Red Hat Directory Server 11

管理ガイド

Directory Server の基本および高度な管理

Last Updated: 2024-07-07

Red Hat Directory Server 11 管理ガイド

Directory Server の基本および高度な管理

Marc Muehlfeld
Red Hat Customer Content Services
mmuehlfeld@redhat.com

Petr Bokoč
Red Hat Customer Content Services

Tomáš Čapek
Red Hat Customer Content Services

Petr Kovář
Red Hat Customer Content Services

Ella Deon Ballard
Red Hat Customer Content Services

法律上の通知

概要

本ガイドでは、Directory Server インスタンスおよびデータベースを管理する GUI およびコマンドラインの手順を説明します。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、「Red Hat CTO である Chris Wright のメッセージ」をご覧ください。

第1章 一般的な DIRECTORY SERVER 管理タスク

本章では、Directory Server インスタンスを管理する一般的なタスクを説明します。

1.1. システム要件

『Red Hat Directory Server 11 リリースノート』 の該当するセクションを参照してください。

1.2. ファイルの場所

『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の該当するセクションを参照してください。

1.3. DIRECTORY SERVER を設定するサポート対象のメソッド

以下を使用して Directory Server を設定できます。

- Directory Server が提供するコマンドラインユーティリティー
- Web コンソール

重要

コンソールのウィンドウの外でユーザーが設定を変更すると、Web コンソールでは最新の設定が自動的に表示されません。たとえば、Web コンソールが開いている間にコマンドラインを使用して設定を変更すると、Web コンソールで新しい設定が自動的に更新されません。これは、別のコンピューターの Web コンソールを使用して設定を変更する場合でも当てはまります。この問題を回避するには、コンソールウィンドウ外で設定が変更した場合に、ブラウザーで Web コンソールを手動で更新します。

1.4. WEB コンソールを使用した DIRECTORY SERVER へのログイン

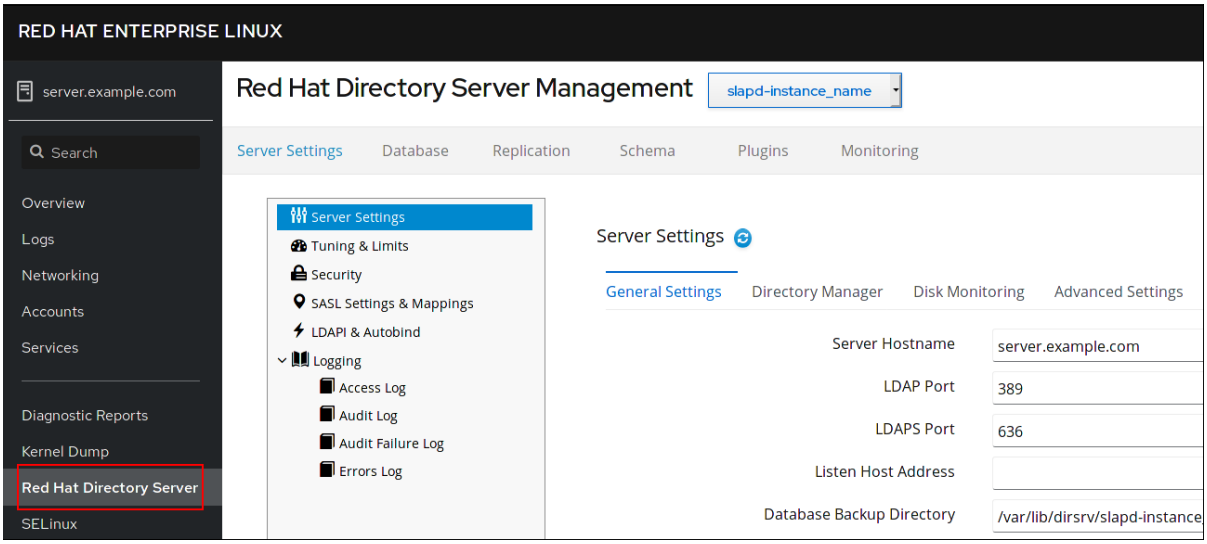
Web コンソールは、ユーザーが管理タスクを実行できるブラウザーベースのグラフィカルユーザーインターフェイス (GUI) です。Directory Server パッケージは、Web コンソールの Directory Server ユーザーインターフェイスを自動的にインストールします。

Web コンソールで Directory Server を開くには、以下を実行します。

1. ブラウザーを使用して、Directory Server ホストのポート 9090 で実行している Web コンソールに接続します。以下に例を示します。

   ```
   https://server.example.com:9090
   ```

2. **root** ユーザーまたは **sudo** 権限を持つユーザーとしてログインします。
3. **Red Hat Directory Server** エントリーを選択します。



1.5. DIRECTORY SERVER インスタンスの起動および停止

1.5.1. コマンドラインを使用した Directory Server インスタンスの起動および停止

dsctl ユーティリティーを使用して、インスタンスを開始、停止、または再起動します。

- インスタンスを起動するには、以下を実行します。

```
# dsctl instance_name start
```

- インスタンスを停止するには、以下を実行します。

```
# dsctl instance_name stop
```

- インスタンスを再起動するには、以下を実行します。

```
# dsctl instance_name restart
```

必要に応じて、システムの起動時に Directory Server インスタンスが自動的に起動するようにすることができます。

- 単一のインスタンスの場合:

```
# systemctl enable dirsrv@instance_name
```

- サーバー上のすべてのインスタンスの場合:

```
# systemctl enable dirsrv.target
```

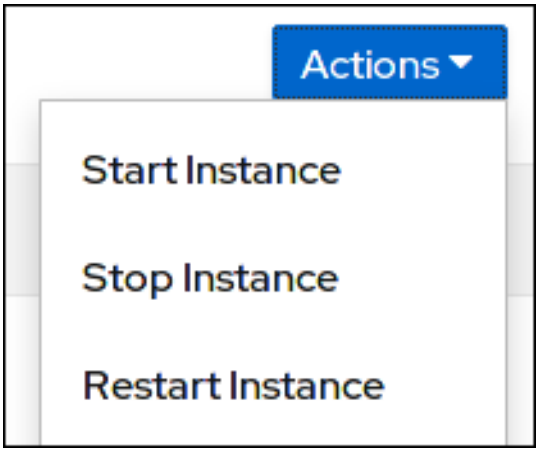
詳細は、『Red Hat システム管理者』 ガイドのシステムサービスの管理 セクションを参照してください。

1.5.2. Web コンソールを使用した Directory Server インスタンスの起動および停止

コマンドラインを除き、Web コンソールを使用してインスタンスの起動、停止、再起動を行うことができます。

Directory Server インスタンスを起動、停止、または再起動するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Actions** ボタンをクリックして、実行するアクションを選択します。
   - **Start Instance**
   - **Stop Instance**
   - **Restart Instance**



1.6. 新規 DIRECTORY SERVER インスタンスの作成

詳細は、『Red Hat Directory Server インストールガイド』の該当するセクションを参照してください。

- .inf ファイルを使用したコマンドラインで新規インスタンスの設定
- インタラクティブインストーラーを使用してコマンドラインで新規インスタンスの設定
- Web コンソールを使用した新規インスタンスの設定

1.7. DIRECTORY SERVER インスタンスの削除

サーバーで複数のインスタンスを実行する場合は、そのインスタンスを個別に削除できます。

インスタンスを削除すると、**/var/lib/dirsrv/slapd-*instance_name*/** および **/etc/dirsrv/slapd-*instance_name*/** ディレクトリーの内容が削除されます。

重要

/var/lib/dirsrv/slapd-*instance_name*/ ディレクトリーには、データベース、バックアップおよびエクスポートディレクトリーが含まれています。**/etc/dirsrv/slapd-*instance_name*/** ディレクトリーには、インスタンス設定とネットワークセキュリティーサービス (NSS) データベースが含まれています。インスタンスを削除する前に、このデータのバックアップを作成します。

1.7.1. コマンドラインを使用したインスタンスの削除

コマンドラインを使用してインスタンスを削除するには、以下を実行します。

```
# dsctl instance_name remove --do-it
Removing instance ...
Completed instance removal
```

1.7.2. Web コンソールを使用したインスタンスの削除

Web コンソールを使用してインスタンスを削除するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Actions** ボタンをクリックし、**Remove instance** を選択します。

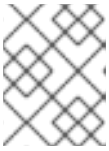
1.8. DIRECTORY SERVER の設定パラメーターの設定

Directory Server は、その設定を **cn=config** ディレクトリーエントリーに保存します。各設定パラメーターは LDAP 属性で、パラメーターの値はこの属性に設定した値になります。

1.8.1. 設定パラメーターの管理

以下を使用すると、設定パラメーターを設定、更新、および削除することができます。

- **dsconf** ユーティリティーの使用:



注記

Red Hat は、**dsconf** ユーティリティーを使用して、Directory Server 設定を管理することを推奨しています。

例1.1 dsconf を使用した設定パラメーターの設定

たとえば、エラーログレベルを **16384** に設定するには、**dsconf** ユーティリティーを使用して、***nsslapd-errorlog-level*** パラメーターを更新します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-errorlog-level=16384
```

dsconf の使用の詳細については、dsconf(8) man ページを参照してください。

- LDAP インターフェイスの使用:

例1.2 LDAP インターフェイスを使用した設定パラメーターの設定

たとえば、エラーログレベルを **16384** に設定するには、LDAP インターフェイスを使用して、***nsslapd-errorlog-level*** パラメーターを更新します。

```
# ldapmodify -D "cn=Directory Manager" -W -x -H ldap://server.example.com:389

dn: cn=config
replace: nsslapd-errorlog-level
nsslapd-errorlog-level: 16384
```

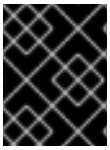
- */etc/dirsrv/slapd-instance_name/dse.ldif* ファイルを編集します。

警告

インスタンスが正常に起動している限り、このファイルは手動で編集しないでください。これは、Directory Server が想定どおりに機能しないか、インスタンスの起動に失敗する可能性があるためです。

1.8.2. Directory Server が設定を保存する場所

Directory Server は、**cn=config** エントリーからの設定を **/etc/dirsrv/slapd-*instance_name*/dse.ldif** ファイルに保存します。サーバーは、このファイルで変更したパラメーターのみを保存します。リストにない属性には、デフォルト値を使用します。これにより、**/etc/dirsrv/slapd-*instance_name*/dse.ldif** ファイルを表示して、このインスタンスに設定したすべての設定パラメーターを識別できます。



重要

インスタンスが正常に起動するかぎり、**/etc/dirsrv/slapd-*instance_name*/dse.ldif** ファイルを手動で編集しないでください。

設定パラメーターの編集方法は、「設定パラメーターの管理」を参照してください。

1.8.3. デフォルト値を使用する利点

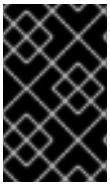
パラメーターが設定されていない場合、Directory Server はこのパラメーターのデフォルト値を使用します。デフォルト値を使用すると、新しいバージョンで最適化された設定が提供され、セキュリティーが強化されます。

たとえば、***passwordStorageScheme*** 属性を指定すると、Directory Server は、サポートされている利用可能な最も強力なパスワード保存スキームを自動的に使用します。今後の更新で、セキュリティーを向上させるデフォルト値を変更すると、パスワードを設定する際に、新しいストレージスキームを使用してパスワードが自動的に暗号化されます。

1.8.3.1. デフォルト値を使用するパラメーターの削除

パラメーターが設定され、代わりにデフォルト値を使用する場合は、パラメーターを削除します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config delete parameter_name
```



重要

nsslapd-secureport など、特定のパラメーターは、削除して、デフォルトにリセットすることはできません。それらを削除しようとすると、サーバーは **Server is unwilling to perform (53)** エラーでリクエストを拒否します。

1.8.4. dsconf config backend コマンドの制限事項

dsconf config backend コマンドは、バックエンド設定を取得および設定します。このコマンドには次の引数があります。

- get
- set

dsconf config backend get コマンドは、設定された値を持つすべてのサーバーバックエンド設定属性を取得します。次に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com:389 backend config get

nsslapd-lookthroughlimit: 5000
nsslapd-mode: 600
nsslapd-idlistscanlimit: 2147483646
…
```

注記

dsconf config backend get コマンドを使用すると、指定された属性の値ではなく、属性値の完全なセットのみを取得できます。

dsconf config backend set コマンドは、バックエンド設定属性を個別に設定します。値を設定するには、LDAP 属性名に一致するオプションを指定します。たとえば、次のようになります。

```
# dsconf -D "cn=Directory Manager"  ldap://server.example.com:389 backend config set --
lookthroughlimit 4000  --cache-autosize-split 24
```

以下に、**dsconf backend config set** コマンドのオプションと LDAP 属性名のマッピングを示します。

表1.1 dsconf backend config set コマンドのオプションと LDAP 属性名のマッピング

dsconf backend config set コマンドのオプション	LDAP 属性名
--lookthroughlimit	nsslapd-lookthroughlimit
--mode	nsslapd-mode
--idlistscanlimit	nsslapd-idlistscanlimit
--directory	nsslapd-directory
--dbcachesize	nsslapd-dbcachesize

dsconf backend config set コマンドのオプション	LDAP 属性名
--logdirectory	nsslapd-db-logdirectory
--txn-wait	nsslapd-db-transaction-wait
--checkpoint-interval	nsslapd-db-checkpoint-interval
--compactdb-interval	nsslapd-db-compactdb-interval
--compactdb-time	nsslapd-db-compactdb-time
--txn-batch-val	nsslapd-db-transaction-batch-val
--txn-batch-min	nsslapd-db-transaction-batch-min-wait
--txn-batch-max	nsslapd-db-transaction-batch-max-wait
--logbufsize	nsslapd-db-logbuf-size
--locks	nsslapd-db-locks
--locks-monitoring-enabled	nsslapd-db-locks-monitoring-enabled
--locks-monitoring-threshold	nsslapd-db-locks-monitoring-threshold
--locks-monitoring-pause	nsslapd-db-locks-monitoring-pause
--import-cache-autosize	nsslapd-import-cache-autosize
--import-cachesize	nsslapd-import-cachesize
--cache-autosize	nsslapd-cache-autosize
--cache-autosize-split	nsslapd-cache-autosize-split
--exclude-from-export	nsslapd-exclude-from-export
--pagedlookthroughlimit	nsslapd-pagedlookthroughlimit
--pagedidlistscanlimit	nsslapd-pagedidlistscanlimit
--rangelookthroughlimit	nsslapd-rangelookthroughlimit
--backend-opt-level	nsslapd-backend-opt-level

dsconf backend config set コマンドのオプション	LDAP 属性名
--deadlock-policy	nsslapd-db-deadlock-policy
--db-home-directory	nsslapd-db-home-directory
--db-lib	nsslapd-backend-implement

1.9. LDAP および LDAPS ポート番号の変更

デフォルトでは、Directory Server は LDAP にポート 389 を使用し、有効な場合は LDAPS プロトコルにポート 636 を使用します。たとえば、1 台のホストで複数の Directory Server インスタンスを実行するなど、これらのポート番号を変更できます。

重要

インスタンス用のプロトコルに割り当てた新しいポートは、他のサービスで使用することができません。

1.9.1. コマンドラインを使用したポート番号の変更

コマンドラインを使用してポート番号を変更するには、以下のパラメーターを更新します。

- ***nsslapd-port***: インスタンスが LDAP プロトコルに使用するポート番号を保存します。
- ***nsslapd-secureport***: インスタンスが LDAPS プロトコルに使用するポート番号を保存します。

コマンドラインを使用して LDAP プロトコルおよび LDAPS プロトコルのポート番号を変更するには、次を実行します。

1. 必要に応じて、インスタンスに現在設定されているポート番号を表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config get nsslapd-port
nsslapd-secureport
nsslapd-port: 389
nsslapd-secureport: 636
```

2. LDAP ポートを変更するには、以下を行います。

   a. LDAP プロトコルのポートを設定します。たとえば、**1389** に設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-port=1389
Successfully replaced "nsslapd-port"
```

   b. 前の手順で割り当てた LDAP ポートの **ldap_port_t** タイプを設定します。

```
# semanage port -a -t ldap_port_t -p tcp 1389
```

3. LDAPS ポートを変更するには、以下を実行します。

    a. LDAPS プロトコルのポートを設定します。たとえば、**1636** に設定するには、以下を実行します。

    ```
    # dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-secureport=1636
    Successfully replaced "nsslapd-secureport"
    ```

    b. 前の手順で割り当てた LDAPS ポートの **ldap_port_t** タイプを設定します。

    ```
    # semanage port -a -t ldap_port_t -p tcp 1636
    ```

4. インスタンスを再起動します。

```
# dsctl instance_name restart
```

1.9.2. Web コンソールを使用したポート番号の変更

Web コンソールを使用して LDAP プロトコルおよび LDAPS プロトコルのポート番号を変更するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. LDAP ポートを変更するには、以下を行います。
    a. **Server Settings** メニューを開きます。
    b. **Server Settings** タブで、新しいポート番号を **LDAP Port** フィールドに入力します。
    c. **Save** をクリックします。
4. LDAPS ポートを変更するには、以下を実行します。
    a. **Server Settings** メニューを開きます。
    b. **General Settings** タブで、新しいポート番号を **LDAPS Port** フィールドに入力します。
    c. **Save** をクリックします。
5. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

1.10. DIRECTORY SERVER プラグインの使用

Directory Server は、レプリケーション、サービスのクラス、属性構文の検証など、コアプラグインを複数提供します。コアプラグインはデフォルトで有効になっています。

さらに、Directory Server パッケージには、属性の一意性や属性リンクなど機能を強化するプラグインが含まれます。ただし、これらすべてのプラグインがデフォルトで有効になっている訳ではありません。

詳細は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 の 『プラグイン実装サーバー機能リファレンス』 の章を参照してください。

1.10.1. 利用可能なプラグインのリスト表示

1.10.1.1. コマンドラインを使用した利用可能なプラグインのリスト表示

コマンドラインを使用して利用可能なプラグインのリストを表示するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin list
7-bit check
Account Policy Plugin
...
```

たとえば、コマンドラインでこれを有効または無効にするには、プラグインの正確な名前が必要です。

1.10.1.2. Web コンソールを使用した利用可能なプラグインのリスト表示

Web コンソールを使用して利用可能なプラグインをすべて表示するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを選択します。

オプションで、**Filter Plugins** フィールドに名前を入力して、プラグインをフィルタリングできます。

1.10.2. プラグインの有効化および無効化

1.10.2.1. コマンドラインでプラグインの有効化および無効化

コマンドラインを使用してプラグインを有効または無効にするには、**dsconf** ユーティリティーを使用します。

注記

dsconf コマンドでは、プラグインの名前を指定する必要があります。すべてのプラグインの名前を表示する場合は、「コマンドラインを使用した利用可能なプラグインのリスト表示」を参照してください。

たとえば、**Automember** プラグインを有効にするには、次のようにします。

1. プラグインを有効にします。

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

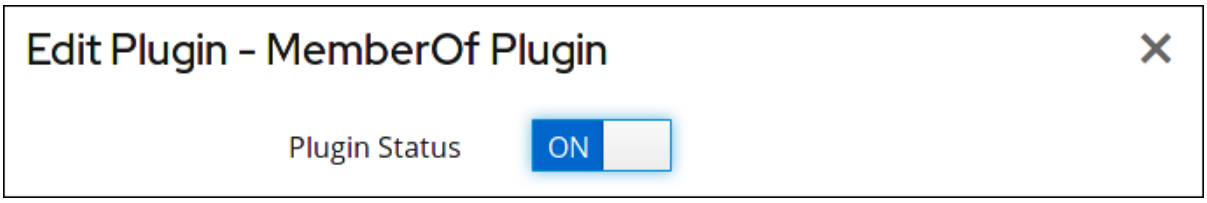
2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

1.10.2.2. Web コンソールを使用したプラグインの有効化および無効化

Web コンソールを使用してプラグインを有効または無効にするには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを選択します。
4. **All Plugins** タブを選択します。
5. 有効または無効にするプラグインの右側にある **Edit Plugin** ボタンをクリックします。
6. ステータスを **ON** に変更してプラグインを有効にするか、**OFF** に変更して無効にします。



7. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

1.10.3. プラグインの設定

1.10.3.1. コマンドラインでプラグインの設定

プラグイン設定を行うには、**dsconf plugin** コマンドを使用します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin \
    plug-in-specific_subcommand ...
```

設定可能なプラグインのリストを表示するには、以下を実行します。

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

1.10.3.2. Web コンソールを使用したプラグインの設定

Web コンソールを使用してプラグインを設定するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを選択します。
4. **All Plugins** タブを選択します。
5. プラグインを選択し、**Show Advanced Settings** をクリックします。

6. プラグイン固有のタブを開きます。
7. 適切な設定を行います。
8. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

1.10.4. プラグインの優先順位の設定

プラグインの優先順位は、プラグインの実行順序にある優先順位です。操作前および操作後のプラグインでは、次のプラグインの開始前にプラグインを実行して完了し、次のプラグインが、その前に実行したプラグインの結果を活用できるようにします。

優先順位は、**1** (最高の優先順位) から **99** (最低の優先順位) までの値に設定できます。優先順位が設定されていない場合、デフォルトは **50** です。

警告

カスタムプラグインでのみ優先順位を設定します。コアプラグインの値を更新すると、Directory Server は期待通りに機能しなくなり、Red Hat では対応していません。

1.10.4.1. コマンドラインを使用したプラグインの優先順位の設定

コマンドラインを使用してプラグインの優先順位を更新するには、次のコマンドを実行します。

1. プラグインの優先順位を設定します。たとえば、**example** プラグインの優先順位を **1** に設定するには、次のようにします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin edit example --precedence 1
```

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

1.10.4.2. Web コンソールを使用したプラグインの優先順位の設定

Web コンソールを使用してプラグインの優先順位を更新するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **All Plugins** を選択します。
5. 優先順位の値を設定するプラグインの横にある **Edit Plugin** ボタンを押します。

6. **Plugin Precedence** フィールドの値を更新します。
7. **Save** をクリックします。
8. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

1.11. .DSRC ファイルを作成して使用し、DIRECTORY SERVER コマンドラインユーティリティーのデフォルトオプションを設定する

~/**.dsrc** ファイルは、Directory Server コマンドラインユーティリティーを使用するコマンドを簡素化します。デフォルトでは、これらのユーティリティーでは、LDAP URL やバインド識別名 (DN) などをコマンドに渡す必要があります。これらの設定を ~/**dsrc** ファイルに保存すると、毎回これらの設定を指定しなくても、コマンドラインユーティリティーを使用できます。

1.11.1. .dsrc ファイルがコマンドを簡素化する方法

以下は、インスタンスの LDAP URL とバインド DN を指定する ~/**.dsrc** ファイルの例です。

```
[server1]
uri = ldap://server1.example.com
binddn = cn=Directory Manager
basedn = dc=example,dc=com
```

これらの設定により、より短い Directory Server コマンドを使用できます。たとえば、ユーザーアカウントを作成するには、次のようにします。

```
# dsidm server1 user create
```

~/**.dsrc** ファイルがない場合は、コマンドでバインド DN、LDAP URL、およびベース DN を指定する必要があります。

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

1.11.2. dsctl ユーティリティーを使用して .dsrc ファイルを作成する

~/**.dsrc** ファイルを手動で作成する代わりに、**dsctl** ユーティリティーを使用して作成できます。

```
# dsctl instance_name dsrc create ...
```

次のオプションをコマンドに渡すことができます。

- **--uri**: ***protocol*://*host_name_or_IP_address_or_socket*** の形式で URL をインスタンスに設定します。

  例 :

  - **--uri ldap://*server.example.com***
  - **--uri = ldaps://*server.example.com***
  - **--uri = ldapi://%%2fvar%%2frun%%2fslapd-*instance_name*.socket**

Directory Server ソケットへのパスを設定する場合は、パスにスラッシュ (/) ではなく **%%02** を使用します。

重要

ldapi URL を使用すると、サーバーは、Directory Server コマンドラインユーティリティーを実行するユーザーのユーザー ID (UID) とグループ ID (GID) を識別します。**root** ユーザーとしてコマンドを実行すると、UID と GID の両方が **0** になり、Directory Server は、対応するパスワードを入力しなくても、**cn=Directory Manager** としてユーザーを自動的に認証します。

- **--starttls**: LDAP ポートに接続し、**STARTTLS** コマンドを送信して、暗号化された接続に切り替えるように、ユーティリティーを設定します。
- **--basedn**: 基本識別名 (DN) を設定します。例: **--basedn *dc=example,dc=com***
- **--binddn**: バインド DN を設定します。例: **--binddn *cn=Directory Manager***
- **--pwdfile**: バインド DN のパスワードを含むファイルへのパスを設定します。例: **--pwdfile */root/rhds.pwd***
- **--tls-cacertdir**: LDAPS 接続を使用する場合、このパラメーターに設定されたパスは、サーバーの証明書を検証するために必要な認証局 (CA) 証明書を含むディレクトリーを定義します。例: **--tls-cacertdir */etc/pki/CA/certs/***

  指定したディレクトリーに CA 証明書をコピーした後、**c_rehash */etc/pki/CA/certs/*** コマンドを使用する必要があることに注意してください。
- **--tls-cert**: サーバーの証明書への絶対パスを設定します。例: **--tls-cert */etc/dirsrv/slapd-instance_name/Server-Cert.crt***
- **--tls-key**: サーバーの秘密鍵への絶対パスを設定します。例: **--tls-key */etc/dirsrv/slapd-instance_name/Server-Cert.key***
- **--tls-reqcert**: クライアントユーティリティーが TLS セッションでサーバー証明書に対して実行するチェックを設定します。例: **--tls-reqcert *hard***

  以下のパラメーターを利用することができます。

  - **never**: ユーティリティーはサーバー証明書を要求または確認しません。
  - **allow**: ユーティリティーは証明書エラーを無視し、接続はとにかく確立されます。
  - **hard**: ユーティリティーは証明書エラーで接続を終了します。
- **--saslmech**: 使用する SASL メカニズムを **PLAIN** または **EXTERNAL** に設定します。例: **--saslmech PLAIN**

1.11.3. Directory Server ユーティリティー使用時のリモートおよびローカル接続の解決

Directory Server 接続を保護する場合、Directory Server コマンドのリモート呼び出しとローカル呼び出しを区別することが重要です。

LDAP URL を指定して Directory Server コマンドを実行すると、サーバーは、それをリモート接続と見なし、コマンドを続行するために、システム全体の設定とともに **/etc/openldap/ldap.conf** 設定ファイルをチェックします。

インスタンス名を指定して Directory Server コマンドを実行すると、サーバーは、~/**.dsrc** ファイルが存在するかどうかを確認し、次のロジックを適用して続行します。

- ~/**.dsrc** ファイルが存在し、インスタンス名と LDAP URL の両方が含まれている場合、Directory Server は、それをリモート接続と見なし、**/etc/openldap/ldap.conf** 設定ファイルとシステム全体の設定をチェックします。

- ~/**.dsrc** ファイルが存在し、指定されたインスタンス名のみが含まれている場合、または ~/**.dsrc** ファイルが存在しない場合、Directory Server は、それをローカル接続と見なし、ローカルの **dse.ldif** ファイルの ***nsslapd-certdir*** 設定を使用して、接続を保護します。***nsslapd-certdir*** が存在しない場合、サーバーは、デフォルトパス **/etc/dirsrv/slapd-*instance_name*/** を使用して、インスタンスの Network Security Services (NSS) データベースを保存します。

nsslapd-certdir パラメーターの詳細については、nsslapd-certdir (証明書と鍵のデータベースディレクトリー) セクションを参照してください。

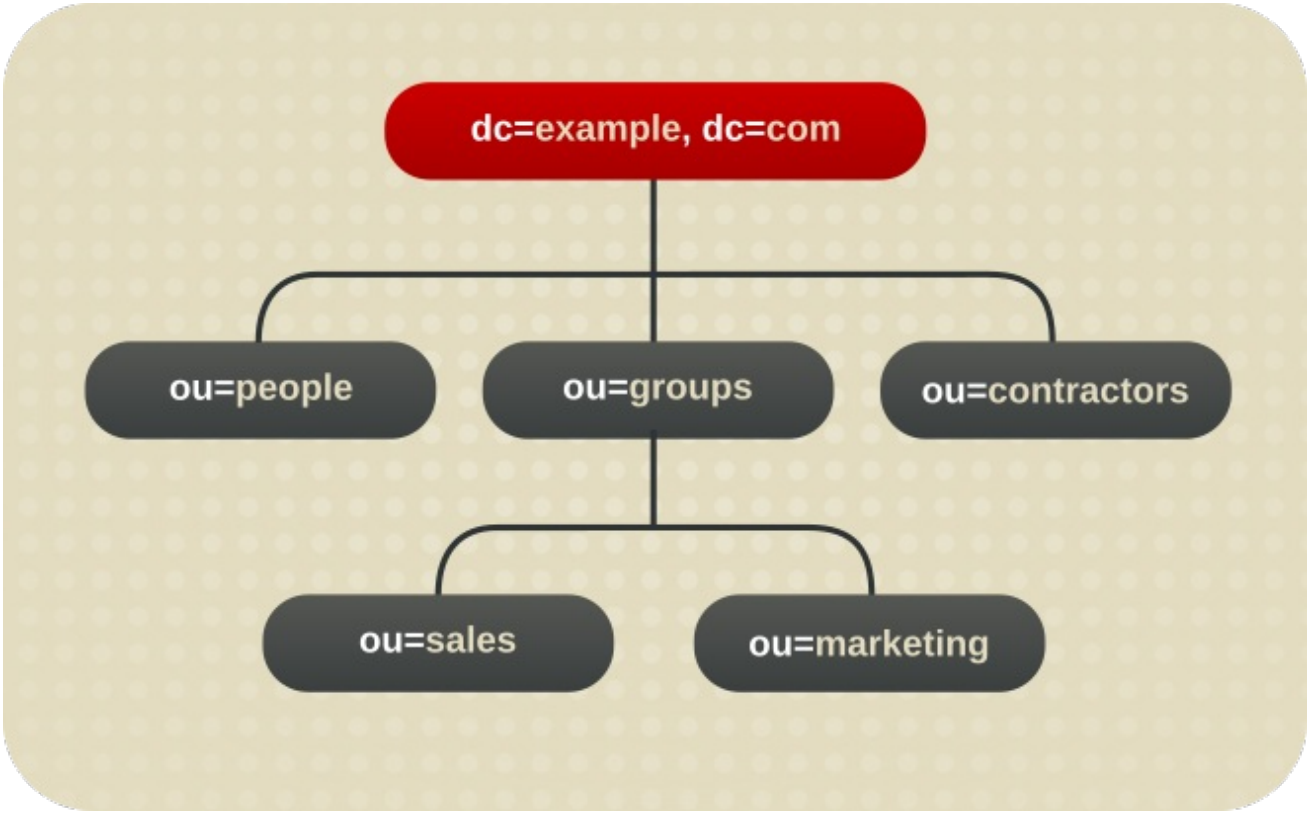
第2章 ディレクトリーデータベースの設定

ディレクトリーはデータベースに保存され、ディレクトリーツリーはデータベース全体に分散されます。本章では、**接尾辞** の作成、ディレクトリーツリーの分岐点、および各接尾辞に関連付けられたデータベースの作成方法を説明します。本章では、リモートサーバーでデータベースを参照するデータベースリンクを作成する方法と、参照を使用してクライアントにディレクトリーデータの外部ソースを指定する方法も説明します。

2.1. 接尾辞の作成および維持

ディレクトリーツリーのさまざまな部分をさまざまなデータベースに保存でき、そのデータベースを複数のサーバーに分散できます。ディレクトリーツリーには、**ノード** と呼ばれる分岐点が含まれます。このノードはデータベースに関連付けられている可能性があります。接尾辞は、特定のデータベースに関連するディレクトリーツリーのノードです。以下は、簡単なディレクトリーツリーです。

図2.1 つのルート接尾辞があるディレクトリーツリー



ou=people 接尾辞とその下のすべてのエントリーとノードが 1 つのデータベースに保存され、**ou=groups** 接尾辞が別のデータベースに保存され、**ou=contractors** 接尾辞がさらに別のデータベースに保存される場合があります。

2.1.1. 接尾辞の作成

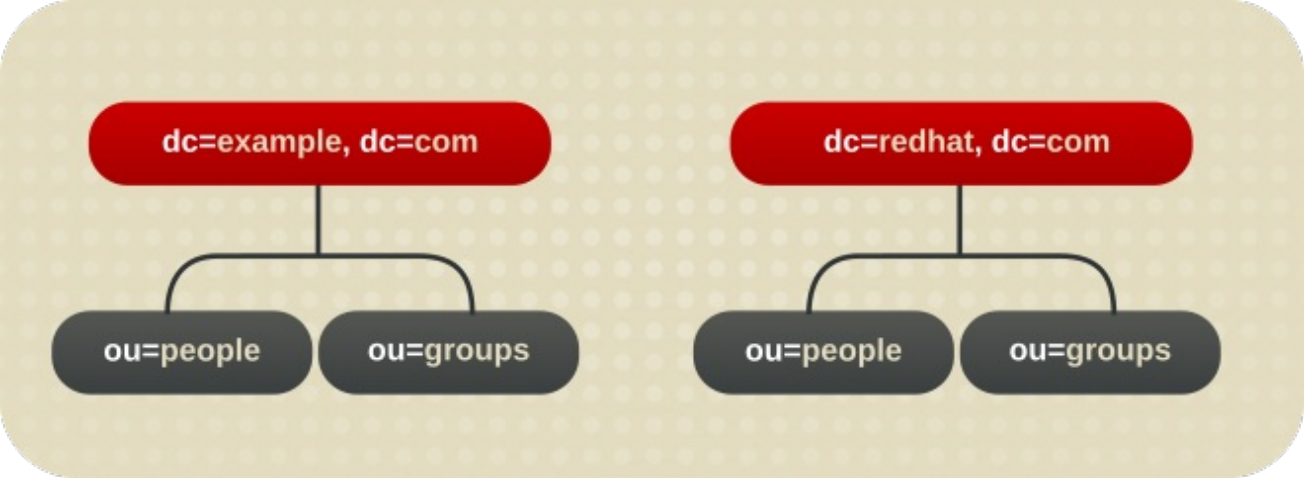
ルートの接尾辞 は、従属接尾辞の親です。これは、Directory Server 用に設計された大規模なツリーの一部になります。**下位スキーム** は、ルート接尾辞の配下にあるブランチです。ディレクトリーツリーのコンテンツの編成には、ルート接尾辞と従属接尾辞の両方が使用されます。ルートおよびサブスキームのデータはデータベースに保存されます。

2.1.1.1. ルート接尾辞の作成

ディレクトリーには、複数のルート接尾辞を含めることができます。たとえば、**example.com** 用と

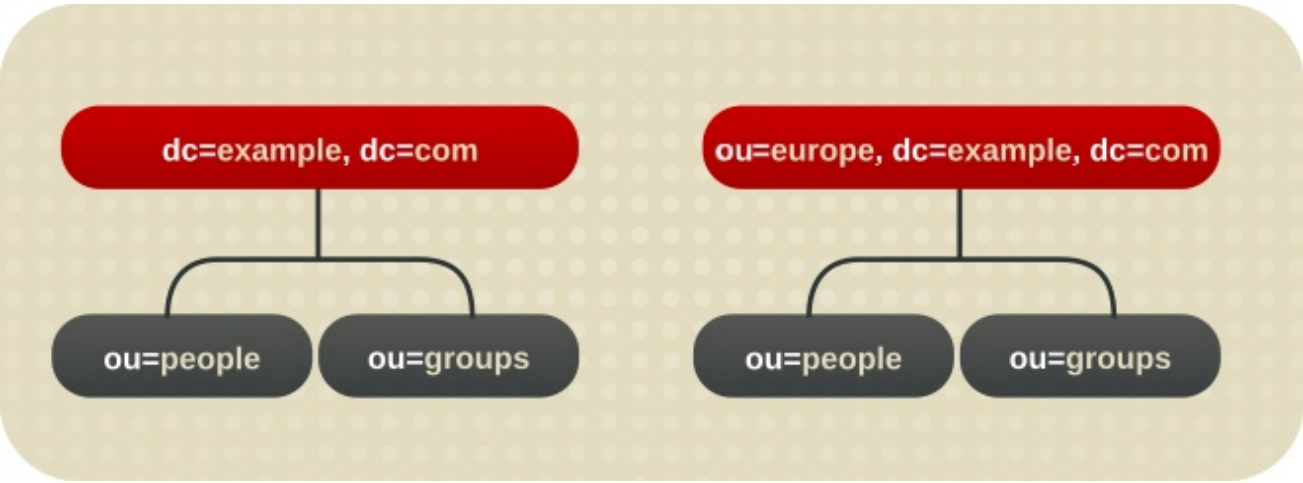
redhat.com 用の複数の Web サイトをホスティングするインターネットサービスプロバイダーなどです。このシナリオでは、2 つのルート接尾辞が必要です。次の図に示すように、1 つは **dc=example,dc=com** 命名コンテキストに対応し、もう 1 つは **dc=redhat,dc=com** 命名コンテキストに対応します。

図2.2 2 つのルート接尾辞があるディレクトリー



また、検索操作からディレクトリーツリーの一部を除外するために、ルート接尾辞を作成することもできます。たとえば、Example Corporation が、一般的な Example Corporation ディレクトリーの検索から、ヨーロッパオフィスを除外する場合などです。これを実装するには、ディレクトリーに 2 つのルート接尾辞が必要です。1 つのルート接尾辞は一般的な Example Corporation ディレクトリーツリー **dc=example,dc=com** に対応し、もう 1 つのルート接尾辞はディレクトリーツリーのヨーロッパブランチ **ou=europe,dc=example,dc=com** に対応します。クライアントアプリケーションの観点では、ディレクトリーツリーは以下の図のように示されます。

図2.3 検索操作への制限がオフのルート接尾辞があるディレクトリー



ディレクトリーの **dc=example,dc=com** ブランチでクライアントアプリケーションによって実行される検索では、ディレクトリーの **ou=europe,dc=example,dc=com** ブランチからのエントリーは返されません。これは、別のルート接尾辞であるためです。

2.1.1.1.1. コマンドラインでルート接尾辞の作成

dsconf backend create コマンドを使用して、新しいルート接尾辞を作成します。

1. 必要に応じて、使用中の接尾辞およびバックエンドデータベースを指定します。

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

括弧内の名前は、対応する接尾辞のデータを保存するバックエンドデータベースです。次の手順でルートの接尾辞を作成する場合は、既存のデータベース名を使用することはできません。

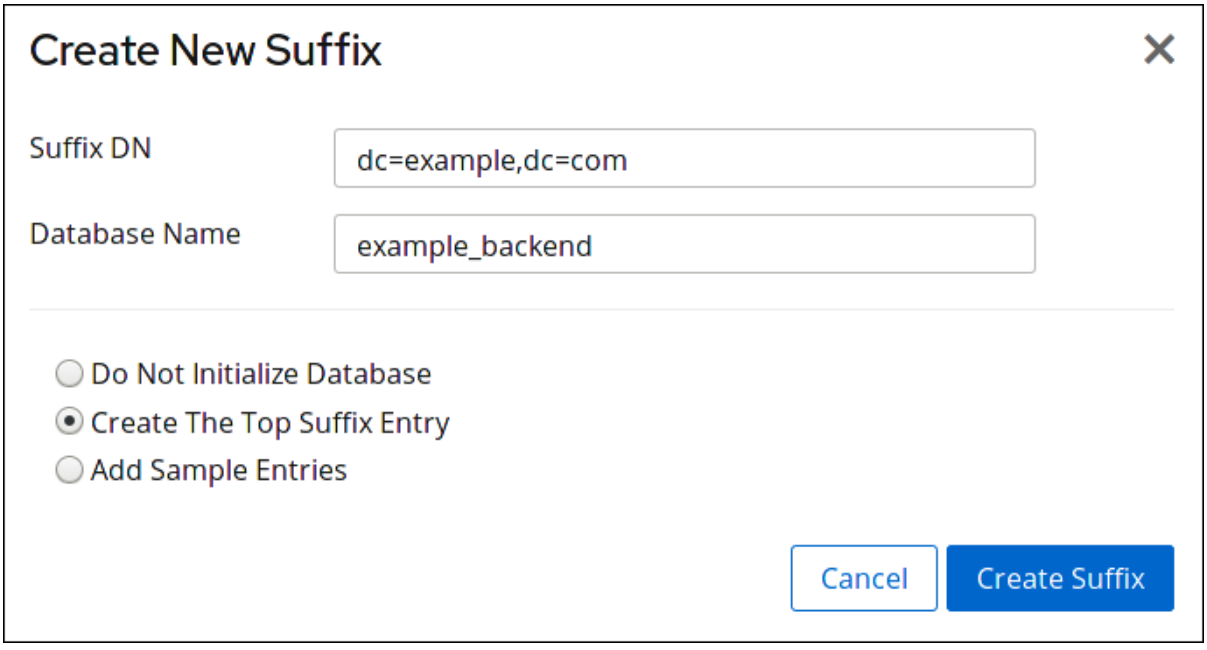
2. **example** バックエンドデータベースに **dc=example,dc=net** ルート接尾辞を作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend create \
    --suffix="dc=example,dc=net" --be-name="example"
```

2.1.1.1.2. Web コンソールを使用したルート接尾辞の作成

Web コンソールを使用して新しいルート接尾辞を作成するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. **Create Suffix** をクリックします。
5. 接尾辞 DN およびバックエンド名を入力します。以下に例を示します。



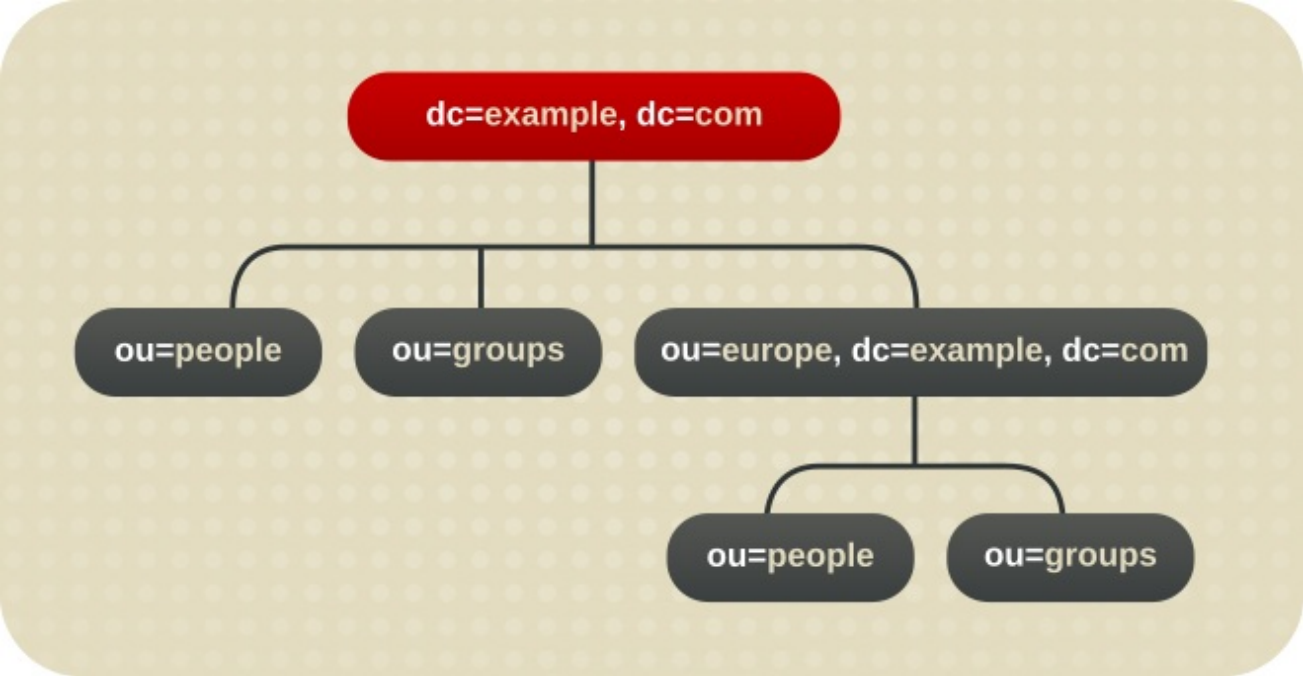
6. **Create The Top Suffix Entry** を選択します。
7. **Create Suffix** をクリックします。

2.1.1.2. 従属接尾辞の作成

特定のシナリオでは、管理者はディレクトリーツリーのブランチを別のデータベースに保存するとします。たとえば、管理者が **ou=europe,dc=example,dc=com** エントリーをサブ接尾辞として作成すると、この接尾辞は別のデータベースに保存されます。同時に、**dc=example,com** ルート接尾辞とその

すべてのサブエントリー (**ou=europe,dc=example,dc=com** とサブエントリーを除く) も別のデータベースに保存されます。

図2.4 従属接尾辞が含まれるディレクトリーツリー



2.1.1.2.1. コマンドラインを使用した従属接尾辞の作成

dsconf backend create コマンドを使用して、新しいサブ接尾辞を作成します。たとえば、**dc=example,dc=com** ルート接尾辞の下にある **people** という名前の新しいデータベースに **ou=People,dc=example,dc=com** サブ接尾辞を作成するには、次のようにします。

1. 必要に応じて、使用中の接尾辞およびバックエンドデータベースを指定します。

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

括弧内の名前は、対応する接尾辞のデータを保存するバックエンドデータベースです。次の手順で従属接尾辞を作成する場合は、既存のデータベース名を使用することはできません。

2. 従属接尾辞を作成します。たとえば、**example** バックエンドデータベースとともに **ou=People,dc=example,dc=com** サブ接尾辞を作成するには、次のように入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend create \
    --suffix="ou=People,dc=example,dc=com" --be-name="example" \
    --parent-suffix="dc=example,dc=com"
```

