホストのみが管理サービスにアクセスできます。

d. オプション: Web インターフェイスで設定ファイルとログファイルへのアクセスを許可する IP アドレスと範囲を設定します。

```
<Location /admin/conf>
  Allow from 192.0.2.15/32
  Allow from [2001:db8:1::22]/128
  ...
</Location>

<Location /admin/log>
  Allow from 192.0.2.15/32
  Allow from [2001:db8:1::22]/128
  ...
</Location>
```

3. **firewalld** サービスを実行し、CUPS へのリモートアクセスを設定する場合は、**firewalld** で CUPS ポートを開きます。

```
# firewall-cmd --permanent --add-port=631/tcp
# firewall-cmd --reload
```

複数のインターフェイスを持つホストで CUPS を実行する場合は、必要なネットワークにアクセスを制限することを検討してください。

4. **cups** サービスを有効にして起動します。

```
# systemctl enable --now cups
```

検証

- ブラウザーを使用して、**http://<hostname>:631** にアクセスします。Web インターフェイスに接続できれば、CUPS は動作しています。
  Administration タブなどの一部の機能では、認証と HTTPS 接続が必要であることに注意してください。デフォルトでは、CUPS は HTTPS アクセスに自己署名証明書を使用するため、認証時の接続はセキュアではありません。

次のステップ

- CUPS サーバーでの TLS 暗号化の設定
- オプション: Web インターフェイスで CUPS サーバーを管理するための管理権限の付与
- Web インターフェイスを使用した CUPS へのプリンターの追加
- firewalld の使用および設定

10.2. CUPS サーバーでの TLS 暗号化の設定

CUPS は TLS 暗号化接続をサポートしており、デフォルトでは、サービスは認証を必要とするすべてのリクエストに対して暗号化接続を強制します。証明書が設定されていない場合、CUPS は秘密鍵と自己署名証明書を作成します。これで十分なのは、ローカルホスト自体から CUPS にアクセスする場合だけです。ネットワーク経由でセキュアに接続するには、認証局 (CA) によって署名されたサーバー証明書を使用します。

警告

暗号化を使用しなかった場合、または自己署名証明書を使用した場合、中間者 (MITM) 攻撃により、次のような情報が漏洩する可能性があります。

- Web インターフェイスを使用して CUPS を設定する際の管理者の認証情報
- ネットワーク経由で印刷ジョブを送信する際の機密データ

前提条件

- CUPS が設定されている。
- 秘密鍵の作成 が完了していて、その秘密鍵に対するサーバー証明書が CA から発行されている。
- 中間証明書をサーバー証明書にアタッチしている (サーバー証明書を検証するために中間証明書が必要な場合)。
- CUPS にはサービスが鍵を読み取るときにパスワードを入力するオプションがないため、秘密鍵はパスワードで保護されません。
- 証明書の正規名 (**CN**) またはサブジェクト代替名 (SAN) フィールドが、次のいずれかと一致する。
  - CUPS サーバーの完全修飾ドメイン名 (FQDN)
  - DNS によってサーバーの IP アドレスに解決されるエイリアス
- 秘密鍵とサーバー証明書ファイルが、Privacy Enhanced Mail (PEM) 形式を使用している。
- クライアントが CA 証明書を信頼している。

手順

1. **/etc/cups/cups-files.conf** ファイルを編集し、次の設定を追加して自己署名証明書の自動作成を無効にします。

```
CreateSelfSignedCerts no
```

2. 自己署名証明書と秘密鍵を削除します。

```
# rm /etc/cups/ssl/<hostname>.crt /etc/cups/ssl/<hostname>.key
```

3. オプション: サーバーの FQDN を表示します。

```
# hostname -f
server.example.com
```

4. オプション: 証明書の **CN** フィールドと SAN フィールドを表示します。

```
# openssl x509 -text -in /etc/cups/ssl/server.example.com.crt
Certificate:
  Data:
    ...
    Subject: CN = server.example.com
    ...
    X509v3 extensions:
      ...
      X509v3 Subject Alternative Name:
        DNS:server.example.com
  ...
```

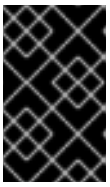
5. サーバー証明書の **CN** または SAN フィールドにサーバーの FQDN とは異なるエイリアスが含まれている場合は、**ServerAlias** パラメーターを **/etc/cups/cupsd.conf** ファイルに追加します。

```
ServerAlias alternative_name.example.com
```

この場合、以降の手順で FQDN の代わりに代替名を使用します。

6. 秘密鍵とサーバー証明書を **/etc/cups/ssl/** ディレクトリーに保存します。次に例を示します。

```
# mv /root/server.key /etc/cups/ssl/server.example.com.key
# mv /root/server.crt /etc/cups/ssl/server.example.com.crt
```



重要

CUPS では、秘密鍵に **<fqdn>.key**、サーバー証明書ファイルに **<fqdn>.crt** という名前を付ける必要があります。エイリアスを使用する場合は、ファイルに **<alias>.key** および **<alias>.crt** という名前を付ける必要があります。

7. **root** ユーザーのみがこのファイルを読み取ることができるように、秘密鍵にセキュアなパーミッションを設定します。

```
# chown root:root /etc/cups/ssl/server.example.com.key
# chmod 600 /etc/cups/ssl/server.example.com.key
```

証明書は、セキュアな接続を確立する前のクライアントとサーバー間の通信の要素であるため、どのクライアントも認証なしで証明書を取得できます。したがって、サーバー証明書ファイルに厳密なパーミッションを設定する必要はありません。

8. SELinux コンテキストを復元します。

```
# restorecon -Rv /etc/cups/ssl/
```

9. デフォルトでは、CUPS は、Web インターフェイスの **/admin** ページで管理タスクを実行する場合など、タスクで認証が必要な場合にのみ暗号化接続を強制します。
CUPS サーバー全体の暗号化を強制するには、**/etc/cups/cupsd.conf** ファイル内のすべての **<Location>** ディレクティブに **Encryption Required** を追加します。次に例を示します。

```
<Location />
  ...
  Encryption Required
</Location>
```

10. CUPS を再起動します。

```
# systemctl restart cups
```

検証

1. ブラウザーを使用して、**https://<hostname>:631/admin/** にアクセスします。接続が成功すれば、CUPS で TLS 暗号化が正しく設定されています。

2. サーバー全体の暗号化を必須とするように設定した場合は、**http://<hostname>:631/** にアクセスします。この場合、CUPS は **Upgrade Required** というエラーを返します。

トラブルシューティング

- **cups** サービスの **systemd** ジャーナルエントリーを表示します。

```
# journalctl -u cups
```

HTTPS プロトコルを使用した Web インターフェイスへの接続に失敗した後、ジャーナルに **Unable to encrypt connection: Error while reading file** というエラーが含まれている場合は、秘密鍵とサーバー証明書ファイルの名前を確認します。