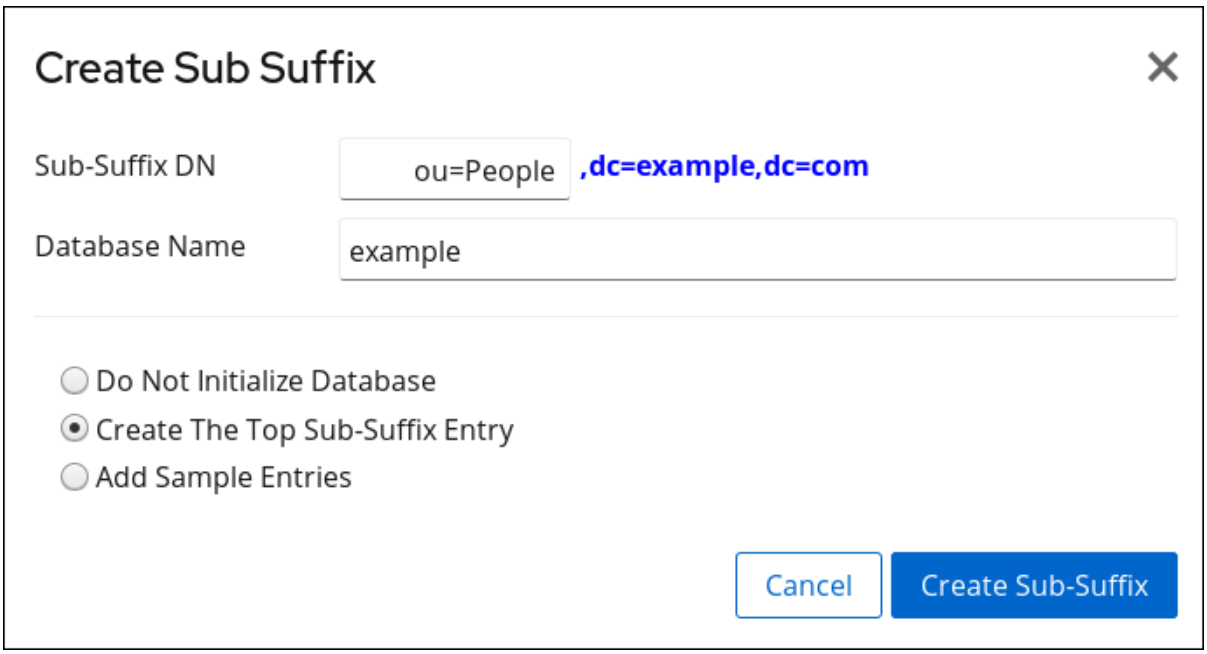
2.1.1.2.2. Web コンソールを使用した副接尾辞の作成

Web コンソールを使用して新しい従属接尾辞を作成するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。

2. インスタンスを選択します。

3. **Database** メニューを開きます。
4. サブ接尾辞を作成する接尾辞を選択し、**Suffix Tasks** をクリックして、**Create Sub-Suffix** を選択します。
5. 従属接尾辞 DN およびバックエンド名を入力します。以下に例を示します。



6. **Create The Top Sub-Suffix Entry** を選択します。
7. **Create Sub-Suffix** をクリックします。

2.1.2. 接尾辞の維持

2.1.2.1. デフォルトの命名コンテキストの表示

命名コンテキストは接尾辞に類似しており、命名ディレクトリーエントリーのルート構造です。ディレクトリーとデータ構造によっては、複数の命名コンテキストが存在する場合があります。たとえば、標準の Directory Server 設定には、**dc=example,dc=com** などのユーザー接尾辞や **cn=config** の設定接尾辞があります。

多くのディレクトリーツリーには複数の命名コンテキストがあり、異なるタイプのエントリーや論理データ分割で使用されます。Directory Server にアクセスするクライアントは、使用する必要がある命名コンテキストを認識しない場合があります。Directory Server には、デフォルトの命名コンテキストが他に認識されていない場合に、デフォルトの命名コンテキストがクライアントに通知するサーバー設定属性があります。

デフォルトの命名コンテキストは、**cn=config** の ***nsslapd-defaultnamingcontext*** 属性に設定されます。この値はルート DSE (Directory Server Agent Service Entry) に伝播され、ルート DSE の ***defaultnamingcontext*** 属性を確認してクライアントが匿名でクエリーできます。

```
# ldapsearch -p 389 -h server.example.com -x -b "" -s base | egrep namingcontext
namingContexts: dc=example,dc=com
namingContexts: dc=example,dc=net
namingContexts: dc=redhat,dc=com
defaultnamingcontext: dc=example,dc=com
```

重要

設定の整合性を維持するには、***nsslapd-allowed-to-delete-attrs*** 一覧から ***nsslapd-defaultnamingcontext*** 属性を削除しないでください。

デフォルトでは、***nsslapd-defaultnamingcontext*** 属性は、***nsslapd-allowed-to-delete-attrs*** 属性に削除 **できる** 属性の一覧に含まれます。これにより、現在のデフォルトの接尾辞を削除してから、適切にサーバー設定を更新できます。

何らかの理由で削除可能な設定属性の一覧から ***nsslapd-defaultnamingcontext*** 属性を削除すると、その属性への変更は保持されません。デフォルトの接尾辞を削除すると、その変更はサーバー設定に伝播できません。つまり、***nsslapd-defaultnamingcontext*** 属性は、空白 (削除) ではなく古い情報を保持することを意味します。これは正しい現在の設定です。

2.1.2.2. 接尾辞の無効化

特定の状況では、ディレクトリーの接尾辞を無効にする必要があります。接尾辞が無効になっていると、その接尾辞に関連するデータベースのコンテンツは、クライアントがアクセスできなくなります。

2.1.2.2.1. コマンドラインでの接尾辞の無効化

コマンドラインで接尾辞を無効にするには、バックエンドデータベース名を **dsconf backend suffix set --disable** コマンドに渡します。たとえば、**o=test** 接尾辞を無効にするには、以下を実行します。

1. 接尾辞とそれに対応するバックエンドを表示します。

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

このコマンドにより、各接尾辞の横にバックエンドデータベースが表示されます。次の手順で、接尾辞のデータベース名が必要です。

2. 接尾辞を無効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend \
    suffix set --disable "test_database"
```

2.1.2.3. 接尾辞の削除

接尾辞が不要になった場合、管理者はその接尾辞をデータベースから削除できます。

警告

接尾辞を削除すると、その接尾辞に関連するデータベースエントリーおよびレプリケーション情報もすべて削除されます。

2.1.2.3.1. コマンドラインを使用した接尾辞の削除

コマンドラインで接尾辞を削除するには、**dsconf backend delete** コマンドを使用します。たとえば、**o=test** 接尾辞を削除するには、以下を実行します。

1. 接尾辞とそれに対応するバックエンドを表示します。

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

このコマンドにより、各接尾辞の横にバックエンドデータベースが表示されます。次の手順で、接尾辞のデータベース名が必要です。

2. バックエンドデータベースと対応する接尾辞を削除します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend delete
test_database
Deleting Backend cn=test_database,cn=ldbm database,cn=plugins,cn=config :
Type 'Yes I am sure' to continue: Yes I am sure
The database, and any sub-suffixes, were successfully deleted
```

2.1.2.3.2. Web コンソールを使用した接尾辞の削除

Web コンソールを使用して接尾辞を削除するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. 接尾辞を選択し、**Suffix Tasks** をクリックして **Delete Suffix.** を選択します。
5. **Yes** をクリックして確定します。

2.2. データベースの作成および維持

ディレクトリーデータを整理するための接尾辞を作成したら、そのディレクトリーのデータを含むデータベースを作成します。

注記

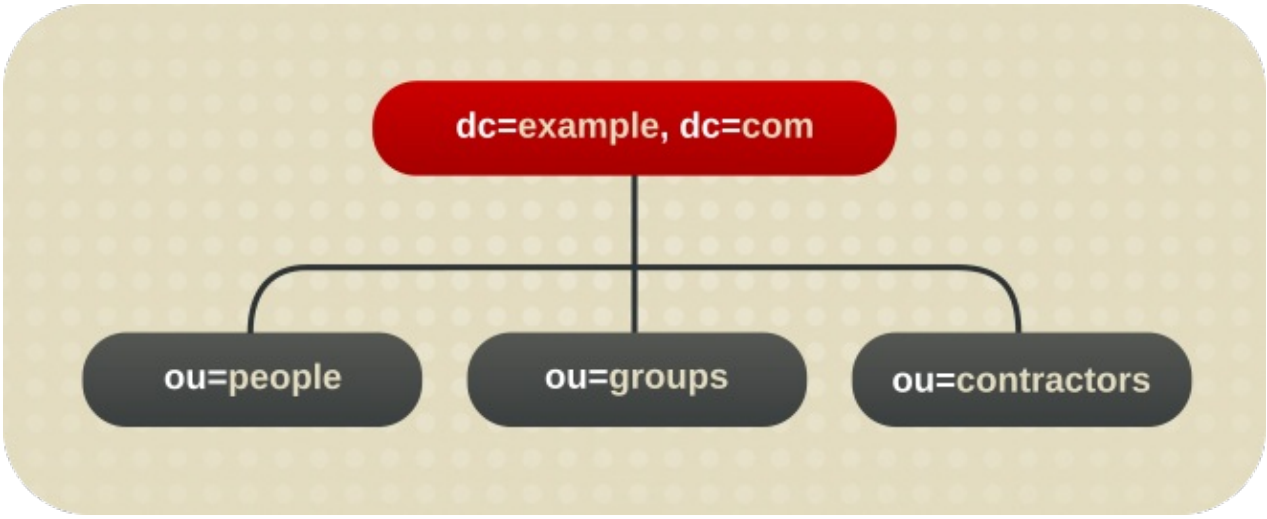
dsconf ユーティリティーまたは Web コンソールを使用して接尾辞を作成すると、Directory Server はデータベースを自動的に作成していました。

2.2.1. データベースの作成

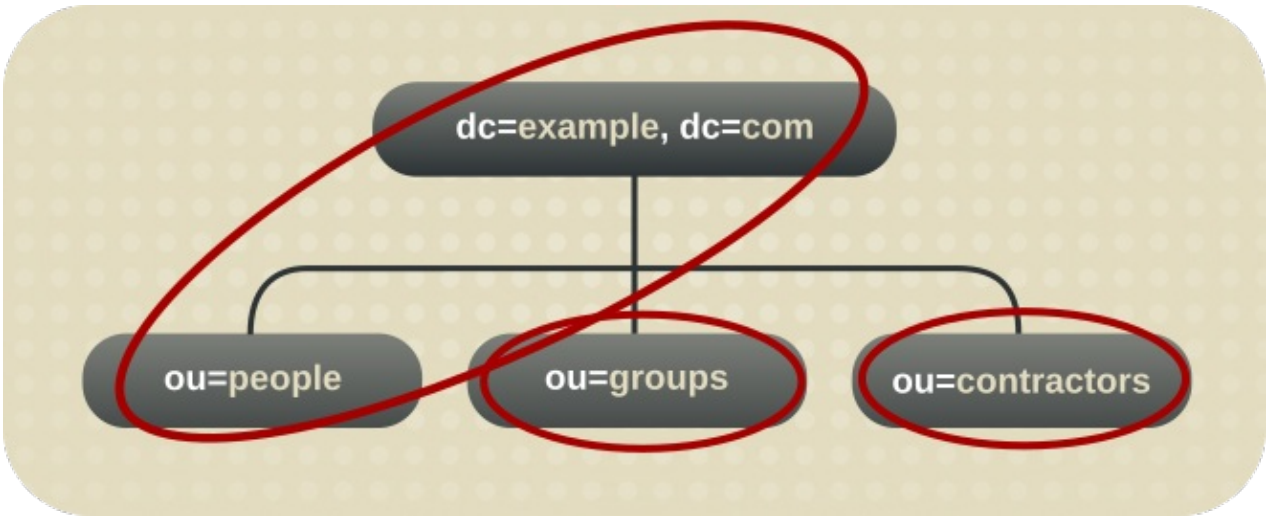
ディレクトリーツリーは、複数の Directory Server データベースに配布できます。複数のデータベースにデータを分散する方法は 2 つあります。

各接尾辞に 1 つのデータベース各接尾辞のデータは個別のデータベースに含まれます。

個別の接尾辞に含まれるデータを格納するために、3 つのデータベースが追加されます。



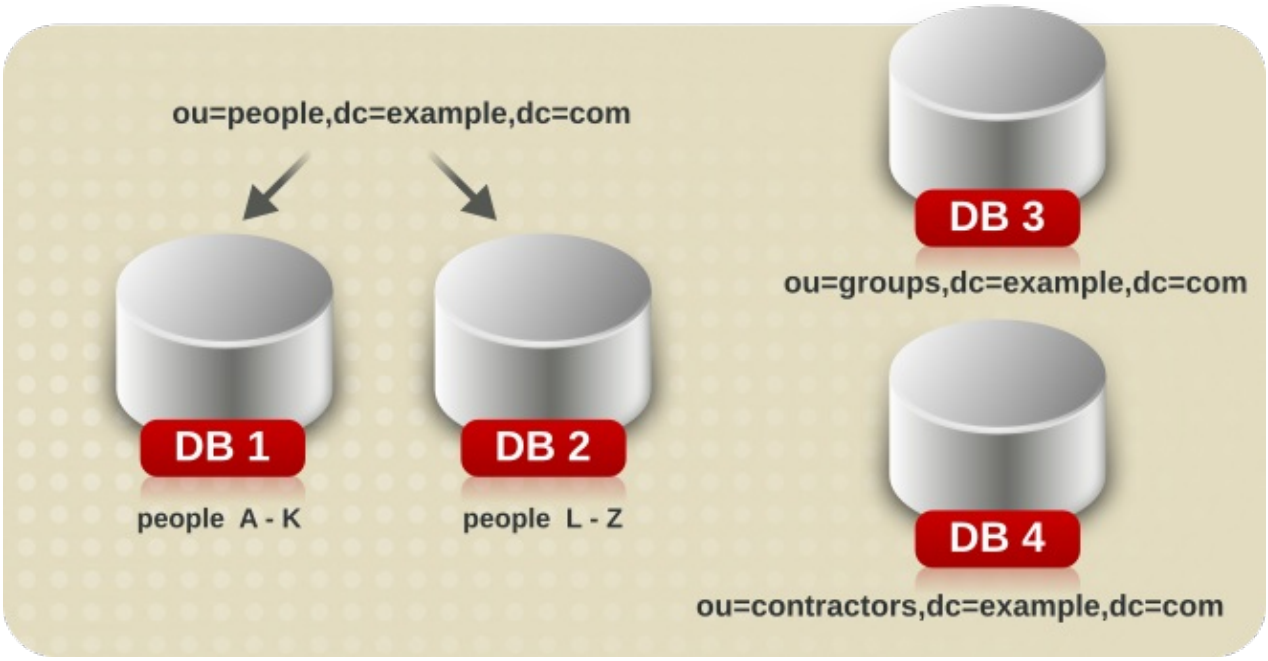
このツリーユニットの分割は、たとえば次の 3 つのデータベースに対応しています。



この例では、DB1 には **ou=people** のデータおよび **dc=example,dc=com** のデータが含まれ、クライアントが **dc=example,dc=com** に基づいて検索を実行できるようにします。ただし、DB2 には **ou=groups** のデータのみが含まれ、DB3 には **ou=contractors** のデータのみが含まれます。

1 つの接尾辞に複数のデータベースがあります。

ディレクトリーツリーの **ou=people** ブランチ内のエントリー数が非常に大きくなると、2 つのデータベースを格納しなければならないとします。この場合、**ou=people** に含まれているデータは 2 つのデータベースに分散できます。



DB1 には **A-K** からの名前の人が含まれ、DB2 には **L-Z** からの名前が含まれます。DB3 には **ou=groups** のデータが含まれ、DB4 には **ou=contractors** のデータが含まれます。

カスタムプラグインは、複数のデータベースにまたがってデータを単一の接尾辞から分散します。Directory Server のディストリビューションロジックの作成方法は、Red Hat コンサルティングにお問い合わせください。

2.2.1.1. コマンドラインを使用した単一の接尾辞用の新規データベースの作成

ldapmodify コマンドラインユーティリティーを使用して、ディレクトリー設定ファイルに新しいデータベースを追加します。データベース設定情報は **cn=ldbm database,cn=plugins,cn=config** エントリーに保存されます。新しいデータベースを追加するには、以下を実行します。

1. **ldapmodify** を実行して、新規データベースのエントリーを作成します。

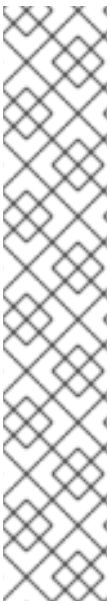
```
# ldapmodify -a -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=UserData,cn=ldbm database,cn=plugins,cn=config
changetype: add
objectclass: extensibleObject
objectclass: nsBackendInstance
nsslapd-suffix: ou=people,dc=example,dc=com
```

   追加されたエントリーは **UserData** という名前のデータベースに対応します。このデータベースには root 接尾辞または副接尾辞 **ou=people,dc=example,dc=com** のデータが含まれます。

2. 「コマンドラインでルート接尾辞の作成」 および 「コマンドラインを使用した従属接尾辞の作成」 で説明されているように、ルートまたは従属接尾辞を作成します。DN 属性に指定されたデータベース名は、接尾辞エントリーの ***nsslapd-backend*** 属性の値に対応している必要があります。

2.2.1.2. 単一の接尾辞に複数のデータベースの追加

1 つの接尾辞は、複数のデータベースに分散できます。ただし、接尾辞を配布するには、ディレクトリーを拡張するためにカスタムディストリビューション機能を作成する必要があります。カスタムディストリビューション機能の作成に関する詳細は、Red Hat コンサルティングにお問い合わせください。



注記

エントリーが分散されたら、再分散できません。以下の制限が適用されます。

- ディストリビューション機能は、エントリーディストリビューションのデプロイ後は変更できません。
- エントリーを異なるデータベースに分散させる可能性がある場合は、LDAP **modrdn** 操作を使用してエントリーの名前を変更することができません。
- 分散ローカルデータベースは複製できません。
- エントリーを異なるデータベースに分散させる可能性がある場合は、**ldapmodify** 操作を使用してエントリーを変更することができません。

これらの制限に違反すると、Directory Server はエントリーを正しく特定して返さないようにします。

カスタムディストリビューションロジックプラグインを作成したら、そのプラグインをディレクトリーに追加します。

ディストリビューションロジックは、接尾辞で宣言された関数です。この関数は、この接尾辞に到達するすべての操作に対して呼び出されます。これには、接尾辞の前に開始するサブツリー検索操作が含まれます。ディストリビューション機能は、Web コンソールとコマンドラインインターフェイスの両方を使用して接尾辞に挿入できます。

カスタムディストリビューション機能を接尾辞に追加するには、以下を指定します。

1. **ldapmodify** を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
```

2. 以下の属性を接尾辞エントリー自体に追加し、カスタムディストリビューションロジックに関する情報を提供します。

```
dn: suffix
changetype: modify
add: nsslapd-backend
nsslapd-backend: Database1
-
add: nsslapd-backend
nsslapd-backend: Database2
-
add: nsslapd-backend
nsslapd-backend: Database3
-
add: nsslapd-distribution-plugin
nsslapd-distribution-plugin: /full/name/of/a/shared/library
```

```
-
add: nsslapd-distribution-funct
nsslapd-distribution-funct: distribution-function-name
```

nsslapd-backend 属性は、この接尾辞に関連付けられたすべてのデータベースを指定します。***nsslapd-distribution-plugin*** 属性は、プラグインが使用するライブラリーの名前を指定します。***nsslapd-distribution-funct*** 属性は、ディストリビューション機能自体の名前を提供します。

2.2.2. Directory データベースの維持

2.2.2.1. 読み取り専用モードでのデータベースの設定

データベースが読み取り専用モードの場合は、エントリーを作成、変更、または削除することはできません。読み取り専用モードが役立つ状況の 1 つは、コンシューマーを手動で初期化する場合や、Directory Server からデータをバックアップまたはエクスポートする前です。読み取り専用モードは、特定の時点でのこれらのデータベースの状態の正確なイメージを保証します。

コマンドラインユーティリティーと Web コンソールは、エクスポートまたはバックアップの操作の前に読み取り専用モードでディレクトリーを自動的に配置しません。これは、ディレクトリーの更新で利用できなくなるためです。ただし、マルチサプライヤーレプリケーションでは、これは問題ではない可能性があります。

2.2.2.1.1. コマンドラインを使用した読み取り専用モードでのデータベースの設定

データベースを読み取り専用モードで設定するには、**dsconf backend suffix set** コマンドを使用します。たとえば、**o=test** 接尾辞のデータベースを読み取り専用モードで設定するには、以下を実行します。

1. 接尾辞とそれに対応するバックエンドを表示します。

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

このコマンドにより、各接尾辞の横にバックエンドデータベースが表示されます。次の手順で、接尾辞のデータベース名が必要です。

2. データベースを読み取り専用モードで設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend suffix set --enable-
readonly "test_database"
```

2.2.2.1.2. Web コンソールを使用した読み取り専用モードでのデータベースの設定

データベースを読み取り専用モードで設定するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。

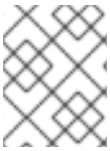
4. 接尾辞エントリーを選択します。
5. **Database Read-Only Mode** を選択します。
6. **Save Configuration** をクリックします。

2.2.2.2. 読み取り専用モードでの Directory Server の配置

Directory Server が複数のデータベースを維持しており、すべてのデータベースを読み取り専用モードで配置する必要がある場合は、1 回の操作で実行できます。

警告

この操作により、Directory Server 設定が読み取り専用になるため、サーバー設定の更新、プラグインの有効化または無効化、読み取り専用モードの場合は Directory Server を再起動することはできません。読み取り専用モードを有効にすると、設定ファイルを手動で変更しない限り、元に戻すことは **できません**。



注記

Directory Server にレプリカが含まれている場合は、レプリケーションを無効にするため、読み取り専用モードを使用 **しないでください**。

2.2.2.2.1. コマンドラインを使用した読み取り専用モードでの Directory Server の配置

Directory Server に対して読み取り専用モードを有効にするには、以下を実行します。

1. ***nsslapd-readonly*** パラメーターを **on** に設定します。

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

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

2.2.2.2.2. Web コンソールを使用した読み取り専用モードでの Entire Directory Server の配置

Directory Server に対して読み取り専用モードを有効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** メニューを開き、**Server Settings** エントリーを選択します。
4. **Advanced Settings** タブで **Server Read-Only** を選択します。
5. **Save** をクリックします。

2.2.2.3. データベースの削除

接尾辞が不要になった場合は、接尾辞を保存するデータベースを削除できます。

2.2.2.3.1. コマンドラインを使用したデータベースの削除

データベースを削除するには、**dsconf backend delete** コマンドを使用します。たとえば、**o=test** 接尾辞のデータベースを削除するには、以下を実行します。

1. 接尾辞とそれに対応するバックエンドを表示します。

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

   次の手順で、接尾辞の横に表示されるバックエンドデータベースの名前が必要です。

2. データベースを削除します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend delete
"test_database"
```

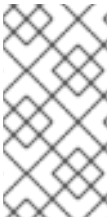
2.2.2.3.2. Web コンソールを使用したデータベースの削除

Web コンソールを使用してデータベースを削除するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. 削除する接尾辞を選択し、**Suffix Tasks** をクリックして **Delete Suffix** を選択します。
5. **Yes** をクリックして確定します。

2.2.2.4. トランザクションログディレクトリーの変更

トランザクションログにより、Directory Server は、インスタンスが予期せずにシャットダウンした後にデータベースを復元できます。特定の状況では、管理者はトランザクションログへのパスを変更したい場合があります。たとえば、Directory Server データベースとは異なる物理ディスクに保存するには、次を実行します。



注記

パフォーマンスを向上させるには、場所を変更する代わりに、トランザクションログが含まれるディレクトリーに高速ディスクをマウントします。詳細は、『Red Hat Directory Server パフォーマンスチューニングガイド』の該当するセクションを参照してください。

トランザクションログディレクトリーの場所を変更するには、以下を行います。

1. Directory Server インスタンスを停止します。

```
# dsctl instance_name stop
```

2. トランザクションログ用に新しい場所を作成します。以下に例を示します。

```
# mkdir -p /srv/dirsrv/instance_name/db/
```

3. Directory Server のみがディレクトリーにアクセスできるように、パーミッションを設定します。

```
# chown dirsrv:dirsrv /srv/dirsrv/instance_name/db/
# chmod 770 /srv/dirsrv/instance_name/db/
```

4. 以前のトランザクションログディレクトリーからすべての **__db.*** ファイルを削除します。以下に例を示します。

```
# rm /var/lib/dirsrv/slapd-instance_name/db/__db.*
```

5. 以前のトランザクションログディレクトリーから新しいトランザクションログディレクトリーに、すべての **log.*** ファイルを移動します。以下に例を示します。

```
# mv /var/lib/dirsrv/slapd-instance_name/db/log.* \
    /srv/dirsrv/instance_name/db/
```

6. SELinux が **enforcing** モードで実行している場合は、ディレクトリーに ***dirsrv_var_lib_t*** コンテキストを設定します。

```
# semanage fcontext -a -t dirsrv_var_lib_t /srv/dirsrv/instance_name/db/
# restorecon -Rv /srv/dirsrv/instance_name/db/
```

7. **/etc/dirsrv/slapd-*instance_name*/dse.ldif** ファイルを編集し、**cn=config,cn=ldbm database,cn=plugins,cn=config** エントリーの ***nsslapd-db-logdirectory*** パラメーターを更新します。以下に例を示します。

```
dn: cn=config,cn=ldbm database,cn=plugins,cn=config
...
nsslapd-db-logdirectory: /srv/dirsrv/instance_name/db/
```

8. インスタンスを起動します。

```
# dsctl instance_name start
```

2.3. データベースリンクの作成および維持

チェーン とは、サーバーがクライアントアプリケーションの代わりに他のサーバーに接続し、組み合わせた結果を返すことを意味します。チェーンは **データベースリンク** を介して実装され、リモートで保存されたデータを参照します。クライアントアプリケーションがデータベースリンクからデータを要求すると、データベースリンクはリモートデータベースからデータを取得し、クライアントに返します。

チェーンに関する一般的な情報は、『Red Hat Directory Server デプロイメントガイド』のディレクトリートポロジーの設計の章を参照してください。「データベースリンクアクティビティーの監視」では、データベースリンクアクティビティーを監視する方法を説明します。

2.3.1. 新規データベースリンクの作成

基本的なデータベースリンクの設定には、以下の情報が必要です。

- **接尾辞の情報。**接尾辞は、通常のデータベースではなく、データベースリンクが管理するディレクトリーツリーに作成されます。この接尾辞は、データが含まれるリモートサーバーの接尾辞に対応します。
- **バインド認証情報。**データベースリンクがリモートサーバーにバインドされると、ユーザーのなりすましが行われ、リモートサーバーとバインドするために使用する各データベースリンクの DN および認証情報を指定します。
- **LDAP URL。**これは、データベースリンクが接続するリモートサーバーの LDAP URL を提供します。URL はプロトコル (ldap または ldaps)、サーバーのホスト名または IP アドレス (IPv4 または IPv6)、およびポートで設定されます。
- **フェイルオーバーサーバーのリスト。**これは、障害発生時にデータベースリンクが接続するための代替サーバーのリストを提供します。この設定項目は任意です。

注記

シンプルなパスワード認証 (「セキュアなバインドの要求」) にセキュアなバインドが必要な場合は、セキュアな接続で行われる場合を除き、チェーン操作は失敗します。いずれの場合も、セキュアな接続 (TLS および STARTTLS 接続または SASL 認証) の使用が推奨されます。

2.3.1.1. コマンドラインを使用した新規データベースリンクの作成

新しいデータベースリンクを作成するには、**dsconf chaining link-create** コマンドを使用します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com chaining link-create --
suffix="ou=Customers,dc=example,dc=com" --server-url="ldap://remote_server.example.com:389" --
bind-mech="" --bind-dn="cn=proxy_user,cn=config" --bind-pw="password" "example_chain_name"
```

これにより、**ou=Customers,dc=example,dc=com** の **example_chain_name** という名前のデータベースリンクが作成されます。リンクはサーバー **ldap://remote_server.example.com:389** を参照し、指定のバインド DN とパスワードを使用して認証を行います。**--bind-mech** が空に設定されているため、リンクは簡易認証を使用します。

注記

proxy_user にデータへのアクセス権を付与するには、リモートサーバーの **dc=example,dc=com** 接尾辞にプロキシー ACI エントリーを作成する必要があります。その方法については、「データベースリンクの作成時に必要な設定に関する追加情報」セクションを参照してください。

データベースリンクの作成時に設定できる追加設定を表示するには、以下を参照してください。

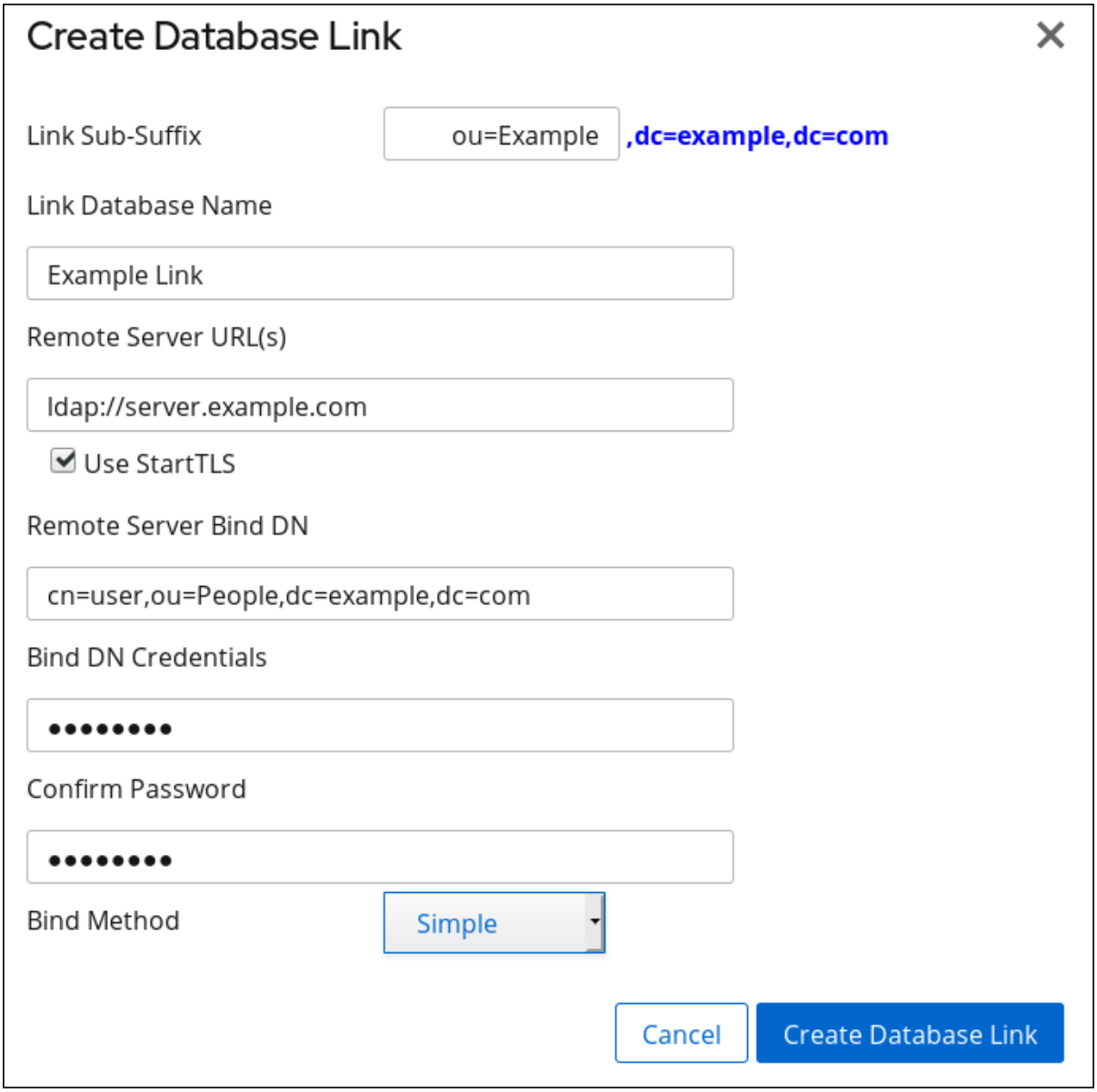
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com chaining link-create --help
```

詳細は、「データベースリンクの作成時に必要な設定に関する追加情報」を参照してください。

2.3.1.2. Web コンソールを使用した新規データベースリンクの作成

新規データベースリンクを作成するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. 「接尾辞の作成」 で説明されているように、新しい接尾辞を作成します。
5. 接尾辞を選択し、**Suffix Tasks** をクリックして **Create Database Link** を選択します。
6. フィールドに、リモートサーバーへのコネクションの詳細を入力します。以下に例を示します。



詳細は、「データベースリンクの作成時に必要な設定に関する追加情報」を参照してください。

7. **Create Database Link** をクリックします。

2.3.1.3. 新規データベースリンクのデフォルト設定の管理

dsconf chaining コマンドを使用すると、データベースリンクのデフォルト設定を管理できます。

現在のデフォルト値を表示するには、以下を参照してください。

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

新しいデータベースリンク設定を変更するには、**dsconf chaining config-set-def** コマンドを使用します。たとえば、***response-delay*** パラメーターを ***30*** に設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com chaining config-set-def --response-delay 30
```

このサンプルコマンドは、すべてのチェーン接続にデフォルトの応答タイムアウトを設定します。**dsconf instance chaining link-set** コマンドを使用すると、特定のチェーンリンクの応答タイムアウトを上書きできます。

設定可能なすべてのパラメーターの一覧を表示するには、以下を実行します。

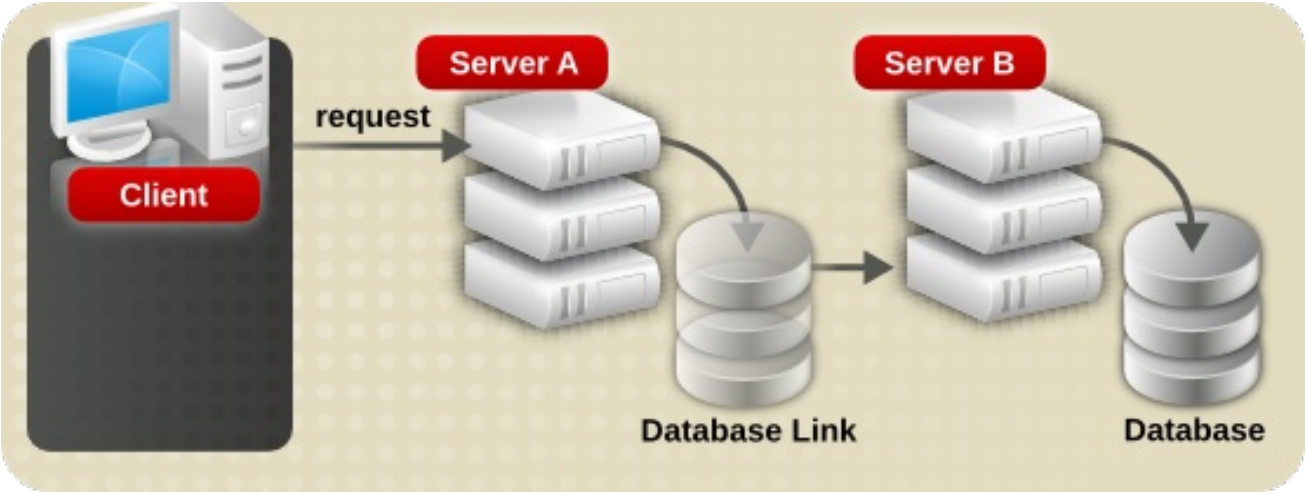
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com chaining config-set-def --help
```

2.3.1.4. データベースリンクの作成時に必要な設定に関する追加情報

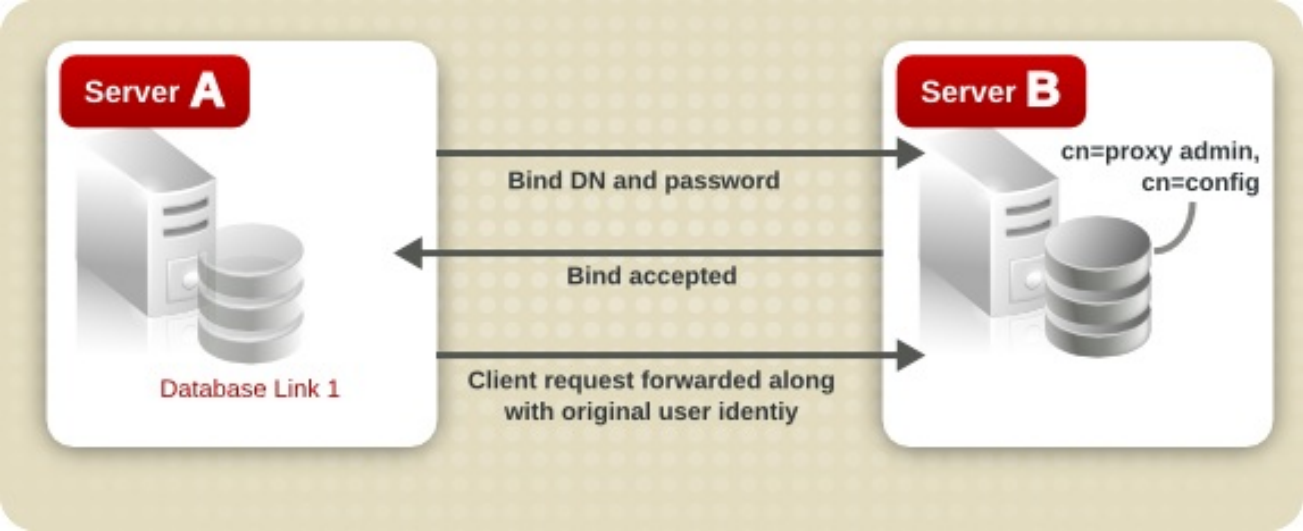
接尾辞情報
接尾辞は、データベースリンクで管理される接尾辞を定義します。

バインド認証情報
クライアントアプリケーションからの要求がリモートサーバーにチェーンされるようにするには、クライアントアプリケーションに特別なバインド認証情報を指定できます。これにより、リモートサーバーでチェーン操作に必要なプロキシー認証権限が付与されます。バインド認証情報がないと、データベースリンクは **anonymous** としてリモートサーバーにバインドされます。

たとえば、クライアントアプリケーションは要求をサーバー A に送信します。サーバー A には、サーバー B のデータベースに要求をチェーンするデータベースリンクが含まれています。



サーバー A のデータベースリンクは、特別なユーザーとパスワードを使用してサーバー B にバインドされます。



サーバー B にはユーザーエントリーが含まれ、このユーザーのプロキシー認証権限を設定する必要があります。プロキシー承認を正しく設定するには、プロキシー ACI をその他の ACI として設定します。

警告

ディレクトリーの制限された領域へのアクセス権限を付与しないようにチェーンを有効にする場合は、アクセス制御を慎重に検討します。たとえば、ブランチにデフォルトのプロキシー ACI が作成されると、データベースリンクを使用して接続するユーザーは、そのブランチの下にあるすべてのエントリーを表示できるようになります。ユーザーがすべてのサブツリーを表示する必要がない場合もあります。セキュリティーホールを回避するには、追加の ACI を作成して、サブツリーへのアクセスを制限します。

ACI の詳細は、*18章 アクセス制御の管理*を参照してください。

注記

エントリーの作成または変更にクライアントアプリケーションでデータベースリンクを使用する場合、***creatorsName*** 属性および ***modifiersName*** 属性にはエントリーの実際の作成者または修正者が反映されません。これらの属性には、リモートデータサーバーでプロキシー認証権限が付与されている管理ユーザーの名前が含まれています。

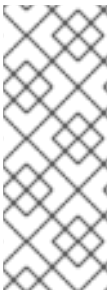
バインド認証情報を指定するには、リモートサーバーで以下の手順が必要です。

1. データベースリンク用に、**cn=proxy_user,cn=config** などの管理ユーザーを作成します。エントリーの追加に関する詳細は、*3章 ディレクトリーエントリーの管理*を参照してください。

2. データベースリンクによってチェーンされたサブツリーの直前の手順で作成した管理ユーザーのプロキシーアクセス権限を提供します。ACI の設定に関する詳細は、*18章 アクセス制御の管理*を参照してください。

   たとえば、以下の ACI は、ACI が設定されたサブツリー内のみにリモートサーバーに含まれるデータにアクセスするために、**cn=proxy_admin,cn=config** ユーザーへの読み取り専用アクセスを付与します。

```
aci: (targetattr = "*")(version 3.0; acl "Proxied authorization
    for database links"; allow (proxy) userdn = "ldap:///cn=proxy_admin
    ,cn=config";)
```



注記

ユーザーがデータベースリンクにバインドすると、ユーザーのアイデンティティーがリモートサーバーに送信されます。アクセス制御は、常にリモートサーバーで評価されます。ユーザーがリモートサーバーにデータを変更または書き込みできるようにするには、リモートサーバーに正しいアクセス制御を設定します。チェーン操作のコンテキストでアクセス制御がどのように評価されるかについての詳細は、「データベースリンクおよびアクセス制御評価」を参照してください。

LDAP URL

データベースリンクが含まれるサーバーで、データベースリンクが **LDAP URL** を使用して接続するリモートサーバーを特定します。標準の LDAP URL 形式とは異なり、リモートサーバーの URL は接尾辞を指定しません。**ldap://*host_name*:*port*** の形式を取ります。

データベースリンクが TLS 経由で LDAP を使用してリモートサーバーに接続する場合、リモートサーバーの LDAP URL は URL の LDAP ではなくプロトコル LDAP を使用し、サーバーのセキュアなポートを参照します。たとえば、以下のようになります。