関連情報

- ソリューション How to configure CUPS to use a CA-signed TLS certificate in RHEL

10.3. WEB インターフェイスで CUPS サーバーを管理するための管理権限の付与

デフォルトでは、**sys**、**root**、および **wheel** グループのメンバーは、Web インターフェイスで管理タスクを実行できます。しかし、他の特定のサービスもこれらのグループを使用します。たとえば、**wheel** グループのメンバーは、デフォルトで、**sudo** を使用して **root** 権限でコマンドを実行できます。CUPS 管理者が他のサービスで予期しない権限を取得しないようにするには、CUPS 管理者専用のグループを使用します。

前提条件

- CUPS が設定されている。

- 使用するクライアントの IP アドレスに、Web インターフェイスの管理領域にアクセスする権限がある。

手順

1. CUPS 管理者用のグループを作成します。

```
# groupadd cups-admins
```

2. Web インターフェイスでサービスを管理する必要があるユーザーを、**cups-admins** グループに追加します。

```
# usermod -a -G cups-admins <username>
```

3. **/etc/cups/cups-files.conf** ファイルの **SystemGroup** パラメーターの値を更新し、**cups-admin** グループを追加します。

```
SystemGroup sys root wheel cups-admins
```

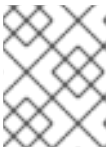
cups-admin グループのみに管理アクセス権を与える必要がある場合は、パラメーターから他のグループ名を削除します。

4. CUPS を再起動します。

```
# systemctl restart cups
```

検証

1. ブラウザーを使用して、**https://<hostname_or_ip_address>:631/admin/** にアクセスします。



注記

ユーザーは HTTPS プロトコルを使用する場合のみ、Web UI の管理領域にアクセスできます。

2. 管理タスクの実行を開始します。たとえば、**Add printer** をクリックします。

3. Web インターフェイスで、ユーザー名とパスワードの入力を求められます。続行するには、**cups-admins** グループのメンバーであるユーザーの認証情報を使用して認証します。認証が成功すると、このユーザーは管理タスクを実行できます。

10.4. プリンタードライバーを含むパッケージの概要

Red Hat Enterprise Linux (RHEL) は、CUPS 用のプリンタードライバーを含むさまざまなパッケージを提供しています。これらのパッケージの概要と、パッケージに含まれるドライバーが対応するベンダーを以下に示します。

表10.1 ドライバーパッケージのリスト

パッケージ名	プリンター用ドライバー
cups	Zebra、Dymo
c2esp	Kodak

パッケージ名	プリンター用ドライバー
foomatic	Brother、Canon、Epson、Gestetner、HP、Infotec、Kyocera、Lanier、Lexmark、NRG、Ricoh、Samsung、Savin、Sharp、Toshiba、Xerox など
gutenprint-cups	Brother、Canon、Epson、Fujitsu、HP、Infotec、Kyocera、Lanier、NRG、Oki、Minolta、Ricoh、Samsung、Savin、Xerox など
hplip	HP
pnm2ppa	HP
splix	Samsung、Xerox など

パッケージによっては、同じプリンターベンダーまたはモデル用の、異なる機能を持つドライバーが含まれている場合があることに注意してください。

必要なパッケージをインストールした後、CUPS Web インターフェイスまたは **lpinfo -m** コマンドを使用してドライバーのリストを表示できます。

10.5. プリンターがドライバーレス印刷をサポートしているかどうかの確認

CUPS はドライバーレス印刷をサポートしています。つまり、ハードウェア固有のソフトウェアをプリンターに提供しなくても印刷できます。ドライバーレス印刷を使用するには、プリンターからクライアントにその機能を通知し、次のいずれかの標準をプリンターで使用する必要があります。

- AirPrint™
- IPP Everywhere™
- Mopria®
- Wi-Fi Direct Print Services

ipptool ユーティリティーを使用すると、プリンターがドライバーレス印刷をサポートしているかどうかを確認できます。

前提条件

- プリンターまたはリモートプリントサーバーが、インターネット印刷プロトコル (IPP) をサポートしている。
- ホストが、プリンターまたはリモートプリントサーバーの IPP ポートに接続できる。デフォルトの IPP ポートは 631 です。

手順

- **ipp-versions-supported** 属性と **document-format-supported** 属性を照会し、**get-printer-attributes** テストに合格することを確認します。
  - リモートプリンターの場合は、次のように入力します。

```
# ipptool -tv ipp://<ip_address_or_hostname>:631/ipp/print get-printer-attributes.test | grep -E "ipp-versions-supported|document-format-supported|get-printer-attributes"
Get printer attributes using get-printer-attributes      [PASS]
    ipp-versions-supported (1setOf keyword) = ...
    document-format-supported (1setOf mimeMediaType) = ...
```

- リモートプリントサーバー上のキューの場合は、次のように入力します。

```
# ipptool -tv ipp://<ip_address_or_hostname>:631/printers/<queue_name> get-printer-attributes.test | grep -E "ipp-versions-supported|document-format-supported|get-printer-attributes"
Get printer attributes using get-printer-attributes      [PASS]
    ipp-versions-supported (1setOf keyword) = ...
    document-format-supported (1setOf mimeMediaType) = ...
```

ドライバーレス印刷が機能することを確認するには、出力で次のことを確認します。

- **get-printer-attributes** テストが **PASS** を返す。
- プリンターがサポートする IPP バージョンが 2.0 以降である。
- 形式のリストに次のいずれかが含まれている。
  - **application/pdf**
  - **image/urf**
  - **image/pwg-raster**
- カラープリンターの場合、前述の形式のいずれかに加えて、**image/jpeg** が出力に含まれている。

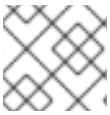
次のステップ:

- Web インターフェイスを使用した CUPS へのプリンターの追加
- lpadmin ユーティリティーを使用した CUPS へのプリンターの追加

10.6. WEB インターフェイスを使用した CUPS へのプリンターの追加

ユーザーが CUPS 経由で印刷できるようにするには、事前にプリンターを追加する必要があります。ネットワークプリンターと、USB 経由などで CUPS ホストに直接接続されているプリンターの両方を使用できます。

CUPS ドライバーレス機能を使用するか、PostScript Printer Description (PPD) ファイルを使用してプリンターを追加できます。



注記

CUPS はドライバーレス印刷を優先しており、ドライバーの使用は非推奨です。

Red Hat Enterprise Linux (RHEL) は、mDNS レスポンダーにクエリーを実行することでリクエストを解決する、Name Service Switch Multicast DNS プラグイン (**nss-mdns**) を提供しません。そのため、