```
ldaps://africa.example.com:636/
```

注記

TLS を介してチェーンするには、ローカル Directory Server とリモート Directory Server で TLS を有効にする必要があります。TLS の有効化に関する詳細は、「TLS の有効化」を参照してください。

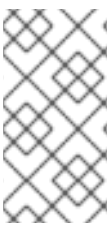
データベースリンクとリモートサーバーが TLS を使用して通信するよう設定されている場合は、操作要求を行うクライアントアプリケーションも TLS を使用して通信する必要があるわけではありません。クライアントは、通常のポートを使用してバインドできます。

バインドメカニズム

ローカルサーバーは、複数の異なる接続タイプと認証メカニズムを使用してリモートサーバーに接続できます。

ローカルサーバーがリモートサーバーに接続する方法は 3 つあります。

- 標準の LDAP ポートを使用する場合
- 専用の LDAPS ポートを使用する場合
- STARTTLS (標準ポートでのセキュアな接続) の使用



注記

シンプルなパスワード認証 (「セキュアなバインドの要求」) にセキュアなバインドが必要な場合は、セキュアな接続で行われる場合を除き、チェーン操作は失敗します。いずれの場合も、セキュアな接続 (TLS および STARTTLS 接続または SASL 認証) の使用が推奨されます。

ローカルサーバーがファームサーバーへの認証に使用できる 4 つの方法があります。

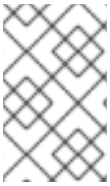
- **empty**: バインドメカニズムが設定されていない場合、サーバーは単純な認証を実行し、バインド DN とパスワードが必要になります。
- **EXTERNAL**: これは TLS 証明書を使用して、ファームサーバーをリモートサーバーに認証します。ファームサーバーをセキュアな URL (**ldaps**) に設定するか、***nsUseStartTLS*** 属性を **on** に設定する必要があります。

  さらに、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 の 『certmap.conf』 セクションで説明されているように、ファームサーバーの証明書をバインド ID にマッピングするようにリモートサーバーを設定する必要があります。
- **DIGEST-MD5**: DIGEST-MD5 暗号化での SASL 認証を使用します。簡易認証と同様、バインド情報を付与するには、***nsMultiplexorBindDN*** および ***nsMultiplexorCredentials*** 属性が必要です。
- **GSSAPI**: SASL 上で Kerberos ベースの認証を使用します。

  ファームサーバーは Kerberos キータブで設定する必要があるため、リモートサーバーには、そのファームサーバーのバインド ID に対して定義された SASL マッピングが必要です。Kerberos キータブおよび SASL マッピングの設定は、「SASL Identity マッピングの設定」 に記載されています。

注記

SASL 接続は、標準の接続または TLS 接続で確立できます。



注記

SASL を使用する場合は、SASL およびパスワードポリシーコンポーネントをチェーンするようにローカルサーバーを設定する必要もあります。「シャーシポリシーの設定」 で説明されているように、データベースリンク設定のコンポーネントを追加します。

2.3.2. シャーシポリシーの設定

この手順では、クライアントアプリケーションによって行われた要求を、データベースリンクを含む Directory Server に、Directory Server が連鎖させる方法の設定を説明します。このチェーンポリシーは、Directory Server で作成されたすべてのデータベースリンクに適用されます。

2.3.2.1. コンポーネントの動作の連鎖

コンポーネントは、内部操作を使用するサーバーの機能ユニットです。たとえば、プラグインはフロントエンドの機能のようにコンポーネントとみなされます。ただし、プラグインは実際には複数のコンポーネントで設定される場合があります (例: ACI プラグイン)。

一部のコンポーネントは、ローカルデータのみにアクセスできることを想定し、内部 LDAP 要求をサーバーに送信します。このようなコンポーネントの場合、チェーンポリシーを制御し、コンポーネントが操作を正常に完了できるようにします。一例として、証明書の検証機能が挙げられます。証明書をチェックするために関数によって行われる LDAP 要求を連鎖させると、リモートサーバーが信頼されていることを意味します。リモートサーバーが信頼されていない場合は、セキュリティーの問題があります。

デフォルトでは、すべての内部操作はチェーンされず、チェーンにはコンポーネントは許可されませんが、これは上書きできます。

また、指定したプラグインがリモートサーバーで操作を実行できるようにするには、リモートサーバーに ACI を作成する必要があります。ACI は、データベースリンクに割り当てられた **接尾辞** に存在している必要があります。

以下は、コンポーネント名、内部操作をチェーンできるようにする可能性のある副次的な影響、およびリモートサーバーの ACI で必要なパーミッションをリスト表示します。

ACI プラグイン

このプラグインはアクセス制御を実装します。ACI 属性の取得および更新に使用される操作は、ローカルおよびリモートの ACI 属性を混在することは安全ではないため、連鎖されません。ただし、ユーザーエントリーの取得に使用されるリクエストは、チェーンコンポーネント属性を設定することでチェーンすることができます。

```
nsActiveChainingComponents: cn=ACI Plugin,cn=plugins,cn=config
```

権限: 読み取り、検索、および比較

リソース制限コンポーネント

このコンポーネントは、ユーザーバインド DN に応じてサーバー制限を設定します。リソース制限コンポーネントを連鎖させることが可能であれば、リソース制限をリモートユーザーに適用できます。リソース制限コンポーネント操作を連鎖させるには、連鎖コンポーネント属性を追加します。

```
nsActiveChainingComponents: cn=resource limits,cn=components,cn=config
```

権限: 読み取り、検索、および比較

証明書ベースの認証チェックコンポーネント

このコンポーネントは、外部バインドメソッドが使用される場合に使用します。リモートサーバーのデータベースからユーザー証明書を取得します。このコンポーネントの連鎖を許可すると、証明書ベースの認証がデータベースリンクと連携できることを意味します。このコンポーネントの動作を連鎖させるには、チェーンコンポーネント属性を追加します。

```
nsActiveChainingComponents: cn=certificate-based authentication,cn=components,cn=config
```

権限: 読み取り、検索、および比較

パスワードポリシーコンポーネント

このコンポーネントは、リモートサーバーへの SASL バインドを許可するために使用されます。SASL 認証の形式によっては、ユーザー名とパスワードを使用した認証が必要になります。パスワードポリシーを有効にすると、サーバーは要求された特定の認証方法を検証および実装し、適切なパスワードポリシーを適用できます。このコンポーネントの動作を連鎖させるには、チェーンコンポーネント属性を追加します。

```
nsActiveChainingComponents: cn=password policy,cn=components,cn=config
```

権限: 読み取り、検索、および比較

SASL コンポーネント

このコンポーネントは、リモートサーバーへの SASL バインドを許可するために使用されます。このコンポーネントの動作を連鎖させるには、チェーンコンポーネント属性を追加します。

```
nsActiveChainingComponents: cn=password policy,cn=components,cn=config
```

権限: 読み取り、検索、および比較

参照整合性プラグイン

このプラグインは、DN を含む属性の更新が、属性へのポインターを含むすべてのエントリーに伝播されるようにします。たとえば、グループのメンバーであるエントリーが削除されると、そのエントリーはグループから自動的に削除されます。このプラグインをチェーンで使用すると、グループメンバーが静的グループ定義にリモートになる場合に静的グループの管理を簡素化できます。このコンポーネントの動作を連鎖させるには、チェーンコンポーネント属性を追加します。

```
nsActiveChainingComponents: cn=referential integrity postoperation,cn=plugins,cn=config
```

権限: 読み取り、検索、および比較

Attribute Uniqueness プラグイン

このプラグインは、指定された属性のすべての値が一意 (重複なし) のものであることを確認します。このプラグインが連鎖されている場合は、データベースリンクで変更された属性でも属性値が一意であることを確認します。このコンポーネントの動作を連鎖させるには、チェーンコンポーネント属性を追加します。

```
nsActiveChainingComponents: cn=attribute uniqueness,cn=plugins,cn=config
```

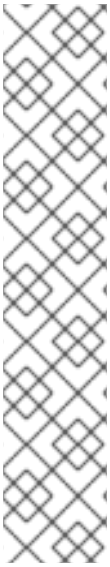
権限: 読み取り、検索、および比較

ロールのコンポーネント

このコンポーネントは、データベースのエントリーのロールおよびロール割り当てを連鎖させます。このコンポーネントの連鎖は、連鎖されたデータベースであってもロールを維持します。このコンポーネントの動作を連鎖させるには、チェーンコンポーネント属性を追加します。

```
nsActiveChainingComponents: cn=roles,cn=components,cn=config
```

権限: 読み取り、検索、および比較



注記

以下のコンポーネントはチェーンできません。

- Roles プラグイン
- パスワードポリシーコンポーネント
- レプリケーションプラグイン
- 参照整合性プラグイン

連鎖要求を発行しているサーバーで Referential Integrity プラグインを有効にする場合は、パフォーマンス、リソース、時間のニーズ、整合性のニーズを分析してください。整合性チェックは時間がかかり、メモリーと CPU を浪費する可能性があります。ACI および連鎖関連の制限の詳細は、「ACI の制限」を参照してください。

2.3.2.1.1. コマンドラインを使用したコンポーネント操作の連鎖

チェーンに許可されているコンポーネントを追加するには、以下を実行します。

1. チェーンに追加するコンポーネントを指定します。たとえば、整合性コンポーネントが連鎖操作を実行できるように設定するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com chaining config-set \
    --add-comp="cn=referential integrity postoperation,cn=components,cn=config"
```

   連鎖が可能なコンポーネントのリストは、「コンポーネントの動作の連鎖」を参照してください。

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

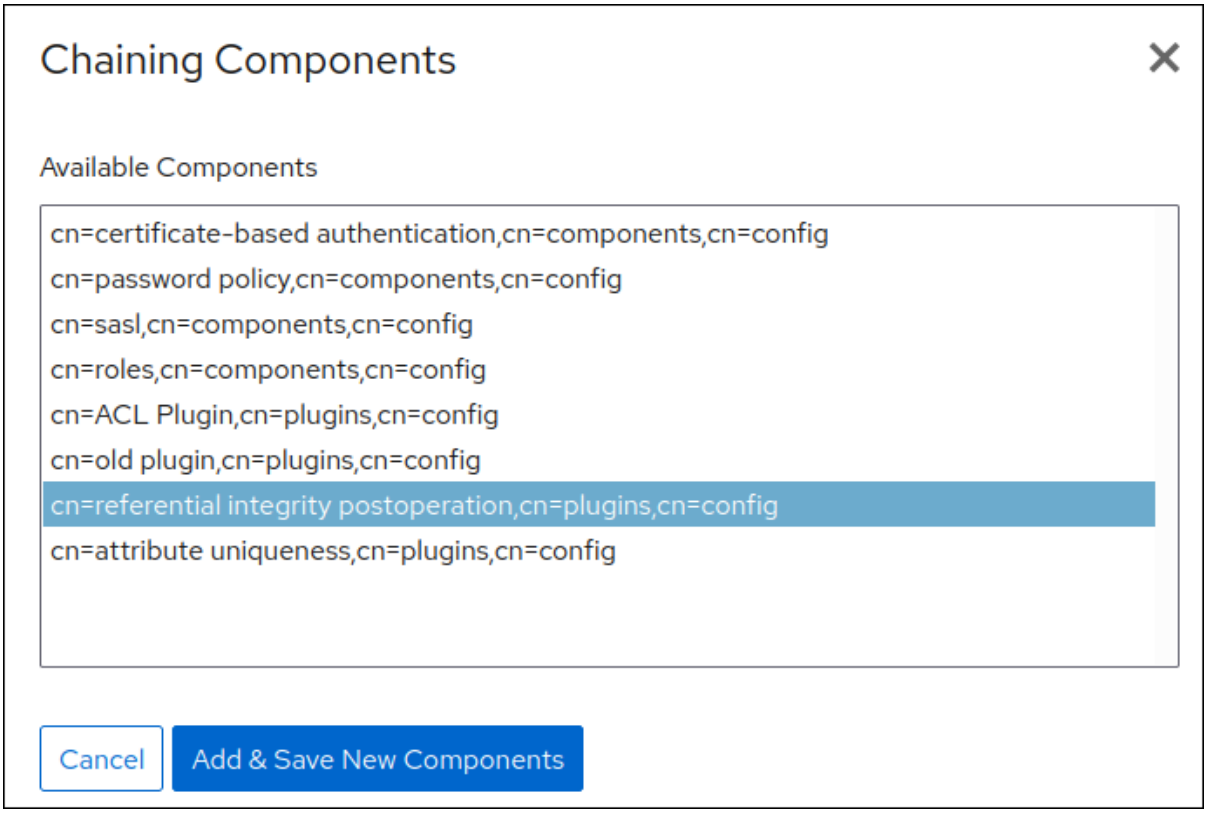
3. 操作を連鎖させるリモートサーバーの接尾辞に ACI を作成します。たとえば、Rerefer Integrity プラグインに ACI を作成するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h remoteserver.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*")(target="ldap:///ou=customers,l=us,dc=example,dc=com")
 (version 3.0; acl "RefInt Access for chaining"; allow
 (read,write,search,compare) userdn = "ldap:///cn=referential
 integrity postoperation,cn=plugins,cn=config";)
```

2.3.2.1.2. Web コンソールを使用したコンポーネントの操作の連鎖

チェーンに許可されているコンポーネントを追加するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** タブを開きます。
4. 左側のナビゲーションで、**Chaining Configuration** エントリーを選択します。
5. **Components to Chain** フィールドの下にある **Add** ボタンをクリックします。
6. コンポーネントを選択し、**Add & Save New Components** をクリックします。



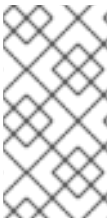
7. 操作を連鎖させるリモートサーバーの接尾辞に ACI を作成します。たとえば、Rerefer Integrity プラグインに ACI を作成するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h remoteserver.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*")(target="ldap:///ou=customers,l=us,dc=example,dc=com")
 (version 3.0; acl "RefInt Access for chaining"; allow
 (read,write,search,compare) userdn = "ldap:///cn=referential
 integrity postoperation,cn=plugins,cn=config";)
```

2.3.2.2. LDAP 制御チェーン

LDAP 制御による操作リクエストを連鎖させることは **できません**。デフォルトでは、以下の制御で行われる要求は、データベースリンクによってリモートサーバーに転送されます。

- **仮想リストビュー (VLV)。**この制御は、すべてのエントリー情報を返すのではなく、エントリーの一部のリストを提供します。
- **サーバー側のソート。**この制御では、通常は特定のマッチングルールを使用して、エントリーを属性値に従ってソートます。
- **逆参照。**この制御では、参照されたエントリーから指定された属性情報を取得し、この情報を残りの検索結果とともに返します。
- **管理 DSA。**この制御は、参照に従うのではなく、スマート参照をエントリーとして返します。そのため、スマートの参照自体は変更または削除できます。
- **ループ検出。**この制御では、別のサーバーとのサーバー連鎖の回数を追跡します。数が設定された数に達すると、ループが検出され、クライアントアプリケーションに通知が送信されます。この制御の使用方法は、「ループの検出」を参照してください。



注記

サーバー側のソートおよび VLV 制御は、1 つのデータベースにクライアントアプリケーション要求が行われている場合にのみサポートされます。データベースリンクは、クライアントアプリケーションが複数のデータベースに要求を行う場合に、これらの制御をサポートしません。

チェーン可能な LDAP 制御とその OID を以下の表に示します。

表2.1 LDAP コントロールとその OID

コントロール名	OID
仮想リストビュー (VLV)	2.16.840.1.113730.3.4.9
サーバー側のソート	1.2.840.113556.1.4.473
管理 DSA	2.16.840.1.113730.3.4.2
ループ検出	1.3.6.1.4.1.1466.29539.12
検索の逆参照	1.3.6.1.4.1.4203.666.5.16

2.3.2.2.1. コマンドラインを使用した LDAP コントロールの連鎖

LDAP 制御をチェーンするには、**dsconf chaining config-set --add-control** コマンドを使用します。たとえば、仮想リストビューコントロールを転送するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com chaining \
    config-set --add-control="2.16.840.1.113730.3.4.9"
```

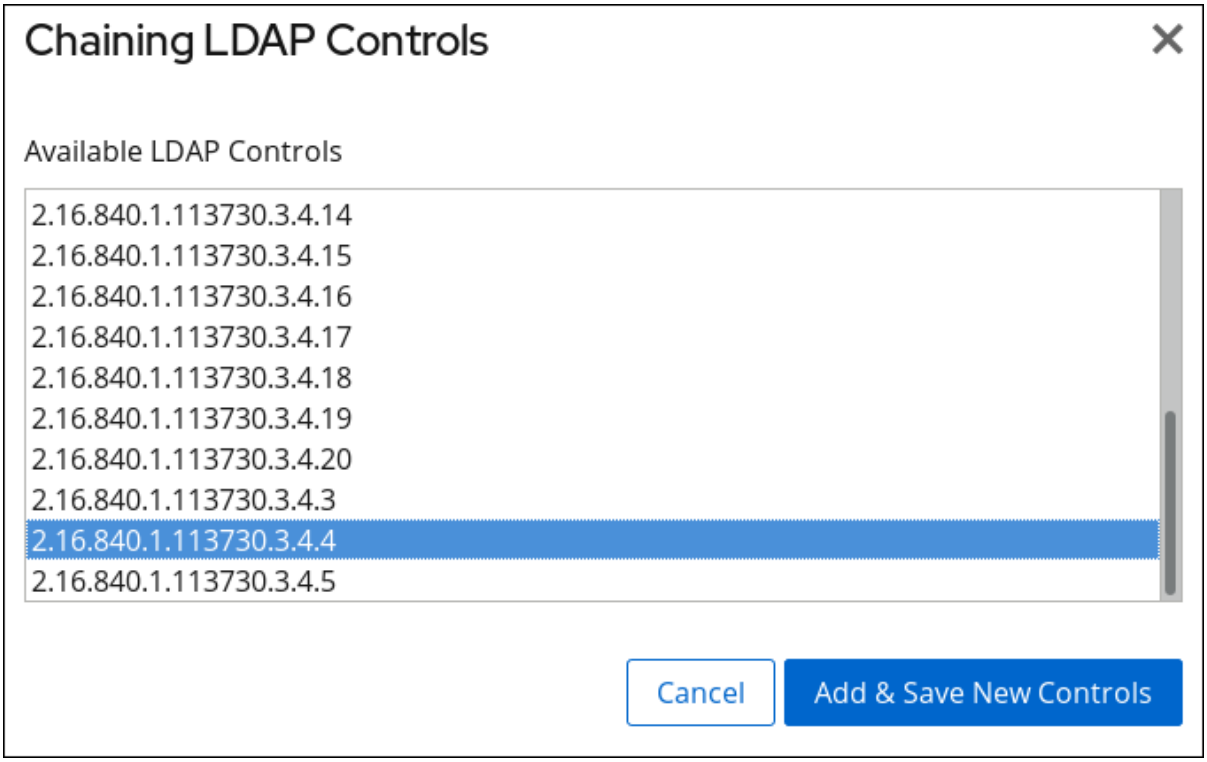
Directory Server のクライアントが独自の制御を作成し、その操作をリモートサーバーにチェーンする必要がある場合は、カスタムコントロールのオブジェクト識別子 (OID) を追加します。

連鎖可能な LDAP コントロールとその OID のリストは、表2.1「LDAP コントロールとその OID」 を参照してください。

2.3.2.2.2. Web コンソールを使用した LDAP 制御の連鎖

Web コンソールを使用して LDAP コントロールを連鎖するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. **Chaining Configuration** エントリーを選択します。
5. **Forwarded LDAP Controls** フィールドの下にある **Add** ボタンをクリックします。
6. LDAP コントロールを選択し、**Add & Save New Controls** をクリックします。



Directory Server のクライアントが独自の制御を作成し、その操作をリモートサーバーにチェーンする必要がある場合は、カスタムコントロールのオブジェクト識別子 (OID) を追加します。

連鎖可能な LDAP コントロールとその OID のリストは、表2.1「LDAP コントロールとその OID」を参照してください。

7. **Save** をクリックします。

2.3.3. データベースリンクおよびアクセス制御評価

ユーザーがデータベースリンクを含むサーバーにバインドすると、データベースリンクがユーザーの ID をリモートサーバーに送信します。アクセス制御は、常にリモートサーバーで評価されます。リモートサーバーで評価されるすべての LDAP 操作は、プロキシーが設定された承認コントロールを使用して渡されたクライアントアプリケーションの元の ID を使用します。ユーザーがリモートサーバーに含まれるサブツリーに正しいアクセス制御がある場合に限り、リモートサーバーで操作に成功します。これには、いくつかの制限があるリモートサーバーに通常のアクセス制御を追加する必要があります。

- すべての種類のアクセス制御を使用できるわけではありません。

  たとえば、ロールベースまたはフィルターベースの ACI はユーザーエントリーへのアクセスを必要とします。データベースリンクを介してデータにアクセスするため、プロキシーコントロールのデータのみが検証できます。ユーザーエントリーがユーザーのデータと同じデータベースに配置されるように、ディレクトリーを設計することを検討してください。

- クライアントの IP アドレスまたは DNS ドメインに基づくすべてのアクセス制御は、チェーン中にクライアントの元のドメインが失われるためです。リモートサーバーは、クライアントアプリケーションをデータベースリンクと同じ IP アドレスと、同じ DNS ドメインにある表示します。

注記

Directory Server は、IPv4 と IPv6 の IP アドレスの両方に対応します。

データベースリンクで使用される ACI には、以下の制限が適用されます。

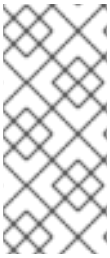
- ACI は、使用する任意のグループと共に配置する必要があります。グループが動的である場合は、グループ内のすべてのユーザーが ACI およびグループで配置される必要があります。グループが静的である場合は、リモートユーザーにリンクされます。
- ACI は、使用するロール定義とそれらのロールを持つユーザーに対して配置する必要があります。
- ユーザーのエントリーの値 (例: **userattr** サブジェクトルール) にリンクする ACI は、ユーザーがリモートの場合に機能します。

アクセス制御は常にリモートサーバーで評価されますが、データベースリンクとリモートサーバーの両方が含まれるサーバーでも評価できます。これにはいくつかの制限があります。

- アクセス制御の評価時に、ユーザーエントリーの内容は必ずしも利用できるとは限りません (たとえば、データベースリンクを含むサーバーでアクセス制御が評価され、エントリーがリモートサーバーにある場合)。

  パフォーマンス上の理由から、クライアントはリモート問い合わせを実行してアクセス制御を評価することはできません。
- データベースリンクは、クライアントアプリケーションによって変更されるエントリーに必ずしもアクセスできるとは限りません。

  変更操作を実行する場合、データベースリンクはリモートサーバーに保存されている全エントリーにアクセスできません。削除操作を実行すると、データベースリンクはエントリーの DN のみを認識します。アクセス制御が特定の属性を指定する場合、データベースリンクを介して実行すると削除操作は失敗します。



注記

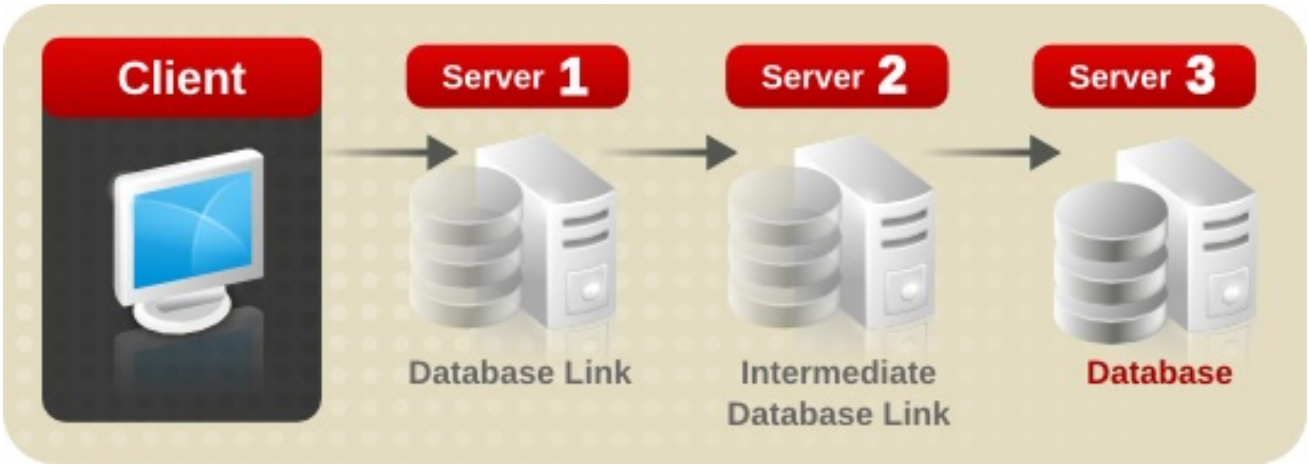
デフォルトでは、データベースリンクが含まれるサーバーに設定されたアクセス制御は評価されません。このデフォルトをオーバーライドするには、**cn=database_link, cn=chaining database,cn=plugins,cn=config** エントリーの ***nsCheckLocalACI*** 属性を使用します。ただし、データベースリンクを含むサーバーでアクセス制御を評価することは、カスケード連鎖を使用する場合を除いて推奨されません。

2.4. カスケード連鎖の設定

データベースリンクは、別のデータベースリンクに指定するように設定でき、カスケード連鎖操作を作成します。ディレクトリーツリー内の全データにアクセスするのに、複数のホップが必要になるとカスケード連鎖がいつでも発生します。

2.4.1. カスケード連鎖の概要

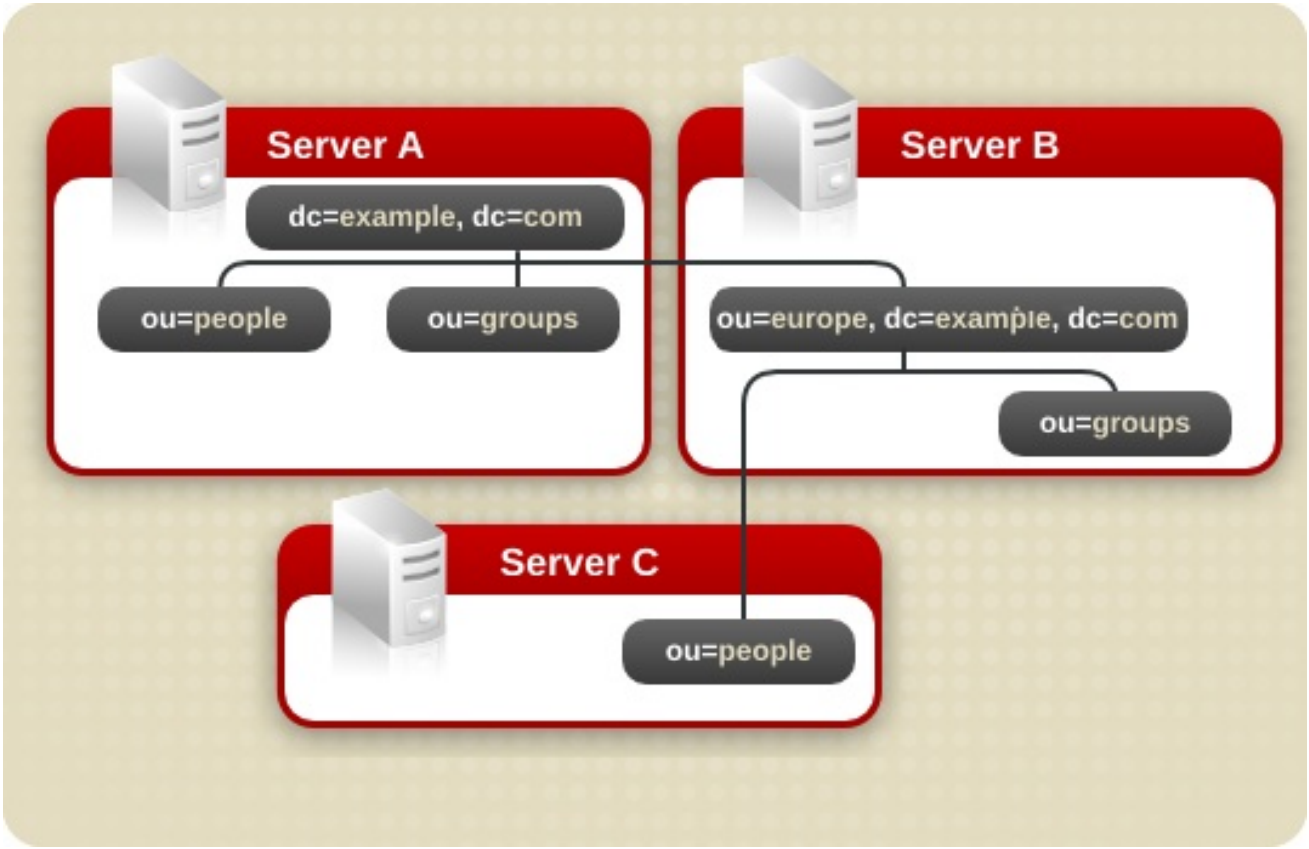
ディレクトリーがクライアントアプリケーションの要求を処理するために必要なホップが複数使用されると、カスケード連鎖が発生します。



クライアントアプリケーションは変更要求を Server 1 に送信します。Server 1 には、別のデータベースリンクが含まれる Server 2 に操作を転送するデータベースリンクが含まれています。Server 2 のデータベースリンクは、クライアントがデータベースに変更するデータが含まれる Server 3 に転送します。クライアントが変更するデータの一部にアクセスするには、2 つのホップが必要になります。

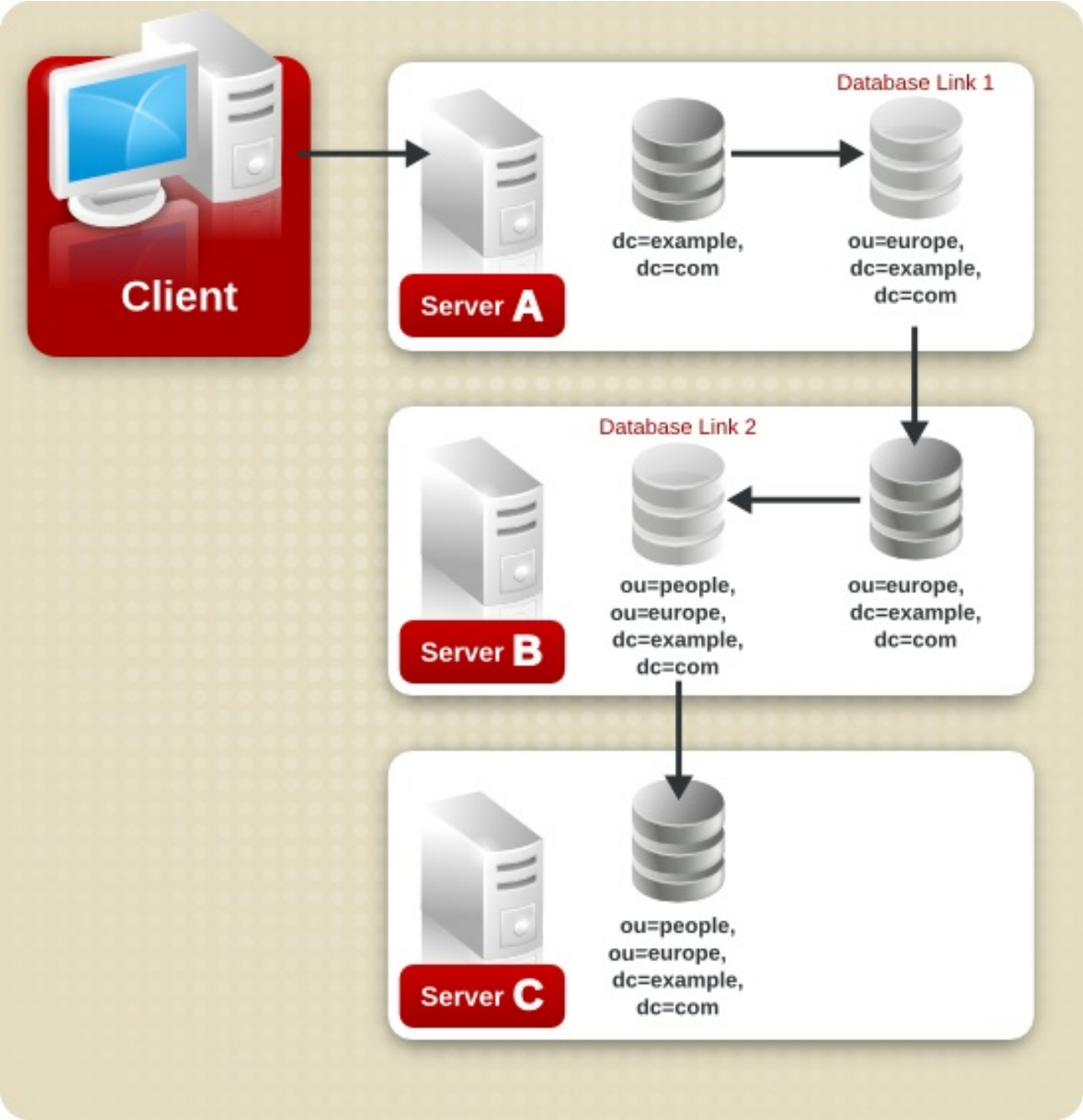
通常の操作要求時に、クライアントはサーバーにバインドし、そのクライアントに適用された ACI が評価されます。カスケード連鎖では、クライアントバインド要求は Server 1 で評価されますが、上の例の Server 2 クライアントに適用される ACI は、要求が宛先サーバーへチェーンされた後にのみ評価されます。

たとえば、サーバー A では、ディレクトリーツリーが分割されます。



ルート接尾辞 **dc=example,dc=com** ならびに **ou=people** および **ou=groups** のサブ接尾辞は、サーバー A に保存されます。**ou=europe,dc=example,dc=com** および **ou=groups** の接尾辞は、サーバー B に保存され、**ou=europe,dc=example,dc=com** 接尾辞の **ou=people** ブランチはサーバー C に保存されます。

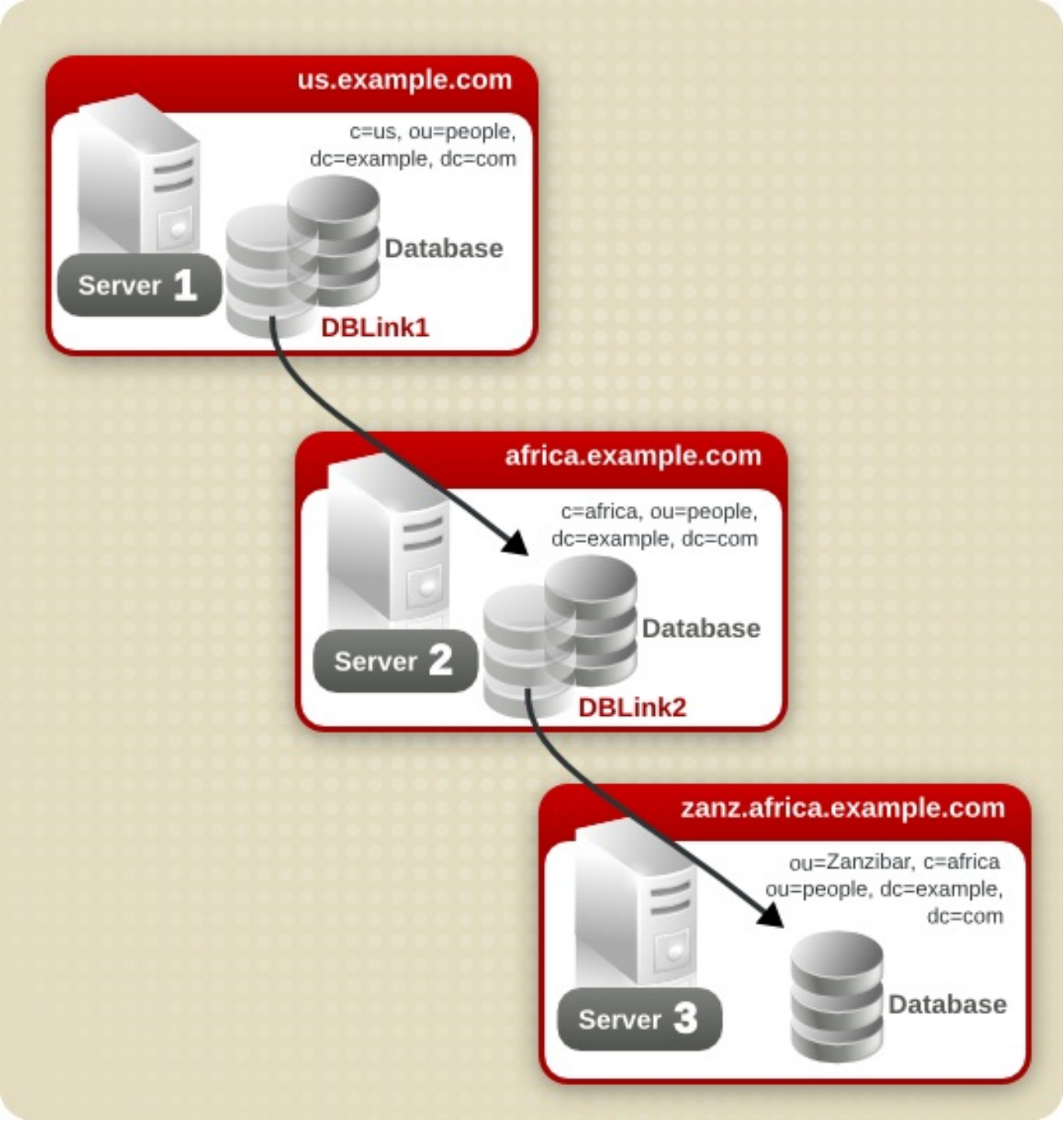
サーバー A、B、および C に設定されたカスケードでは、以下のように **ou=people,ou=europe,dc=example,dc=com** エントリーでターゲットとなるクライアント要求が以下のようにルーティングされます。



まず、クライアントは、Database Link 1 を使用して Server A にバインドし、Server B に連鎖します。その後、Server B は、Database Link 2 を使用して **ou=people,ou=europe,dc=example,dc=com** ブランチのデータにアクセスする Server C のターゲットデータベースに連鎖されます。ディレクトリーがクライアント要求を処理するには少なくとも 2 つのホップが必要であるため、これはカスケード連鎖とみなされます。

2.4.2. コマンドラインを使用したカスケード連鎖の設定

本セクションでは、以下の図に示すように、3 つのサーバーを使用してカスケード連鎖を設定する方法を説明します。



Server 1 の設定手順

1. 接尾辞 **c=africa,ou=people,dc=example,dc=com** を作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server1.example.com backend create --parent-suffix="ou=people,dc=example,dc=com" --suffix="c=africa,ou=people,dc=example,dc=com"
```

2. **DBLink1** データベースリンクを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server1.example.com chaining link-create --suffix="c=africa,ou=people,dc=example,dc=com" --server-url="ldap://africa.example.com:389/" --bind-mech="" --bind-dn="cn=server1 proxy admin,cn=config" --bind-pw="password" --check-aci="off" "DBLink1"
```

3. ループ検出を有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server1.example.com chaining config-set --add-control="1.3.6.1.4.1.1466.29539.12"
```

Server 2 上の設定手順

1. Server 1 をプロキシー承認に使用するプロキシー管理ユーザーを Server 2 に作成します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server2.example.com -x
dn: cn=server1 proxy admin,cn=config
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: server1 proxy admin
sn: server1 proxy admin
userPassword: password
description: Entry for use by database links
```

重要

セキュリティー上の理由から、**cn=Directory Manager** アカウントは使用しないでください。

2. 接尾辞 **ou=Zanzibar,c=africa,ou=people,dc=example,dc=com** を作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server2.example.com backend create --parent-suffix="c=africaou=people,dc=example,dc=com" --suffix="ou=Zanzibar,c=africa,ou=people,dc=example,dc=com"
```

3. **DBLink2** データベースリンクを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server2.example.com chaining link-create --suffix="ou=Zanzibar,c=africa,ou=people,dc=example,dc=com" --server-url="ldap://zanz.africa.example.com:389/" --bind-mech="" --bind-dn="server2 proxy admin,cn=config" --bind-pw="password" --check-aci="on "DBLink2"
```

DBLink2 リンクはカスケード連鎖設定の中間データベースリンクであるため、ACL チェックを有効にして、クライアントとプロキシーの管理ユーザーによるデータベースリンクへのアクセスを許可するかどうかをサーバーが確認できるようにします。

4. ループ検出を有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server2.example.com chaining config-set --add-control="1.3.6.1.4.1.1466.29539.12"
```

5. プロキシー認証制御を有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server2.example.com chaining config-set --add-control="2.16.840.1.113730.3.4.12"
```

6. ローカルプロキシー認証 ACI を追加します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server2.example.com -x
```

```
dn: c=africa,ou=people,dc=example,dc=com
changetype: modify
add: aci
aci:(targetattr="*")(target="lou=Zanzibar,c=africa,ou=people,dc=example,dc=com")
    (version 3.0; acl "Proxied authorization for database links"; allow (proxy)
    userdn = "ldap:///cn=server1 proxy admin,cn=config";)
```

7. Server 1 の **c=us,ou=people,dc=example,dc=com** にいる ***uid*** 属性セットを持つユーザーが、Server 3 の **ou=Zanzibar,c=africa,ou=people,dc=example,dc=com** の接尾辞ツリーに対して、任意のタイプの操作を実行できるようにする ACI を追加します。

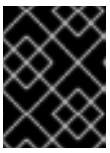
```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server2.example.com -x
dn: c=africa,ou=people,dc=example,dc=com
changetype: modify
add: aci
aci:(targetattr="*")(target="ou=Zanzibar,c=africa,ou=people,dc=example,dc=com")
    (version 3.0; acl "Client authorization for database links"; allow (all)
    userdn = "ldap:///uid=*,c=us,ou=people,dc=example,dc=com";)
```

Server 3 に、Server 3 で追加の権限を必要とする別の接尾辞のユーザーが存在する場合は、Server 2 にさらにクライアント ACI を追加する必要があります。

Server 3 の設定手順

1. Server 3 でプロキシー管理ユーザーを作成し、Server 2 をプロキシー承認に使用するプロキシー管理ユーザーを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server3.example.com -x
dn: cn=server2 proxy admin,cn=config
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: server2 proxy admin
sn: server2 proxy admin
userPassword: password
description: Entry for use by database links
```



重要

セキュリティー上の理由から、**cn=Directory Manager** アカウントは使用しないでください。

2. ローカルプロキシー認証 ACI を追加します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server3.example.com -x
dn: ou=Zanzibar,ou=people,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*")(version 3.0; acl "Proxied authorization
    for database links"; allow (proxy) userdn = "ldap:///cn=server2
    proxy admin,cn=config";)
```

3. Server 1 の **c=us,ou=people,dc=example,dc=com** にいる ***uid*** 属性セットを持つユーザーが、Server 3 の **ou=Zanzibar,c=africa,ou=people,dc=example,dc=com** の接尾辞ツリーに対して、任意のタイプの操作を実行できるようにする ACI を追加します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server3.example.com -x
dn: ou=Zanzibar,ou=people,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr ="*")(target="ou=Zanzibar,c=africa,ou=people,dc=example,dc=com")
    (version 3.0; acl "Client authentication for database link users"; allow (all)
    userdn = "ldap:///uid=*,c=us,ou=people,dc=example,dc=com";)
```

Server 3 に、Server 3 で追加の権限を必要とする別の接尾辞のユーザーが存在する場合は、Server 2 にさらにクライアント ACI を追加する必要があります。

これで、カスケード連鎖は設定されました。このカスケード設定により、ユーザーは Server 1 にバインドし、Server 3 の **ou=Zanzibar,c=africa,ou=people,dc=example,dc=com** ブランチの情報を変更できます。セキュリティーのニーズに応じて、アクセス制御をより詳細に提供する必要があります。

2.4.3. ループの検出

Directory Server に含まれる LDAP 制御により、ループが回避されます。サーバーは、最初にチェーンを試行するときに、このコントロールを、使用できる最大ホップ数または連鎖接続に設定します。後続の各サーバーでカウントが減ります。サーバーが **0** の数を受信すると、ループが検出されたと判断し、クライアントアプリケーションに通知します。

コントロールを使用するには、**1.3.6.1.4.1.1466.29539.12** OID を追加します。LDAP コントロールの追加に関する詳細は、「LDAP 制御チェーン」を参照してください。各データベースリンクの設定ファイルに制御がない場合、ループ検出は実装されません。

許可されるホップ数は、***nsHopLimit*** パラメーターを使用して定義されます。デフォルトでは、パラメーターは **10** に設定されます。たとえば、**example** チェーンのホップ制限を **5** に設定するには、以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com chaining link-set --hop-limit 5
example
```

2.5. 参照の使用

リファーラルは、特定の情報についてどのサーバーに接続するかをクライアントアプリケーションに通知します。このリダイレクトは、クライアントアプリケーションが、ローカルサーバーに存在しないディレクトリーエントリーを要求するか、メンテナンスのためにデータベースがオフラインになったときに発生します。本セクションでは、リファーラルに関する以下の情報を提供します。

- 「リファーラルモードでのサーバーの起動」
- 「デフォルト参照の設定」
- 「スマートリファーラルの作成」
- 「バグ修正参照の作成」

ディレクトリーでリファーラル部分を使用する方法に関する概念情報は、『Red Hat Directory Server デプロイメントガイド』 を参照してください。

2.5.1. リファーラルモードでのサーバーの起動

リファーラルは、現在のサーバーが利用できない場合や、クライアントが現在のサーバーに保持されない情報を要求する場合に、クライアントアプリケーションを別のサーバーにリダイレクトするために使用されます。たとえば、Directory Server の設定変更中に Directory Server を起動すると、そのサーバーが利用できない場合に、すべてのクライアントを別のサプライヤーへ参照します。リファーラルモードで Directory Server を起動するには、**refer** コマンドを使用します。

refer オプションを指定して **nsslapd** を実行します。

```
# ns-slapd refer -D /etc/dirsrv/slapd-instance_name [-p port] -r referral_url
```

- **/etc/dirsrv/slapd-*instance_name*/** は、Directory Server 設定ファイルがあるディレクトリーです。これは、Red Hat Enterprise Linux でのデフォルトの場所です。
- **port** は、リファーラルモードで開始する Directory Server のオプションのポート番号です。
- **referral_url** は、クライアントに返される参照先です。LDAP URL の形式は、付録C *LDAP URL*を参照してください。

2.5.2. デフォルト参照の設定

Directory Server は、ディレクトリーが維持する接尾辞にない DN で操作を送信するクライアントアプリケーションにデフォルトのリファーラル情報を返します。以下の手順では、コマンドラインを使用してディレクトリーのデフォルトリファーラル情報を設定する方法を説明します。

2.5.2.1. コマンドラインを使用したデフォルトのリファーラルの設定

dsconf config replace コマンドを使用して、***nsslapd-referral*** パラメーターのデフォルトのリファーラルを設定します。たとえば、**ldap://directory.example.com/** をデフォルトのリファーラルとして設定するには、以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-
referral="ldap://directory.example.com/"
```

2.5.3. スマートリファーラルの作成

スマートリファーラルは、ディレクトリーエントリーまたはディレクトリーツリーを特定の LDAP URL にマッピングします。スマートリファーラルを使用すると、クライアントアプリケーションは特定のサーバーまたは特定のサーバーの特定のエントリーを参照できます。

たとえば、クライアントアプリケーションは、ディレクトリーエントリー **uid=jdoe,ou=people,dc=example,dc=com** を要求します。サーバー **directory.europe.example.com** のエントリー **cn=john doe,o=people,ou=europe,dc=example,dc=com** を参照するクライアントにスマートリファーラルが返されます。

ディレクトリーがスマートリファーラルを使用する方法は、RFC 2251 セクション 4.1.11 で指定された標準仕様に準拠します。RFC は、http://www.ietf.org/rfc/rfc2251.txt でダウンロードできます。

2.5.3.1. コマンドラインを使用したスマートリファーラルの作成

スマートリファーラルを作成するには、**referral** オブジェクトクラスで関連ディレクトリーエントリーを作成し、***ref*** 属性をリファーラル LDAP URL に設定します。

たとえば、**ldap://directory.europe.example.com/cn=user,ou=people,ou=europe,dc=example,dc=com** を参照する **uid=user,ou=people,dc=example,dc=com** という名前のスマートカードを作成するには、次のコマンドを実行します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server2.example.com -x
dn: uid=user,ou=people,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
objectclass: referral
sn: user
uid: user
cn: user
ref: ldap://directory.europe.example.com/cn=user,ou=people,ou=europe,dc=example,dc=com
```

注記

Directory Server は、LDAP URL 領域後の情報を無視します。このため、リファーラルとして使用する LDAP URL 内の領域の代わりに **%20** を使用します。

DN パスにすでにリファーラルがある場合は、**ldapadd** に **-M** オプションを使用します。スマートリファーラルの詳細は、『Directory Server デプロイメントガイド』 を参照してください。

2.5.4. バグ修正参照の作成

以下の手順では、**接尾辞** にリファーラルを作成する方法を説明します。これは、接尾辞のプロセスがデータベースまたはデータベースリンクではなく、リファーラルを使用して処理されることを意味します。

警告

リファーラルを返すように接尾辞を設定すると、接尾辞に関連付けられたデータベースに含まれる ACI は無視されます。さらに、接尾辞の参照を作成すると、複製されていない接尾辞にのみ適用されます。

2.5.4.1. コマンドラインを使用した接尾辞リファーラルの作成

接尾辞リファーラルを作成するには、以下を実行します。

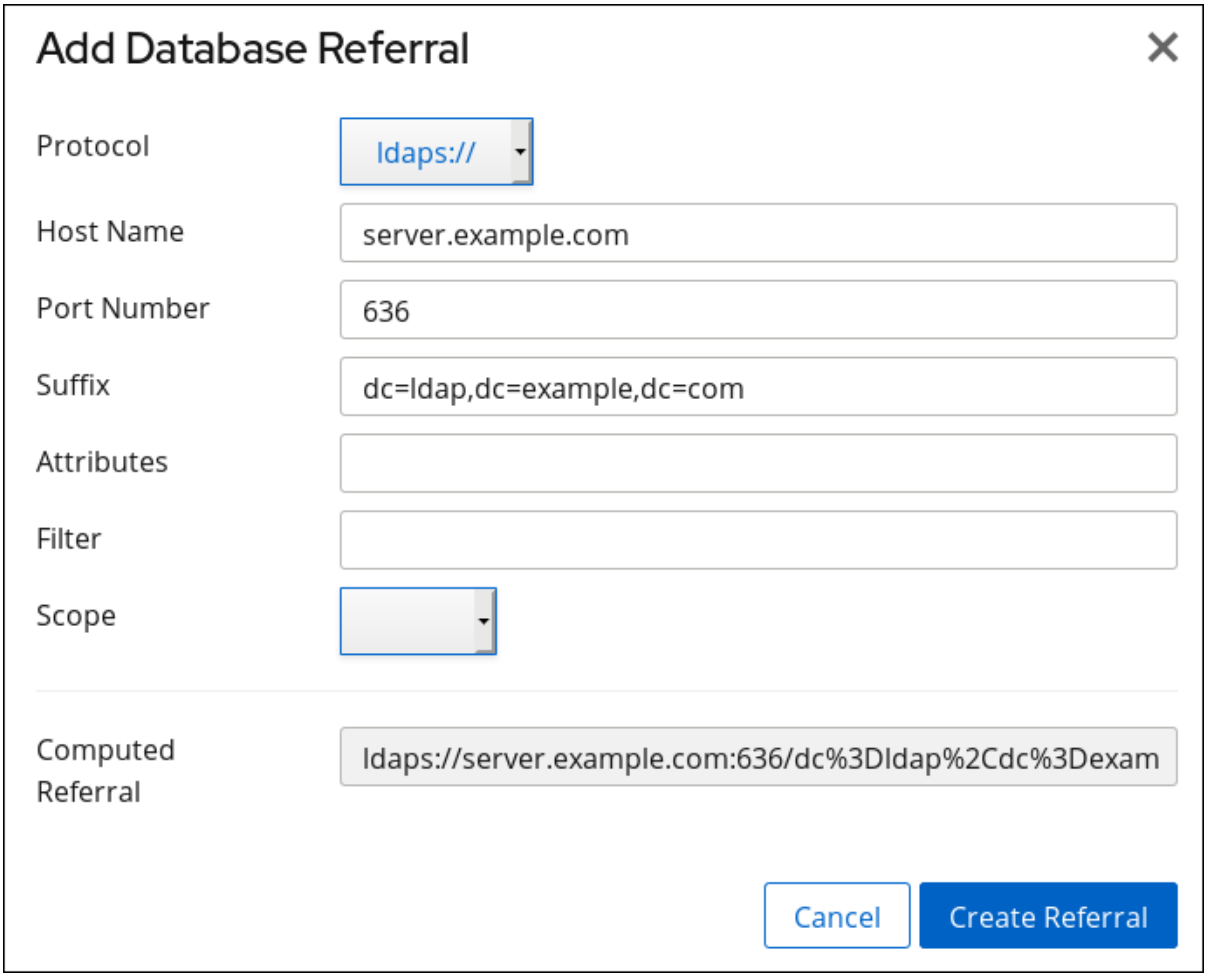
1. 必要に応じて、ルートまたは従属接尾辞が存在しない場合は作成します。詳細については、「接尾辞の作成」 を参照してください。
2. 接尾辞にリファーラルを追加します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend suffix set --add-
referral="ldap://directory.example.com/" database_name
```

2.5.4.2. Web コンソールを使用した接尾辞リファーラルの作成

接尾辞リファーラルを作成するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. 必要に応じて、ルートまたは従属接尾辞が存在しない場合は作成します。詳細については、「接尾辞の作成」 を参照してください。
5. 一覧で接尾辞を選択し、**Referrals** タブを開きます。
6. **Create Referral** をクリックします。
7. リファーラル URL を作成するために、フィールドに入力します。



8. **Create Referral** をクリックします。

2.6. バックエンドデータベースの整合性の確認

dsctl dbverify コマンドを使用すると、管理者はバックエンドデータベースの整合性を検証できます。たとえば、**userroot** データベースを確認するには、以下のコマンドを実行します。

1. 必要に応じて、インスタンスのバックエンドデータベースをリスト表示します。

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

後のステップでデータベースの名前が必要になります。

2. Directory Server インスタンスを停止します。

```
# dsctl instance_name stop
```

3. データベースを確認します。

```
# dsctl instance_name dbverify userroot
[04/Feb/2020:13:11:02.453624171 +0100] - INFO -
ldbm_instance_config_cachememsize_set - force a minimal value 512000
[04/Feb/2020:13:11:02.465339507 +0100] - WARN -
ldbm_instance_add_instance_entry_callback - ldbm instance userroot already exists
[04/Feb/2020:13:11:02.468060144 +0100] - ERR - ldbm_config_read_instance_entries -
Failed to add instance entry cn=userroot,cn=ldbm database,cn=plugins,cn=config
[04/Feb/2020:13:11:02.471079045 +0100] - ERR - bdb_config_load_dse_info - failed to read
instance entries
[04/Feb/2020:13:11:02.476173304 +0100] - ERR - libdb - BDB0522 Page 0: metadata page
corrupted
[04/Feb/2020:13:11:02.481684604 +0100] - ERR - libdb - BDB0523 Page 0: could not check
metadata page
[04/Feb/2020:13:11:02.484113053 +0100] - ERR - libdb - /var/lib/dirsrv/slapd-
instance_name/db/userroot/entryrdn.db: BDB0090 DB_VERIFY_BAD: Database verification
failed
[04/Feb/2020:13:11:02.486449603 +0100] - ERR - dbverify_ext - verify failed(-30970):
/var/lib/dirsrv/slapd-instance_name/db/userroot/entryrdn.db
dbverify failed
```

4. 検証プロセスで問題が報告された場合は、手動で修正するか、バックアップを復元します。

5. Directory Server インスタンスを起動します。

```
# dsctl instance_name start
```

第3章 ディレクトリーエントリーの管理

コマンドラインまたは Web コンソールを使用してディレクトリーエントリーを管理できます。

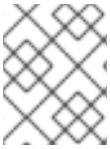
3.1. コマンドラインを使用したディレクトリーエントリーの管理

コマンドラインを使用して LDAP 操作を実行するには、openldap-clients パッケージをインストールします。このパッケージによりインストールされるユーティリティーを使用すると、以下が可能になります。

- 新規エントリーの追加
- 既存のエントリーへの新規属性の追加
- 既存のエントリーおよび属性の更新
- エントリーからエントリーおよび属性を削除します
- 一括操作の実行

openldap-clients パッケージをインストールするには、以下を実行します。

```
# yum install openldap-clients
```



注記

LDAP 操作を実行するには、適切なパーミッションが必要です。アクセス制御の詳細は、18章 *アクセス制御の管理*を参照してください。

3.1.1. ldapadd、ldapmodify、および ldapdelete ユーティリティーへの入力の指定

ディレクトリー内のエントリーまたは属性を追加、更新、または削除する場合は、ユーティリティーのインタラクティブモードを使用して LDAP データ交換形式 (LDIF) ステートメントを入力するか、LDIF ファイルをこれらのファイルに渡します。

LDIF の詳細は、「LDIF ファイルの形式の概要」を参照してください。

3.1.1.1. インタラクティブモードでの入力の提供

インタラクティブモードでは、**ldapadd**、**ldapmodify**、および **ldapdelete** ユーティリティーはコマンドラインから入力を読み取ります。インタラクティブモードを終了するには、**Ctrl**+**D** (**^D**) のキーの組み合わせを押して、end-of-file (EOF) エスケープシーケンスを送信します。

インタラクティブモードでは、ユーティリティーは、**Enter** を 2 回押したときに、または EOF シーケンスを送信するときに、ステートメントを LDAP サーバーに送信します。

対話型モードを使用します。

- ファイルを作成せずに LDIF ステートメントに入るには、次を実行します。

例3.1 ldapmodify インタラクティブモードを使用した LDIF ステートメントの開始

以下の例では、インタラクティブモードで **ldapmodify** を開始し、***telephoneNumber*** 属性を削除し、**cn=*manager_name*,ou=people,dc=example,dc=com** 値を持つ manager 属性を

uid=*user*,ou=people,dc=example,dc=com エントリーに追加します。最後のステートメントの後に **Ctrl**+**D** を押して、インタラクティブモードを終了します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=people,dc=example,dc=com
changetype: modify
delete: telephoneNumber
-
add: manager
manager: cn=manager_name,ou=people,dc=example,dc=com
^D
```

- 別のコマンドで出力される LDIF ステートメントを Directory Server にリダイレクトするには、次を実行します。

例3.2 リダイレクトされたコンテンツでの ldapmodify インタラクティブモードの使用

以下の例では、**command_that_outputs_LDIF** コマンドの出力を **ldapmodify** にリダイレクトします。対話モードは、リダイレクトされたコマンドの終了後に自動的に終了します。

```
# command_that_outputs_LDIF | ldapmodify -D "cn=Directory Manager" \
    -W -p 389 -h server.example.com -x
```

3.1.1.2. LDIF ファイルを使用した入力の提供

インタラクティブモードでは、**ldapadd**、**ldapmodify**、および **ldapdelete** ユーティリティーは、ファイルから LDIF ステートメントを読み取ります。このモードを使用して、多数の LDIF ステートメントを Directory Server に送信します。

例3.3 LDIF ステートメントを持つファイルを ldapmodify に渡す

1. LDIF ステートメントでファイルを作成します。たとえば、以下のステートメントで **~/example.ldif** ファイルを作成します。

```
dn: uid=user,ou=people,dc=example,dc=com
changetype: modify
delete: telephoneNumber
-
add: manager
manager: cn=manager_name,ou=people,dc=example,dc=com
```

この例では、***telephoneNumber*** 属性を削除し、**cn=*manager_name*,ou=people,dc=example,dc=com** 値を持つ manager 属性を **uid=*user*,ou=people,dc=example,dc=com** エントリーに追加します。

2. **-f *file_name*** オプションを使用して、ファイルを **ldapmodify** コマンドに渡します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x \
    -f ~/example.ldif
```

3.1.2. 継続的操作モード

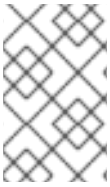
複数の LDIF ステートメントを Directory Server に送信し、1 回の操作に失敗すると、プロセスは停止します。ただし、エラーが発生する前に処理されるエントリーは、正常に追加、変更、または削除されています。

エラーを無視してバッチでさらに LDIF ステートメントの処理を続けるには、**-c** オプションを **ldapadd** および **ldapmodify** に渡します。以下に例を示します。

```
# ldpamodify -c -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
```

3.1.3. エントリーの追加

新しいエントリーをディレクトリーに追加するには、**ldapadd** ユーティリティーまたは **ldapmodify** ユーティリティーを使用します。**ldapadd** は **/bin/ldapmodify** へのシンボリックリンクであることに注意してください。そのため、**ldapadd** は **ldapmodify -a** と同じ操作を実行します。



注記

親エントリーがすでに存在する場合のみ、新しいディレクトリーエントリーを追加できます。たとえば、**ou=people,dc=example,dc=com** の親エントリーが存在しない場合は、**cn=user,ou=people,dc=example,dc=com** エントリーを追加できません。

3.1.3.1. ldapadd を使用したエントリーの追加

ldapadd ユーティリティーを使用して、たとえば **cn=user,ou=people,dc=example,dc=com** ユーザーエントリーを追加するには、以下を実行します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
uid: user
givenName: given_name
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
sn: surname
cn: user
```

注記

ldapadd を実行すると、**changetype: add** 操作が自動的に実行されます。そのため、LDIF ステートメントで **changetype: add** を指定する必要はありません。

このコマンドで使用されるパラメーターの詳細は、ldapadd(1) の man ページを参照してください。

3.1.3.2. ldapmodify を使用したエントリーの追加

ldapmodify ユーティリティーを使用して、たとえば **cn=user,ou=people,dc=example,dc=com** ユーザーエントリーを追加するには、以下を実行します。

```
# ldapmodify -a -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
uid: user
givenName: given_name
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
sn: surname
cn: user
```

注記

-a オプションを **ldapmodify** コマンドに渡すと、ユーティリティーは **changetype: add** 操作を自動的に実行します。そのため、LDIF ステートメントで **changetype: add** を指定する必要はありません。

このコマンドで使用されるパラメーターの詳細は、ldapmodify(1) の man ページを参照してください。

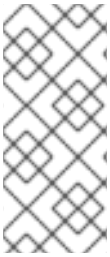
3.1.3.3. ルートエントリーの作成

dc=example,dc=com などのデータベース接尾辞のルートエントリーを作成するには、**cn=Directory Manager** ユーザーとしてバインドし、エントリーを追加します。

DN は、データベースのルートまたは従属接尾辞の DN に対応します。

たとえば、**dc=example,dc=com** 接尾辞を追加するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: dc=example,dc=com
changetype: add
objectClass: top
objectClass: domain
dc: example
```



注記

ルートオブジェクトは、接尾辞に 1 つのデータベースがある場合にのみ追加できます。複数のデータベースに保存されている接尾辞を作成する場合は、**-n *back_end*** オプションを指定して **ldif2db** ユーティリティーを使用し、新しいエントリーを保持するデータベースを設定する必要があります。詳細については、「コマンドラインでのインポート」 を参照してください。

3.1.4. ディレクトリーエントリーの更新

ディレクトリーエントリーを変更する場合は、**changetype: modify** ステートメントを使用します。change 操作に応じて、エントリーから属性を追加、変更、または削除できます。

ldapmodify ユーティリティーを使用して、LDIF ステートメントを Directory Server に送信します。たとえば、インタラクティブモードでは、以下のようになります。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
```

ldapmodify コマンドで使用されるパラメーターの詳細は、ldapmodify(1) の man ページを参照してください。

3.1.4.1. エントリーへの属性の追加

エントリーに属性を追加するには、**add** 操作を使用します。

たとえば、**uid=*user*,ou=People,dc=example,dc=com** エントリーに **555-1234567** の値を持つ ***telephoneNumber*** 属性:

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
add: telephoneNumber
telephoneNumber: 555-1234567
```

属性が多値である場合、属性名を複数回指定して、1 つの操作ですべての値を追加できます。たとえば、**uid=*user*,ou=People,dc=example,dc=com** に 2 つの ***telephoneNumber*** 属性を同時に追加するには、以下のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
add: telephoneNumber
telephoneNumber: 555-1234567
telephoneNumber: 555-7654321
```

3.1.4.2. 属性の値の更新

属性の値を更新する手順は、属性が単値であるか多値であるかによって異なります。

単値属性の更新
単値属性を更新する場合は、**replace** 操作を使用して既存の値を上書きします。次のコマンドは、**uid=*user*,ou=People,dc=example,dc=com** エントリーの ***manager*** 属性:

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
replace: manager
manager: uid=manager_name,ou=People,dc=example,dc=com
```

多値属性の特定値の更新
多値属性の特定の値を更新するには、最初に置き換えるエントリーを削除してから、新しい値を追加する必要があります。次のコマンドは、**uid=*user*,ou=People,dc=example,dc=com** エントリーで現在 **555-1234567** に設定されている ***telephoneNumber*** 属性:

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
```

```
dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
delete: telephoneNumber
telephoneNumber: 555-1234567
-
add: telephoneNumber
telephoneNumber: 555-9876543
```

3.1.4.3. エントリーからの属性の削除

エントリーから属性を削除するには、**delete** 操作を実行します。

属性の削除
たとえば、**uid=*user*,ou=People,dc=example,dc=com** エントリーの ***manager*** 属性:

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
delete: manager
```

注記

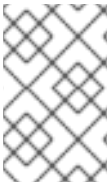
属性に複数の値が含まれる場合、この操作によりすべての値が削除されます。

多値属性から特定の値の削除
多値属性から特定の値を削除する場合は、LDIF ステートメントに属性とその値をリスト表示します。たとえば、**uid=*user*,ou=People,dc=example,dc=com** エントリーから **555-1234567** に設定されている ***telephoneNumber*** 属性:

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
delete: telephoneNumber
telephoneNumber: 555-1234567
```

3.1.5. エントリーの削除

エントリーを削除すると、ディレクトリーからエントリーが削除されます。



注記

子エントリーのないエントリーのみを削除できます。たとえば、**uid=*user*,ou=People,dc=example,dc=com** エントリーがまだ存在している場合は、**ou=People,dc=example,dc=com** エントリーを削除できません。

3.1.5.1. ldapdelete を使用したエントリーの削除

ldapdelete ユーティリティーを使用すると、1 つまたは複数のエントリーを削除できます。たとえば、**uid=*user*,ou=People,dc=example,dc=com** エントリーを削除するには、次のコマンドを実行します。

```
# ldapdelete -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
"uid=user,ou=People,dc=example,dc=com"
```

1 つの操作で複数のエントリーを削除するには、コマンドに追加します。以下に例を示します。

```
# ldapdelete -D "cn=Directory Manager" -W -p 389 -h server.example.com -x \
    "uid=user1,ou=People,dc=example,dc=com" \
    "uid=user2,ou=People,dc=example,dc=com"
```

使用されるパラメーターの詳細は、ldapdelete(1) の man ページを参照してください。

3.1.5.2. ldapmodify を使用したエントリーの削除

ldapmodify ユーティリティーを使用してエントリーを削除するには、**changetype: delete** 操作を使用します。たとえば、**uid=*user*,ou=People,dc=example,dc=com** エントリーを削除するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

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

3.1.6. エントリーの名前変更および変更

本セクションでは、エントリーの名前変更または移動の方法を説明します。

注記

moddn アクセス制御リスト (ACL) を使用して、エントリーを移動するパーミッションを付与します。詳細については、「ソースおよび宛先 DN のターゲット」 を参照してください。

以下の名前変更操作が存在します。

エントリーの名前変更

エントリーの名前を変更すると、**modrdn** 操作によってエントリーの相対識別名 (RDN) が変更されます。

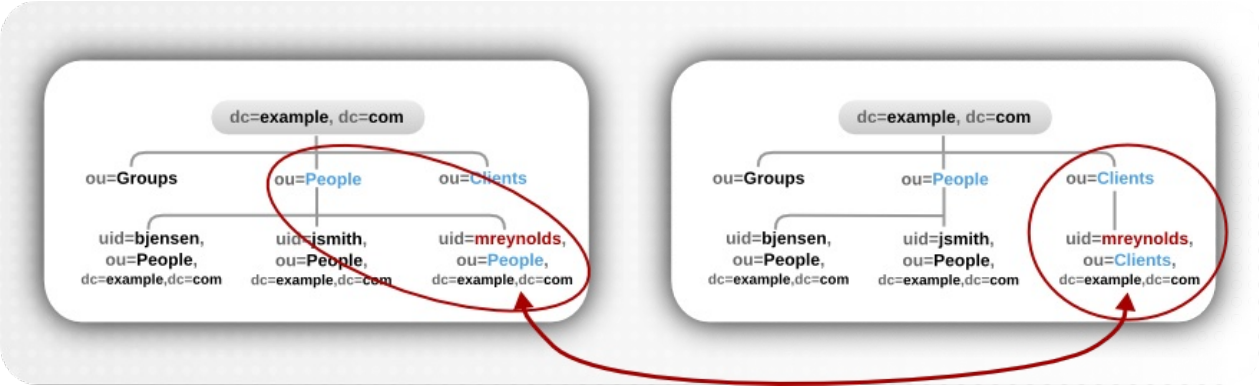
サブエントリーの名前変更

サブツリーエントリーの場合、**modrdn** 操作はサブツリーと子エントリーの DN コンポーネントの名前を変更します。

大規模なサブツリーでは、このプロセスに多くの時間とリソースが必要になる可能性があることに注意してください。

エントリーを新しい親へ移動

サブツリーの名前を変更する同様のアクションは、エントリーをあるサブツリーから別のサブツリーに移動することです。これは、**modrdn** 操作の拡張タイプで、エントリーの名前を同時に変更し、***newSuperior*** 属性を設定して、エントリーを別の親に移動します。



3.1.6.1. エントリーの名前変更に関する考慮事項

名前変更の操作を実行する場合は、以下の点に留意してください。

- root 接尾辞の名前を変更することはできません。
- サブツリー名前変更操作によるレプリケーションへの影響は最小限に抑えられます。レプリカ合意は、データベースのサブツリーではなく、データベース全体に適用されます。そのため、サブツリーの名前変更操作ではレプリカ合意の再設定は必要ありません。サブツリーの名前変更操作後のすべての名前の変更は、通常どおり複製されます。
- サブツリーの名前を変更し、同期合意を再設定する必要がある場合があります。同期合意は、接尾辞またはサブツリーレベルで設定されます。そのため、サブツリーの名前を変更すると、同期が中断する可能性があります。
- サブツリーの名前を変更するには、サブツリーに設定されたサブツリーレベルのアクセス制御命令 (ACI) を手動で再設定し、サブツリーの子エントリーに設定されたエントリーレベルの ACI (エントリーレベルの ACI) を手動で再設定する必要があります。

- ***ou*** から ***dc*** への移行など、サブツリーのコンポーネントを変更しようとすると、スキーマ違反で失敗する可能性があります。たとえば、**organizationalUnit** オブジェクトクラスには ***ou*** 属性が必要です。この属性がサブツリーの名前の一部として削除されると、操作は失敗します。

- グループを移動すると、MemberOf プラグインは ***memberOf*** 属性を自動的に更新します。ただし、グループが含まれるサブツリーを移動する場合は、**cn=memberof task** エントリーでタスクを手動で作成するか、**fixup-memberof.pl** を使用して関連する ***memberOf*** 属性を更新する必要があります。

  memberOf 属性参照のクリーンアップに関する詳細は、「***memberOf*** 値の再生成」 を参照してください。

3.1.6.2. ユーザー、グループ、POSIX グループ、および OU の名前変更

dsidm ユーティリティーは、複数のタイプのオブジェクトの名前を変更できます。

- ユーザー:

  ```
  # dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com" user rename current_user_name new_user_name
  ```

  dsidm user rename コマンドは、指定したベース DN の前に **ou=People** を自動的に配置することに注意してください。

- グループ:

  ```
  # dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com" group rename current_group_name new_group_name
  ```

  dsidm group rename コマンドは、指定したベース DN の前に **ou=Groups** を自動的に配置することに注意してください。

- POSIX グループ:

  ```
  # dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com" posixgroup rename current_posix_group_name new_posix_group_name
  ```

  dsidm posixgroup rename コマンドは、指定したベース DN の前に **ou=Groups** を自動的に配置することに注意してください。

- 組織単位 (OU)

  ```
  # dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com" organizationalunit rename current_ou_name new_ou_name
  ```

  dsidm organizationalunit rename コマンドは、指定したベース DN で直接名前変更の操作を実行します。

3.1.6.3. LDIF ステートメントを使用してエントリーの名前を変更する際の *deleteOldRDN* パラメーター

エントリーの名前を変更すると、***deleteOldRDN*** パラメーターは古い RDN が削除されるかどうかを制御します。

***deleteOldRDN*: 0**

既存の RDN は、新しいエントリーの値として保持されます。生成されるエントリーには、古い属性と新しい共通名 (CN) を持つ 2 つの ***cn*** 属性が含まれます。

たとえば、以下の属性は、**deleteOldRDN: 0** パラメーターを設定して、**cn=old_group,dc=example,dc=com** から**cn=new_group,dc=example,dc=com** に名前を変更したグループに属しています。

```
dn: cn=new_group,ou=Groups,dc=example,dc=com
objectClass: top
objectClass: groupOfUniqueNames
cn: old_group
cn: new_group
```

***deleteOldRDN*: 1**

Directory Server は古いエントリーを削除し、新しい RDN を使用して新しいエントリーを作成します。新しいエントリーには、新しいエントリーの ***cn*** 属性のみが含まれます。

たとえば、以下のグループは、**deleteOldRDN: 1** パラメーターを設定して、**cn=new_group,dc=example,dc=com** に名前を変更しました。

```
dn: cn=new_group,ou=Groups,dc=example,dc=com
objectClass: top
objectClass: groupofuniquenames
cn: new_group
```

3.1.6.4. LDIF ステートメントを使用したエントリーまたはサブツリーの名前変更

エントリーまたはサブツリーの名前変更には、**changetype: modrdn** 操作を使用し、***newrdn*** 属性に新しい RDN を設定します。

たとえば、**cn=demo1,dc=example,dc=com** エントリーの名前を**cn=example_user,dc=example,dc=com** に変更するには、次のようにします。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=example_user,cn=ldap_connect,dc=example,dc=com
changetype: modrdn
newrdn: cn=example_user
deleteOldRDN: 1
newSuperior: dc=example,dc=com
```

deleteOldRDN の詳細は、「LDIF ステートメントを使用してエントリーの名前を変更する際の ***deleteOldRDN*** パラメーター」を参照してください。

3.1.6.5. LDIF ステートメントを使用した新しい親へのエントリーの移動

エントリーを新しい親に移動するには、**changetype: modrdn** 操作を使用して、以下の属性を設定します。

newrdn

移動したエントリーの RDN を設定します。RDN が同じままであっても、このエントリーを設定する必要があります。

newSuperior

新しい親エントリーの DN を設定します。

たとえば、**cn=demo** エントリーを **ou=Germany,dc=example,dc=com** から **ou=France,dc=example,dc=com** に移動するには、以下のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=demo,ou=Germany,dc=example,dc=com
changetype: modrdn
newrdn: cn=demo
deleteOldRDN: 1
newSuperior: ou=France,dc=example,dc=com
```

deleteOldRDN の詳細は、「LDIF ステートメントを使用してエントリーの名前を変更する際の ***deleteOldRDN*** パラメーター」を参照してください。

3.1.7. 特殊文字の使用

コマンドラインを使用する場合は、スペース ( )、アスタリスク (*)、バックスラッシュ (\) などのコマンドラインインタープリターに特別な意味を持つ文字を引用符で囲みます。コマンドラインインタープリターに応じて、一重引用符または二重引用符を使用します。

たとえば、**cn=Directory Manager** ユーザーとして認証するには、ユーザーの DN を引用符で囲みます。

```
# ldapmodify -a -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
```

また、DN にコンポーネントのコンマが含まれる場合は、バックスラッシュを使用してエスケープします。たとえば、**uid=*user*,ou=People,dc=example.com Chicago, IL** ユーザーとして認証するには、次のコマンドを実行します。

```
# ldapmodify -a -D "cn=uid=user,ou=People,dc=example.com Chicago\, IL" \
    -W -p 389 -h server.example.com -x
```

3.1.8. Binary 属性の使用

特定の属性は、***jpegPhoto*** 属性などのバイナリー値をサポートします。このような属性を追加または更新すると、ユーティリティーはファイルから属性の値を読み取ります。このような属性を追加または更新するには、**ldapmodify** ユーティリティーを使用できます。

たとえば、**uid=*user*,ou=People,dc=example,dc=com** エントリーに ***jpegPhoto*** 属性を追加し、**/home/*user_name*/photo.jpg** ファイルから属性の値を読み取るには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
```

```
changetype: modify
add: jpegPhoto
jpegPhoto:< file:///home/user_name/photo.jpg
```

重要

: と **<** の間には、スペースがないことに注意してください。

3.1.9. 国際化されたディレクトリーにおけるエントリーの更新

属性の値を英語以外の言語で使用するには、属性の値を言語タグに関連付けます。

ldapmodify を使用して言語タグが設定されている属性を更新する場合は、値と言語タグを正確に一致させる必要があります。そうでないと、操作は失敗します。

たとえば、**lang-fr** 言語タグが設定された属性値を変更するには、**modify** 操作にタグを追加します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
replace: homePostalAddress;lang-fr
homePostalAddress;lang-fr: 34 rue de Seine
```

3.2. WEB コンソールを使用したディレクトリーエントリーの管理

Web コンソールを使用して、LDAP エントリーを追加、編集、削除、および名前の変更を実行できます。

3.2.1. Web コンソールを使用した LDAP エントリーの追加

Web コンソールで LDAP ブラウザーを使用して、ディレクトリーサーバーデータベースのエントリーを検索できます。

Web コンソールを使用して、次のエントリーを作成できます。

- ユーザー
- グループ
- roles
- 組織単位 (OU)
- カスタムエントリー

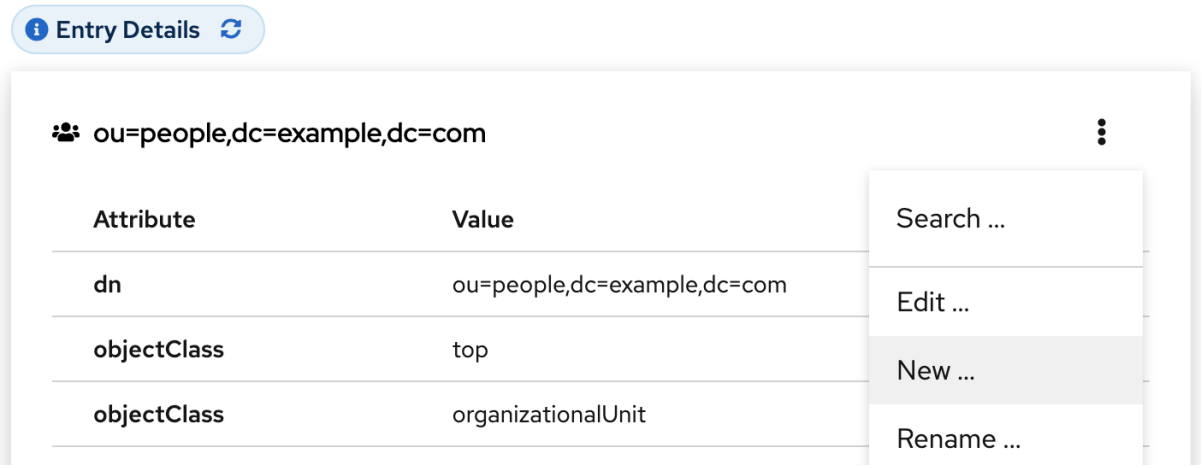
たとえば、***cn=John Smith,ou=people,dc=example,dc=com*** のパスワードを使用して POSIX ユーザーを作成するとします。

前提条件

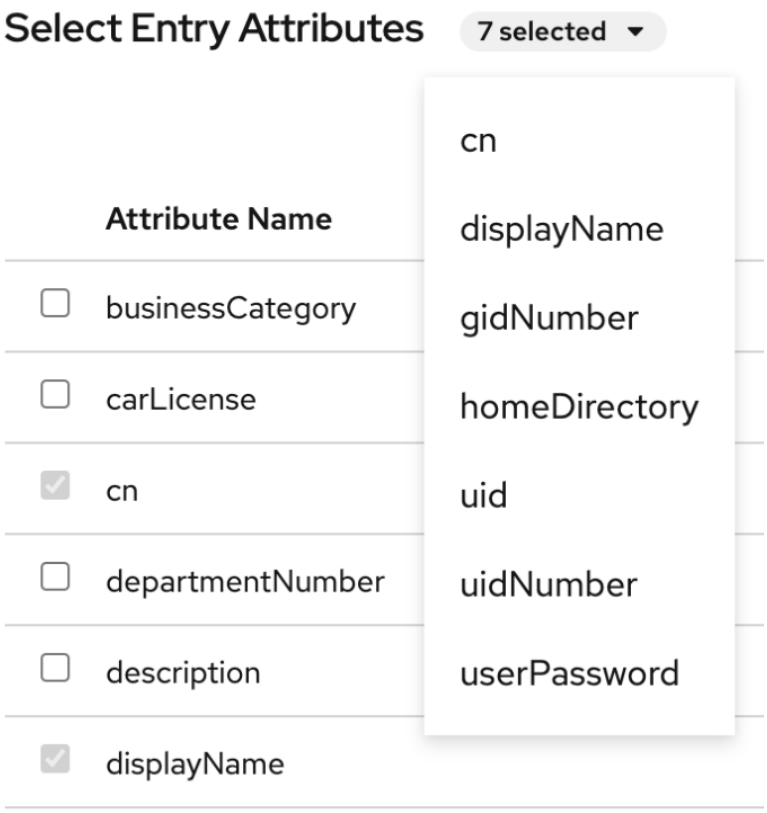
- ディレクトリーサーバー Web コンソールにログインしている。
- 親エントリーが存在する。例: ***ou=people,dc=example,dc=com***

手順

1. Web コンソールで **LDAP Browser** メニューを開き、既存の接尾辞のリストを表示します。
2. **Tree** ビューまたは **Table** ビューを使用して、ユーザーを作成する親エントリー ***ou=people,dc=example,dc=com*** を展開します。
3. **Options** メニュー (⋮) をクリックし、**New** を選択してウィザードウィンドウを開きます。

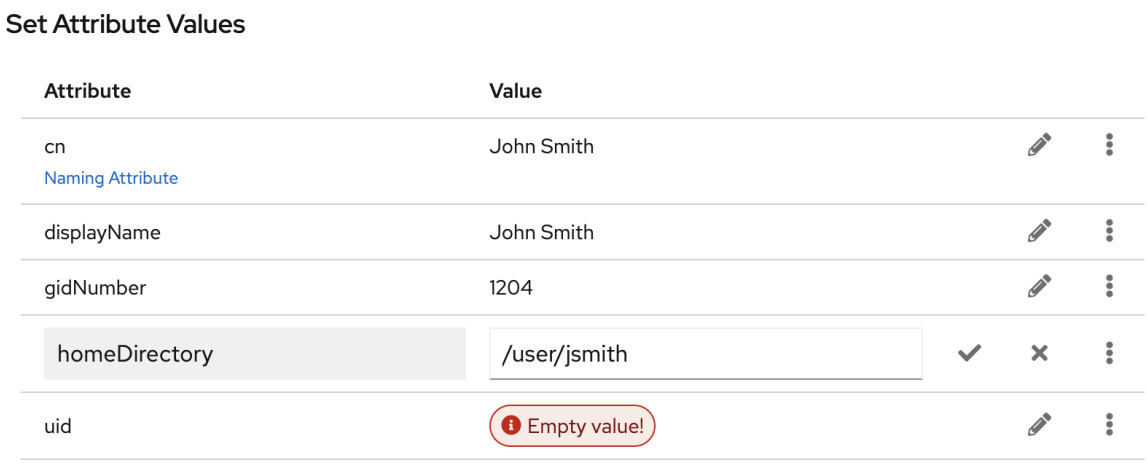


4. **Create a new User** オプションを選択し、**Next** をクリックします。
5. ユーザーエントリーで **Posix Account** タイプを選択し、**Next** をクリックします。
6. オプション: ***userPassword*** のような追加属性を選択し、**Next** をクリックします。ステップ名の近くにあるドロップダウンリストを展開すると、選択されたすべての属性を表示できます。



7. 各属性の値を設定します。

a. 属性の鉛筆ボタンをクリックし、値を追加します。



userPassword 値を設定すると、別のメニューが開くことに注意してください。プレーンテキストを非表示にするために、値にはアスタリスク (*) が入力されます。

b. チェックボタンをクリックして変更を保存します。

c. オプション: **Options メニュー (▮) → Add Another Value** をクリックして、追加の属性値を設定します。

d. すべての値を設定したら、**Next** をクリックします。

8. エントリーの詳細がすべて正しいことを確認し、**Create User** をクリックします。ディレクトリーサーバーは、POSIX ユーザーの必須属性を持つエントリーを作成し、それにパスワードを設定します。**Back** をクリックしてエントリーの設定を変更するか、**Cancel** をクリックしてエントリーの作成をキャンセルできます。

9. **Result for Entry Creation** 表示し、**Finish** をクリックします。

検証

1. **LDAP Browser → Search** に移動します。
2. エントリーを含むデータベース接尾辞 (例: ***dc=example,cd=com***) を選択します。
3. 検索基準 (例: ***John***) をフィールドに入力し、**Enter** を押します。
4. エントリーのリストで最近作成したエントリーを見つけます。

3.2.2. Web コンソールを使用した LDAP エントリーの編集

Web コンソールを使用してディレクトリーエントリーを変更できます。この例では、ユーザーエントリー ***cn=John Smith,ou=people,dc=example,dc=com*** を次のとおり変更します。

- 電話番号 ***556778987*** および ***556897445*** を追加します。
- 電子メール ***jsmith@example.com*** を追加します。
- パスワードを変更します。

前提条件
ディレクトリーサーバー Web コンソールにログインしている。

手順

1. Web コンソールで **LDAP Browser** メニューを開き、既存の接尾辞のリストを表示します。
2. **Tree** ビューまたは **Table** ビューを使用して、編集するエントリー (例: ***cn=John Smith,ou=people,dc=example,dc=com***) を展開します。
3. **Options メニュー (▮)** をクリックし、**Edit** を選択してウィザードウィンドウを開きます。
4. オプション: **Select ObjectClasses** の手順で、エントリーのオブジェクトクラスを追加または削除します。**Next** をクリックします。
5. **Select Attributes** の手順で、エントリーに ***telephoneNumber*** と ***mail*** の属性を追加し、**Next** をクリックします。エントリーに追加する属性が表示されない場合は、前の手順で対応するオブジェクトクラスを追加していないことを意味します。