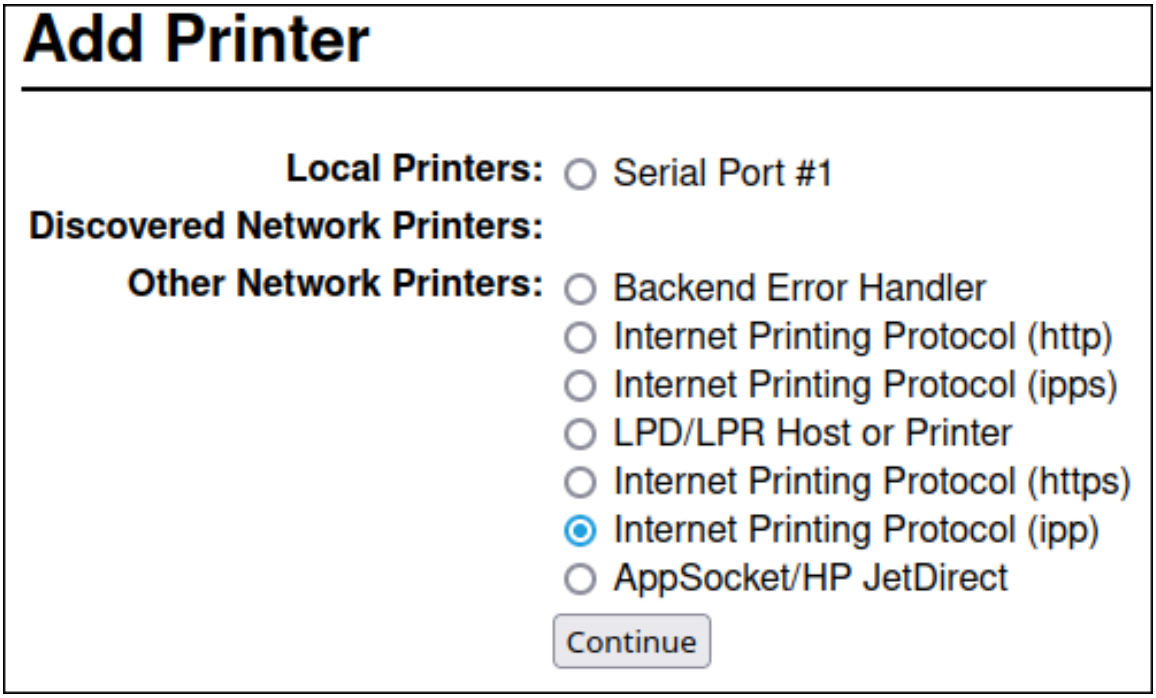
RHEL では、mDNS を使用したローカルドライバーレスプリンターの自動検出とインストールは利用できません。この問題を回避するには、1台のプリンターを手動でインストールするか、**cups-browsed** を使用してリモートプリントサーバーで使用可能な多数の印刷キューを自動的にインストールします。

前提条件

- CUPS が設定されている。
- CUPS でプリンターを管理する権限を持っている。
- CUPS をプリントサーバーとして使用する場合は、ネットワーク上でデータをセキュアに送信するために TLS 暗号化を設定 している。
- プリンターがドライバーレス印刷をサポートしている (この機能を使用する場合)。

手順

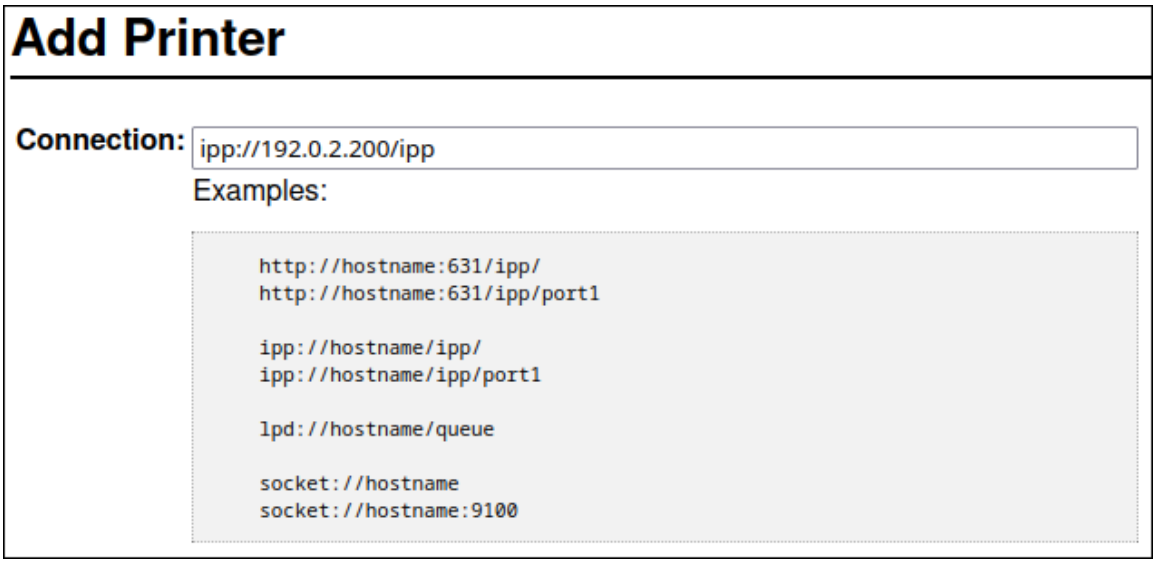
1. ブラウザーを使用して、**https://<hostname>:631/admin/** にアクセスします。
   Web インターフェイスには HTTPS プロトコルを使用して接続する必要があります。使用しないと、セキュリティー上の理由から、CUPS が後のステップで認証できなくなります。
2. **Add printer** をクリックします。
3. ユーザーがまだ認証されていない場合、CUPS は管理ユーザーの認証情報の入力を求めます。許可されたユーザーのユーザー名とパスワードを入力します。
4. ドライバーレス印刷を使用せず、追加するプリンターが自動的に検出された場合は、それを選択し、**Continue** をクリックします。
5. プリンターが検出されなかった場合は、以下を実行します。
   a. プリンターがサポートするプロトコルを選択します。