注記

この手順では、選択したオブジェクトクラスの必須属性は **削除できません**。

6. **Edit Attribute Values** の手順で、 ***telephoneNumber*** を ***556778987*** および ***556897445***、***mail*** を ***jsmith@example.com*** に設定し、 ***userPassword*** 値を変更します。
   a. 属性の鉛筆ボタンをクリックして、新しい値を追加または変更します。
   b. チェックボタンをクリックして変更を保存します。
   c. オプション: **Options メニュー (▮) → Add Another Value** をクリックして、属性に追加の値を設定します。この例では、***telephoneNumber*** 属性には値が 2 つあります。すべての値を設定したら、**Next** をクリックします。
7. 変更内容を確認し、**Next** をクリックします。
8. エントリーを編集するには、**Modify Entry** をクリックします。**Back** をクリックしてエントリーの設定を変更するか、**Cancel** をクリックしてエントリーの編集をキャンセルできます。
9. **Result for Entry Modification**表示し、**Finish** をクリックします。

検証

- エントリーの詳細を展開し、エントリー属性に表示される新しく変更された内容を確認します。

3.2.3. Web コンソールを使用した LDAP エントリーまたはサブツリーの名前変更と再配置

Web コンソールを使用して、ディレクトリーエントリーまたはサブツリーの名前を変更または再配置できます。この例では、エントリー ***cn=John Smith,ou=people,dc=example,dc=com*** の名前と配置を ***cn=Tom Smith,ou=clients,dc=example,dc=com*** に変更します。

前提条件
ディレクトリーサーバー Web コンソールにログインしている。

手順

1. Web コンソールで **LDAP Browser** メニューを開き、既存の接尾辞のリストを表示します。

2. Tree ビューまたは Table ビューを使用して、変更するエントリー (例: ***cn=John Smith,ou=people,dc=example,dc=com***) を展開します。

3. Options メニュー (▮) をクリックし、Rename を選択してウィザードウィンドウを開きます。

4. Select The Naming Attribute And Value の手順で以下を実行します。

   a. 命名属性 ***cn*** に新しい値 ***Tom Smith*** を設定し、**Next** をクリックします。

   b. オプション: ドロップダウンメニューから別の命名属性を選択します。

   c. オプション: 古いエントリーを削除し、新しい RDN を使用して新規エントリーを作成する場合は、Delete the old RDN をオンにします。

5. Select The Entry Location の手順で新しい場所の親エントリーを選択し、**Next** をクリックします。

6. エントリーに加えた変更を確認し、**Next** をクリックします。

7. エントリーの詳細が正しい場合は、**Change Entry Name** をクリックします。**Back** をクリックしてエントリーに別の変更を加えるか、**Cancel** をクリックしてエントリーの変更をキャンセルできます。

8. Result for Entry Modification を表示し、**Finish** をクリックします。

検証

- エントリーの詳細を展開し、更新されたエントリーを確認します。

3.2.4. Web コンソールを使用した LDAP エントリーの削除

Web コンソールを使用して、ディレクトリーエントリーまたはサブツリーを削除できます。この例では、エントリー ***cn=Tom Smith,ou=clients,dc=example,dc=com*** を削除します。

前提条件
ディレクトリーサーバー Web コンソールにログインしている。

手順

1. Web コンソールで LDAP Browser メニューを開き、既存の接尾辞のリストを表示します。

2. Tree ビューまたは Table ビューを使用して、削除するエントリー (例: ***cn=Tom Smith,ou=clients,dc=example,dc=com***) を展開します。

3. Options メニュー (▮) をクリックし、**Delete** を選択してウィザードウィンドウを開きます。

4. 削除するエントリーに関するデータを確認し、**Next** をクリックします。

5. Deletion の手順でスイッチを **Yes, I'm sure** の位置に切り替え、**Delete** をクリックします。**Cancel** をクリックすると、エントリーの削除をキャンセルできます。

6. Result for Entry Deletion を表示し、**Finish** をクリックします。

検証

1. LDAP Browser → Search に移動します。

2. 前にエントリーが存在していた接尾辞 (例: ***dc=example,cd=com***) を選択します。

3. 検索基準 (例: ***Tom***) をフィールドに入力し、**Enter** を押します。

4. 削除されたエントリーが存在しないことを確認します。

第4章 ディレクトリーエントリーの変更の追跡

特定の状況では、エントリーに変更が加えられるタイミングを追跡すると便利です。Directory Server が追跡するエントリーの変更には、以下の 2 つの側面があります。

- 変更シーケンス番号を使用してデータベースへの変更を追跡します。これは、レプリケーションおよび同期で使用されるシーケンス番号の変更に類似しています。通常のディレクトリー操作はすべて、シーケンス番号がトリガーされます。
- 作成および変更の情報を割り当てます。これらの属性は、エントリーを作成して直近に変更したユーザーの名前と、エントリーの作成および修正時のタイムスタンプを記録します。

注記

エントリー更新シーケンス番号 (USN)、時間および名前の変更、および時間および作成はすべて操作属性であり、通常の **ldapsearch** では返されません。操作属性の検索実行に関する詳細は、「操作属性の検索」を参照してください。

4.1. 更新シーケンス番号でデータベースへの変更の追跡

USN プラグインにより、エントリーが変更されたかどうかを LDAP クライアントおよびサーバーが特定できるようになります。

4.1.1. エントリーシーケンス番号の概要

USN プラグインが有効な場合は、エントリーに対して書き込み操作を実行するたびに、エントリーに割り当てられるシーケンス番号 (USN) を更新します。書き込み操作には、add、modify、modrdn、および delete 操作が含まれます。エクスポート操作などの内部データベース操作は、更新シーケンスでカウントされません。 USN カウンターは、最近割り当てられた USN を追跡します。

4.1.1.1. ローカルおよびグローバルの USN

USN は、単一のエントリーではなく、データベース全体に対してグローバルに評価されます。USN は、データベースまたはディレクトリーの変更を追跡するために単に上向きにチェックするという点で、レプリケーションと同期の変更シーケンス番号に似ています。ただし、エントリー USN は CSN と別個に維持され、USN は複製されません。

このエントリーには、***entryUSN*** 操作属性のエントリーへの最後の変更の変更番号が表示されます。操作属性の詳細は、「操作属性の検索」を参照してください。

例4.1 エントリー USN の例

uid=*example*,ou=People,dc=example,dc=com ユーザーエントリーの ***entryusn*** 属性を表示するには、以下のコマンドを実行します。

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com:389 -x -b
"uid=example,ou=People,dc=example,dc=com" -s base -x entryusn

dn: uid=example,ou=People,dc=example,dc=com
entryusn: 17653
```

USN プラグインには、ローカルモードとグローバルモードという 2 つのモードがあります。

- ローカルモードでは、各バックエンドデータベースには、そのバックエンドデータベースに固有の USN カウンターを持つ USN プラグインのインスタンスがあります。これはデフォルト設定です。
- グローバルモードでは、ディレクトリー全体に追加された変更に適用されるグローバル USN カウンターを使用する USN プラグインのグローバルインスタンスがあります。

USN プラグインをローカルモードに設定すると、結果はローカルのバックエンドデータベースに限定されます。USN プラグインをグローバルモードに設定すると、返される結果はディレクトリー全体に対して行われます。

ルート DSE は、***lastusn*** 属性のデータベースのエントリーに割り当てられた最新の USN を表示します。USN プラグインがローカルモードに設定されているため、各データベースに独自のローカル USN カウンターがある場合、***lastUSN*** は、USN が割り当てられているデータベースと、USN の両方を表示します。

```
lastusn;database_name:USN
```

以下に例を示します。

```
lastusn;example1: 2130
lastusn;example2: 2070
```

グローバルモードでは、データベースが共有 USN カウンターを使用する場合、***lastUSN*** 属性は最新の USN のみを表示します。

```
lastusn: 4200
```

4.1.1.2. USN エントリーのインポート

エントリーがインポートされると、USN プラグインは ***nsslapd-entryusn-import-initval*** 属性を使用して、エントリーに USN が割り当てられているかどうかを確認します。***nsslapd-entryusn-import-initval*** の値が数値である場合、インポートされたエントリーはこの数字をエントリーの USN として使用します。***nsslapd-entryusn-import-initval*** の値が数値でない場合、USN プラグインは ***lastUSN*** 属性の値を使用して、インポートしたエントリーの USN で増やします。

4.1.2. USN プラグインの有効化

本セクションでは、**USN** プラグインがエントリーに USN を記録できるようにする方法を説明します。

4.1.2.1. コマンドラインで USN プラグインの有効化

コマンドラインを使用して **USN** プラグインを有効にするには、以下を実行します。

1. **dsconf** ユーティリティーを使用してプラグインを有効にします。

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

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

4.1.2.2. Web コンソールを使用した USN プラグインの有効化

Web コンソールを使用して **USN** プラグインを有効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを選択します。
4. **USN** プラグインを選択します。
5. ステータスを **ON** に変更し、プラグインを有効にします。
6. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

4.1.3. グローバルの USN

デフォルト設定では、Directory Server は各バックエンドデータベースに一意の更新シーケンス番号 (USN) を使用します。あるいは、すべてのバックエンドデータベースで一意の USN を有効にすることができます。

注記

この機能を使用するには、**USN** プラグインを有効にする必要があります。「USN プラグインの有効化」を参照してください。

4.1.3.1. グローバル USN が有効になっているかどうかの特定

本セクションでは、すべてのバックエンドデータベースで USN を有効にするかどうかを特定する方法を説明します。

4.1.3.1.1. コマンドラインでグローバル USN が有効になっているかどうかの特定

コマンドラインを使用して、グローバル USN 機能の現在のステータスを表示するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin usn global
USN global mode is disabled
```

4.1.3.1.2. Web コンソールを使用してグローバル USN が有効になっているかどうかの特定

Web コンソールを使用してグローバル USN 機能の現在のステータスを表示するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを選択します。
4. **USN** プラグインを選択します。

5. **USN Global** スイッチが **On** に設定されていることを確認します。

4.1.3.2. グローバル USN の有効化

4.1.3.2.1. コマンドラインでグローバル USN の有効化

コマンドラインを使用してグローバル USN を有効にするには、以下を実行します。

1. グローバル USN を有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin usn global on
```

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

4.1.3.2.2. Web コンソールを使用したグローバル USN の有効化

Web コンソールを使用してグローバル USN を有効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **USN** プラグインを選択します。
5. プラグインのステータスを **On** に変更します。
6. **USN Global** ステータスを **On** に変更します。
7. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

4.1.4. USN Tombstone エントリーのクリーンアップ

エントリーが削除されると、**USN** プラグインは、エントリーを tombstone エントリーに移動します。レプリケーションが有効な場合は、**USN** および **Replication** プラグインによって個別の tombstone エントリーが保持されます。tombstone エントリーはレプリケーションプロセスで削除されますが、サーバーのパフォーマンスには、USN tombstones を削除することが有益です。

- サーバーをレプリカに変換する前に
- サーバーのメモリーを解放する

4.1.4.1. コマンドラインを使用した USN tombstone エントリーのクリーンアップ

コマンドラインで、**dc=example,dc=com** 接尾辞から USN tombstone エントリーをすべて削除するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin usn cleanup -s
"dc=example,dc=com"
```

-

必要に応じて、**-o *max_USN*** オプションをコマンドに渡して、指定した値まで USN tombstone エントリーを削除します。

4.1.4.2. Web コンソールを使用した USN tombstone エントリーのクリーンアップ

Web コンソールを使用して、**dc=example,dc=com** 接尾辞からすべての USN tombstone エントリーを削除するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **USN** プラグインを選択します。
5. **Run Fixup Task** ボタンをクリックします。
6. フィールドを入力し、**Run** を押します。

4.2. 操作属性によるエントリー変更の追跡

デフォルト設定を使用すると、Directory Server は、全エントリーで以下の操作属性を追跡します。

- ***creatorsName***: エントリーを最初に作成したユーザーの識別名 (DN) です。
- ***createTimestamp***: エントリーの作成時にグリニッジ標準時 (GMT) 形式のタイムスタンプ。
- ***modifiersName***: エントリーを最後に変更したユーザーの識別名。
- ***modifyTimestamp***: エントリーが最後に修正された時点の GMT 形式のタイムスタンプ。

デフォルトの検索では操作属性が返されないことに注意してください。これらの属性はクエリーで明示的に要求する必要があります。詳細については、「操作属性の検索」 を参照してください。

重要

これらの操作属性の追跡は、無効にしないことが推奨されます。無効にすると、エントリーは ***nsUniqueID*** 属性に割り当てられた一意の ID を取得しなくなり、レプリケーションは機能しません。

4.2.1. データベースリンクにより変更されたエントリーまたは作成済みエントリー

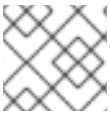
データベースリンク上でエントリーが作成または変更されると、***creatorsName*** および ***modifiersName*** 属性には、リモートサーバーのプロキシー認可権限を付与されたユーザー名が含まれます。この場合、属性はエントリーの元の作成者または最新の変更者を表示しません。ただし、アクセスログには、プロキシーユーザー (**dn**) と元のユーザー (**authzid**) の両方が表示されます。以下に例を示します。

```
[23/May/2018:18:13:56.145747965 +051800] conn=1175 op=0 BIND dn="cn=proxy
admin,ou=People,dc=example,dc=com" method=128 version=3
[23/May/2018:18:13:56.575439751 +051800] conn=1175 op=0 RESULT err=0 tag=97 nentries=0
etime=0 dn="cn=proxy admin,ou=people,dc=example,dc=com"
```

```
[23/May/2018:18:13:56.744359706 +051800] conn=1175 op=1 SRCH base="dc=example,dc=com"
scope=2 filter="(objectClass=*)" attrs=ALL
authzid="uid=user_name,ou=People,dc=example,dc=com"
```

4.2.2. 変更のトラッキングの有効化

デフォルトでは、Directory Server は操作属性の変更を追跡します。



注記

Red Hat は、この機能を無効にしないことが推奨されます。

本セクションでは、この機能を無効にした場合は変更の追跡を再度有効にする方法を説明します。

4.2.2.1. コマンドラインを使用した変更の追跡の有効化

コマンドラインでエントリー変更の追跡を再度有効にするには、次のコマンドを実行します。

1. ***nsslapd-lastmod*** パラメーターを **on** に設定します。

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

2. 必要に応じて、不足している ***nsUniqueID*** 属性を再生成するには、以下を実行します。

   a. データベースを LDAP データ交換形式 (LDIF) ファイルにエクスポートします。「コマンドラインを使用した LDIF ファイルへのデータのエクスポート」を参照してください。

   b. LDIF ファイルからデータベースをインポートします。「コマンドラインでのインポート」を参照してください。

4.3. プラグイン開始更新のバインド DN の追跡

エントリーへの変更の 1 つで、ディレクトリーツリー全体で、他の自動変更をトリガーすることができます。たとえば、ユーザーが削除されると、そのユーザーは **Referential Integrity Postoperation** プラグインが属するグループから自動的に削除されます。

最初のアクションは、サーバーにバインドされているユーザーアカウントによって実行されているものとしてエントリーに表示されますが、関連するすべての更新 (デフォルト) はプラグインによって実行されているものとして表示され、どのユーザーがその更新を開始したかについての情報はありません。たとえば、MemberOf プラグインを使用してグループメンバーシップでユーザーエントリーを更新し、グループアカウントの更新はバインドされたユーザーが実行済みとして表示されますが、ユーザーエントリーの編集は MemberOf プラグインによって実行されると表示されます。

```
dn: cn=example_group,ou=groups,dc=example,dc=com
modifiersname: uid=example,ou=people,dc=example,dc=com

dn: uid=example,ou=people,dc=example,dc=com
modifiersname: cn=memberOf plugin,cn=plugins,cn=config
```

nsslapd-plugin-binddn-tracking パラメーターにより、サーバーは、更新操作を開始したユーザーと、実際に実行した内部プラグインを追跡できます。バインドされたユーザーは ***modifiersname*** 操作属性および ***creatorsname*** 操作属性に表示されますが、実行されたプラグインは ***internalModifiersname*** 操

作属性および ***internalCreatorsname*** 操作属性に表示されます。以下に例を示します。

```
dn: uid=example,ou=people,dc=example,dc=com
modifiersname: uid=admin,ou=people,dc=example,dc=com
internalModifiersname: cn=memberOf plugin,cn=plugins,cn=config
```

nsslapd-plugin-binddn-tracking パラメーターは、バインドされたユーザーと、その接続に対して実行される更新の関係を追跡し、維持します。

注記

internalModifiersname 属性および ***internalCreatorsname*** 属性は、常にプラグインをアイデンティティーとして表示します。このプラグインは、MemberOf プラグインなどの追加のプラグインである可能性があります。コア Directory Server により変更が加えられると、プラグインはデータベースプラグイン (**cn=ldbm database,cn=plugins,cn=config**) になります。

4.3.1. コマンドラインで開始した更新のバインド DN の追跡の有効化

コマンドラインを使用して、プラグインを開始する更新のバインド DN の追跡を有効にするには、以下を実行します。

1. ***nsslapd-plugin-binddn-tracking*** パラメーターを **on** に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-plugin-binddn-tracking=on
```

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

4.3.2. Web コンソールを使用したプラグイン開始の更新のバインド DN の追跡の有効化

Web コンソールを使用して、プラグインを開始する更新のバインド DN の追跡を有効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** メニューを開き、**Server Settings** エントリーを選択します。
4. **Advanced Settings** タブで **Enable Plugin Bind DN Tracking** を選択します。
5. **Save** をクリックします。
6. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

4.4. パスワード変更時間の追跡

パスワードの変更操作は、通常、エントリーに対するその他の変更として扱われるため、更新時間は

lastModified 操作属性に記録されます。ただし、Active Directory 同期におけるパスワードの更新や、他の LDAP クライアントへの接続を容易にするため、パスワード変更の時間は別々に記録しないといけない場合があります。

パスワードポリシー内の ***passwordTrackUpdateTime*** 属性は、エントリーのパスワードが最後に更新された日時のタイムスタンプを記録するようサーバーに指示します。パスワードの変更時間はユーザーエントリーで ***pwdUpdateTime*** 操作属性 (***modifyTimestamp*** または ***lastModified*** 操作属性とは別) として保存されます。

passwordTrackUpdateTime 属性は、パスワード変更時間にアクセスする必要があるクライアントに応じて、グローバルパスワードポリシー、サブツリー、またはユーザーレベルのポリシーの一部として設定できます。パスワードポリシーの設定は、「パスワードポリシーの管理」を参照してください。

第5章 参照整合性の維持

参照整合性 は、関連するエントリー間の関係を維持するデータベースメカニズムです。Directory Server では、参照整合性を使用して、ディレクトリー内の 1 つのエントリーへの更新が、更新されたエントリーを参照するその他のエントリーに適切に反映されるようにします。

たとえば、ユーザーのエントリーがディレクトリーから削除され、参照整合性が有効になると、サーバーはユーザーがメンバーとなるグループからユーザーも削除します。参照整合性が有効になっていないと、ユーザーは管理者が手動で削除するまでグループのメンバーのままになります。これは、ユーザーおよびグループの管理のディレクトリーに依存する他の製品と Directory Server を統合する場合に重要な機能です。

5.1. 参照整合性の仕組み

Referential Integrity Postoperation プラグインを有効にすると、削除または名前変更の操作直後に、指定した属性で整合性の更新を実行します。デフォルトでは、**Referential Integrity Postoperation** プラグインは無効になっています。

注記

マルチサプライヤーレプリケーション環境では、すべてのサプライヤーで **Referential Integrity Postoperation** プラグインを有効にする必要があります。

ディレクトリー内のユーザーまたはグループエントリーを削除、名前変更、または移動すると、その操作は Referential Integrity ログファイルに記録されます。ログファイル内の識別名 (DN) の場合、Directory Server はプラグイン設定に設定された属性を定期的に検索および更新します。

- エントリーの場合は、ログファイルで削除済みと表示され、ディレクトリーの対応する属性が削除されます。
- エントリーの場合は、ログファイルで名前が変更または移動済みと表示され、ディレクトリーの対応する属性の名前が変わります。

デフォルトでは、**Referential Integrity Postoperation** プラグインを有効にすると、**削除** または **名前変更** 操作の直後に ***member***、***uniquemember***、***owner***、および***seeAlso*** 属性の整合性更新が実行されます。ただし、**Referential Integrity Postoperation** プラグインの動作は、ディレクトリーのニーズに応じて複数の方法で設定できます。

- レプリケーション変更ログに整合性の更新を記録します。
- 更新間隔を変更します。
- 参照整合性を適用する属性を選択します。
- 参照整合性を無効にします。

参照整合性で使用される属性はすべて存在、等価性、従属文字列のために、***必ず*** インデックス化する必要があります。これらの属性をインデックス化しないと、変更操作および削除操作のためにサーバーのパフォーマンスが低下します。

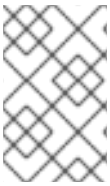
```
nsIndexType: pres
nsIndexType: eq
nsIndexType: sub
```

インデックスの確認および作成に関する詳細は、「標準インデックスの作成」を参照してください。

5.2. レプリケーションによる参照整合性の使用

レプリケーション環境で **Referential Integrity Postoperation** プラグインを使用する場合は、特定の制限があります。

- 専用のコンシューマーサーバー (読み取り専用レプリカのみが含まれるサーバー) では有効に **しないでください**。
- 読み書きレプリカと読み取り専用レプリカの組み合わせが含まれるサーバーで有効に **しないでください**。
- 読み取り/書き込みレプリカ **のみ** を含むサプライヤーサーバーで有効にします。
- マルチサプライヤーレプリケーショントポロジーの各サプライヤーサーバーに対してプラグインを有効にします。プラグイン設定は、すべてのサプライヤーサーバーで同じである必要があります。



注記

サプライヤーサーバーが **Referential Integrity Postoperation** プラグインの変更をコンシューマーサーバーに送信するため、コンシューマーサーバーとハブサーバーで **Referential Integrity Postoperation** プラグインを実行する必要はありません。

5.3. 参照整合性の有効化

このセクションでは、**Referential Integrity Postoperation** プラグインを有効にする方法を説明します。

5.3.1. コマンドラインを使用した参照整合性の有効化

コマンドラインを使用して **Referential Integrity Postoperation** プラグインを有効にするには、以下を実行します。

1. **dsconf** ユーティリティーを使用してプラグインを有効にします。

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

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

5.3.2. Web コンソールを使用した参照整合性の有効化

Web コンソールを使用して **Referential Integrity** プラグインを有効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを選択します。
4. **Referential Integrity** プラグインを選択し、**Show Advanced Settings** をクリックします。

5. ステータスを **ON** に変更し、プラグインを有効にします。

5.4. 参照整合性の更新間隔

デフォルトでは、サーバーは、**削除** または **名前変更** 操作の直後に Referential Integrity の更新を実行します。操作の量によっては、パフォーマンスに影響する可能性があります。パフォーマンスへの影響を軽減するために、更新間の時間を増やすことができます。

更新間隔を秒単位で設定できます。あるいは、以下の値を設定できます。

- **0**: 参照整合性の確認は即座に実行されます。
- **-1**: 参照整合性の確認は実行されません。

重要

マルチサプライヤーのレプリケーション環境では、Red Hat はすべてのサプライヤーで更新間隔を **0** に設定することを推奨しています。

注記

サプライヤーで間隔を 0 より大きい値 (たとえば 5) に設定すると、サプライヤーが直接の **削除** または **名前変更** 操作を受け取り、その操作を複製して、ターゲットエントリーへの参照をクリーンアップする前に **オフライン** になる可能性があります。このような場合、トポロジーの残りの部分には、サーバーが再び稼働するまで (おそらく 5 秒超) ターゲットエントリーへの参照が含まれます。

5.4.1. コマンドラインを使用した更新間隔の表示

コマンドラインを使用して更新間隔を表示するには、以下を行います。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity show
referint-update-delay: 0
...
```

5.4.2. Web コンソールを使用した更新間隔の表示

Web コンソールを使用して更新間隔を表示するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **Referential Integrity** プラグインを選択します。
5. 更新間隔については、**Update Delay** フィールドを参照してください。

5.4.3. コマンドラインを使用した更新間隔の変更

たとえば、コマンドラインを使用して更新間隔を即座更新に設定するには、次を実行します。

1. 更新間隔を **0** に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity set
--update-delay=0
```

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

5.4.4. Web コンソールを使用した更新間隔の変更

たとえば、Web コンソールを使用して更新間隔を即座更新に設定するには、次を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **Referential Integrity** プラグインを選択します。
5. **Update Delay** フィールドに間隔を設定します。
6. **Save Config** を押します。
7. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

5.5. 属性リストの表示および修正

デフォルトでは、参照整合性プラグインは ***member*** 属性、***uniquemember*** 属性、***owner*** 属性、および ***seeAlso*** 属性を確認し、更新します。コマンドラインまたは Web コンソールを使用して、更新する属性を追加または削除できます。

注記

Referential Integrity プラグインのパラメーターリストに設定される属性には、全データベースで等価インデックスが必要です。そうでない場合、プラグインは削除済みまたは変更された DN に一致するためにデータベースのすべてのエントリーをスキャンします。これにより、パフォーマンスに大きく影響する可能性があります。インデックスの確認および作成に関する詳細は、「標準インデックスの作成」を参照してください。

5.5.1. コマンドラインを使用した属性リストの表示

コマンドラインを使用して属性リストを表示するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity show
```

5.5.2. Web コンソールを使用した属性リストの表示

Web コンソールを使用して属性リストを表示するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **Referential Integrity** プラグインを選択します。
5. 属性のリストは、**Membership Attribute** フィールドを参照してください。

5.5.3. コマンドラインで属性リストの設定

コマンドラインを使用して属性リストを更新するには、以下を実行します。

1. 必要に応じて、属性の現在のリストを表示します。「コマンドラインを使用した属性リストの表示」を参照してください。
2. 属性リストを更新します。
   - プラグインが確認および更新される必要がある属性リストを設定するには、以下を実行します。

     ```
     # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity set --membership-attr attribute_name_1 attribute_name_2
     ```

   - プラグインで確認および更新されなくなった属性をすべて削除するには、以下を実行します。

     ```
     # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity set --membership-attr delete
     ```

3. インスタンスを再起動します。

   ```
   # dsctl instance_name restart
   ```

5.5.4. Web コンソールを使用した属性リストの設定

Web コンソールを使用して属性リストを更新するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **Referential Integrity** プラグインを選択します。
5. **Membership Attribute** フィールドを更新して、属性を設定します。
   - 属性を追加するには、**Membership Attribute** フィールドに名前を入力します。
   - 属性を削除するには、**Membership Attribute** フィールドの属性名の横にある **X** ボタンをクリックします。

6. **Save Config** を押します。

5.6. 参照整合性のためのスコープの設定

エントリーを削除すると、エントリーへの参照は削除または変更されます。この更新がすべてのエントリーおよびすべてのグループに適用されると、パフォーマンスに影響が及ぶ可能性があり、選択されたサブツリーに参照整合性を制限できなくなる可能性があります。この問題の *スコープアドレス*を定義します。

たとえば、接尾辞 **dc=example,dc=com** に **ou=active users,dc=example,dc=com** と **ou=deleted users,dc=example,dc=com** の 2 つのサブツリーが含まれる場合があります。**deleted users** のエントリーは、参照整合性の確保のために処理しないでください。

5.6.1. 参照整合性の範囲を制御するパラメーター

Referential Integrity Postoperation プラグイン設定でスコープを定義するために、以下の 3 つのパラメーターを使用することができます。

nsslapd-pluginEntryScope

この多値パラメーターは、削除または名前変更のエントリーの範囲を制御します。これは、Referential Integrity Postoperation プラグインがユーザーエントリーの削除操作または名前変更操作を検索するサブツリーを定義します。定義されたサブツリーに存在しないユーザーを削除または名前変更した場合、プラグインは操作を無視します。このパラメーターを使用すると、プラグインが操作を適用するデータベースのブランチを指定できます。

nsslapd-pluginExcludeEntryScope

このパラメーターは、削除または名前変更のエントリーの範囲も制御します。これは、Referential Integrity Postoperation プラグインがユーザーの削除や名前変更の操作を無視するサブツリーを定義します。

nsslapd-pluginContainerScope

このパラメーターは、参照を更新するグループのスコープを制御します。ユーザーの削除後、Referential Integrity Postoperation プラグインはユーザーが属するグループを検索し、それに応じて更新します。このパラメーターは、プラグインがユーザーが属するグループを検索するブランチを指定します。Referential Integrity Postoperation プラグインは、指定のコンテナーブランチにあるグループのみを更新し、その他のグループは更新されないままにします。

5.6.2. コマンドラインを使用した参照整合性範囲の表示

以下のコマンドは、コマンドラインを使用してスコープ設定を表示する方法を示しています。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity show
...
nsslapd-pluginEntryScope: DN
nsslapd-pluginExcludeEntryScope: DN
nsslapd-pluginContainerScope: DN
```

5.6.3. Web コンソールを使用した参照整合性範囲の表示

以下の手順では、Web コンソールを使用してスコープ設定を表示する方法を説明します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **Referential Integrity** プラグインを選択します。
5. 現在設定されているスコープについては、**Entry Scope** フィールド、**Exclude Entry Scope** フィールド、および **Container Scope** フィールドを参照してください。

5.6.4. コマンドラインを使用した参照整合性範囲の設定

コマンドラインを使用して参照整合性の範囲を設定するには、次のコマンドを実行します。

1. 必要に応じて、スコープ設定を表示します。「コマンドラインを使用した参照整合性範囲の表示」を参照してください。
2. 以下のコマンドは、コマンドラインを使用して個別の参照整合性スコープを設定する方法を示しています。
   - 識別名 (DN) を設定するには、以下を実行します。
     - ***nsslapd-pluginEntryScope*** パラメーターに対して以下を実行します。

       ```
       # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity set --entry-scope="DN"
       ```

     - ***nsslapd-pluginExcludeEntryScope*** パラメーターに対して以下を実行します。

       ```
       # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity set --exclude-entry-scope="DN"
       ```

     - ***nsslapd-pluginContainerScope*** パラメーターに対して以下を実行します。

       ```
       # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity set --container-scope="DN"
       ```

   - DN を削除するには、以下を実行します。
     - ***nsslapd-pluginEntryScope*** パラメーターから、以下を行います。

       ```
       # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity set --entry-scope=delete
       ```

     - ***nsslapd-pluginExcludeEntryScope*** パラメーターから、以下を行います。

       ```
       # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity set --exclude-entry-scope=delete
       ```

     - ***nsslapd-pluginContainerScope*** パラメーターから、以下を行います。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin referential-integrity set --container-scope=delete
```

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

5.6.5. Web コンソールを使用した参照整合性範囲の設定

Web コンソールを使用して参照整合性スコープを設定するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを選択します。
4. **Referential Integrity** プラグインを選択します。
5. **Entry Scope** フィールド、**Exclude Entry Scope** フィールドおよび **Container Scope** フィールドにスコープを設定します。
6. **Save Config** をクリックします。

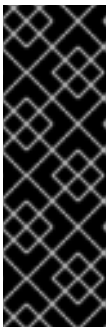
第6章 DIRECTORY DATABASE への入力

データベースには、Red Hat Directory Server が管理するディレクトリーデータが含まれます。

6.1. データのインポート

Directory Server には、以下を行ってデータベースにデータを入力できます。

- データのインポート



重要

データをインポートするには、インポートする LDIF ファイルを **/var/lib/dirsrv/slapd-*instance_name*/ldif/** ディレクトリーに保存する必要があります。Directory Server はデフォルトで **PrivateTmp** systemd ディレクティブを使用します。そのため、LDIF ファイルを **/tmp/** または **/var/tmp/** システムディレクトリーにエクスポートすると、Directory Server はインポート中にこれらの LDIF ファイルを認識しません。**PrivateTmp** の詳細は、**systemd.exec(5)** の man ページを参照してください。

- レプリケーションのデータベースの初期化

次の表に、データベースのインポートと初期化の違いを示します。

表6.1 インポート方法の比較

動作	インポート	データベースの初期化
データベースの上書き	いいえ	はい
LDAP 操作	追加、変更、削除	追加のみ
パフォーマンス	より時間がかかる	速い
パーティション特長	すべてのパーティションで機能	ローカルパーティションのみ
サーバー障害への応答	ベストエフォート (障害が発生した時点までに行われたすべての変更が残る)	アトミック (障害発生後にすべての変更が失われる)
LDIF ファイルの場所	Web コンソールへのローカル	Web コンソールまたはローカルサーバーへのローカル
設定情報 (**cn=config**) をインポートする	はい	いいえ

6.1.1. インポート中の EntryUSN 初期値の設定

エントリーがサーバーからエクスポートされ、別のサーバーにインポートされた場合には、エントリーの更新シーケンス番号 (USN) が保持されません。「更新シーケンス番号でデータベースへの変更の追跡」で説明されているように、エントリー USN はローカルサーバーで発生する操作に割り当てられる

ため、これらの USN を別のサーバーにインポートすることは意味がありません。

ただし、データベースのインポート時やデータベースの初期化時にエントリーに USN 値を設定することが可能です (レプリカがレプリケーションに対して初期化される場合など)。これは、***nsslapd-entryusn-import-initval*** パラメーターを設定します。これにより、インポートされたすべてのエントリーの USN が開始されます。

nsslapd-entryusn-import-initval には 2 つの値を指定することができます。

- 整数。インポートされたすべてのエントリーに使用される明示的な開始番号です。
- **next**。つまり、インポートされたエントリーはすべて、インポート操作の前にサーバー上にあった最大のエントリー USN 値を、1 つずつインクリメントして使用します。

nsslapd-entryusn-import-initval が設定されていない場合、すべてのエントリー USN はゼロで始まります。

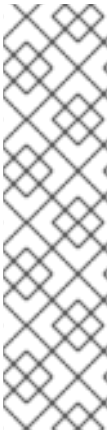
例 6.1 *nsslapd-entryusn-import-initval* パラメーターの仕組み

たとえば、インポートまたは初期化操作の前にサーバーの最大値が **1000** で、***nsslapd-entryusn-import-initval*** の値が **next** の場合、インポートされたすべてのエントリーには **1001** の USN が割り当てられます。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -x "(cn=*)" entryusn

dn: dc=example,dc=com
entryusn: 1001
dn: ou=Accounting,dc=example,dc=com
entryusn: 1001
dn: ou=Product Development,dc=example,dc=com
entryusn: 1001
...
dn: uid=user_name,ou=people,dc=example,dc=com
entryusn: 1001
...
```

エントリー USN の初期値を設定するには、***nsslapd-entryusn-import-initval*** パラメーターを、データをインポートするサーバーまたは初期化を実行するサプライヤーサーバーに追加します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-entryusn-import-initval=next
```



注記

マルチサプライヤーレプリケーションでは、***nsslapd-entryusn-import-initval*** パラメーターはサーバー間で複製 **されません**。つまり、値は、レプリカの初期化に使用しているすべてのサプライヤーサーバーに対して設定する必要があります。

たとえば、**Supplier1** ホストの ***nsslapd-entryusn-import-initval*** が next に設定され、レプリカの初期化に使用される場合は、インポートされたエントリーのエントリー USN は最も高い値に 1 が追加されます。**Supplier2** ホストに ***nsslapd-entryusn-import-initval*** が設定されておらず、レプリカの初期化に使用される場合は、**Supplier1** および **Supplier2** にマルチサプライヤーレプリカ合意がある場合でも、インポートされたエントリーのすべてのエントリー USN はゼロで始まります。

6.1.2. コマンドラインでのインポート

Directory Server は、インスタンスの実行中またはオフライン時にデータのインポートをサポートします。

- インスタンスが実行している場合には、以下のいずれかの方法を使用します。
  - **dsconf backend import** コマンドを使用します。「**dsconf backend import** コマンドを使用したインポート」を参照してください。
  - **cn=tasks** エントリーを作成します。「**cn=tasks** エントリーを使用したデータのインポート」を参照してください。
- インスタンスがオフラインの場合は、**dsctl ldif2db** コマンドを使用します。「サーバーがオフライン時のデータのインポート」を参照してください。

警告

インポート操作を開始すると、Directory Server はまずデータベースから既存のデータをすべて削除し、その後 LDIF ファイルからデータをインポートします。LDIF ファイルが存在しない場合など、インポートに失敗すると、サーバーは以前のデータをデータベースから削除しています。

インポート操作に使用される LDIF ファイルは、**UTF-8** 文字セットエンコーディングを使用する必要があります。インポート操作は、データをローカル文字セットエンコーディングから **UTF-8** に変換しません。また、インポートされたすべての LDIF ファイルにルート接尾辞エントリーが含まれている必要があります。

Directory Server は、**dirsrv** ユーザーとしてインポート操作を実行します。したがって、LDIF ファイルのパーミッションにより、このユーザーがファイルの読み取りを許可する必要があります。

6.1.2.1. サーバーの実行中にデータのインポート

本セクションでは、Directory Server の実行中にデータをインポートする方法を説明します。

6.1.2.1.1. dsconf backend import コマンドを使用したインポート

dsconf backend import コマンドを使用して、LDIF ファイルからデータをインポートするタスクを自

動的に作成します。たとえば、**/var/lib/dirsrv/slapd-*instance_name*/ldif/*instance_name*-*database_name*-*time_stamp*.ldif** ファイルを **userRoot** データベースにインポートするには、以下のコマンドを実行します。

1. 接尾辞が存在しない場合は作成します。詳細については、「接尾辞の作成」 を参照してください。
2. インポートする LDIF に接尾辞エントリーを追加するステートメントが含まれていない場合は、「ルートエントリーの作成」 で説明されているように、このエントリーを手動で作成します。
3. LDIF ファイルをインポートします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend import userRoot
/var/lib/dirsrv/slapd-instance_name/ldif/instance_name-database_name-time_stamp.ldif
The import task has finished successfully
```

dsconf backend import コマンドは、特定の接尾辞を除外するなど、追加オプションに対応します。利用可能なオプションをすべて表示するには、以下を入力します。

```
# dsconf ldap://server.example.com backend import --help
```

6.1.2.1.2. cn=tasks エントリーを使用したデータのインポート

Directory Server 設定の **cn=tasks,cn=config** エントリーは、サーバーがタスクの管理に使用する一時的なエントリー用のコンテナーエントリーです。インポート操作を開始するには、**cn=import,cn=tasks,cn=config** エントリーにタスクを作成します。

インポートタスクエントリーには以下の属性が必要です。

- ***cn***: タスクの一意の名前を設定します。
- ***nsFilename***: インポートする LDIF ファイルの名前を設定します。
- ***nsInstance***: ファイルをインポートするデータベースの名前を設定します。

インポートタスクは、接尾辞を除外するための追加パラメーターをサポートします。完全な一覧は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の 『cn=import』 セクションを参照してください。

たとえば、**/var/lib/dirsrv/slapd-*instance_name*/ldif/example.ldif** ファイルの内容を **userRoot** データベースにインポートするタスクを追加するには:

1. 接尾辞が存在しない場合は作成します。詳細については、「接尾辞の作成」 を参照してください。
2. インポートする LDIF に接尾辞エントリーを追加するステートメントが含まれていない場合は、「ルートエントリーの作成」 で説明されているように、このエントリーを手動で作成します。
3. インポートタスクを追加します。

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

dn: cn=example_import,cn=import,cn=tasks,cn=config
```

```
changetype: add
objectclass: extensibleObject
cn: example_import
nsFilename: /var/lib/dirsrv/slapd-instance_name/ldif/example.ldif
nsInstance: userRoot
```

タスクが完了すると、エントリーはディレクトリー設定から削除されます。

6.1.2.2. サーバーがオフライン時のデータのインポート

データのインポート時にサーバーがオフラインの場合は、**dsctl ldif2db** コマンドを使用します。

1. 接尾辞が存在しない場合は作成します。詳細については、「接尾辞の作成」 を参照してください。
2. インポートする LDIF に接尾辞エントリーを追加するステートメントが含まれていない場合は、「ルートエントリーの作成」 で説明されているように、このエントリーを手動で作成します。
3. インスタンスを停止します。

```
# dsctl instance_name stop
```

4. LDIF ファイルからデータをインポートします。たとえば、**/var/lib/dirsrv/slapd-*instance_name*/ldif/example.ldif** ファイルを **userRoot** データベースにインポートするには、以下を実行します。

```
# dsctl instance_name ldif2db userroot /var/lib/dirsrv/slapd-instance_name/ldif/example.ldif
OK group dirsrv exists
OK user dirsrv exists
[17/Jul/2018:13:42:42.015554231 +0200] - INFO - ldbm_instance_config_cachememsize_set
- force a minimal value 512000
...
[17/Jul/2018:13:42:44.302630629 +0200] - INFO - import_main_offline - import userroot:
Import complete.  Processed 160 entries in 2 seconds. (80.00 entries/sec)
ldif2db successful
```

警告

コマンドで指定したデータベースが、LDIF ファイルに含まれる接尾辞と一致しない場合は、データベースに含まれるすべてのデータが削除され、インポートに失敗します。

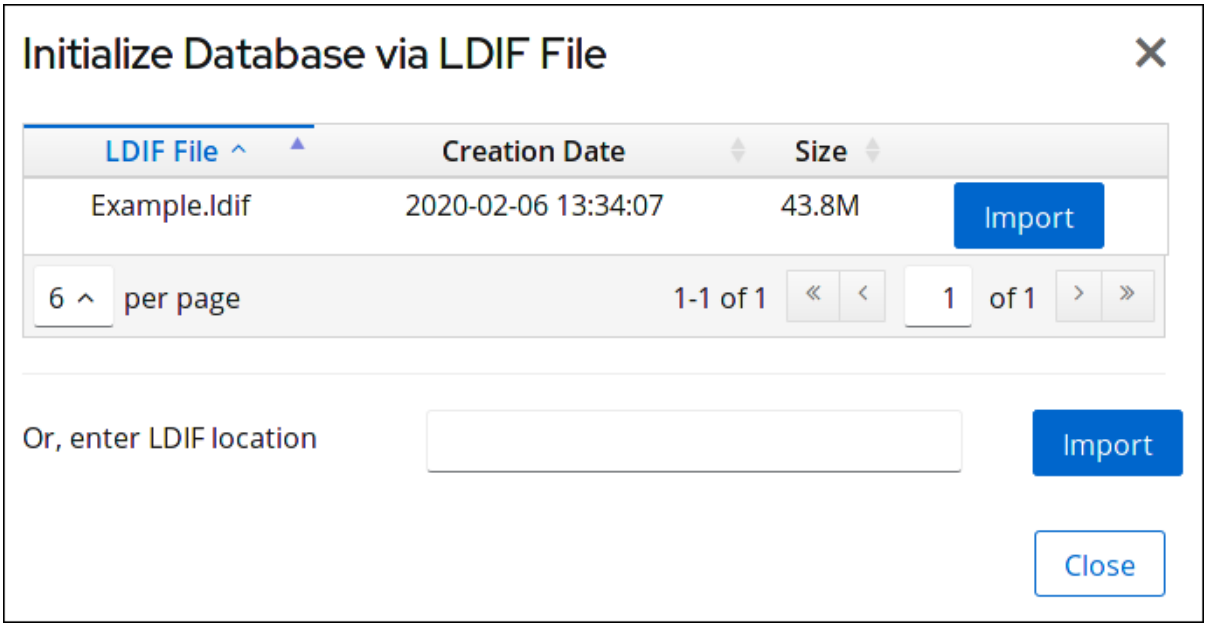
5. インスタンスを起動します。

```
# dsctl instance_name start
```

6.1.3. Web コンソールを使用したデータのインポート

Web コンソールを使用して LDIF ファイルからデータをインポートするには、以下を行います。

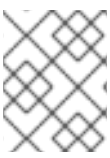
1. 接尾辞が存在しない場合は作成します。詳細については、「接尾辞の作成」 を参照してください。
2. インポートする LDIF に接尾辞エントリーを追加するステートメントが含まれていない場合は、「ルートエントリーの作成」 で説明されているように、このエントリーを手動で作成します。
3. インポートする LDIF ファイルを **/var/lib/dirsrv/slapd-*instance_name*/ldif/** ディレクトリーに保存します。
4. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
5. インスタンスを選択します。
6. **Database** メニューを開きます。
7. 接尾辞エントリーを選択します。
8. **Suffix Tasks** をクリックし、**Initialize Suffix** を選択します。
9. インポートする LDIF ファイルを選択するか、ファイルへの完全パスを入力します。