   プリンターがドライバーレス印刷をサポートしており、この機能を使用する場合は、**ipp** または **ipps** プロトコルを選択します。

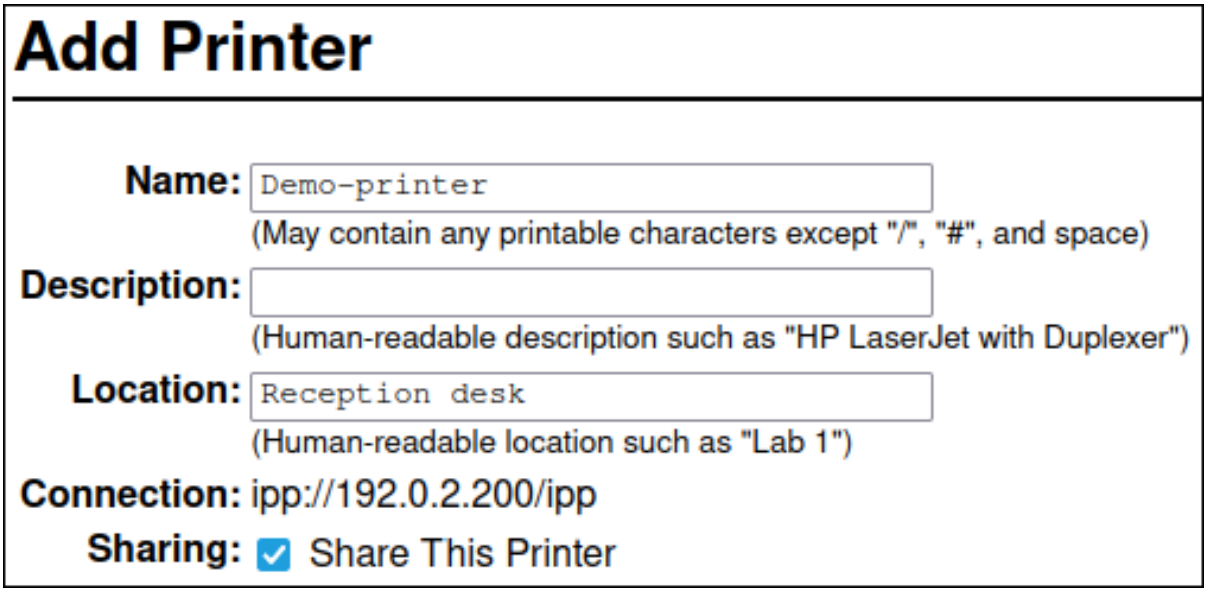
   b. **Continue** をクリックします。

c. プリンターまたはリモートプリントサーバー上のキューへの URL を入力します。

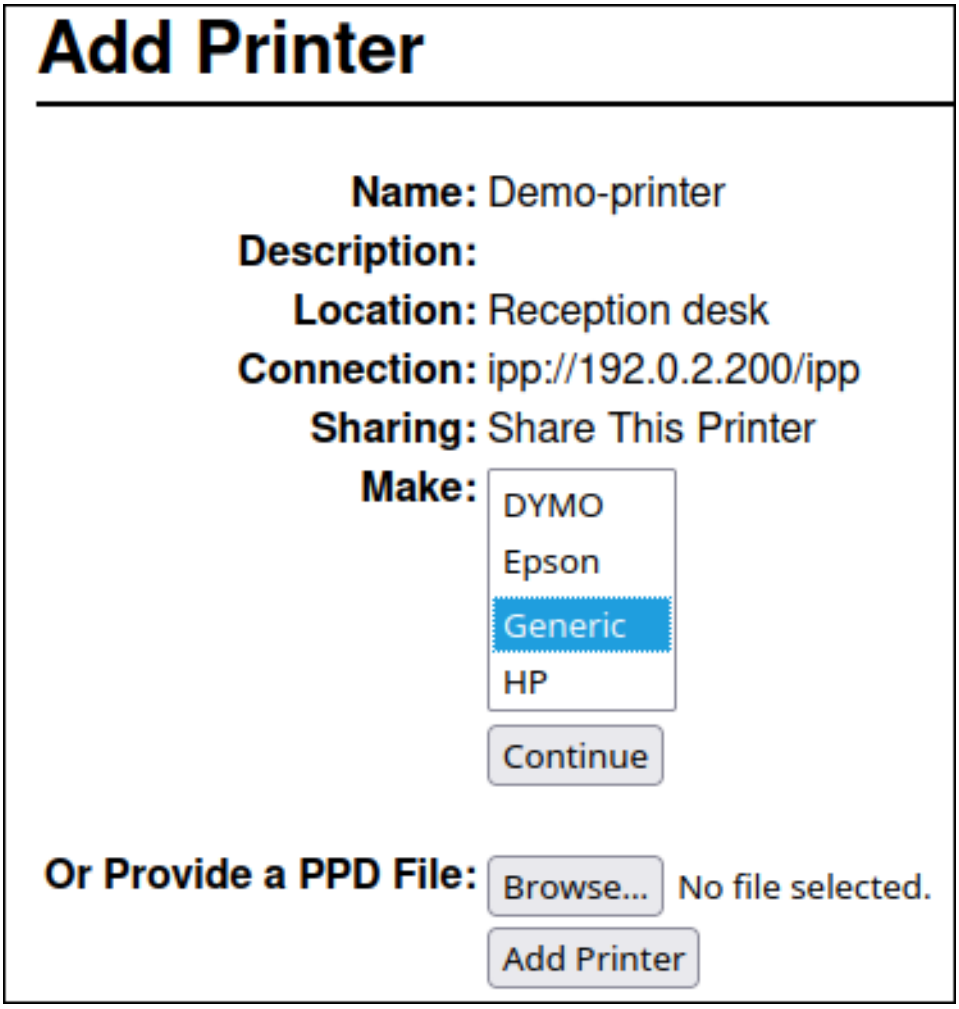


d. **Continue** をクリックします。

6. 名前を入力し、必要に応じて説明と場所を入力します。CUPS をプリントサーバーとして使用し、他のクライアントが CUPS を介してこのプリンターで印刷できるようにする場合は、**Share this printer** も選択します。



7. **Make** リストからプリンターの製造元を選択します。プリンターの製造元がリストにない場合は、**Generic** を選択するか、プリンターの PPD ファイルをアップロードします。

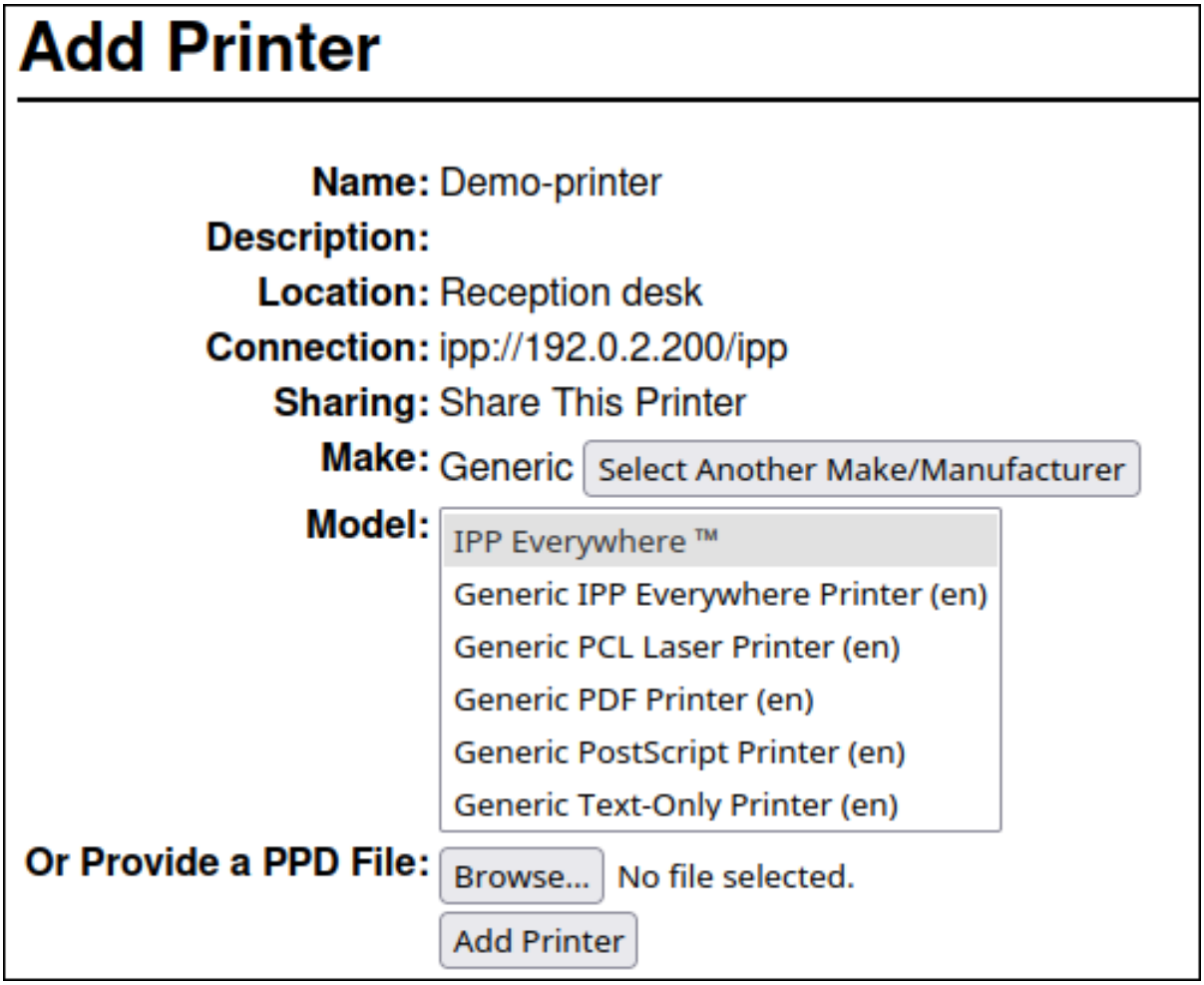


8. **Continue** をクリックします。

9. プリンターのモデルを選択します。

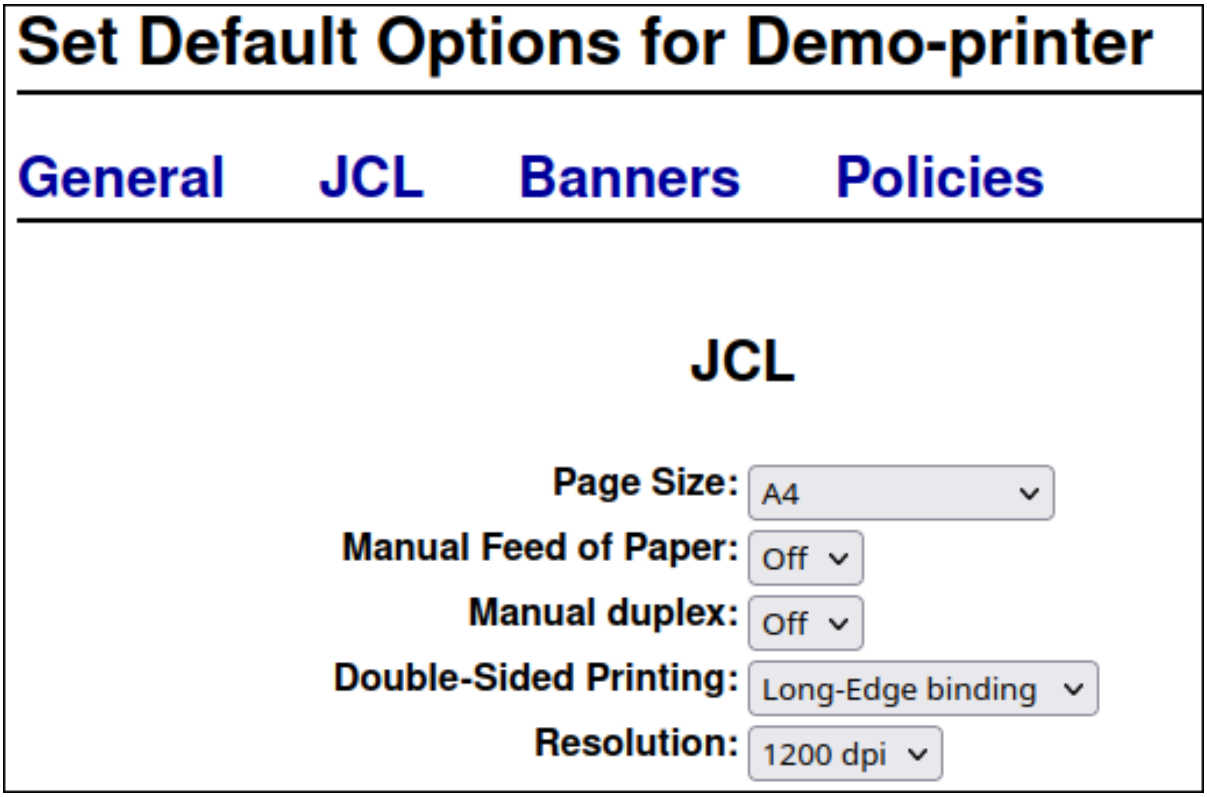
   - プリンターがドライバーレス印刷をサポートしている場合は、**IPP Everywhere** を選択します。なお、以前にプリンター固有のドライバーをローカルにインストールしていた場合は、リストに **<printer_name> - IPP Everywhere** などのエントリーも含まれている可能性があります。

   - プリンターがドライバーレス印刷をサポートしていない場合は、モデルを選択するか、プリンターの PPD ファイルをアップロードします。



10. **Add Printer** をクリックします。

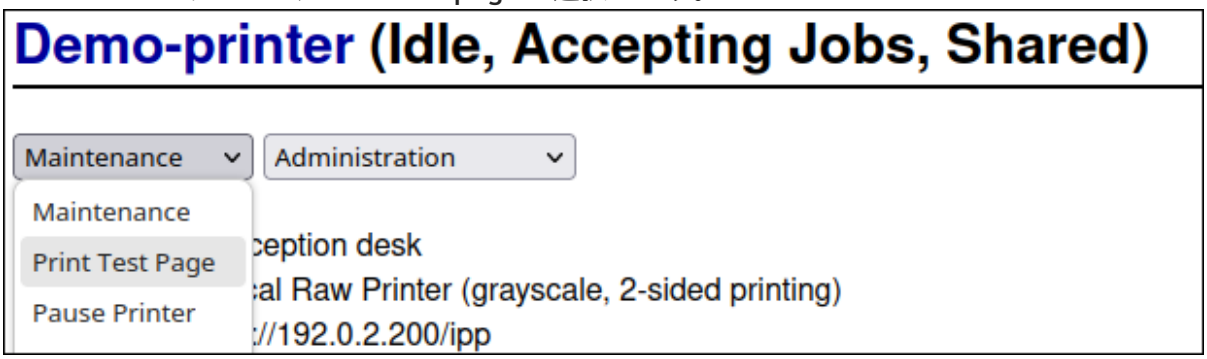
11. **Set printer options** ページの設定とタブは、ドライバーとプリンターがサポートする機能によって異なります。このページを使用して、用紙サイズなどのデフォルトのオプションを設定します。