10. **Yes, I am sure** を選択し、**Initialize Database** をクリックして確定します。

6.2. データのエクスポート

LDAP データ交換形式 (LDIF) ファイルは、Directory Server データベースからデータベースエントリーをエクスポートするために使用されます。LDIF は RFC 2849 で説明されている標準形式です。



注記

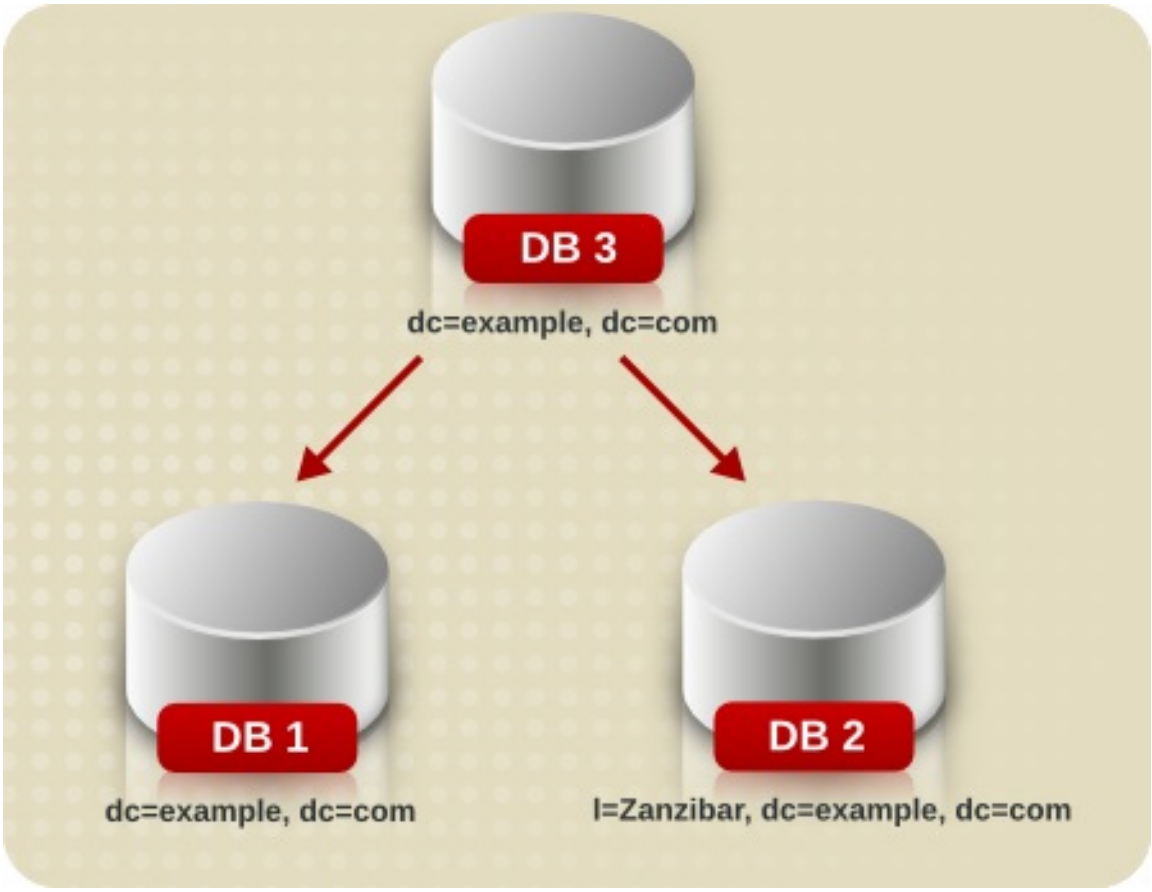
エクスポート操作は、設定情報 (**cn=config**)、スキーマ情報 (**cn=schema**)、または監視情報 (**cn=monitor**) をエクスポートしません。

データのエクスポートは、以下の場合に役に立ちます。

- データベースのデータのバックアップを作成します。
- 別の Directory Server にデータをコピーします。
- 別のアプリケーションへのデータのエクスポート。
- ディレクトリートポロジーの変更後にデータベースを再作成します。

  たとえば、ディレクトリーに 1 つのデータベースが含まれ、その内容を 2 つのデータベースに分割する必要がある場合は、図6.1「データベースコンテンツの 2 つのデータベースへの分割」で説明されるように、2 つの新しいデータベースのコンテンツをエクスポートし、それを 2 つの新しいデータベースにインポートます。

図6.1 データベースコンテンツの 2 つのデータベースへの分割



警告

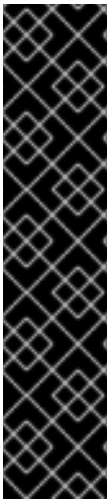
エクスポート操作中はサーバーを停止しないでください。

Directory Server は、エクスポート操作を **dirsrv** ユーザーとして実行します。したがって、移行先ディレクトリーのパーミッションでは、このユーザーがこのファイルを作成できるようにする必要があります。

6.2.1. コマンドラインを使用した LDIF ファイルへのデータのエクスポート

Directory Server は、インスタンスの実行中またはオフライン時にデータのエクスポートをサポートします。

- インスタンスが実行している場合には、以下のいずれかの方法を使用します。
  - **dsconf backend export** コマンドを使用します。「**dsconf backend export** コマンドを使用したデータベースのエクスポート」を参照してください。
  - **cn=tasks** エントリーを作成します。「**cn=tasks** エントリーを使用したデータベースのエクスポート」を参照してください。
- インスタンスがオフラインの場合は、**dsctl db2ldif** コマンドを使用します。「サーバーがオフライン時のデータベースのエクスポート」を参照してください。



重要

LDIF ファイルを **/tmp** または **/var/tmp/** ディレクトリーにエクスポートしないでください。理由は以下のとおりです。

- Directory Server は、デフォルトで **systemd** の **PrivateTmp** 機能を使用します。LDIF ファイルを **/tmp** または **/var/tmp/** システムディレクトリーに配置すると、Directory Server はインポート中にこれらの LDIF ファイルを認識しません。**PrivateTmp** の詳細は、**systemd.exec(5)** の man ページを参照してください。
- LDIF ファイルには、しばしばユーザーパスワードなどの機密データが含まれます。したがって、これらのファイルを保存するために一時システムディレクトリーを使用しないでください。

6.2.1.1. サーバーの実行中にデータベースのエクスポート

6.2.1.1.1. dsconf backend export コマンドを使用したデータベースのエクスポート

dsconf backend export コマンドを使用して、LDIF ファイルにデータをエクスポートするタスクを自動的に作成します。

たとえば、**userRoot** データベースをエクスポートするには、以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend export userRoot
The export task has finished successfully
```

デフォルトでは、**dsconf** は、エクスポートを **/var/lib/dirsrv/slapd-*instance_name*/export/** ディレクトリーの ***instance_name_database_name-time_stamp*.ldif** という名前のファイルに保存します。または、コマンドに **-l file_name** オプションを追加して、別の場所を指定します。

dsconf backend export コマンドは、特定の接尾辞を除外するなど、追加オプションに対応します。利用可能なオプションをすべて表示するには、以下を入力します。

```
# dsconf ldap://server.example.com backend export --help
```

6.2.1.1.2. cn=tasks エントリーを使用したデータベースのエクスポート

Directory Server 設定の **cn=tasks,cn=config** エントリーは、サーバーがタスクの管理に使用する一時的なエントリー用のコンテナーエントリーです。エクスポート操作を開始するには、**cn=export,cn=tasks,cn=config** エントリーにタスクを作成します。

タスクエントリーを使用すると、サーバーの実行中にデータをエクスポートできます。

エクスポートタスクエントリーには以下の属性が必要です。

- ***cn***: タスクの一意の名前を設定します。
- ***nsInstance***: エクスポートするデータベースの名前を設定します。
- ***nsFilename***: エクスポートを保存するファイルの名前を設定します。

エクスポートタスクは、接尾辞を除外するなどの追加のパラメーターをサポートします。完全なリストは、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の 『cn=export』 セクションを参照してください。

たとえば、**userRoot** データベースのコンテンツを **/var/lib/dirsrv/slapd-*instance_name*/ldif/example.ldif** ファイルにエクスポートするタスクを追加するには、次のようにします。

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

dn: cn=example_export,cn=export,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn: example_export
nsInstance: userRoot
nsFilename: /var/lib/dirsrv/slapd-instance_name/ldif/example.ldif
```

タスクが完了すると、エントリーはディレクトリー設定から削除されます。

6.2.1.2. サーバーがオフライン時のデータベースのエクスポート

データのエクスポート時にサーバーがオフラインの場合は、**dsctl db2ldif** コマンドを使用します。

1. インスタンスを停止します。

```
# dsctl instance_name stop
```

2. データベースを LDIF ファイルにエクスポートします。たとえば、**userRoot** データベースを **/var/lib/dirsrv/slapd-*instance_name*/ldif/example.ldif** ファイルにエクスポートするには:

```
# dsctl instance_name db2ldif userroot /var/lib/dirsrv/slapd-instance_name/ldif/example.ldif
OK group dirsrv exists
OK user dirsrv exists
ldiffile: /var/lib/dirsrv/slapd-instance_name/ldif/example.ldif
[18/Jul/2018:10:46:03.353656777 +0200] - INFO - ldbm_instance_config_cachememsize_set
- force a minimal value 512000
[18/Jul/2018:10:46:03.383101305 +0200] - INFO - ldbm_back_ldbm2ldif - export userroot:
Processed 160 entries (100%).
[18/Jul/2018:10:46:03.391553963 +0200] - INFO - dblayer_pre_close - All database threads
now stopped
db2ldif successful
```

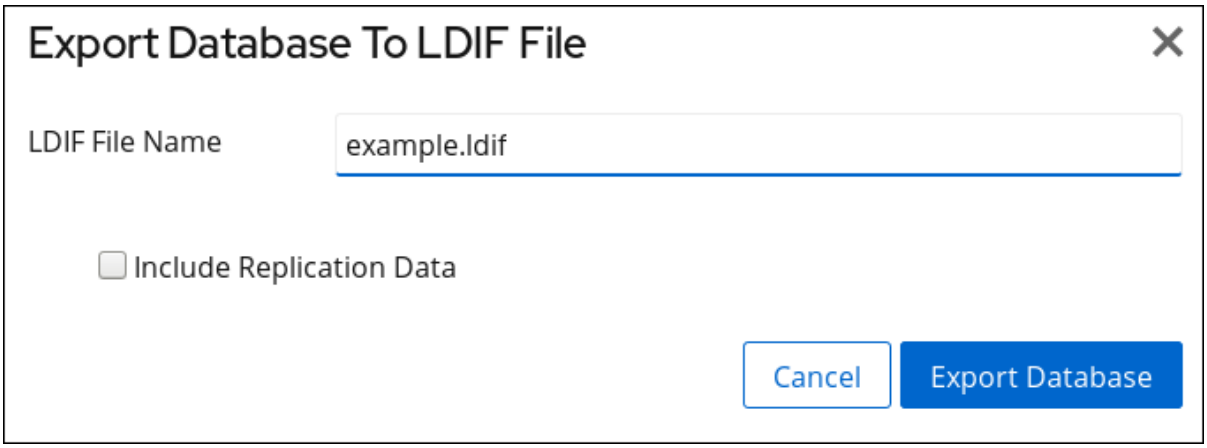
3. インスタンスを起動します。

```
# dsctl instance_name start
```

6.2.2. Web コンソールを使用した LDIF ファイルへの接尾辞のエクスポート

Web コンソールを使用して接尾辞をエクスポートするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. 接尾辞エントリーを選択します。
5. **Suffix Tasks** をクリックし、**Export Suffix** を選択します。
6. エクスポートを保存する LDIF ファイルの名前を入力します。Directory Server は、指定したファイル名を使用して、ファイルを **/var/lib/dirsrv/slapd-*instance_name*/ldif/** ディレクトリーに保存します。



7. **Export Database** をクリックします。

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

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

6.2.3.1. グループがデータをエクスポートすることの許可

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

手順

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

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

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

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

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

3. ユーザーを作成します。

   a. ユーザーアカウントを作成します。

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

   b. ユーザーアカウントのパスワードを設定します。

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

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

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

検証

- **cn=config** に設定した ACI を表示します。

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

```
ldifdir")(version 3.0 ; acl "permission: Allow export_users group to access ldifdir attribute" ;
allow (read,search) groupdn = "ldap:///cn=export_users,ou=groups,dc=example,dc=com";)
...
```

6.2.3.2. 通常のユーザーとしてのエクスポートの実行

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

前提条件

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

手順

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

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

  - タスクの手動での作成:

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

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

検証

- バックアップが作成されたことを確認します。

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

6.3. DIRECTORY SERVER のバックアップ

Directory Server のバックアップには、以下が含まれます。

- これらのデータベース内に格納されているデータを含むすべてのデータベースファイル

注記

Directory Server は、個別のデータベースのバックアップをサポートしません。

- トランザクションログ
- インデックス

バックアップとは対照的に、「データのエクスポート」 で説明されているように、データをエクスポートできます。エクスポート機能を使用して、LDAP Data Interchange Format (LDIF) 形式のサーバーからサブツリーなどの特定のデータをエクスポートします。

警告

バックアップ操作中にサーバーを停止しないでください。

Directory Server は、バックアップタスクを **dirsrv** ユーザーとして実行します。したがって、移行先ディレクトリーのパーミッションでは、このユーザーがファイルを作成できるようにする必要があります。

6.3.1. コマンドラインを使用した全データベースのバックアップ

Directory Server は、インスタンスの実行中またはオフライン時にデータベースのバックアップをサポートします。

- インスタンスが実行している場合には、以下のいずれかの方法を使用します。
  - **dsconf backup create** コマンドを使用します。「**dsconf backup create** コマンドを使用した全データベースのバックアップ」を参照してください。
  - **cn=tasks** エントリーを作成します。「**cn=tasks** エントリーを使用した全データベースのバックアップ」を参照してください。
- インスタンスがオフラインの場合は、**dsctl db2bak** コマンドを使用します。「サーバーがオフライン時のすべてのデータベースのバックアップ」を参照してください。

重要

これらのメソッドはデータベースのみのバックアップを作成します。設定などの他の重要なファイルのバックアップに関する詳細は、「設定ファイル、証明書データベース、およびカスタムスキーマファイルのバックアップ」を参照してください。

6.3.1.1. サーバーの実行中にすべてのデータベースのバックアップ

6.3.1.1.1. dsconf backup create コマンドを使用した全データベースのバックアップ

dsconf backup create コマンドを使用して、全データベースをバックアップするタスクを自動的に作成します。

重要

データベースがオンラインバックアップから復元されると、Directory Server は changelog をクリーンアップします。したがって、オンラインバックアップを使用する場合は、データベースの復元後にレプリカを再初期化する必要があります。再初期化を回避するには、オフラインバックアップを使用します。

たとえば、すべてのデータベースのバックアップを作成するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backup create
The backup create task has finished successfully
```

デフォルトでは、**dsconf** は、バックアップを **/var/lib/dirsrv/slapd-*instance_name*/bak/** ディレクトリーの ***instance_name*-*time_stamp*** というサブディレクトリーに保存します。別の場所を指定するには、コマンドにディレクトリー名を追加します。

6.3.1.1.2. cn=tasks エントリーを使用した全データベースのバックアップ

Directory Server 設定の **cn=tasks,cn=config** エントリーは、サーバーがタスクの管理に使用する一時的なエントリー用のコンテナーエントリーです。バックアップ操作を開始するには、**cn=backup,cn=tasks,cn=config** エントリーでタスクを作成します。

タスクエントリーを使用すると、サーバーの実行中にデータベースのバックアップを作成できます。

バックアップタスクエントリーには以下の属性が必要です。

- ***cn***: タスクの一意の名前を設定します。
- ***nsDatabaseType***: バックアップするデータベースのタイプを設定します。Directory Server は、この属性の **ldbm database** 値のみをサポートします。

バックアップタスクは、デフォルトとして別の宛先ディレクトリーを指定するなど、追加のパラメーターをサポートします (**/var/lib/dirsrv/slapd-*instance_name*/bak/**)。完全なリストは、『Red Hat Directory Server Configuration, Command, and File Reference』の『cn=backup』セクションを参照してください。

たとえば、すべてのデータベースをバックアップし、アーカイブをデフォルトのバックアップディレクトリーに保存するには、以下を実行します。

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

dn: cn=example_backup,cn=export,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn: example_backup
nsDatabaseType: ldbm database
```

nsArchiveDir 属性を指定しないと、サーバーは **/var/lib/dirsrv/slapd-*instance_name*/bak/** ディレクトリーの ***instance_name*-*time_stamp*** という名前のサブディレクトリーにバックアップを保存します。

タスクが完了すると、エントリーはディレクトリー設定から削除されます。

6.3.1.2. サーバーがオフライン時のすべてのデータベースのバックアップ

データベースのバックアップ時にサーバーがオフラインの場合は、**dsctl db2bak** コマンドを使用します。

1. インスタンスを停止します。

```
# dsctl instance_name stop
```

2. データベースのバックアップを作成します。

```
# dsctl instance_name db2bak
db2bak successful
```

注記

dsctl db2bak コマンドは、**dirsrv** ユーザーとしてバックアップとして実行されます。したがって、インストール先ディレクトリーのパーミッションでは、このユーザーがファイルとディレクトリーを作成できるようにする必要があります。

宛先ディレクトリーをコマンドに追加しないと、サーバーは、**/var/lib/dirsrv/slapd-*instance_name*/bak/** ディレクトリーのサブディレクトリー ***instance_name*-*time_stamp*** にバックアップを保存します。

3. インスタンスを起動します。

```
# dsctl instance_name start
```

6.3.2. Web コンソールを使用した全データベースのバックアップ

Web コンソールを使用して、オンラインバックアップを実行できます。

重要

データベースがオンラインバックアップから復元されると、Directory Server は changelog をクリーンアップします。したがって、オンラインバックアップを使用する場合は、データベースの復元後にレプリカを再初期化する必要があります。再初期化を回避するには、オフラインバックアップを使用します。

Web コンソールを使用してインスタンスのデータベースをすべてバックアップするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Actions** ボタンをクリックして、**Manage Backup** を選択します。
4. **Create Backup** をクリックします。
5. バックアップの作成日時を示すタイムスタンプなど、バックアップの名前を入力します。
6. **Create Backup** をクリックします。

サーバーは、バックアップを **/var/lib/dirsrv/slapd-*instance_name*/bak/** ディレクトリー内の指定した名前のサブディレクトリーに保存します。

6.3.3. 設定ファイル、証明書データベース、およびカスタムスキーマファイルのバックアップ

Directory Server に統合されているバックアップメカニズムは、データベースのみのバックアップを作成します。ただし、**/etc/dirsrv/slapd-*instance_name*/** ディレクトリーには追加のファイルが保存されています。これらのファイルは、たとえば、ハードウェア障害後に別のサーバー上でインスタンスを復元するのに必要になります。

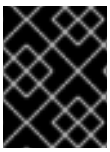
注記

Web コンソールで設定ディレクトリーのバックアップはサポートされていません。

例6.2 /etc/dirsrv/slapd-*instance_name*/ ディレクトリーのバックアップ方法

/etc/dirsrv/slapd-*instance_name*/ のコンテンツをバックアップするには、ディレクトリーをコピーするか、アーカイブファイルに保存します。たとえば、**/etc/dirsrv/slapd-*instance_name*/** ディレクトリーのコンテンツを **/root/config_slapd-*instance_name_time_stamp*.tar.gz** ファイルに保存するには、次のコマンドを実行します。

```
# cd /etc/dirsrv/
# tar -zcvf /root/config_slapd-instance_name_$(date +%Y-%m-%d_%H-%M-%S).tar.gz slapd-instance_name/
```



重要

バックアップ時に、証明書データベースを更新しないでください。更新してしまうと、バックアップ内のデータベースで一貫性がなくなる可能性があります。

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

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

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

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

手順

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

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

2. **cn=backup_users,ou=groups,dc=example,dc=com** グループのメンバーがバックアップタスクを作成するのを許可するアクセス制御手順 (ACI) を追加します。

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

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

3. ユーザーを作成します。

   a. ユーザーアカウントを作成します。

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

   b. ユーザーアカウントのパスワードを設定します。

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

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

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

検証

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

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

6.3.4.2. 通常のユーザーとしてのバックアップの実行

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

前提条件

- **cn=backup_users,ou=groups,dc=example,dc=com** グループのメンバーがデータをバックアップするのを許可している。「グループが Directory Server をバックアップすることの許可」を参照してください。
- バックアップの実行に使用するユーザーが **cn=backup_users,ou=groups,dc=example,dc=com** グループのメンバーである。

手順

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

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

  - タスクの手動での作成:

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

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

検証

- バックアップが作成されたことを確認します。

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

6.4. DIRECTORY SERVER の復元

特定の状況では、管理者がハードウェア障害後など Directory Server を復元する必要があります。本セクションでは、サポートされている復元方法を説明します。

注記

Directory Server は、個別のデータベースの復元には対応していません。

Directory Server は、復元操作を **dirsrv** ユーザーとして実行します。そのため、バックアップを含むディレクトリーのパーミッションにより、このユーザーがファイルを読み取ることができます。

6.4.1. コマンドラインでのすべてのデータベースの復元

Directory Server は、インスタンスの実行中またはオフライン時にデータベースの復元をサポートします。

- インスタンスが実行している場合には、以下のいずれかの方法を使用します。
  - **dsconf backup restore** コマンドを使用します。「**dsconf backup restore** コマンドを使用した全データベースの復元」を参照してください。
  - **cn=tasks** エントリーを作成します。「**cn=tasks** エントリーを使用した全データベースの復元」を参照してください。
- インスタンスがオフラインの場合は、**dsctl bak2db** コマンドを使用します。「サーバーがオフライン時の全データベースの復元」を参照してください。

6.4.1.1. サーバーの実行中にすべてのデータベースの復元

6.4.1.1.1. dsconf backup restore コマンドを使用した全データベースの復元

dsconf backup restore コマンドを使用して、バックアップディレクトリーからすべてのデータベースを復元するタスクを自動的に作成します。

たとえば、**/var/lib/dirsrv/slapd-instance_name/bak/*instance_name-time_stamp*/** ディレクトリーに保存されているバックアップを復元するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backup restore /var/lib/dirsrv/slapd-instance_name/bak/instance_name-time_stamp/
The backup restore task has finished successfully
```

6.4.1.1.2. cn=tasks エントリーを使用した全データベースの復元

Directory Server 設定の **cn=tasks,cn=config** エントリーは、サーバーがタスクの管理に使用する一時的なエントリー用のコンテナーエントリーです。復元操作を開始するには、**cn=restore,cn=tasks,cn=config** エントリーでタスクを作成します。

警告

復元タスクを使用すると、インスタンス内のすべてのデータが上書きされます。

復元タスクエントリーには以下の属性が必要です。

- ***cn***: タスクの一意の名前を設定します。
- ***nsArchiveDir***: バックアップが含まれるディレクトリーへのパスを設定します。
- ***nsDatabaseType***: 復元するデータベースのタイプを設定します。Directory Server は、この属性の **ldbm database** 値のみをサポートします。

たとえは、**/var/lib/dirsrv/slapd-*instance_name*/bak/*instance_name-time_stamp*/** ディレクトリーに保存されているバックアップからすべてのデータベースを復元するタスクを追加するには、次のコマンドを実行します。

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

dn: cn=example_restore,cn=import,cn=tasks,cn=config
changetype: add
objectclass: extensibleObject
cn: example_restore
nsArchiveDir: /var/lib/dirsrv/slapd-instance_name/bak/instance_name-time_stamp/
nsDatabaseType: ldbm database
```

タスクが完了すると、エントリーはディレクトリー設定から削除されます。

6.4.1.2. サーバーがオフライン時の全データベースの復元

データベースの復元時にサーバーがオフラインの場合は、**dsctl bak2db** コマンドを使用します。

1. インスタンスを停止します。

   ```
   # dsctl instance_name stop
   ```

2. データベースを復元します。たとえば、**/var/lib/dirsrv/slapd-*instance_name*/bak/*instance_name-time_stamp*/** ディレクトリーに保存されているバックアップからすべてのデータベースを復元するタスクを追加するには、次のコマンドを実行します。

   ```
   # dsctl instance_name bak2db
   /var/lib/dirsrv/slapd-instance_name/bak/instance_name-time_stamp/
   bak2db successful
   ```

   注記

   dsctl bak2db コマンドは、**dirsrv** ユーザーとして復元として実行されます。したがって、ソースディレクトリーのパーミッションでは、このユーザーがファイルとディレクトリーを読み取りできるようにする必要があります。

3. インスタンスを起動します。

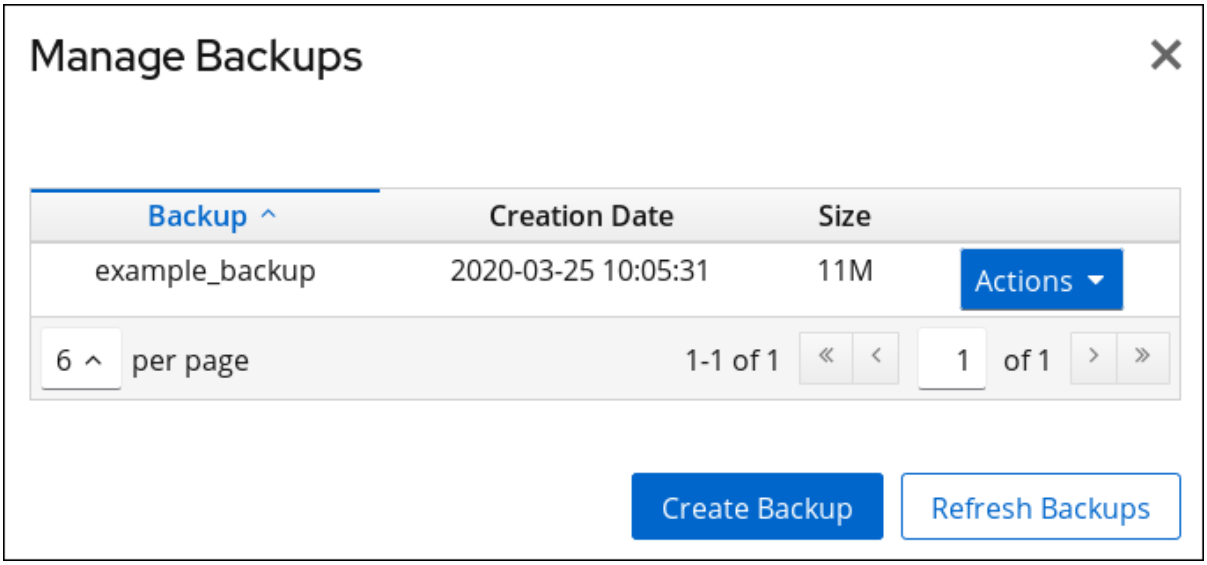
   ```
   # dsctl instance_name start
   ```

6.4.2. Web コンソールを使用した全データベースの復元

Web コンソールを使用してすべてのデータベースを復元するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Actions** ボタンをクリックして、**Manage Backups** を選択します。

表示されるウィンドウには、**/var/lib/dirsrv/slapd-*instance_name*/bak/** ディレクトリーで利用可能なバックアップが表示されます。



4. 復元するバックアップの横にある **Actions** メニューを開き、**Restore Backup** を選択します。
5. **Yes** をクリックして確定します。

6.4.3. 複製されたエントリーが含まれるデータベースの復元

サプライヤーサーバーを復元すると、いくつかの状況が発生する可能性があります。

- コンシューマーサーバーも復元される。

  (データが同期されるように) 全く同じ時間に作成されたバックアップからすべてのデータベースを復元するような非常にまれな状況においては、コンシューマーはサプライヤーと同期が取れた状態のままであるため、特に何もする必要はありません。レプリケーションは中断せずに再開します。

- サプライヤーだけが復元します。

  サプライヤーのみが復元された場合や、コンシューマーが別のバックアップから復元された場合は、サプライヤーがコンシューマーを再初期化して、データベースのデータを更新します。
  サプライヤーのみが復元された場合や、コンシューマーが別のバックアップから復元された場合は、サプライヤーがコンシューマーを再初期化して、データベースのデータを更新します。

- サプライヤーサーバーでチェンジログエントリーの有効期限が切れていません。

  データベースのバックアップの取得後にサプライヤーの変更ログが期限切れになっていない場合は、ローカルコンシューマーを復元し、通常の操作を継続します。この状態は、**cn=changelog5,cn=config** エントリーで、最大 changelog age 属性 ***nsslapd-changelogmaxage*** に設定された値よりも短い期間内にバックアップを取得した場合に限り発生します。このオプションの詳細は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 を参照してください。

  Directory Server は、レプリカとその変更ログの間の互換性を自動的に検出します。不一致が検出されると、サーバーは古い変更ログファイルを削除し、空のファイルを新たに作成します。

- 変更ログエントリーが、ローカルバックアップを作成した後にサプライヤーサーバー上で期限切れになる。

変更ログエントリーの有効期限が切れている場合は、コンシューマーが再初期化される。コンシューマーの再初期化に関する詳細は、「コンシューマーの初期化」を参照してください。

例6.3 Directory Server のレプリケーショントポロジーの復元

たとえば、2 つのサプライヤーと 2 つのコンシューマーサーバーで設定されるレプリケーション環境のサーバーをすべて復元するには、以下を実行します。

1. 最初のサプライヤーを復元します。**dsconf backend import** コマンドを使用して、データをインポートします。「コマンドラインでのインポート」を参照してください。
2. レプリケーションを使用して残りのサーバーをオンラインに初期化します。
   a. 最初のサプライヤーから 2 番目のサプライヤーを初期化します。
   b. サプライヤーからコンシューマーを初期化します。

   詳細については、「コンシューマーの初期化」 を参照してください。
3. 各サーバーでレプリケーションのステータスを表示し、レプリケーションが正しく機能していることを確認します。詳細については、「特定のレプリカ合意の状態の表示」 を参照してください。

復元操作中に、復元されたデータベースに関連する変更ログが削除されます。再初期化が必要であることを示すメッセージが、サプライヤーサーバーのログファイルに記録されます。

レプリケーションの管理に関する情報は、15章 *レプリケーションの管理*を参照してください。

第7章 属性および値の管理

Red Hat Directory Server は、ディレクトリーエントリーで一部の属性タイプを動的かつ自動的に維持するためのさまざまなメカニズムを提供します。これらのプラグインおよび設定オプションを使用すると、ディレクトリーデータの管理やエントリー間の関係の表現が容易になります。

エントリーの特徴の一部は、相互 **関係** です。とうぜん。マネージャーには従業員がいるため、この 2 つのエントリーには関連性があります。グループはメンバーに関連付けられます。共通の物理的な場所を共有するエントリー間のように、あまり明白でない関係もあります。

Red Hat Directory Server は、このようなエントリー間の関係をスムーズにかつ一貫して維持する方法を複数提供します。複数のプラグインは、ディレクトリー内のデータの一部として属性を自動的に適用または生成できます。これには、サービスのクラス、属性のリンク、一意の数値属性値の生成が含まれます。

7.1. 属性の一意性の有効化

ディレクトリーまたはサブツリー全体で属性の値が一意になるように、**Attribute Uniqueness** プラグインを使用します。

複数の属性を一意にしたい場合や、異なる条件を使用する場合は、プラグインに複数の設定レコードを作成します。

7.1.1. Attribute Uniqueness プラグインの新規設定レコードの作成

値が一意である必要がある属性ごとに、**Attribute Uniqueness** プラグインの新しい設定レコードを作成します。

注記

コマンドラインからプラグインの新しい設定レコードのみを作成できます。

Example **Attribute Uniqueness** という名前のプラグインの設定解除および無効にした新しい設定レコードを作成するには、以下を実行します。

```
dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq add "Example" --attr-name uid
```

7.1.2. 接尾辞またはサブツリーにおける属性一意の設定

Attribute Uniqueness プラグインを設定して、特定の接尾辞、サブツリー、または接尾辞およびサブツリーで属性の値が一意になるようにすることができます。

7.1.2.1. コマンドラインで接尾辞またはサブツリーに対する属性一意の設定

たとえば、**mail** 属性に保存される値が一意となるように設定するには、以下を実行します。

1. たとえば **mail Attribute Uniqueness** という名前の Attribute Uniqueness プラグインの新たな設定レコードを作成します。詳細については、「Attribute Uniqueness プラグインの新規設定レコードの作成」 を参照してください。
2. プラグイン設定レコードを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq enable "mail
Attribute Uniqueness"
```

3. ***mail*** 属性に保存されている値は、内部で一意である必要があります。たとえば、**ou=Engineering,dc=example,dc=com** および **ou=Sales,dc=example,dc=com** サブツリーなどです。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq set "mail
Attribute Uniqueness" --attr-name mail --subtree ou=Engineering,dc=example,dc=com
ou=Sales,dc=example,dc=com
```

4. 必要に応じて、このプラグイン設定レコードに設定されたすべてのサブツリーで一意性を設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq set "mail
Attribute Uniqueness" --across--all-subtrees=on
```

5. インスタンスを再起動します。

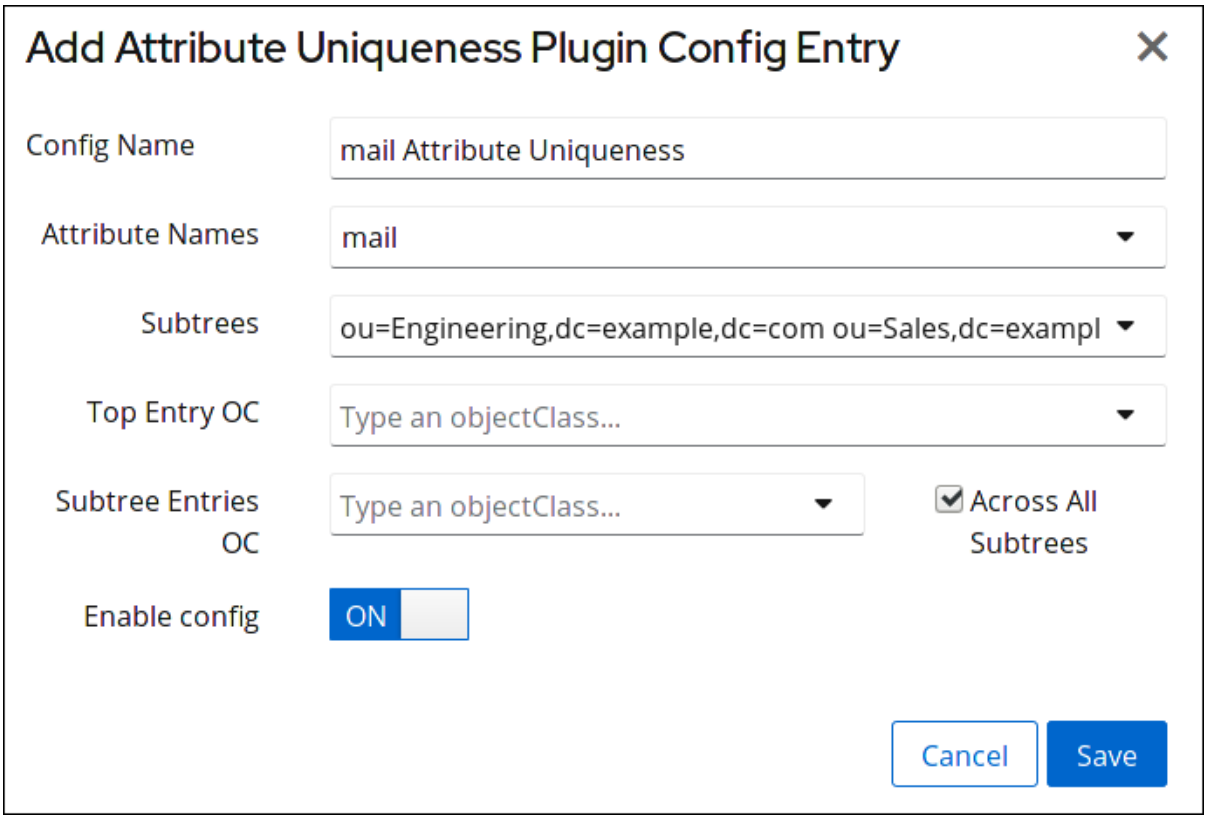
```
# dsctl instance_name restart
```

7.1.2.2. Web コンソールを使用した接尾辞またはサブツリーに対する属性の一意の設定

たとえば、**mail** 属性に保存される値が一意となるように設定するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **Attribute Uniqueness** プラグインを選択します。
5. **Add Config** をクリックします。
6. フィールドを入力し、設定を有効にします。以下に例を示します。

図7.1 属性一意設定の追加



7. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

7.1.3. オブジェクトクラスに対する属性の一意性の設定

Attribute Uniqueness プラグインを設定して、特定のオブジェクトクラスが含まれるサブツリーエントリーで属性の値が一意になるようにすることができます。Directory Server は、更新されたオブジェクトの親エントリーでこのオブジェクトクラスを検索します。Directory Server でオブジェクトクラスが見つからなかった場合、検索はディレクトリーツリーのルートまで次の上位レベルのエントリーで続行されます。オブジェクトクラスが見つかった場合、Directory Server は、***uniqueness-attribute-name*** に設定された属性の値がこのサブツリー内で一意であることを確認します。

たとえば、**mail** 属性に保存されている値が、**nsContainer** オブジェクトクラスが含まれるエントリーで一意となるように設定するには、以下を実行します。

1. たとえば **mail Attribute Uniqueness** という名前の Attribute Uniqueness プラグインの新たな設定レコードを作成します。詳細については、「Attribute Uniqueness プラグインの新規設定レコードの作成」を参照してください。

2. プラグイン設定レコードを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq enable "mail Attribute Uniqueness"
```

3. ***mail*** 属性に保存される値は、**nsContainer** オブジェクトクラスが含まれるエントリーで一意である必要があります。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq set "mail Attribute Uniqueness" --top-entry-oc=nsContainer
```

4. 必要に応じて、チェックされるオブジェクトの範囲を制限できます。サーバーが **nsContainer** オブジェクトクラスを含むエントリーの下にあるエントリーのサブセットのみをチェックするようにするには、***uniqueness-subtree-entries-oc*** パラメーターに追加のオブジェクトクラスを設定します。この追加クラスも存在している必要があります。

   たとえば、**nsContainer** オブジェクトクラスセットが含まれるエントリーにあるすべてのエントリーで ***mail*** 属性を一意にする必要があります。ただし、プラグインは、**inetOrgPerson** など、この属性を提供するオブジェクトクラスが含まれるエントリーで ***mail*** のみを検索します。この場合は、以下を入力します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin attr-uniq set "mail Attribute Uniqueness" --subtree-entries-oc=inetOrgPerson
   ```

5. インスタンスを再起動します。

   ```
   # dsctl instance_name restart
   ```

7.1.4. 属性の一意性プラグイン設定パラメーター

Attribute Uniqueness プラグインの設定レコードを設定するには、**cn=*attribute_uniqueness_configuration_record_name*,cn=plugins,cn=config** エントリーでプラグインの設定属性を設定します。

例7.1 プラグイン固有の属性を使用した属性の一意性プラグイン設定

```
dn: cn=Example Attribute Uniqueness,cn=plugins,cn=config
nsslapd-pluginEnabled: on
uniqueness-attribute-name: attribute_name
uniqueness-top-entry-oc: objectclass1
uniqueness-subtree-entries-oc: objectclass2
```

Attribute Uniqueness プラグインを設定するために設定できるパラメーターの一覧は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』の該当するセクションを参照してください。

7.2. サービスのクラスの割り当て

サービス定義 (CoS) は、アプリケーションに透過的な方法でエントリー間で属性を共有します。CoS はエントリー管理を簡素化し、ストレージ要件を削減します。

Directory Server のクライアントは、ユーザーのエントリーの属性を読み取ります。CoS では、一部の属性値はエントリー自体に保存されない可能性があります。代わりに、エントリーがクライアントアプリケーションに送信されるため、これらの属性の値はサービスロジックのクラスによって生成されます。

各 CoS は、ディレクトリー内に 2 種類のエントリーで設定されます。

- **CoS 定義エントリー。**CoS 定義エントリーは、使用される CoS のタイプを識別します。ロール定義エントリーと同様に、**LDAPsubentry** オブジェクトクラスから継承されます。CoS 定義エントリーは、有効なブランチの下にあります。

- **テンプレートエントリー。**CoS テンプレートエントリーには、共有属性値のリストが含まれま

す。テンプレートエントリー属性値への変更は、CoS の範囲内のすべてのエントリーに自動的に適用されます。1 つの CoS に、複数のテンプレートエントリーが関連付けられている場合があります。

CoS 定義エントリーとテンプレートエントリーは、CoS の範囲内で任意のターゲットエントリーに属性情報を提供するために対話します。

7.2.1. CoS 定義エントリーの概要

CoS 定義エントリーは、**cosSuperDefinition** オブジェクトクラスのインスタンスです。CoS 定義エントリーには、エントリーを生成するために使用するテンプレートエントリーのタイプを指定する 3 つのオブジェクトクラスのいずれか 1 つも含まれています。CoS と対話するターゲットエントリーは、CoS 定義エントリーと同じ親を共有します。

CoS には 3 つのタイプの CoS 定義エントリーを使用して定義されます。

- **ポインター CoS。** ポインター CoS は、テンプレート DN のみを使用してテンプレートエントリーを特定します。
- **間接的な CoS。** 間接 CoS は、ターゲットエントリーの属性の 1 つを使用してテンプレートエントリーを識別します。たとえば、間接的な CoS はターゲットエントリーの ***manager*** 属性を指定する場合があります。次に、***manager*** 属性の値を使用してテンプレートエントリーを特定します。

  ターゲットエントリーの属性は単値であり、DN が含まれる必要があります。
- **Classic CoS。** Classic CoS は、テンプレートエントリーのベース DN とターゲットエントリーの属性の 1 つの値を使用してテンプレートエントリーを特定します。

CoS の各タイプのオブジェクトクラスおよび属性に関する詳細は、「コマンドラインでの CoS の管理」を参照してください。

CoS ロジックが、CoS が値を生成する属性を含むことを検出すると、デフォルトでは CoS が値を生成する属性を検出すると、クライアントアプリケーションにエントリー自体の属性値が提供されます。ただし、CoS 定義エントリーはこの動作を制御できます。

7.2.2. CoS テンプレートエントリーの概要

CoS テンプレートエントリーには、CoS ロジックによって生成された属性の値 (1 つまたは複数) が含まれます。CoS テンプレートエントリーには、一般的なオブジェクトクラス **cosTemplate** が含まれます。特定の CoS テンプレートエントリーは、CoS 定義とともにディレクトリーツリーに保存されます。