12. **Set default options** をクリックします。

検証

1. Web インターフェイスで **Printers** タブを開きます。
2. プリンターの名前をクリックします。
3. **Maintenance** リストで、**Print test page** を選択します。



トラブルシューティング

- ドライバーレス印刷を使用していて印刷がうまくいかない場合は、**lpadmin** ユーティリティーを使用してコマンドラインでプリンターを追加します。詳細は、lpadmin ユーティリティーを使用した CUPS へのプリンターの追加 を参照してください。

10.7. LPADMIN ユーティリティーを使用した CUPS へのプリンターの追加

ユーザーが CUPS 経由で印刷できるようにするには、事前にプリンターを追加する必要があります。ネットワークプリンターと、USB 経由などで CUPS ホストに直接接続されているプリンターの両方を使用できます。

CUPS ドライバーレス機能を使用するか、PostScript Printer Description (PPD) ファイルを使用してプリンターを追加できます。

注記

CUPS はドライバーレス印刷を優先しており、ドライバーの使用は非推奨です。

Red Hat Enterprise Linux (RHEL) は、mDNS レスポンダーにクエリーを実行することでリクエストを解決する、Name Service Switch Multicast DNS プラグイン (**nss-mdns**) を提供しません。そのため、RHEL では、mDNS を使用したローカルドライバーレスプリンターの自動検出とインストールは利用できません。この問題を回避するには、1 台のプリンターを手動でインストールするか、**cups-browsed** を使用してリモートプリントサーバーで使用可能な多数の印刷キューを自動的にインストールします。

前提条件

- CUPS が設定されている。
- プリンターがドライバーレス印刷をサポートしている (この機能を使用する場合)。
- プリンターが、ポート 631 (IPP)、9100 (ソケット)、または 515 (LPD) でデータを受信する。ポートは、プリンターへの接続に使用する方法によって異なります。

手順

- CUPS にプリンターを追加します。
  - ドライバーレスサポートを使用してプリンターを追加するには、次のように入力します。

    ```
    # lpadmin -p Demo-printer -E -v ipp://192.0.2.200/ipp/print -m everywhere
    ```

    お使いのプリンターで **-m everything** オプションが機能しない場合は、**-m driverless:<uri>** を試してください (例: **-m driverless:ipp://192.0.2.200/ipp/print**)。

  - ドライバーレスサポートを使用してリモートプリントサーバーのキューを追加するには、次のように入力します。

    ```
    # lpadmin -p Demo-printer -E -v ipp://192.0.2.201/printers/example-queue -m everywhere
    ```

    お使いのプリンターで **-m everything** オプションが機能しない場合は、**-m driverless:<uri>** を試してください (例: **-m driverless:ipp://192.0.2.200/printers/example-queue**)。

  - ファイル内のドライバーを使用してプリンターを追加するには、次のように入力します。

    ```
    # lpadmin -p Demo-printer -E -v socket://192.0.2.200/ -P /root/example.ppd
    ```

  - ファイル内のドライバーを使用してリモートプリントサーバーのキューを追加するには、次のように入力します。

    ```
    # lpadmin -p Demo-printer -E -v ipp://192.0.2.201/printers/example-queue -P /root/example.ppd
    ```

  - ローカルドライバーデータベース内のドライバーを使用してプリンターを追加するには、次の手順を実行します。

    i. データベース内のドライバーをリスト表示します。

    ```
    # lpinfo -m
    ...
    drv:///sample.drv/generpcl.ppd Generic PCL Laser Printer
    ...
    ```

    ii. データベース内のドライバーへの URI を使用してプリンターを追加します。

    ```
    # lpadmin -p Demo-printer -E -v socket://192.0.2.200/ -m drv:///sample.drv/generpcl.ppd
    ```

  これらのコマンドでは、次のオプションを使用します。

  - **-p <printer_name>**: CUPS 内のプリンターの名前を設定します。
  - **-E**: プリンターを有効にします。CUPS はそのプリンターのジョブを受け付けます。このオプションは **-p** の後に指定する必要があることに注意してください。詳細は、man ページのオプションの説明を参照してください。
  - **-v <uri>**: プリンターまたはリモートプリントサーバーキューへの URI を設定します。

- **-m <driver_uri>**: ローカルドライバーデータベースから取得した指定のドライバー URI に基づいて PPD ファイルを設定します。
- **-P <PPD_file>**: PPD ファイルへのパスを設定します。

検証

1. 使用可能なプリンターを表示します。

```
# lpstat -p
printer Demo-printer is idle. enabled since Fri 23 Jun 2023 09:36:40 AM CEST
```

2. テストページを印刷します。

```
# lp -d Demo-printer /usr/share/cups/data/default-testpage.pdf
```

10.8. WEB インターフェイスを使用した CUPS プリンターのメンテナンスおよび管理タスクの実行

プリンター管理者は、場合によってはプリントサーバー上でさまざまなタスクを実行する必要があります。以下に例を示します。

- 技術者がプリンターを修理している間のプリンターの一時停止などのメンテナンスタスク
- プリンターのデフォルト設定の変更などの管理タスク

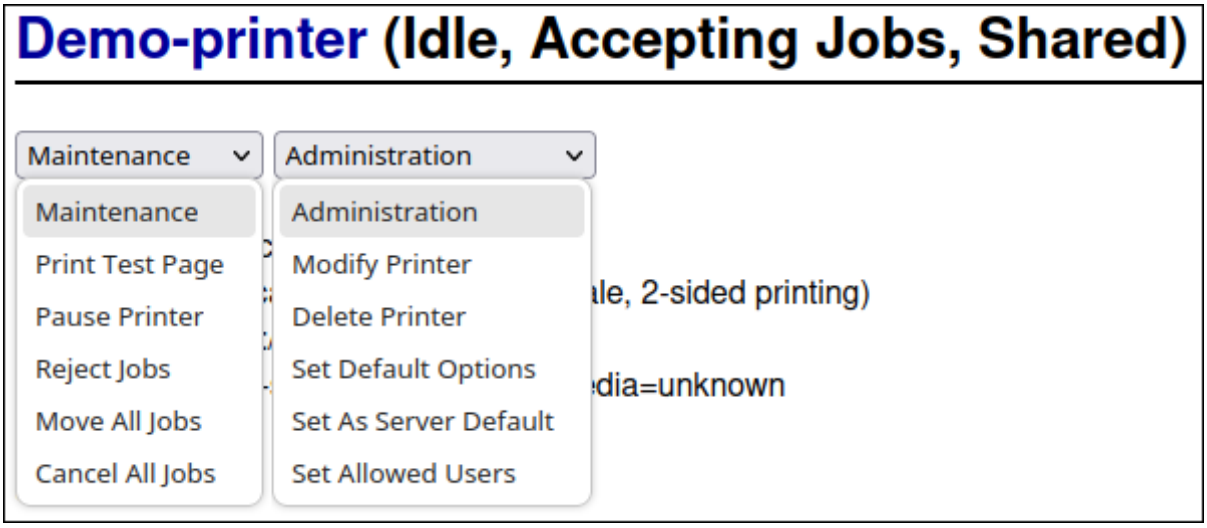
これらのタスクは、CUPS の Web インターフェイスを使用して実行できます。

前提条件

- CUPS が設定されている。
- CUPS でプリンターを管理する権限を持っている。
- CUPS をプリントサーバーとして使用する場合は、ネットワーク上で認証情報をプレーンテキストで送信しないように TLS 暗号化を設定 している。
- プリンターが CUPS にすでに存在する。

手順

1. ブラウザーを使用して、**https://<hostname>:631/printers/** にアクセスします。
   Web インターフェイスには HTTPS プロトコルを使用して接続する必要があります。使用しないと、セキュリティー上の理由から、CUPS が後のステップで認証できなくなります。
2. 設定するプリンターの名前をクリックします。
3. メンテナンスタスクを実行するか管理タスクを実行するかに応じて、対応するリストから必要なアクションを選択します。



4. ユーザーがまだ認証されていない場合、CUPS は管理ユーザーの認証情報の入力を求めます。許可されたユーザーのユーザー名とパスワードを入力します。

5. タスクを実行します。

10.9. SAMBA を使用した KERBEROS 認証による WINDOWS プリントサーバーへの印刷

samba-krb5-printing ラッパーを使用すると、Red Hat Enterprise Linux (RHEL) にログインした Active Directory (AD) ユーザーは、Kerberos を使用して Active Directory (AD) に対して認証して、Windows プリントサーバーに印刷ジョブを転送するローカルの CUPS プリントサーバーに印刷を出力できます。

この設定の利点は、RHEL 上の CUPS の管理者が、設定で固定ユーザー名およびパスワードを保存する必要がないことです。CUPS は、印刷ジョブを送信するユーザーの Kerberos チケットで AD に対して認証します。

注記

Red Hat は、ローカルシステムから CUPS への印刷ジョブの送信のみをサポートしています。Samba プリントサーバー上のプリンターの再共有はサポートしていません。

前提条件

- ローカルの CUPS インスタンスに追加するプリンターが、AD プリントサーバーで共有されている。
- AD のメンバーとして RHEL ホストに参加している。
- CUPS が RHEL にインストールされており、**cups** サービスが実行されている。
- プリンターの PostScript Printer description (PPD) ファイルが **/usr/share/cups/model/** ディレクトリーに保存されている。

手順

1. **samba-krb5-printing**、**samba-client**、および **krb5-workstation** パッケージをインストールします。

```
# yum install samba-krb5-printing samba-client krb5-workstation
```

2. 必要に応じて、ドメイン管理者として認証を行い、Windows プリントサーバーで共有されるプリンターの一覧を表示します。

```
# smbclient -L win_print_srv.ad.example.com -U
administrator@AD_KERBEROS_REALM --use-kerberos=required

 Sharename       Type      Comment
 ---------       ----      -------
 ...
 Example         Printer   Example
 ...
```

3. 必要に応じて、CUPS モデルのリストを表示して、プリンターの PPD 名を指定します。

```
lpinfo -m
...
samsung.ppd Samsung M267x 287x Series PXL
...
```

次のステップでプリンターを追加する場合は、PPD ファイルの名前が必要です。

4. CUPS にプリンターを追加します。

```
# lpadmin -p "example_printer" -v smb://win_print_srv.ad.example.com/Example -m
samsung.ppd -o auth-info-required=negotiate -E
```

このコマンドでは次のオプションを使用します。

- **-p printer_name** は、CUPS にプリンターの名前を設定します。
- **-v URI_to_Windows_printer** は、Windows プリンターに URI を設定します。**smb://host_name/printer_share_name** という形式を使用します。
- **-m PPD_file** は、プリンターが使用する PPD ファイルを設定します。
- **-o auth-info-required=negotiate** は、印刷ジョブをリモートサーバーに転送する際に CUPS が Kerberos 認証を使用するように設定します。
- **-E** はプリンターを有効にします。CUPS はそのプリンターのジョブを受け付けます。

検証

1. AD ドメインユーザーとして RHEL ホストにログインします。
2. AD ドメインユーザーとして認証します。

```
# kinit domain_user_name@AD_KERBEROS_REALM
```

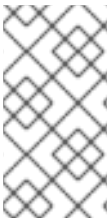
3. ローカルの CUPS プリントサーバーに追加したプリンターにファイルを出力します。

```
# lp -d example_printer file
```

10.10. CUPS-BROWSED を使用してリモートプリントサーバーのプリンターをローカルに統合する

cups-browsed サービスは、DNS サービス検出 (DNS-SD) と CUPS ブラウジングを使用して、共有リモートプリンターのすべて、またはフィルタリング後のサブセットを、ローカルの CUPS サービスで自動的に利用できるようにします。

たとえば、管理者はワークステーションでこの機能を使用して、信頼できるプリントサーバーのプリンターのみをアプリケーションの印刷ダイアログで使用できるようにすることができます。プリントサーバーが多数のプリンターを共有している場合、リストに表示されるプリンターの数を減らすために、参照するプリンターを特定の条件でフィルタリングするように **cup-browsed** を設定することもできます。



注記

アプリケーションの印刷ダイアログが DNS-SD などの他のメカニズムを使用してリモートプリンターのリストを表示している場合、**cups-browsed** は影響を及ぼしません。また、**cups-browsed** サービスは、ユーザーがリストにないプリンターに手動でアクセスすることを妨げるものではありません。

前提条件

- CUPS サービスがローカルホストで設定されている。
- リモート CUPS プリントサーバーが存在する。このサーバーには次の条件が適用されます。
  - サーバーが、クライアントからアクセス可能なインターフェイスでリッスンしている。
  - **/etc/cups/cups.conf** ファイル内のサーバーの **<Location />** ディレクティブにある **Allow from** パラメーターにより、クライアントの IP アドレスからのアクセスが許可されている。
  - サーバーがプリンターを共有している。
  - ファイアウォールルールにより、クライアントからサーバーの CUPS ポートへのアクセスが許可されている。

手順

1. **/etc/cups/cups-browsed.conf** ファイルを編集し、次の変更を加えます。
   a. ポーリングするリモート CUPS サーバーごとに **BrowsePoll** パラメーターを追加します。

   ```
   BrowsePoll remote_cups_server.example.com
   BrowsePoll 192.0.2.100:1631
   ```

   リモート CUPS サーバーが 631 とは異なるポートでリッスンする場合は、ホスト名または IP アドレスに **<port>** を追加します。

   b. オプション: フィルターを設定して、ローカル CUPS サービスに表示されるプリンターを制限します。たとえば、名前に **sales_** が含まれるキューをフィルタリングするには、次のパラメーターを追加します。