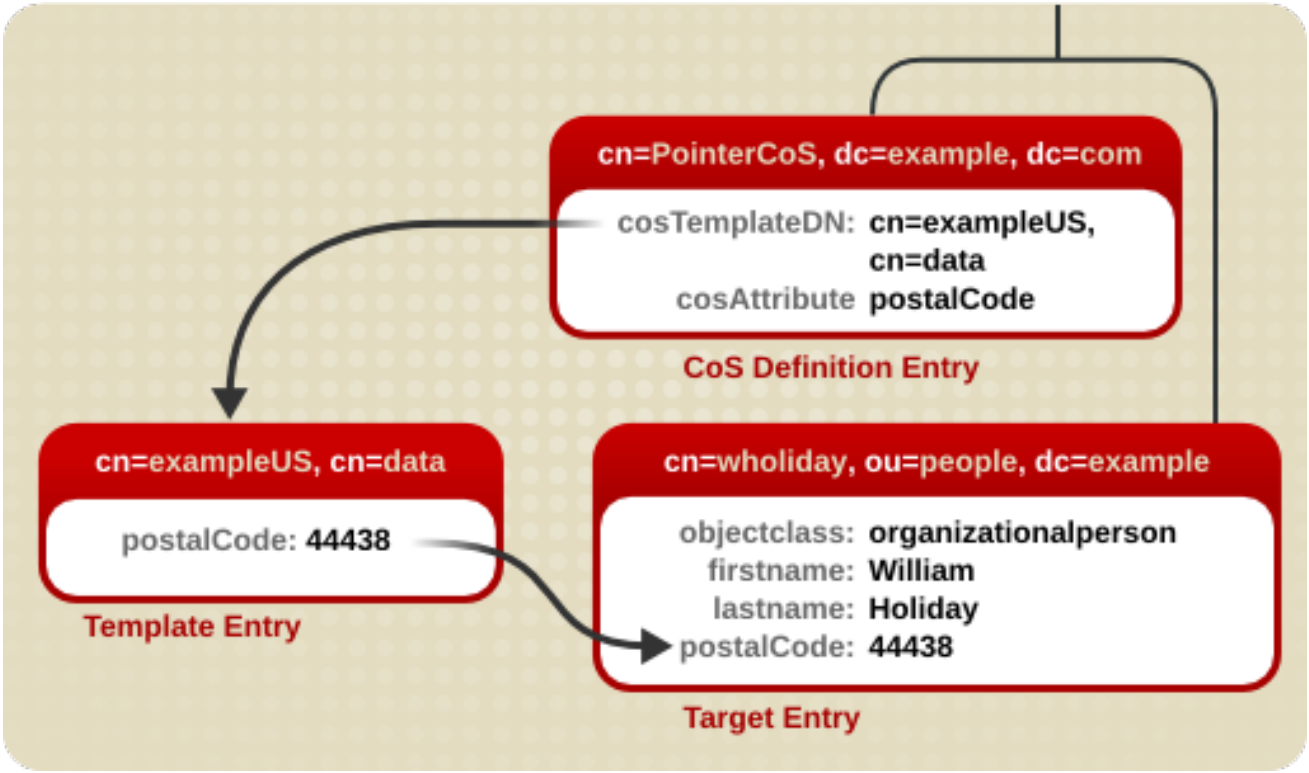
テンプレートエントリーの相対識別名 (RDN) は、以下のいずれかで決定されます。

- テンプレートエントリーの DN のみ。このタイプのテンプレートは Pointer CoS 定義に関連付けられます。
- ターゲットエントリーの属性の 1 つの値。テンプレートエントリーに相対 DN を提供するために使用される属性は、***cosIndirectSpecifier*** 属性を使用して CoS 定義エントリーに指定されます。このタイプのテンプレートは、間接 CoS 定義に関連付けられます。
- CoS がテンプレートの 1 つのレベル検索を実行するサブツリーの DN と、ターゲットエントリーの属性の 1 つの値の組み合わせ。このタイプのテンプレートは、Classic CoS 定義に関連付けられます。

7.2.3. Pointer CoS の仕組み

管理者は、**dc=example,dc=com** に保存されているすべてのエントリーと共通の郵便番号を共有するポインター CoS を作成します。図7.2「Pointer CoS のサンプル」に示されるように、この CoS の 3 つのエントリーが表示されます。

図7.2 Pointer CoS のサンプル

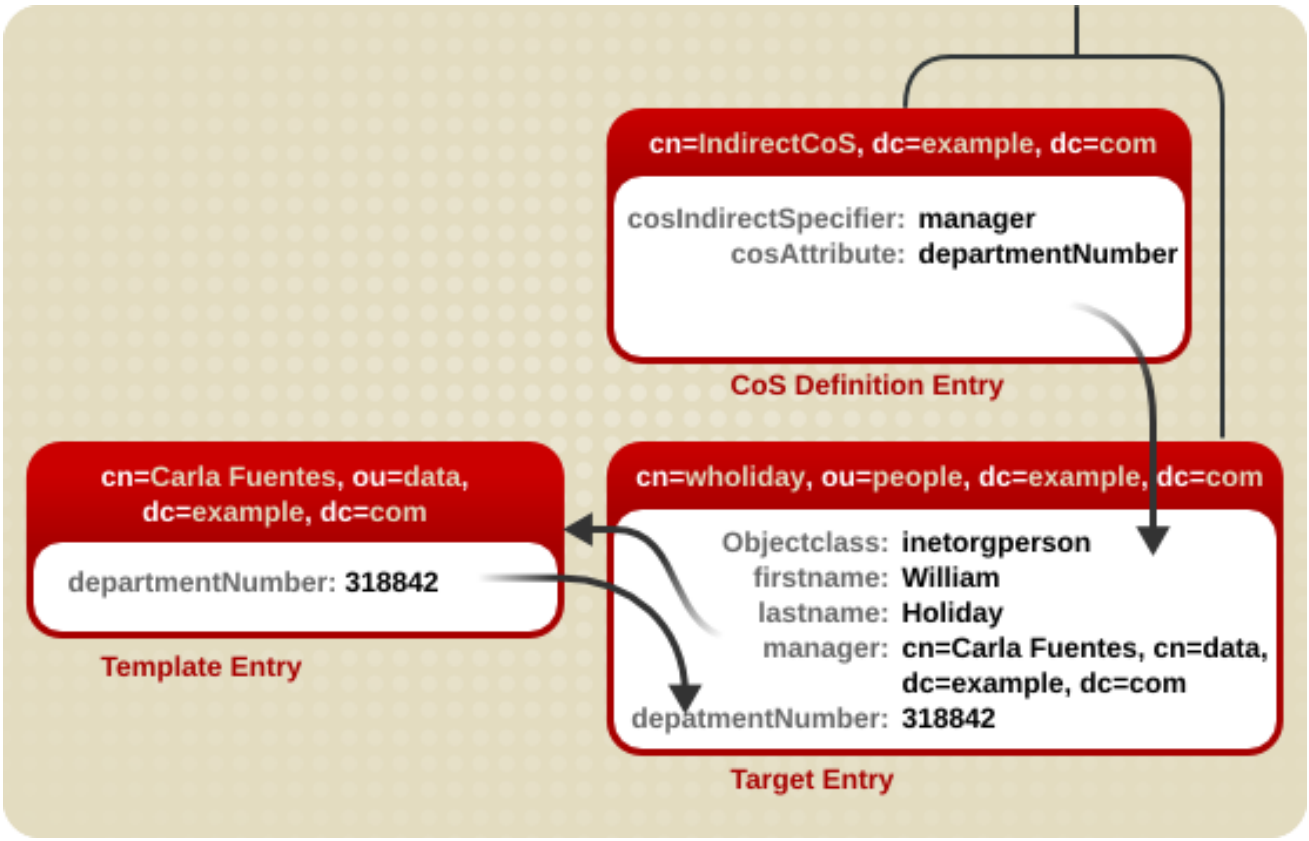


この例では、テンプレートエントリーが CoS 定義エントリーの DN **cn=exampleUS,cn=data** で識別されます。***postalCode*** 属性がエントリー **cn=wholiday,ou=people,dc=example,dc=com** に対してクエリーされるたびに、Directory Server は、テンプレートエントリー **cn=exampleUS,cn=data** で使用可能な値を返します。

7.2.4. 間接的な CoS の仕組み

管理者は、ターゲットエントリーの ***manager*** 属性を使用してテンプレートエントリーを識別する間接的な CoS を作成します。図7.3「間接的な CoS の例」に示されるように、3 つの CoS エントリーが表示されます。

図7.3 間接的な CoS の例

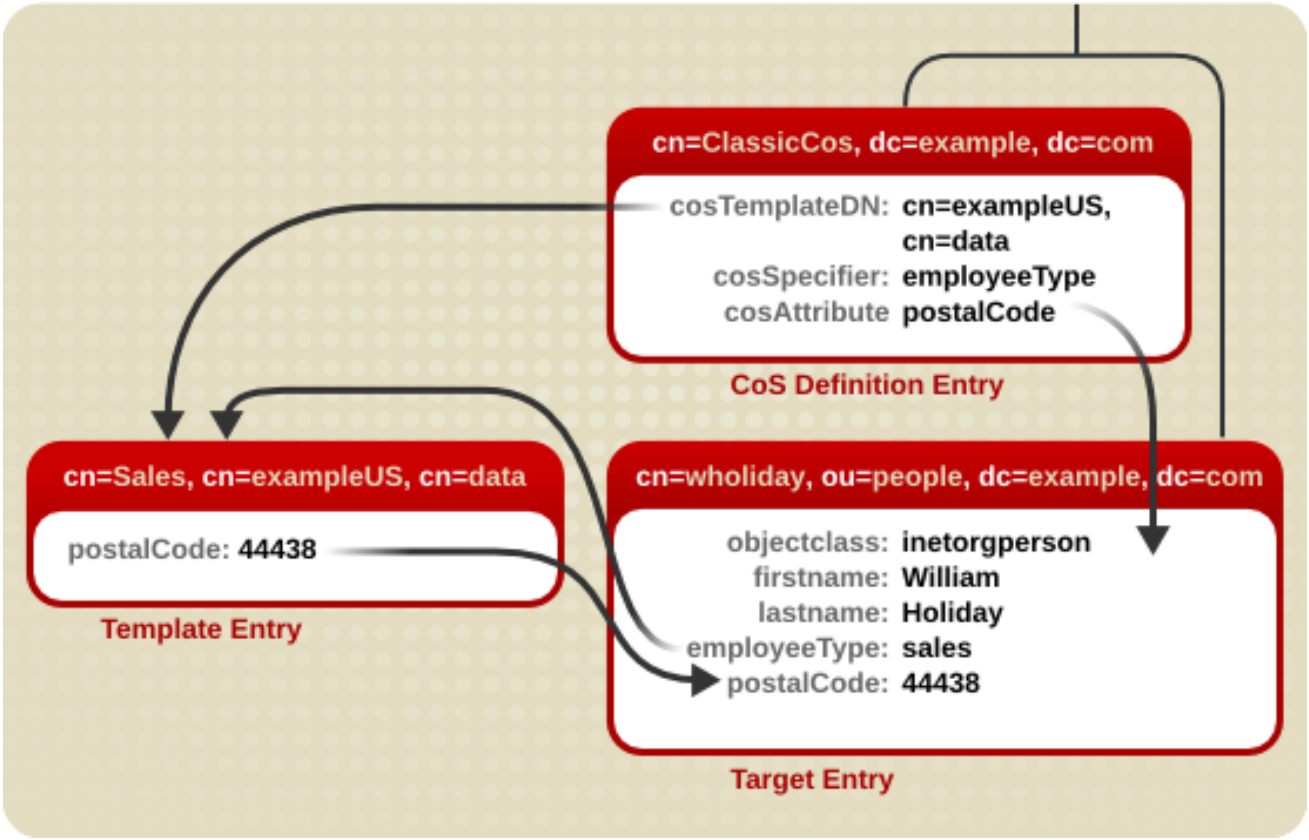


この例では、Wiiam Holiday のターゲットエントリーには間接指定子 (***manager*** 属性) が含まれます。William のマネージャーは Carla Fuentes であるため、***manager*** 属性にはテンプレートエントリーの DN へのポインター **cn=Carla Fuentes,ou=people,dc=example,dc=com** が含まれます。テンプレートエントリーは次に、**318842** の ***departmentNumber*** 属性値を提供します。

7.2.5. Classic CoS の仕組み

管理者は、テンプレート DN と CoS 指定子の組み合わせを使用して、有番号を含むテンプレートエントリーを特定します。図7.4「Classic CoS のサンプル」 に示されるように、3 つの CoS エントリーが表示されます。

図7.4 Classic CoS のサンプル



この例では、CoS 定義エントリーの ***cosSpecifier*** 属性は、***employeeType*** 属性を指定しています。この属性は、テンプレート DN と組み合わせて、テンプレートエントリーを **cn=sales,cn=exampleUS,cn=data** として特定します。その後、テンプレートエントリーは、***postalCode*** 属性値をターゲットエントリーに提供します。

7.2.6. 物理属性値の処理

cosAttribute 属性には、サービスのクラスによって管理される別の属性の名前が含まれます。この属性は、属性値の後に **override** 修飾子を許可します。属性値の生成時に、CoS がエントリーの既存の属性値を処理する方法を設定します。

```
cosAttribute: attribute_name override
```

override 修飾子は 4 つあります。

- **default**: エントリーに対応する属性値が格納されてない場合のみ、生成された値を返します。
- **override**: エントリーに値が保存されている場合でも、常に CoS によって生成された値を返します。
- **operational**: 生成された属性が検索で明示的に要求されている場合にのみ、生成された属性を返します。操作属性は、返されるためにスキーマチェックに合格する必要はありません。**operational** を使用すると、既存の属性値も上書きされます。

注記

属性は、スキーマで操作可能として定義されている場合にのみ操作可能にすることができます。たとえば、CoS が ***description*** 属性の値を生成する場合、この属性はスキーマで稼働していないため、**operational** 修飾子を使用することはできません。

- **operational-default**: エントリーに対応する属性値が格納されておらず、検索で明示的に要求された場合にのみ、生成された値を返します。

修飾子が設定されていない場合は、**default** と仮定されます。

たとえば、このポインター CoS 定義エントリーは、***postalCode*** 属性の値を生成するテンプレートエントリー **cn=exampleUS,ou=data,dc=example,dc=com** に関連付けられていることを示しています。**override** 修飾子は、この値が ***postalCode*** 属性のエントリーによって保存される値よりも優先されることを示しています。

```
dn: cn=pointerCoS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=exampleUS,ou=data,dc=example,dc=com
cosAttribute: postalCode override
```

注記

エントリーに CoS によって生成された属性値が含まれる場合、操作修飾子または上書き修飾子で定義された場合には、属性の値を手動で更新することは **できません**。

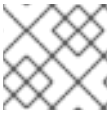
CoS 属性の詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 を参照してください。

7.2.7. CoS を使用した多値属性の処理

属性は、サービスのクラスを使用して生成できます。これには複数値の属性が含まれます。これにより、混乱を生じさせる可能性があります。どの CoS が値を提供しますか。これらのいずれか、またはすべてですか。競合 CoS テンプレートからどのように値が選択されていますか。生成された属性は単値または多値を使用しますか。

これを解決する方法は 2 つあります。

- 複数の CoS が生成する属性をターゲットエントリーにマージするルールを作成。これにより、ターゲットエントリーの値が複数表示されます。
- 競合する CoS 定義の中から 1 つの CoS 値を選択するように優先度を設定。これにより、ターゲットエントリーに 1 つの値が生成されます。

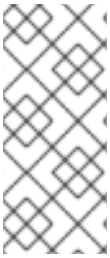


注記

Indirect CoS は ***cosPriority*** 属性をサポートしません。

CoS が CoS 属性の複数の値を処理する方法は、**merge-schemes** 修飾子を使用するかどうかで定義されます。

```
cosAttribute: attribute override merge-schemes
```



注記

merge-schemes 修飾子は、CoS による物理属性値や **override** 修飾子の処理方法に影響を与えません。競合する CoS テンプレートまたは定義が複数ある場合は、競合するすべての CoS 定義のすべての ***cosAttribute*** に、同じ **merge-schemes** および **override** 修飾子を設定する必要があります。それ以外の場合は、1 つの組み合わせが可能なすべての CoS 定義から任意に選択されます。

merge-schemes 修飾子を使用すると、CoS に、マネージド属性に対して複数の値を生成する、または生成できることを通知します。多値 CoS 属性を持つシナリオは 2 つあります。

- 1 つの CoS テンプレートエントリーに管理 CoS 属性のインスタンスが複数含まれるため、ターゲットエントリーに多値が作成されます。以下に例を示します。

```
dn: cn=server access template,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
accessTo: mail.example.com
accessTo: irc.example.com
```

注記

このメソッドは、Classic CoS でのみ動作します。

- 複数の CoS 定義が同じターゲット属性にサービスクラスを定義する可能性があるため、複数のテンプレートエントリーがあります。以下に例を示します。

```
dn: cn=mail template,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
accessTo: mail.example.com

dn: cn=chat template,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
accessTo: irc.example.com
```

ただし、CoS 定義が複数ある場合でも、属性には 1 つの値のみが生成されることがあります。CoS 定義が複数ある場合、値は任意に選択されます。これは予測不可で、意図しないオプションです。どの CoS テンプレートを使用するかを制御する方法は、テンプレートに順位 (**優先順位**) を設定することであり、優先順位の高い CoS が常に勝ち、値を提供します。

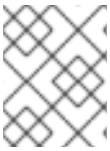
値を提供するために複数のテンプレートが完成することはかなり一般的です。たとえば、CoS 定義エントリーには複数値の ***cosSpecifier*** 属性を使用できます。テンプレートの優先度は、***cosPriority*** 属性を使用して設定します。この属性は、特定のテンプレートのグローバル優先度を表します。0 の優先度が最も優先されます。

たとえば、部門番号を生成する CoS テンプレートエントリーは、以下のようになります。

```
dn: cn=data,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
departmentNumber: 71776
cosPriority: 0
```

このテンプレートエントリーには、***departmentNumber*** 属性の値が含まれます。優先度はゼロであるため、このテンプレートは、別の ***departmentNumber*** 値を定義する他の競合するテンプレートよりも優先されます。

cosPriority 属性が含まれないテンプレートは、優先度が最も低いとみなされます。2 つ以上のテンプレートが属性値を提供し、優先度が同じ (または優先順位がない) 場合は、値が任意に選択されます。



注記

負の ***cosPriority*** 値の動作は Directory Server では定義されないため、負の値を入力しないでください。

7.2.8. CoS 指定の属性の検索

CoS 定義はエントリーで属性の値を指定します。たとえば、CoS はサブツリー内のすべてのエントリーに ***postalCode*** 属性を設定できます。ただし、これらの CoS 定義属性に対する検索は、通常のエントリーに対する検索のように動作しません。

CoS が定義する属性が、あらゆる種類のインデックス (存在を含む) でインデックス化されている場合、CoS によって設定される値を持つ属性は検索で返されません。以下に例を示します。

- Ted Morris の ***postalCode*** 属性は、CoS によって定義されます。
- Barbara Jensen の ***postalCode*** 属性は、Barbara Jensen 自身のエントリーに設定されます。
- ***postalCode*** 属性はインデックス化されます。

ldapsearch コマンドでフィルター **(postalCode=*)** が使用される場合、Barbara Jensen のエントリーが返されますが、Tedris のエントリーは返されません。

CoS が定義されている属性がインデックス化されて **いない** 場合には、属性の値がローカルで設定されるか、CoS と設定されている場合にかかわらず、一致するすべてのエントリーが検索で返されます。以下に例を示します。

- Ted Morris の ***postalCode*** 属性は、CoS によって定義されます。
- Barbara Jensen の ***postalCode*** 属性は、Barbara Jensen 自身のエントリーに設定されます。
- ***postalCode*** 属性はインデックス化 **されません**。

ldapsearch コマンドでフィルター **(postalCode=*)** を使用する場合は、Barbara Jensen のエントリーと Ted Morris のエントリーの両方が返されます。

CoS は **override** を許可します。これは、CoS エントリーの ***cosAttribute*** 属性に指定された ID です。つまり、属性のローカル値は CoS 値を上書きできます。CoS に上書きが設定されていると、エントリーのローカル値があるかぎり、**ldapsearch** 操作は属性がインデックス化された場合でもエントリーの値を返します。CoS を持ち、ローカル値を持たない他のエントリーは、**ldapsearch** 操作では返されません。

CoS で定義した属性で LDAP 検索要求を実行する際に問題が発生する可能性があるため、CoS を使用して生成する属性を決定する際には注意してください。

7.2.9. アクセス制御と CoS

サーバーは、通常の保存属性と同じように、CoS が生成した属性へのアクセスを制御します。ただし、CoS によって生成された属性の値によってはアクセス制御ルールは機能しません。これは、検索フィルターで CoS が生成する属性の使用に適用される制限と同じです。

7.2.10. コマンドラインでの CoS の管理

すべての設定情報とテンプレートデータはディレクトリーにエントリーとして格納されるため、標準の LDAP ツールを使用して CoS 設定および管理に使用できます。

- 「コマンドラインでの CoS 定義エントリーの作成」
- 「コマンドラインでの CoS テンプレートエントリーの作成」
- 「Pointer CoS の例」
- 「間接的な CoS の例」
- 「Classic CoS の例」
- 「CoS エントリーの検索」

7.2.10.1. コマンドラインでの CoS 定義エントリーの作成

各タイプの CoS では、定義エントリーに特定のオブジェクトクラスを指定する必要があります。すべての CoS 定義オブジェクトクラスは、**LDAPsubentry** オブジェクトクラスと **cosSuperDefinition** オブジェクトクラスから継承されます。

ポインター CoS は **cosPointerDefinition** オブジェクトクラスを使用します。例7.2「Pointer CoS エントリーの例」 に示されるように、このオブジェクトクラスは、***cosTemplateDn*** 属性で指定されたエントリー DN 値を使用してテンプレートエントリーを識別します。

例7.2 Pointer CoS エントリーの例

```
dn: cn=pointerCoS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn:DN_string
cosAttribute:list_of_attributes qualifier
cn: pointerCoS
```

間接 CoS は **cosIndirectDefinition** オブジェクトクラスを使用します。このタイプの CoS は、***cosIndirectSpecifier*** 属性で指定されているターゲットエントリーの属性のいずれかの値に基づいてテンプレートエントリーを識別します。これは、例7.3「間接的な CoS エントリーの例」 で説明されています。

例7.3 間接的な CoS エントリーの例

```
dn: cn=indirectCoS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosIndirectDefinition
cosIndirectSpecifier:attribute_name
cosAttribute:list_of_attributes qualifier
cn: indirectCoS
```

クラス CoS は **cosClassicDefinition** オブジェクトクラスを使用します。これは、テンプレートエントリーの DN (***cosTemplateDn*** 属性に設定) と、ターゲットエントリーの属性のいずれかの値 (***cosSpecifier*** 属性に設定) の両方を使用してテンプレートエントリーを特定します。これは、例 7.4「Classic CoS エントリーの例」 で説明されています。

例7.4 Classic CoS エントリーの例

```
dn: cn=classicCoS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosClassicDefinition
cosTemplateDn:DN_string
cosSpecifier:attribute_name
cosAttribute:list_of_attributes qualifier
cn: classicCoS
```

サービスのクラスの場合、オブジェクトクラスは CoS のタイプを定義し、サポート属性は CoS テンプレートを定義することで影響を受けるディレクトリーエントリーを識別します。各 CoS には、定義できる属性が 1 つあります (***cosAttribute***)。CoS の目的は、複数のエントリーに属性値を提供することです。***cosAttribute*** 属性は、CoS が生成する属性を定義します。

7.2.10.2. コマンドラインでの CoS テンプレートエントリーの作成

各テンプレートエントリーは、**cosTemplate** オブジェクトクラスのインスタンスです。

注記

新しいテンプレートエントリーに **LDAPsubentry** オブジェクトクラスを追加することを検討してください。CoS テンプレートエントリーを **LDAPsubentry** オブジェクトクラスのインスタンスに設定すると、通常の検索を設定エントリーに妨げられずに実行することができます。ただし、テンプレートエントリーがすでに存在し、ユーザーエントリーなどの他のものに使用される場合は、**LDAPsubentry** オブジェクトクラスをテンプレートエントリーに追加する必要はありません。

CoS テンプレートエントリーには、CoS (CoS 定義エントリーの ***cosAttribute*** 属性で指定) によって生成された属性とその属性の値も含まれます。

たとえば、***postalCode*** 属性の値を提供する CoS テンプレートエントリーは、以下のようになります。

```
dn:cn=exampleUS,ou=data,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
```

```
objectclass: cosTemplate
postalCode: 44438
```

以下のセクションでは、テンプレートエントリーの例と、各タイプの CoS 定義エントリーの例を示します。

- 「Pointer CoS の例」
- 「間接的な CoS の例」
- 「Classic CoS の例」

7.2.10.3. Pointer CoS の例

Corporation の管理者の例では、**dc=example,dc=com** ツリー内のすべてのエントリーと共通の郵便番号コードを共有するポインター CoS を作成します。

1. **ldapmodify** を使用して、新しい pointer CoS 定義エントリーを **dc=example,dc=com** 接尾辞に追加します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=pointerCoS,dc=example,dc=com
changetype: add
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=exampleUS,ou=data,dc=example,dc=com
cosAttribute: postalCode
```

2. テンプレートエントリーを作成します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=exampleUS,ou=data,dc=example,dc=com
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
postalCode: 44438
```

CoS テンプレートエントリー (**cn=exampleUS,ou=data,dc=example,dc=com**) は、 ***postalCode*** 属性に保存された値を **dc=example,dc=com** 接尾辞に置かれているエントリーに提供します。これらのエントリーはターゲットエントリーです。

7.2.10.4. 間接的な CoS の例

この間接 CoS は、ターゲットエントリーの ***manager*** 属性を使用して CoS テンプレートエントリーを識別します。これは、属性の値によって異なります。

1. **ldapmodify** を使用して、**dc=example,dc=com** 接尾辞に新しい間接 CoS 定義エントリーを追加します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=indirectCoS,dc=example,dc=com
changetype: add
```

```
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosIndirectDefinition
cosIndirectSpecifier: manager
cosAttribute: departmentNumber
```

ディレクトリーまたはマネージャーエントリーに ***departmentNumber*** 属性がすでに含まれる場合は、他の属性をマネージャーエントリーに追加する必要はありません。この属性は定義エントリーの ***cosIndirectSpecifier*** 属性に指定されるため、定義エントリーは ***manager*** 属性が含まれるエントリーのターゲット接尾辞 (**dc=example,dc=com** 下のエントリー) を検索します。次に、一覧表示された manager エントリーの ***departmentNumber*** 値を確認します。 ***departmentNumber*** 属性の値は、***manager*** 属性を持つマネージャーの下位すべてに自動的にリレーされます。***departmentNumber*** の値は、異なるマネージャーのエントリーに記載されている部門番号によって異なります。

7.2.10.5. Classic CoS の例

Corporation の例の管理者は、テンプレート DN と ***cosSpecifier*** 属性で指定された属性の組み合わせを使用して、郵便番号コードを自動的に生成する Classic CoS を作成します。

1. **ldapmodify** を使用して、新しいクラス CoS 定義エントリーを **dc=example,dc=com** 接尾辞に追加します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=classicCoS,dc=example,dc=com
changetype: add
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosClassicDefinition
cosTemplateDn: cn=classicCoS,dc=example,dc=com
cosSpecifier: businessCategory
cosAttribute: postalCode override
```

2. 営業部門およびマーケティング部門のテンプレートエントリーを作成します。CoS 属性をテンプレートエントリーに追加します。テンプレートの ***cn*** は、ターゲットエントリーの ***businessCategory*** 属性の値を設定し、テンプレートの値に応じて属性を追加または上書きされます。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=sales,cn=classicCoS,dc=example,dc=com
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
postalCode: 44438
-
dn: cn=marketing,cn=classicCoS,dc=example,dc=com
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
postalCode: 99111
```

Classic CoS 定義エントリーは、**dc=example,dc=com** 接尾辞の下にあるすべてのエントリーに適用されます。エントリー内の ***businessCategory*** 属性と ***cosTemplateDn*** の組み合わせによって、2 つのテンプレートのいずれかに到達できます。1 つ目は営業テンプレートで、営業部門に従業員固有の郵便番

号コードを提供します。マーケティングテンプレートは、マーケティング部門の従業員固有の郵便番号コードを提供します。

7.2.10.6. CoS エントリーの検索

CoS 定義エントリーは **操作上** のエントリーで、デフォルトでは、通常の検索では返されません。検索の CoS 定義エントリーを返すには、**ldapSubEntry** オブジェクトクラスを CoS 定義エントリーに追加します。以下に例を示します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=pointerCoS,ou=People,dc=example,dc=com
changetype: add
objectclass: ldapSubEntry
```

次に、**ldapsearch** ユーティリティーで **(objectclass=ldapSubEntry)** フィルターを使用して、**ldapSubEntry** オブジェクトクラスを含むエントリーを検索します。以下に例を示します。

```
# ldapsearch -x -s sub -b ou=People,dc=example,dc=com "(|
(objectclass=*)(objectclass=ldapSubEntry))"
```

この検索は、**ou=People,dc=example,dc=com** サブツリーの CoS 定義エントリーに加えて、すべての通常のエントリーを返します。

7.2.10.7. costargettree 属性

costargettree 属性は、CoS スキーマが適用されるサブツリーを定義します。スキーマと複数の CoS スキーマの ***costargettree*** の値は、任意にターゲットツリーと重複する可能性があります。

表7.1 costargettree 属性

OID	2.16.840.1.113730.3.1.552
構文	DirectoryString
多値または単一値	単一値
定義される場所	Directory Server

7.2.11. ロールベースの属性の作成

Classic CoS スキームは、エントリーが所有するロールに基づいてエントリーの属性値を生成します。たとえば、ロールベースの属性を使用して、エントリーごとにサーバーのルックスルー制限を設定できます。

ロールベースの属性を作成するには、Classic CoS の CoS 定義エントリーで ***nsRole*** 属性を ***cosSpecifier*** として使用します。***nsRole*** 属性が多値指定できるため、複数のテンプレートエントリーを持つ CoS スキームを定義できます。使用するテンプレートエントリーの曖昧さを解決するには、CoS テンプレートエントリーに ***cosPriority*** 属性を追加します。

たとえば、この CoS は manager ロールのメンバーが標準のメールボックスクォータを超過できるようにします。マネージャーのロールエントリーは次のとおりです。

```
dn: cn=ManagerRole,ou=people,dc=example,dc=com
objectclass: top
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsFilteredRoleDefinition
cn: ManagerRole
nsRoleFilter: ou=managers
Description: filtered role for managers
```

警告

nsRoleFilter 属性は仮想属性の値を受け付けません。

仮想属性値にインデックスを付けないでください。仮想属性で検索を実行すると、予期しないシステム動作や不正確な検索結果が発生する可能性があります。インデックス付けされていない検索は、検索フィルターで仮想属性を使用する検索アクションを中断します。仮想属性は動的に生成され、Directory Server バックエンドには保存されません。したがって、仮想属性はインデックス付けをサポートしていません。

Classic CoS 定義エントリーは以下のようになります。

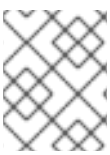
```
dn: cn=managerCOS,dc=example,dc=com
objectclass: top
objectclass: cosSuperDefinition
objectclass: cosClassicDefinition
cosTemplateDn: cn=managerCOS,dc=example,dc=com
cosSpecifier: nsRole
cosAttribute: mailboxquota override
```

cosTemplateDn 属性は、***cosSpecifier*** 属性で指定された属性 (ターゲットエントリーの ***nsRole*** 属性) とともに CoS テンプレートエントリーを識別する値を提供します。CoS テンプレートエントリーは、***mailboxquota*** 属性の値を提供します。**override** の他の修飾子は、ターゲットエントリーの既存の ***mailboxquota*** 属性値をオーバーライドするように、CoS に指示します。

対応する CoS テンプレートエントリーは以下のようになります。

```
dn:cn="cn=ManagerRole,ou=people,dc=example,dc=com",cn=managerCOS,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: cosTemplate
mailboxquota: 1000000
```

テンプレートは、***mailboxquota*** 属性の値 **1000000** を指定します。



注記

ロールエントリーと CoS 定義およびテンプレートエントリーは、ディレクトリーツリーの同じレベルにある必要があります。

7.3. 属性値の管理属性のリンク

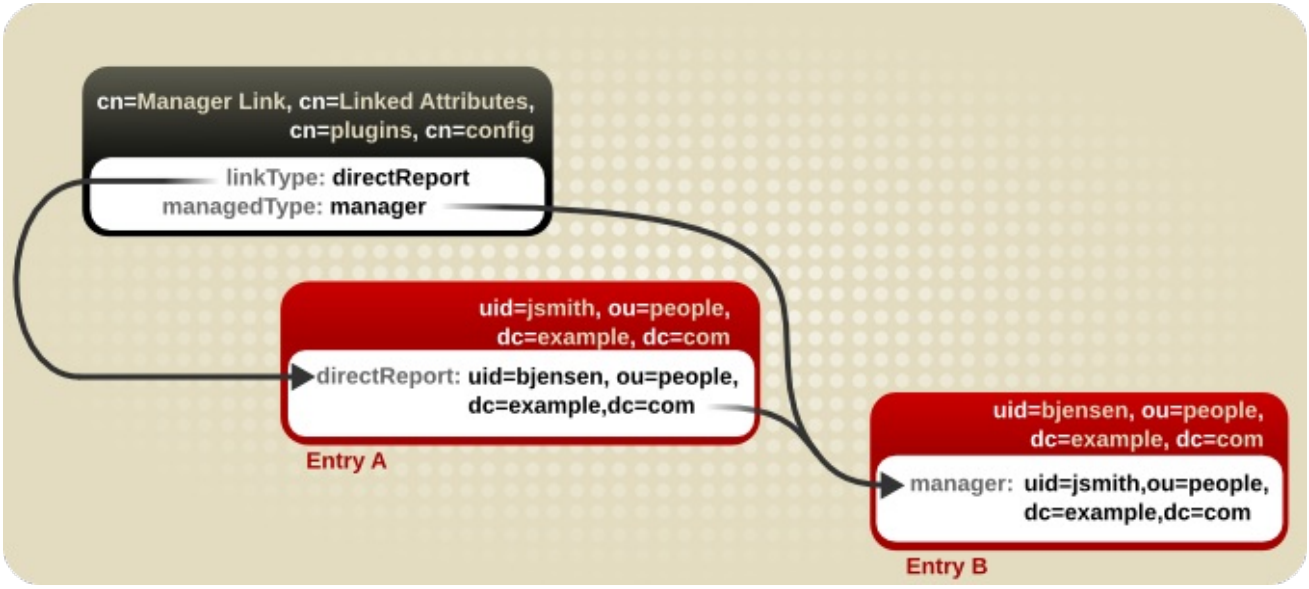
サービスのクラスは、住所、郵便番号コード、主なオフィス番号など、すべて **同じ値** の属性を持つエントリーの属性値を動的に提供します。これらは共有属性値であり、単一のテンプレートエントリーで更新されます。

多くの場合、エントリー間には、それらの間のリンクを表現する方法が必要な関係がありますが、その関係を表現する値 (および場合によっては属性) は異なります。Red Hat Directory Server は、指定された属性を繋ぎ合わせる方法を提供するため、1 つのエントリーの属性が変更すると、関連するエントリーの対応する属性が自動的に更新されます。(リンクおよび管理属性の両方に DN の値があります。リンク属性の値には、更新するプラグインのエントリーの DN が含まれます。2 つ目のエントリーの管理属性には、元のリンクエントリーを参照する DN の値があります。)

7.3.1. リンク元属性の概要

リンク先属性プラグインは、プラグインの複数のインスタンスを許可します。各インスタンスは、管理者によって手動で維持される 1 つの属性 (***linkType***) およびプラグインによって自動的に維持される 1 つの属性 (***managedType***) を設定します。

図7.5 リンク先属性の基本設定

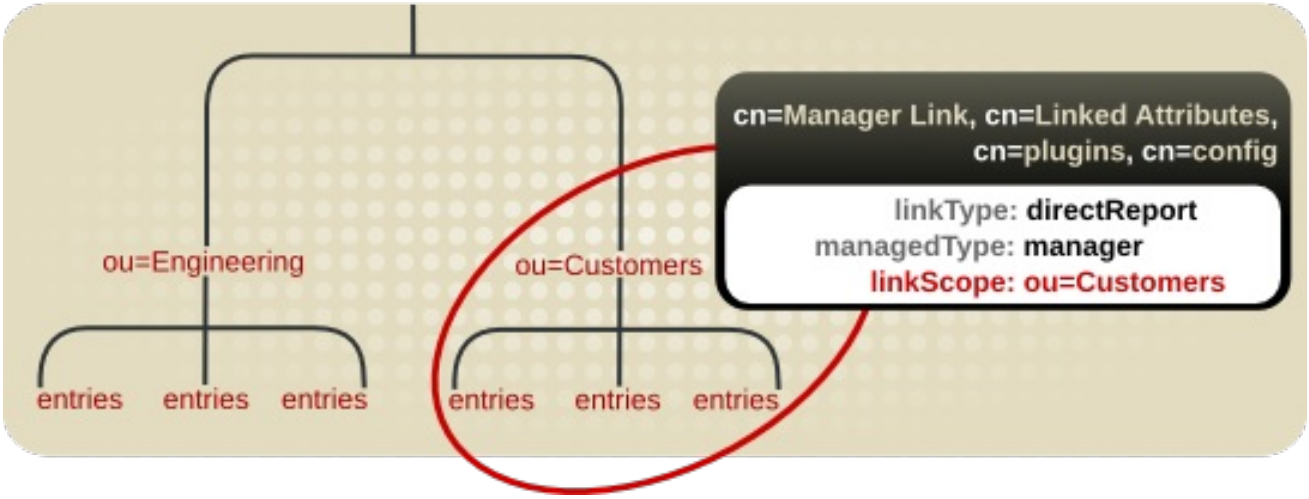


注記

データの一貫性を維持するには、プラグインプロセスのみが管理属性を維持する必要があります。管理属性へのすべての書き込みアクセスを制限する ACI の作成を検討してください。ACI の設定に関する詳細は、「ACI の追加」を参照してください。

リンク先属性プラグインインスタンスは、ディレクトリー内の単一のサブツリーに制限できます。これにより、属性の組み合わせと影響を受けるエントリーのより柔軟なカスタマイズが可能になります。スコープが設定されていない場合、プラグインはディレクトリー全体で動作します。

図7.6 リンク先属性プラグインを特定のサブツリーに制限



リンク先属性プラグインインスタンスを設定する場合は、特定の設定が必要です。

- 管理属性とリンク先属性の両方で、属性定義で識別名の構文が必要です。リンク先属性は基本的にクロス参照で管理されます。プラグインがこれらの相互参照を処理する方法は、属性値からエントリーの DN をプルすることで行われます。

  カスタムスキーマ要素のプランニングに関する情報は、*12章 ディレクトリースキーマの管理*を参照してください。

- 各リンク先属性プラグインインスタンスはローカルで、**管理** 属性は一部レプリケーションを使用してレプリケーションからブロックする必要があります。

  あるサプライヤーに加えられた変更は、プラグインが自動的に発生し、対応するディレクトリーエントリーの値を管理するため、データはサーバー全体で一貫性を維持します。ただし、リンクされたエントリー間でデータの一貫性を保つには、プラグインインスタンスで管理属性を維持する必要があります。つまり、管理属性値は、マルチサプライヤーレプリケーション環境であっても、レプリケーションプロセスではなく、プラグインプロセスによってのみ維持される必要があります。

  一部レプリケーションの使用方法は、「一部レプリケーションを使用した属性のサブセットの複製」を参照してください。

7.3.2. リンク元属性プラグイン構文の確認

デフォルトのリンク先属性プラグインエントリーは、各プラグインインスタンスのコンテナーエントリーです。これは、次のセクションの password 構文プラグインや DNA プラグインに類似します。このコンテナーエントリーの下にある各エントリーは、異なるリンク管理属性ペアを定義します。

新しいリンク元属性ペアを作成するには、コンテナーエントリーの下に新しいプラグインインスタンスを作成します。基本的なリンク元属性プラグインインスタンスでは、以下の 2 つの項目を定義する必要があります。

- ***linkType*** 属性において、管理者が手動で管理する属性
- ***managedType*** 属性に含まれる、プラグインによって動的に作成される属性
- 必要に応じて、プラグインを ***linkScope*** 属性のディレクトリーツリーの特定の部分に制限するスコープ

例7.5 リンク先属性のプラグインインスタンスエントリーの例

```
dn: cn=Manager Link,cn=Linked Attributes,cn=plugins,cn=config
objectClass: top
objectClass: extensibleObject
cn: Manager Link
linkType: directReport
managedType: manager
linkScope: ou=people,dc=example,dc=com
```

Linked Attributes プラグインのインスタンスで利用可能な属性の一覧は、Red Hat Directory Server Configuration, Command, and File Referenceの該当するセクションを参照してください。

7.3.3. 属性リンクの設定

1. これが有効になっていない場合は、**Linked Attributes** プラグインを有効にします。詳細については、「プラグインの有効化および無効化」 を参照してください。
2. プラグインインスタンスを作成します。***--managed-type*** パラメーターおよび ***--link-type*** パラメーターの両方が必要です。以下の例は、**dsconf** を使用して作成したプラグインインスタンスを示しています。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin linked-attr config
"Manager Link" add --link-type=directReport --managed-type=manager
```

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

7.3.4. 属性リンクのクリーンアップ

管理属性とリンク先属性は同期しなくなる可能性があります。たとえば、リンク先属性をサーバーにインポートまたは複製できましたが、リンク属性が適切に設定されていなかったため、対応するマネージド属性はインポートされませんでした。管理属性とリンク先属性ペアは、**dsconf plugin linked-attr fixup** コマンドを実行するか、修正タスクを起動することで修正できます。

修正タスクは、参照エントリーに対応するリンク属性 (管理者が管理する属性) のない管理属性 (プラグインによって管理される属性) を削除します。逆に、エントリーにリンク属性が存在する場合は、タスクで不明な管理属性が追加されます。

7.3.4.1. リンク先属性の再生成

dsconf plugin linked-attr fixup コマンドは特殊なタスクを起動し、ディレクトリーエントリー上の管理属性とリンク属性のペアをすべて再生成します。特定の状況では、1 つまたはもう 1 つが失われる可能性があります。リンク属性がエントリーに存在する場合、タスクは利用可能な属性でクロス参照されている DN を追跡し、参照されたエントリーで対応する管理属性を作成します。対応するリンク属性のない管理属性が存在する場合は、管理属性値が削除されます。

プラグインのスコープ全体に設定されたリンク属性ペアをすべて修復するには、Directory Manager で以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin linked-attr fixup
```

また、ベース DN をコマンドに渡して、修正タスクを単一のリンク管理属性ペアに制限することもできます。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin linked-attr fixup "cn=Manager Link,cn=Linked Attributes,cn=plugins,cn=config"
```

7.3.4.2. ldapmodify を使用したリンク先属性の再生成

リンクされた属性の修復は、特別なタスク設定エントリーで管理できるタスクの1つです。タスクエントリーは、**dse.ldif** ファイルの **cn=tasks** 設定エントリーで発生するため、**ldapmodify** を使用してエントリーを追加してタスクを開始することもできます。タスクが完了すると、エントリーはディレクトリーから削除されます。

このタスクは、実行時に **dsconf plugin linked-attr fixup** コマンドによって自動的に作成されるタスクと同じです。

リンク付きの属性修正タスクを開始するには、**cn=fixup linked attributes,cn=tasks,cn=config** エントリーの下にエントリーを追加します。必要な属性は特定タスクの ***cn*** のみですが、***ttl*** 属性にはタイムアウト期間を設定できます。**ldapmodify** の使用:

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=example,cn=fixup linked attributes,cn=tasks,cn=config
changetype: add
cn:example
ttl: 5
```

タスクが完了すると、エントリーは **dse.ldif** 設定から削除されるため、同じタスクエントリーを継続的に再利用できます。

cn=fixup linked attributes タスク設定は、『Configuration, Command, and File Reference』を参照してください。

7.4. 一意の数値属性値の割り当ておよび管理

エントリー属性によっては、***uidNumber*** や ***gidNumber*** などの一意の番号が必要です。Directory Server は、DNA (Distributed Numeric Assignment) プラグインを使用して、指定された属性に一意の番号を自動的に生成して提供できます。

注記

属性の一意性 は、DNA プラグインで維持されるとは限りません。プラグインは、オーバーランディングの範囲のみを割り当てますが、管理属性に手動で割り当てる数字を許可し、手動で割り当てられた数字が一意であることを検証せず、必要とされません。