   ```
   BrowseFilter name sales_
   ```

さまざまなフィールド名でフィルタリングしたり、フィルターを無効にしたり、正確な値を照合したりできます。詳細は、**cups-browsed.conf(5)** マニュアルページのパラメーターの説明と例を参照してください。

c. オプション: ポーリング間隔とタイムアウトを変更して、参照サイクルの数を制限します。

```
BrowseInterval 1200
BrowseTimeout 6000
```

プリンターが参照リストから消えるのを避けるために、**BrowseInterval** と **BrowseTimeout** の両方を同じ比率で増やしてください。つまり、**BrowseInterval** の値を 5 以上の整数で乗算し、その計算結果の値を **BrowseTimeout** に使用します。

デフォルトでは、**cups-browsed** は 60 秒ごとにリモートサーバーをポーリングし、タイムアウトは 300 秒です。ただし、キューの数が多いプリントサーバーでこれらのデフォルト値を使用すると、大量のリソースが消費される可能性があります。

2. **cups-browsed** サービスを有効にして起動します。

```
# systemctl enable --now cups-browsed
```

検証

- 使用可能なプリンターをリスト表示します。

```
# lpstat -v
device for Demo-printer: implicitclass://Demo-printer/
...
```

プリンターの出力に **implicitclass** が含まれていれば、**cups-browsed** が CUPS でプリンターを管理しています。

関連情報

- **cups-browsed.conf(5)** man ページ

10.11. SYSTEMD ジャーナルの CUPS ログへのアクセス

デフォルトでは、CUPS はログメッセージを **systemd** ジャーナルに保存します。これには、以下のものが含まれます。

- エラーメッセージ
- アクセスログエントリー
- ページログエントリー

前提条件

- CUPS がインストールされている。

手順

- ログエントリーを表示します。

- すべてのログエントリーを表示するには、次のように入力します。

```
# journalctl -u cups
```

- 特定の印刷ジョブのログエントリーを表示するには、次のように入力します。

```
# journalctl -u cups JID=<print_job_id>
```

- 特定の期間内のログエントリーを表示するには、次のように入力します。

```
# journalectl -u cups --since=<YYYY-MM-DD> --until=<YYYY-MM-DD>
```

YYYY は年に、**MM** は月に、**DD** は日に置き換えます。

関連情報

- **journalctl(1)** の man ページ

10.12. SYSTEMD ジャーナルではなくファイルにログを保存するように CUPS を設定する

デフォルトでは、CUPS はログメッセージを **systemd** ジャーナルに保存します。ログメッセージをファイルに保存するように CUPS を設定することもできます。

前提条件

- CUPS がインストールされている。

手順

1. **/etc/cups/cups-files.conf** ファイルを編集し、**AccessLog**、**ErrorLog**、および **PageLog** パラメーターを、これらのログファイルを保存するパスに設定します。

```
AccessLog /var/log/cups/access_log
ErrorLog /var/log/cups/error_log
PageLog /var/log/cups/page_log
```

2. **/var/log/cups/** 以外のディレクトリーにログを保存するように CUPS を設定する場合は、このディレクトリーに **cupd_log_t** SELinux コンテキストを設定します。たとえば、次のように設定します。

```
# semanage fcontext -a -t cupsd_log_t "/var/log/printing(/.*)?"
# restorecon -Rv /var/log/printing/
```

3. **cups** サービスを再起動します。

```
# systemctl restart cups
```

検証

1. ログファイルを表示します。

```
# cat /var/log/cups/access_log
# cat /var/log/cups/error_log
# cat /var/log/cups/page_log
```

2. **/var/log/cups/** 以外のディレクトリーにログを保存するように CUPS を設定した場合は、ログディレクトリーの SELinux コンテキストが **cupd_log_t** であることを確認します。

```
# ls -ldZ /var/log/printing/
drwxr-xr-x. 2 lp sys unconfined_u:object_r:cupsd_log_t:s0 6 Jun 20 15:55 /var/log/printing/
```

10.13. CUPS ドキュメントへのアクセス

CUPS では、CUPS サーバーにインストールされているサービスのドキュメントにブラウザーからアクセスできます。このドキュメントには次のものが含まれます。

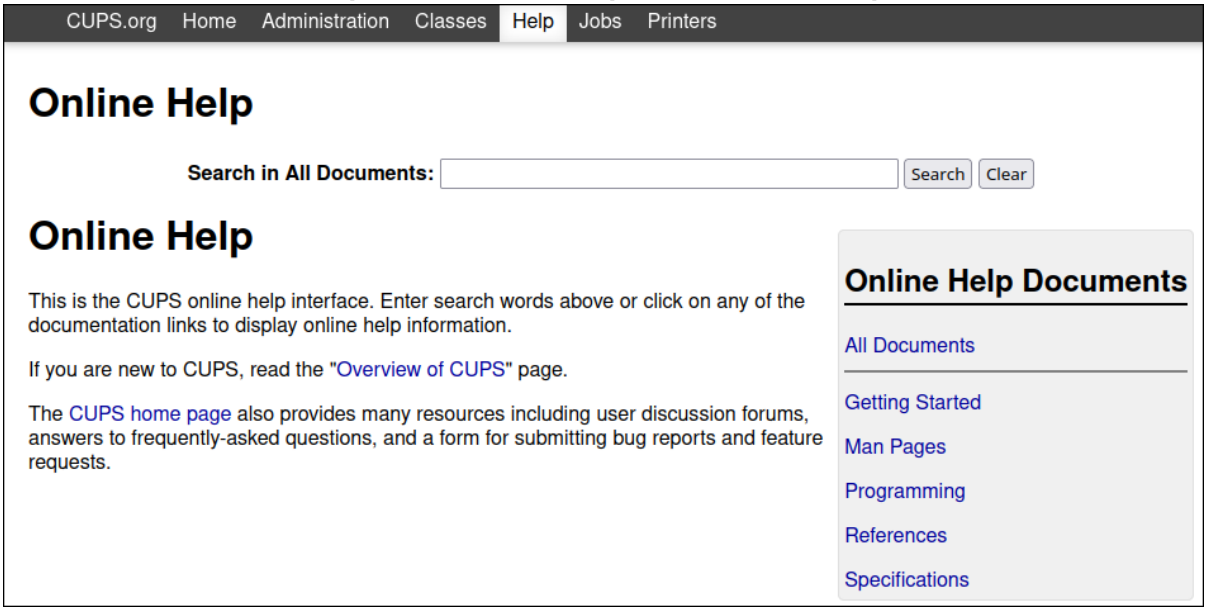
- コマンドラインによるプリンターの管理やアカウンティングなどに関する管理ドキュメント
- man ページ
- 管理 API などのプログラミングドキュメント
- 参考資料
- Specifications

前提条件

- CUPS がインストールされ、実行されている。
- 使用するクライアントの IP アドレスに、Web インターフェイスにアクセスする権限がある。

手順

1. ブラウザーを使用して、**http://<hostname_or_ip_address>:631/help/** にアクセスします。



2. **Online Help Documents** のエントリーを展開し、参照するドキュメントを選択します。