一意の数字を割り当てる問題は、数字の生成ではなく、レプリケーションの競合を回避することで問題となります。DNA プラグインは、**単一** のバックエンド全体に一意の番号を割り当てます。マルチプライヤーレプリケーションの場合、各プライヤーがローカル DNA プラグインインスタンスを実行しているときに、各インスタンスが真に一意の番号セットを使用していることを確認する方法が必要です。これは、割り当てる各サーバーに異なる **範囲** を割り当てることで行われます。

7.4.1. 動的番号の割り当ての概要

サーバーの DNA プラグインは、そのインスタンスが発行することのできる利用可能な数字の範囲を割

り当てます。範囲の定義は非常にシンプルで、サーバーの次に利用可能な番号 (範囲の下限) と最大値 (範囲の最後) の 2 つの属性で設定されます。初期の下限範囲は、プラグインインスタンスを設定する際に設定されます。その後、下部の値はプラグインによって更新されます。利用可能な数を各レプリカの複数の範囲に分割することで、サーバーはすべて、互いに重複することなく、継続的に番号を割り当てることができます。

7.4.1.1. フィルター、検索、およびターゲットエントリー

(「標準インデックスの作成」 で説明されているように) サーバーは、内部的にソートされた検索を実行し、次に指定された範囲がすでに取得されているかどうかを確認し、管理属性に適切な順序のマッチングルールと同じインデックスを割り当てる必要があります。

DNA プラグインは、常にディレクトリーツリーの特定領域 (**スコープ**) と、そのサブツリー内の特定のエントリータイプ (**フィルター**) に適用されます。

重要

DNA プラグインは 1 つのバックエンドでのみ機能します。複数のデータベースの番号割り当ては管理できません。DNA プラグインは、DNA プラグイン以外で値がすでに割り当てられているかどうかを確認する際に、ソートコントロールを使用します。この検証は、ソートコントロールを使用して単一のバックエンドでのみ機能します。

7.4.1.2. 範囲および割り当て番号

Directory Server が属性値の生成を処理する方法は複数あります。

- 最も単純なケースでは、属性がない場合に、unique-number 属性を必要とするオブジェクトクラスを持つディレクトリーにユーザーエントリーが追加されます。管理属性に値を持たないエントリーを追加すると、DNA プラグインによる値の割り当てが発生します。このオプションは、一意の値を 1 つの属性に割り当てるように DNA プラグインが設定されている場合に限り機能します。

- 同様の管理可能なオプションは、**マジック番号** を使用することです。このマジックナンバーは、マネージド属性のテンプレート値であり、サーバーの範囲外のもの、数字、または単語でさえあり、プラグインは新しい割り当て値に置き換える必要があると認識します。マジック値でエントリーが追加され、エントリーが設定された DNA プラグインの範囲およびフィルター内にある場合は、プラグインでマジック番号を使用した新しい値の生成が自動的に発生します。下の例では、**ldapmodify** の使用に基づいて、0 をマジックナンバーとして追加します。

```
dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: posixAccount
uid: jsmith
cn: John Smith
uidNumber: 0
gidNumber: 0
....
```

DNA プラグインは、新規の一意の値のみを生成します。DNA プラグインが制御する属性に特定の値を使用するためにエントリーを追加または変更した場合には、指定した番号が使用されます。DNA プラグインは、その番号を上書きしません。

7.4.1.3. 同じ範囲の複数の属性

DNA プラグインは、1つの属性タイプに、または1つの範囲の一意の番号から複数の属性タイプに一意の番号を割り当てることができます。

これにより、属性に一意の数字を割り当てるためのオプションが複数提供されます。

- 一意の番号の1つの範囲から、1つの属性タイプに割り当てられた1つの番号。
- 1つのエントリーの 2 つの属性に割り当てられた同じ一意の番号。
- 2 つの異なる属性は、同じ範囲の一意の数字から 2 つの異なる数字を割り当てていました。

多くの場合は、属性タイプごとに一意の番号を割り当てるだけで十分です。新しい従業員エントリーに ***employeeID*** を割り当てる際には、各従業員エントリーに一意の ***employeeID*** が割り当てられます。

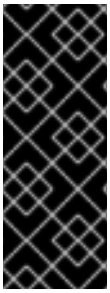
ただし、同じ範囲の数字から複数の属性に一意の番号を割り当てることが役に立つ場合もあります。たとえば、***uidNumber*** と ***gidNumber*** を **posixAccount** エントリーに割り当てると、DNA プラグインは同じ数を両方の属性に割り当てます。これを行うには、マジック値を指定して、両方の管理属性を変更操作に渡します。**ldapmodify** の使用:

```
#  ldapmodify -D "cn=Directory Manager" -W -x

dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
add: uidNumber
uidNumber: 0
-
add:gidNumber
gidNumber: 0
```

DNA プラグインで複数の属性を処理する場合は、オブジェクトクラスが1つしか許可されない場合、プラグインはこれらの属性の1つにのみ一意の値を割り当てることができます。たとえば、**posixGroup** オブジェクトクラスは ***uidNumber*** 属性を許可しませんが、***gidNumber*** を許可します。DNA プラグインが ***uidNumber*** と ***gidNumber*** の両方を管理する場合は、**posixGroup** エントリーが作成されると、***gidNumber*** の一意の番号が ***uidNumber*** および ***gidNumber*** 属性と同じ範囲から割り当てられます。プラグインが管理するすべての属性に同じプールを使用すると、一意の数字の割り当てを維持し、異なるエントリーの ***uidNumber*** と ***gidNumber*** が異なる範囲から割り当てられ、同じ **一意** の番号になる状況を防ぐことができます。

複数の属性が DNA プラグインで処理される場合は、1つの変更操作のエントリーで指定の管理属性のすべてに同じ値が割り当てられます。同じ範囲から **別** の数字を割り当てるには、別の変更操作を実施する必要があります。以下の例では、**ldapmodify** を使用してこれを行います。

```
# ldapmodify -D "cn=Directory Manager" -W -x
dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
add: uidNumber
uidNumber: 0
^D

# ldapmodify -D "cn=Directory Manager" -W -x
dn: uid=jsmith,ou=people,dc=example,dc=com
changetype: modify
add: employeeId
employeeId: magic
```



重要

DNA プラグインが一意の数字を複数の属性に割り当てるように設定する場合は、一意の番号を必要とする各属性にマジック値を指定する必要があります。1 つの属性に一意の番号を提供するように DNA プラグインが設定されている場合には、これは必須ではありませんが、複数の属性に必要です。エントリーが範囲に対して定義された各タイプの属性を許可しない場合や、さらに重要なことに、定義されたすべての属性タイプをエントリーが許可しますが、属性のサブセットのみが一意の値を必要とする場合があります。

例7.6 DNA および一意の銀行口座番号

銀行の例では、顧客の ***primaryAccount*** 属性および ***customerID*** 属性に同じ一意の番号を使用します。銀行の例の管理者は、DNA プラグインが同じ範囲から両方の属性に一意の値を割り当てるよう設定していました。

また、銀行では、顧客 ID とプライマリーの口座番号と同じ範囲のセカンダリー口座に番号を割り当てますが、これらの数字をプライマリーの口座番号と同じにすることはできません。銀行の例の管理者は、DNA プラグインも ***secondaryAccount*** 属性を管理するように設定しますが、エントリーの作成、***secondaryAccount*** 属性および ***primaryAccount*** 属性の割り当ての**後** にのみ ***customerID*** 属性を追加します。これにより、***primaryAccount*** および ***customerID*** は、同じ一意の番号を共有し、***secondaryAccount*** 番号は完全に一意ですが、それでも同じ範囲の番号からのものです。

7.4.2. DNA プラグイン構文の確認

DNA プラグイン自体は、パスワードストレージスキームプラグインと同様のコンテナーエントリーです。DNA プラグインエントリーの下にある各 DNA エントリーは、DNA プラグインの新しい管理範囲を定義します。

DNA プラグインの新しい管理範囲を設定するには、コンテナーエントリーの下にエントリーを作成します。

最も基本的な設定は、1 台のサーバーで分散数値の割り当てを設定することです。つまり、その範囲はサーバー間で共有されず、転送されません。基本的な DNA 設定エントリーでは、以下の 4 つの項目を定義します。

- 値が管理される属性で、***dnaType*** 属性に設定されます。
- ***dnaScope*** 属性に設定される、エントリーを検索するためのベースとして使用するエントリー DN
- ***dnaFilter*** 属性に設定される、管理するエントリーの特定に使用する検索フィルター
- ***dnaNextValue*** 属性に設定される、次に割り当てることができる値 (エントリーの作成後、プラグインにより処理される)

cn=*DNA_config_entry*,cn=Distributed Numeric Assignment Plugin,cn=plugins,cn=config エントリーでサポートされる属性の一覧は、『Red Hat Directory Server Configuration, Command, and File Reference』を参照してください。

1 つの属性タイプに対して、1 台のサーバーに分散数値割り当てを設定するには、以下を実行します。

```
dn: cn=Account UIDs,cn=Distributed Numeric Assignment Plugin,cn=plugins,cn=config
objectClass: top
objectClass: dnaPluginConfig
```

```
cn: Account UIDs
dnatype: uidNumber
dnafilter: (objectclass=posixAccount)
dnascope: ou=people,dc=example,dc=com
dnaNextValue: 1
```

複数のサプライヤーが分散数値の割り当てに設定されている場合は、エントリーに範囲を転送するために必要な情報が含まれます。

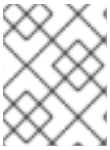
- サーバーが割り当てることができる最大数。これにより、範囲の上限が設定されます。これは、複数のサーバーが番号を割り当てるときに論理的に必要です。これは ***dnaMaxValue*** 属性で設定されます。
- ***dnaThreshold*** 属性に設定された範囲転送を発生させるのに範囲が十分低いしきい値です。これが設定されていない場合、デフォルト値は **1** になります。
- ***dnaRangeRequestTimeout*** 属性に設定される、サーバーが転送を待ってハングしないようにするためのタイムアウト期間。これを設定しないと、デフォルト値は **10** (10 秒) になります。
- ***dnaSharedCfgDN*** 属性に設定した各サプライヤーの範囲情報を保存するすべてのサプライヤーサーバー間で共有される設定エントリー DN。

サーバーで割り当て可能な特定の数字の範囲は、***dnaNextRange*** 属性で定義されます。これは、次に利用可能な範囲を表示し、サーバーが範囲が割り当てられているか、使用しているため、プラグインにより自動的に管理されます。この範囲はデッキ上にあります。 別のサーバーには割り当てられておらず、ローカルの Directory Server が使用する場合は引き続き利用できます。

```
dn: cn=Account UIDs,cn=Distributed Numeric Assignment Plugin,cn=plugins,cn=config
objectClass: top
objectClass: dnaPluginConfig
cn: Account UIDs
dnatype: uidNumber
dnafilter: (objectclass=posixAccount)
dnascope: ou=People,dc=example,dc=com
dnanextvalue: 1
dnaMaxValue: 1300
dnasharedcfgdn: cn=Account UIDs,ou=Ranges,dc=example,dc=com
dnathreshold: 100
dnaRangeRequestTimeout: 60
dnaNextRange: 1301-2301
```

dnaNextRange 属性は、個別の特定範囲を他のサーバーに割り当てる必要がある場合にのみ明示的に設定する必要があります。***dnaNextRange*** 属性に設定した範囲は、重複を避けるために、他のサーバーで利用可能な範囲から一意でなければなりません。他のサーバーからの要求がなく、***dnaNextRange*** が設定されているサーバーが設定 ***dnaMaxValue*** に到達した場合は、次に値 (***dnaNextRange*** の一部) がこのデッキから割り当てられます。

dnaNextRange 割り当ては、DNA 設定に設定されている ***dnaThreshold*** 属性によっても制限されます。***dnaNextRange*** 用に別のサーバーに割り当てられる範囲は、範囲が ***dnaNextRange*** のデッキで利用可能であっても、サーバーのしきい値に違反できません。



注記

dnaNextRange 属性が明示的に設定されていない場合は、内部的に処理されます。自動的に処理される場合、***dnaMaxValue*** 属性は次の範囲の上限として機能します。

各サプライヤーは、範囲およびその接続設定に関する情報を含む、現在の範囲を別の設定エントリーで追跡します。このエントリーは、***dnasharedcfgdn*** の場所の子です。設定エントリーは他のすべてのサプライヤーに複製されるため、各サプライヤーが設定を確認して、新しい範囲で問い合わせるサーバーを見つけることができます。以下に例を示します。

```
dn: dnaHostname=ldap1.example.com+dnaPortNum=389,cn=Account
UIDs,ou=Ranges,dc=example,dc=com
objectClass: dnaSharedConfig
objectClass: top
dnahostname: ldap1.example.com
dnaPortNum: 389
dnaSecurePortNum: 636
dnaRemainingValues: 1000
```

7.4.3. 一意の番号割り当ての設定

一意の数字分散は、DNA プラグインの異なるインスタンスを作成して設定されます。

7.4.3.1. DNA プラグインの新規インスタンスの作成

複数の設定で **DNA** を使用するには、設定ごとにプラグインの新しいインスタンスを作成します。

注記

コマンドラインでのみ、プラグインの新しいインスタンスを作成できます。ただし、コマンドラインと Web コンソールの両方を使用して設定を編集できます。

プラグインの新しいインスタンスを作成して有効にするには、以下を実行します。

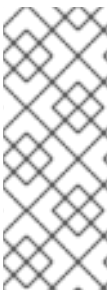
1. たとえば、プラグインの新規インスタンスを作成するには、以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin dna config "Account
UIDs" add --type uidNumber --filter "(objectclass=posixAccount)" --scope
ou=People,dc=example,dc=com --next-value 1 --max-value 1300 --shared-config-entry
"cn=Account UIDs,ou=Ranges,dc=example,dc=com" --threshold 100 --range-request-timeout
60 --magic-regen magic
```

   で設定できる値の詳細については、***--magic-regen*** パラメーターについては、『Configuration, Command and File Reference』 の ***dnaMagicRegen***属性の説明。

2. **DNA** プラグインを有効にするには、以下を実行します。詳細については、「プラグインの有効化および無効化」 を参照してください。

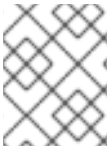
7.4.3.2. コマンドラインでの一意の番号割り当ての設定



注記

一意の番号が割り当てられている属性には、同等インデックスが設定されている必要があります。***dnaNextvalue*** がすでに取得されているかどうかを確認するために、サーバーは、内部的にソートされた検索を実行する必要があります。これには、適切な順序のマッチングルールとともに整数属性で等価インデックスが必要であるかを確認します。

インデックスの作成については、「標準インデックスの作成」を参照してください。



注記

すべてのサプライヤーサーバーに DNA プラグインを設定し、数字の範囲の値と重複しないように注意してください。

1. プラグインの新規インスタンスを作成します。「DNA プラグインの新規インスタンスの作成」を参照してください。
2. 複製されたサブツリーに共有コンテナーエントリーを作成します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=Ranges,dc=example,dc=com
changetype: add
objectclass: top
objectclass: extensibleObject
objectclass: organizationalUnit
ou: Ranges
-
dn: cn=Account UIDs,ou=Ranges,dc=example,dc=com
changetype: add
objectclass: top
objectclass: extensibleObject
cn: Account UIDs
```

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

7.4.3.3. Web コンソールを使用した一意の番号割り当ての設定

Web コンソールを使用して **DNA** プラグインを有効にして設定するには、以下を行います。

1. プラグインの新規インスタンスを作成します。「DNA プラグインの新規インスタンスの作成」を参照してください。
2. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
3. インスタンスを選択します。
4. **Plugins** メニューを開きます。
5. **DNA** プラグインを選択します。
6. ステータスを **ON** に変更し、プラグインを有効にします。
7. **Add Config** をクリックします。
8. フィールドを入力し、設定を有効にします。
9. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

7.4.4. Distributed Number Assignment プラグインのパフォーマンスに関する注意事項

DNA 設定が動的に変更されると、スレッドロックの問題が発生する可能性があります。そのため、新しい設定のスレッドは解放されないため、DNA 設定にアクセスする新しい操作 (DNA タスクや DNA 設定への追加の変更など) は古い設定にアクセスします。これにより、操作が古い設定を使用するか、操作がハングする可能性があります。

これを回避するには、動的 DNA 設定の変更の間隔を 35 秒に設定します。これは、DNA 設定の変更と、DNA プラグイン操作を発生させるディレクトリーエントリーの変更の両方の間にスリープ状態または遅延があることを意味します。

第8章 エントリーの編成とグループ化

ディレクトリーに含まれるエントリーは、ユーザーアカウントの管理を簡素化するために、さまざまな方法でグループ化できます。Red Hat Directory Server は、エントリーのグループ化やエントリー間で属性を共有するさまざまな方法に対応します。ロールおよびサービスのクラスによって提供される機能を完全に活用するには、ディレクトリーのデプロイメントを計画するときにディレクトリートポロジーを決定します。

8.1. グループの使用

オペレーティングシステムと同様に、Directory Server のグループにユーザーを追加できます。グループはロールとして他の方法で機能します。ロールを使用している場合には、割り当てられたロールの DN はユーザーオブジェクトの ***nsRoleDN*** 属性に保存されます。グループを使用する場合は、このグループのメンバーであるユーザーの DN は、グループオブジェクトの ***member*** 属性に保存されます。**memberOf** プラグインを有効にした場合、次にユーザーがメンバーであるグループは、ユーザーオブジェクトの ***memberOf*** 属性に追加で保存されます。このプラグインを有効にすると、グループにもロールの利点があり、ロールの使用時と同様にユーザーのグループメンバーシップをリスト表示できます。また、グループはロールよりも高速です。

memberOf プラグインの使用方法は、「ユーザーエントリーにおけるグループメンバーシップのリスト表示」を参照してください。

8.1.1. 各種グループ

コマンドラインから静的グループと動的グループの両方を作成するプロセスは、同様のプロセスです。グループエントリーには、グループ名、グループの種類、およびメンバー属性が含まれます。

グループタイプにはいくつかのオプションがあります。詳細は、『Red Hat Directory Server 10 Configuration, Command, and File Reference』を参照してください。この場合の **グループのタイプ** は、所有するメンバー属性を定義するタイプを指します。

- **groupOfNames** (推奨) は、任意のエントリーの追加を可能にする単純なグループです。これのメンバーを判断するために使用される属性は ***member*** です。
- **groupOfNames** などの **groupOfUniqueNames** は、ユーザー DN をメンバーとして一覧表示しますが、メンバーは一意でなければなりません。これにより、ユーザーをグループメンバーとして複数回追加しないようにできます。これは、セルフ参照グループメンバーシップを防ぐ 1 つの方法です。これのメンバーを判断するために使用される属性は ***uniqueMember*** です。
- **groupOfURLs** は、LDAP URL の一覧を使用して、メンバーシップの一覧をフィルタリングして生成します。このオブジェクトクラスはすべての動的グループに必要で、**groupOfNames** および **groupOfUniqueNames** とともに使用できます。
- **groupOfCertificates** は、グループメンバーを識別するための証明書 (実際には証明書名) を検索して識別するために LDAP フィルターを使用するという点で、**groupOfURLs** と似ています。これは、グループに特別なアクセスパーミッションを付与できるため、グループベースのアクセス制御に役立ちます。これのメンバーを判断するために使用される属性は ***memberCertificate*** です。

以下の表は、グループのデフォルト属性を示しています。

表8.1 動的および静的のグループスキーマ

グループのタイプ	グループオブジェクトクラス	member 属性
静的	***groupOfNames***[a]	***member***
	groupOfUniqueNames[a]	***uniqueMember***
動的	***groupOfURLs***	***memberURL***
	groupOfCertificates	***memberCertificate***

[a] このオブジェクトクラスが動的オブジェクトクラスの 1 つとともに使用されると、グループは動的になります。

以下の 2 つの例は、静的および動的のグループエントリーを示しています。

例8.1 静的グループエントリー

静的グループエントリーは、グループの特定のメンバーをリスト表示します。以下に例を示します。

```
objectClass: top
objectClass: groupOfUniqueNames
cn: static group
description: Example static group.
uniqueMember: uid=mwhite,ou=People,dc=example,dc=com
uniqueMember: uid=awhite,ou=People,dc=example,dc=com
```

例8.2 動的グループエントリー

動的グループは、少なくとも 1 つの LDAP URL を使用して、グループに属するエントリーを識別し、複数の LDAP URL を指定できます。または、**groupOfUniqueNames** のような別のグループオブジェクトクラスで使用する場合は、動的 LDAP URL とともにいくつかのグループメンバーを明示的に一覧表示できます。以下に例を示します。

```
objectClass: top
objectClass: groupOfUniqueNames
objectClass: groupOfURLs
cn: dynamic group
description: Example dynamic group.
memberURL: ldap:///dc=example,dc=com??sub?(&(objectclass=person)(cn=*sen*))
```

注記

memberOf プラグインは、動的に生成されるグループメンバーシップをサポートしません。属性にグループメンバーを一覧表示する代わりに ***memberURL*** 属性を設定すると、**memberOf** プラグインはフィルターに一致するユーザーオブジェクトに ***memberOf*** 属性を追加しません。

8.1.2. 静的グループの作成

Directory Server は、コマンドラインを使用した静的グループの作成のみに対応します。

8.1.2.1. コマンドラインで静的グループの作成

本セクションでは、コマンドラインを使用して異なるタイプの静的グループを作成する方法を説明します。

異なる静的グループの詳細は、「各種グループ」を参照してください。

groupOfNames オブジェクトクラスを使用した静的グループの作成
dsidm ユーティリティーは、指定したベース DN の **cn=Groups** エントリーに静的グループを作成します。

たとえば、**cn=Groups,dc=example,dc=com** エントリーで **groupOfNames** オブジェクトクラスを使用して静的 **example_group** グループを作成するには、次のコマンドを実行します。

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

groupOfUniqueNames オブジェクトクラスを使用した静的グループの作成
groupOfUniqueNames オブジェクトクラスで静的グループを作成するには、**ldapmodify** ユーティリティーを使用してエントリーを追加します。

たとえば、**cn=Groups,dc=example,dc=com** エントリーで **groupOfUniqueNames** オブジェクトクラスを使用して静的 **example_group** グループを作成するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=example_group,cn=Groups,dc=example,dc=com
changetype: add
objectClass: top
objectClass: groupOfUniqueNames
cn: example_group
description: Example static group with unique members
```

8.1.3. 動的グループの作成

Directory Server は、コマンドラインを使用した動的グループの作成のみをサポートします。

8.1.3.1. コマンドラインで動的グループの作成

本セクションでは、コマンドラインを使用してさまざまなタイプの動的グループを作成する方法を説明します。

さまざまな動的グループの詳細は、「各種グループ」を参照してください。

groupOfURLs オブジェクトクラスを使用した動的グループの作成
たとえば、**cn=Groups,dc=example,dc=com** エントリーで **groupOfURLs** オブジェクトクラスを使用して動的 **example_group** グループを作成するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=example_group,cn=Groups,dc=example,dc=com
changetype: add
```

```
objectClass: top
objectClass: groupOfURLs
cn: example_group
description: Example dynamic group for user entries
memberURL: ldap:///dc=example,dc=com??sub?(&(objectclass=person)(cn=*sen*))
```

groupOfCertificates オブジェクトクラスを使用した動的グループの作成
たとえば、**cn=Groups,dc=example,dc=com** エントリーで **groupOfCertificates** オブジェクトクラスを使用して動的 **example_group** グループを作成するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=example_group,cn=Groups,dc=example,dc=com
changetype: add
objectClass: top
objectClass: groupOfURLs
cn: example_group
description: Example dynamic group for certificate entries
memberCertificate: ...
```

8.1.4. ユーザーエントリーにおけるグループメンバーシップのリスト表示

グループに属するエントリーは、グループエントリー自体で定義されます。これにより、グループを確認し、そのメンバーを確認し、グループメンバーシップを一元管理できるようになります。ただし、1 つのユーザーが属するグループを確認することは適切な方法ではありません。ロールがあるため、メンバーシップを示すユーザーエントリーには何もありません。

MemberOf プラグインは、グループメンバーシップのリストを対応するユーザーエントリーに関連付けます。

MemberOf プラグインはグループエントリーのメンバー属性を分析し、メンバーのエントリーに対応する ***memberOf*** 属性を自動的に書き込みます。(デフォルトでは、これにより ***member*** 属性を確認しますが、複数の属性インスタンスを使用して複数の異なるグループタイプをサポートすることができます。)

メンバーシップが変更になると、プラグインはユーザーエントリーの ***memberOf*** 属性を更新します。MemberOf プラグインは、ネスト化されたグループメンバーシップを含むエントリーを確認して、ユーザーが属するグループを表示する方法を提供します。ネスト化されたグループを介してメンバーシップを追跡することは非常に困難ですが、MemberOf プラグインには、すべてのグループの (直接および間接的な) メンバーシップが表示されます。

MemberOf プラグインは、動的グループや循環グループではなく、静的グループのメンバー属性を管理します。

8.1.4.1. memberOf プラグインを使用する場合の考慮事項

本セクションでは、**memberOf** プラグインを使用する際に重要な考慮事項を説明します。

レプリケーショントポロジーでの memberOf プラグインの使用

レプリケーショントポロジーで ***memberOf*** 属性を管理する方法は 2 つあります。

- トポロジー内のすべてのサプライヤーサーバーおよび読み取り専用レプリカサーバーで、**memberOf** プラグインを有効にします。この場合、すべてのレプリカ合意で、レプリケーションから ***memberOf*** 属性を除外する必要があります。属性の除外に関する詳細は、「一部レプリケーションを使用した属性のサブセットの複製」を参照してください。

- **memberOf** プラグインは、トポロジー内のすべてのサプライヤーサーバーでのみ有効にします。そのためには、以下を実行します。
  - レプリカ合意ですべての書き込みが有効なサプライヤーに対する ***memberOf*** 属性のレプリケーションを無効にする必要があります。属性の除外に関する詳細は、「一部レプリケーションを使用した属性のサブセットの複製」を参照してください。
  - ***memberOf*** 属性のレプリケーションを、すべての読み取り専用レプリカに対して、レプリカ合意で有効にする必要があります。
  - 読み取り専用レプリカでは、**memberOf** プラグインを有効にしないでください。

分散データベースでの memberOf プラグインの使用

「データベースの作成」 で説明されているように、ディレクトリーのサブツリーを個別のデータベースに保存できます。デフォルトでは、**memberOf** プラグインはグループと同じデータベース内に保存されるユーザーエントリーのみを更新します。プラグインがグループとして異なるデータベースのユーザーも更新できるようにするには、***memberOfAllBackends*** パラメーターを **on** に設定する必要があります。「Web コンソールを使用した各サーバーでの MemberOf プラグインの設定」を参照してください。

8.1.4.2. memberOf プラグインで必要なオブジェクトクラス

memberOf プラグイン。デフォルトで、**memberOf** プラグインは、**MemberOf** オブジェクトクラスをオブジェクトに追加し、***memberOf*** 属性を提供します。このオブジェクトクラスは、この目的のために任意のオブジェクトに安全に追加でき、このプラグインを正しく動作させるためにこれ以上のアクションは必要ありません。代わりに、**inetUser** または **inetAdmin** オブジェクトクラスが含まれるユーザーオブジェクトを作成できます。どちらのオブジェクトクラスも ***memberOf*** 属性をサポートします。

ネスト化されたグループを設定するには、グループは **extensibleObject** オブジェクトクラスを使用する必要があります。

注記

ディレクトリーエントリーに必要な属性をサポートするオブジェクトクラスが含まれていない場合、操作は以下のエラーで失敗します。

```
LDAP: error code 65 - Object Class Violation
```

8.1.4.3. MemberOf プラグイン構文

MemberOf プラグインインスタンスは、ポーリングするグループメンバー属性 (***memberOfGroupAttr***) と、メンバーのユーザーエントリー (***memberOfAttr***) で作成および管理する属性の 2 つの属性を定義します。

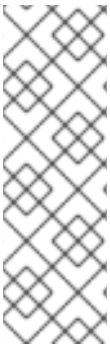
memberOfGroupAttr 属性は複数値です。異なるタイプのグループは異なるメンバー属性を使用するため、複数の ***memberOfGroupAttr*** 属性を使用すると、プラグインで複数のグループタイプを管理できます。

プラグインインスタンスは、MemberOf プラグインを識別するためのプラグインパスと関数も提供し、プラグインを有効にするための状態設定が含まれます。これらの両方は、すべてのプラグインに必要です。デフォルトの MemberOf プラグインが、例8.3「デフォルトの MemberOf プラグインエンティティー」 に表示されます。

例8.3 デフォルトの MemberOf プラグインエンティティー

```
dn: cn=MemberOf Plugin,cn=plugins,cn=config
objectClass: top
objectClass: nsSlapdPlugin
objectClass: extensibleObject
cn: MemberOf Plugin
nsslapd-pluginPath: libmemberof-plugin
nsslapd-pluginInitfunc: memberof_postop_init
nsslapd-pluginType: postoperation
nsslapd-pluginEnabled: on
nsslapd-plugin-depends-on-type: database
memberOfGroupAttr: member
memberOfGroupAttr: uniqueMember
memberOfAttr: memberOf
memberOfAllBackends: on
nsslapd-pluginId: memberOf
nsslapd-pluginVersion: X.Y.Z
nsslapd-pluginVendor: Red Hat, Inc.
nsslapd-pluginDescription: memberOf plugin
```

例で使用するパラメーターの詳細と、設定可能なパラメーターの詳細は、『Red Hat Directory Server Command, Configuration, and File Reference』の『MemberOf Plug-in Attributes』セクションを参照してください。



注記

1 つのメンバー属性 (デフォルトでは ***member***) のみを許可した古いバージョンの Directory Server との後方互換性を維持するために、プラグイン設定で使用される新しいメンバー属性に加えて、***member*** グループ属性または以前のメンバー属性を含める必要がある場合があります。

```
memberOfGroupAttr: member
memberOfGroupAttr: uniqueMember
```

8.1.4.4. MemberOf の有効化

本セクションでは、**MemberOf** プラグインを有効にする方法を説明します。

8.1.4.4.1. コマンドラインを使用した MemberOf プラグインの有効化

コマンドラインで **MemberOf** プラグインを有効にします。

1. **dsconf** ユーティリティーを使用してプラグインを有効にします。

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

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

8.1.4.4.2. Web コンソールを使用した MemberOf プラグインの有効化

Web コンソールを使用して **MemberOf** プラグインを有効にします。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを選択します。
4. **MemberOf** プラグインを選択します。
5. ステータスを **ON** に変更し、プラグインを有効にします。
6. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

8.1.4.5. 各サーバーでの MemberOf プラグインの設定

MemberOf プラグインの設定を複製しない場合は、各サーバーでプラグインを手動で設定します。

8.1.4.5.1. コマンドラインを使用した各サーバーでの MemberOf プラグインの設定

コマンドラインを使用して **MemberOf** プラグインを設定するには、以下を実行します。

1. プラグインを有効にします。「コマンドラインを使用した **MemberOf** プラグインの有効化」を参照してください。
2. デフォルトである ***member*** 以外の属性からグループのメンバーを取得するには、***memberOfGroupAttr*** パラメーターをそれぞれの属性名に設定します。

   たとえば、***uniqueMember*** 属性からグループメンバーを読み取るには、***memberOfGroupAttr*** の現在の値を置き換えます。

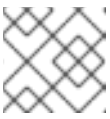
   a. 必要に応じて、現在設定されている属性を表示します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof show
   ...
   memberofgroupattr: member
   ...
   ```

   このコマンドは、現在 ***member*** 属性のみがグループのメンバーを取得するように設定されていることを示しています。

   b. 現在設定されている設定からすべての属性を削除します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof set --
   groupattr delete
   Successfully changed the cn=MemberOf Plugin,cn=plugins,cn=config
   ```



注記

特定のグループ属性を削除できません。

c. 設定に ***uniqueMember*** 属性を追加します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof set --
groupattr uniqueMember
successfully added memberOfGroupAttr value "uniqueMember"
```

複数の属性を設定するには、すべて ***--groupattr*** パラメーターに渡します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof set --
groupattr member uniqueMember ...
```

3. デフォルトでは、MemberOf プラグインは ***memberOf*** 属性をユーザーエントリーに追加します。別の属性を使用するには、***memberOfAttr*** パラメーターで属性の名前を設定します。

たとえば、***customMemberOf*** 属性をユーザーレコードに追加するには、***memberOfAttr*** の現在の値を置き換えます。

a. 必要に応じて、現在設定されている属性を表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof show
...
memberofattr: memberOf
...
```

b. MemberOf プラグインを設定し、***customMemberOf*** 属性をユーザーエントリーに追加します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof set --
attr customMemberOf
memberOfAttr set to "customMemberOf"
```

注記

このパラメーターは、DN 構文をサポートする属性にのみ設定できます。

4. 分散データベースを使用する環境では、ローカルデータベースのみではなく、すべてのデータベースでユーザーエントリーを検索するようにプラグインを設定できます。

```
dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof set --
allbackends on
memberOfAllBackends enabled successfully
```

5. インスタンスを再起動します。

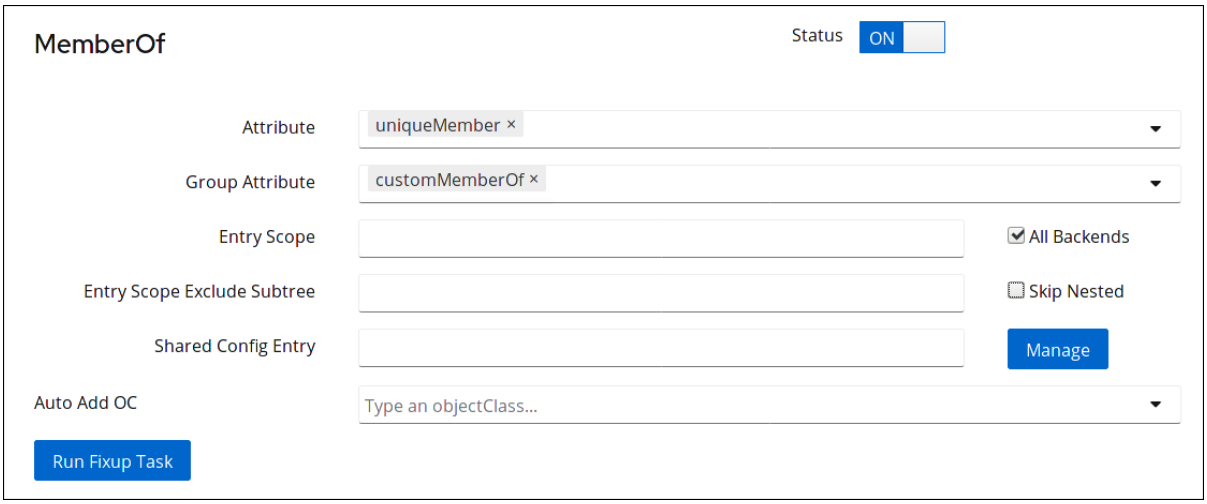
```
# dsctl instance_name restart
```

8.1.4.5.2. Web コンソールを使用した各サーバーでの MemberOf プラグインの設定

コマンドラインを使用して MemberOf プラグインを設定するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソール

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **memberOf** プラグインを選択します。
5. ステータスを **ON** に変更し、プラグインを有効にします。
6. フィールドに入力してプラグインを設定します。たとえば、***uniqueMember*** 属性がグループに追加されると、プラグインがユーザーエントリーに ***customMemberOf*** 属性を追加するように設定するには、以下を実行します。



7. **Save** をクリックします。
8. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

8.1.4.6. MemberOf プラグイン共有設定の使用

デフォルトでは、**MemberOf** プラグインの設定は各サーバーに保存されます。プラグインの共有設定機能を使用すると、設定は **cn=config** 接尾辞の外に格納され、複製されます。管理者は、各サーバーでプラグインを手動で設定せずに、同じ設定を使用できます。

1. プラグインを有効にします。「**MemberOf** の有効化」を参照してください。
2. **MemberOf** プラグインの共有設定エントリーを追加します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof config-entry add "cn=shared_MemberOf_config,dc=example,dc=com" --groupattr "member" --attr "memberOf"
```

これにより、コマンドを実行するサーバーで共有設定エントリーが自動的に有効になります。

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

4. 共有設定を使用する必要があるレプリケーショントポロジー内の他のすべてのサーバーで、共有設定を有効にします。

   a. プラグインを有効にします。「MemberOf の有効化」を参照してください。

   b. 共有設定を保存する DN を設定します。以下に例を示します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof set --config-entry cn=shared_MemberOf_config,dc=example,dc=com
   ```

   c. インスタンスを再起動します。

   ```
   # dsctl instance_name restart
   ```

重要

共有設定を有効にすると、プラグインは **cn=MemberOf Plugin,cn=plugins,cn=config** プラグインエントリーに設定されたすべてのパラメーターを無視し、共有設定エントリーの設定のみを使用します。

8.1.4.7. MemberOf プラグインのスコープの設定

複数のバックエンドまたは複数のネストされた接尾辞を設定した場合は、***memberOfEntryScope*** パラメーターおよび ***memberOfEntryScopeExcludeSubtree*** パラメーターを使用して、**MemberOf** プラグインが動作する接尾辞を設定できます。

ユーザーをグループに追加する場合、**MemberOf** プラグインは、ユーザーおよびグループの両方がプラグインのスコープにある場合に限り ***memberOf*** 属性をグループに追加します。たとえば、***dc=example,dc=com*** 内のすべてのエントリーで機能するように **MemberOf** プラグインを設定し、***ou=private,dc=example,dc=com*** のエントリーを除外するには、以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof set --scope "dc=example,com"
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof set --exclude "dc=group,dc=example,com"
```

--scope DN パラメーターを使用してユーザーエントリーを範囲外に移動した場合:

- ***member*** などのメンバーシップ属性は、グループエントリーで更新され、ユーザー DN 値が削除されます。
- ***memberOf*** 属性は、グループ DN 値を削除するためにユーザーエントリーで更新されます。

注記

--exclude パラメーターに設定した値は、***--scope*** に設定した値よりも高くなります。両方のパラメーターで設定したスコープが重複する場合、**MemberOf** プラグインは、非オーバーラッピングディレクトリーエントリーでのみ機能します。

8.1.4.8. *memberOf* 値の再生成

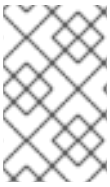
MemberOf プラグインは、グループエントリー自体の設定に基づいて、グループメンバーエントリーで ***memberOf*** 属性を自動的に管理します。ただし、***memberOf*** 属性はユーザーエントリーで手動で編集でき、新しいエントリーは ***memberOf*** 属性がすでに設定されているサーバーにインポートまたは複

製できます。このような状況では、サーバープラグインによって管理される ***memberOf*** 設定と、エントリーで定義された実際のメンバーシップとの間に不整合が生じます。

たとえば、**dc=example,dc=com** エントリーおよびサブエントリーで ***memberOf*** の値を再生成するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin memberof fixup -f "(|
(objectclass=inetuser)(objectclass=inetadmin)(objectclass=nsmemberof))" "dc=example,dc=com"
Attempting to add task entry...
Successfully added task entry
```

-f *filter* オプションは任意です。フィルターを使用して、フィルターに一致するユーザーエントリーの ***memberOf*** 属性を再生成します。フィルターを指定しない場合、タスクは **inetUser** オブジェクトクラス、**inetAdmin** オブジェクトクラス、または **nsMemberOf** オブジェクトクラスを含むすべてのエントリーで属性を再生成します。



注記

エントリー自体が複製された場合でも、再生成タスクをローカルに実行します。つまり、更新されたエントリーが複製されるまで、他のサーバーのエントリーの ***memberOf*** 属性は更新されません。

8.1.5. 指定したグループへのエントリーの自動追加

- 「Automembership ルールの構造の確認」
- 「Automembership ルールの例」
- 「**Auto Membership** 定義の設定」

グループ管理は、特に Directory Server データおよび組織を使用するクライアントや、グループを使用してエントリーに機能を適用するクライアントなど、ディレクトリーデータを管理する上で重要な要素となります。グループにより、ディレクトリー全体で一貫して、信頼できるポリシーの適用が容易になります。パスワードポリシー、アクセス制御リスト、その他のルールはすべてグループメンバーシップに基づいて設定できます。

アカウントの作成時に、新しいエントリーをグループに自動的に割り当てることができるため、管理者の介入なしに、適切なポリシーと機能がそれらのエントリーに即座に適用されるようにします。

動的グループは、一致するエントリーがグループに自動的に含まれるため、グループを作成してメンバーを自動的に割り当てる 1 つの方法です。Directory Server のポリシーおよび設定を適用するには、これで十分です。ただし、LDAP アプリケーションとクライアントには、通常、必要な操作を行うためにグループメンバーの静的リストおよび明示的なリストが必要です。静的グループのすべてのメンバーは、このグループに手動で追加する必要があります。

静的グループ自体は動的グループのようなメンバーを検索できませんが、静的グループに自動的にメンバーを追加できるようになります (**Auto Membership プラグイン**)。

自動メンバーシップにより、基本的に、静的グループが動的グループのように動作できるようにします。異なる自動メンバー定義により、すべての新規ディレクトリーエントリーで自動的に実行される検索が作成されます。自動メンバールールは、動的検索フィルターと同様に、一致するエントリーを検索し、特定します。次に、これらのエントリーをメンバーとして静的グループに追加します。

注記

デフォルトでは、**cn=Auto Membership Plugin,cn=plugins,cn=config** エントリーの ***autoMemberProcessModifyOps*** パラメーターは **on** に設定されます。この設定では、Automembership プラグインは、ユーザーエントリーを編集して管理者が別のグループにユーザーを移動する際にグループメンバーシップも更新します。

autoMemberProcessModifyOps を **off** に設定すると、Directory Server は、ユーザーにグループエントリーを追加する場合にのみプラグインを起動し、グループメンバーシップを更新するために手動で修正タスクを実行する必要があります。

Automembership は、ディレクトリーに保存されているオブジェクトタイプ (ユーザー、マシン、ネットワークデバイス、顧客データ、またはその他のアセット) をターゲットにすることができます。

注記

Automembership は、定義した基準に基づいて既存のグループに新しいメンバーを追加します。新しいエントリー用のグループは作成されません。

特定タイプの新規エントリーの作成時に対応するグループエントリーを作成するには、Managed Entries プラグインを使用します。詳細は、「デュアルエントリーの自動作成」を参照してください。

8.1.5.1. Automembership ルールの構造の確認

Auto Membership プラグイン自体は、**cn=plugins,cn=config** のコンテナーエントリーです。グループ割り当ては、子エントリーで定義されます。

8.1.5.1.1. Automembership 設定エントリー

自動メンバー割り当ては、Automembership プラグインエントリーの子であるメインの定義エントリーを使用して作成されます。各定義エントリーは 3 つの要素を定義します。

- 検索スコープと検索フィルターの両方を含むエントリーを識別する LDAP 検索 (***autoMemberScope*** および ***autoMemberFilter***)
- メンバーエントリーを追加するデフォルトグループ (***autoMemberDefaultGroup***)
- メンバーエントリーの形式: ***member*** などのグループエントリーの属性、および ***dn*** などの属性値 (***autoMemberGroupingAttr***)

定義は、automember ルールの基本設定です。必要な情報をすべて識別し、一致するメンバーエントリーとそのメンバーの所属先のグループを特定します。

たとえば、以下の定義は、オブジェクトクラスが **ntUser** に設定されているすべてのユーザーを **cn=windows-users** グループに割り当てます。

```
dn: cn=Windows Users,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
autoMemberScope: ou=People,dc=example,dc=com
autoMemberFilter: objectclass=ntUser
autoMemberDefaultGroup: cn=windows-group,cn=groups,dc=example,dc=com
autoMemberGroupingAttr: member:dn
```

この例に使用される属性と、このエントリーに設定できるその他の属性の詳細は、『Red Hat Directory Server Configuration, Command, and File Reference』の **cn=Auto Membership Plugin,cn=plugins,cn=config** エントリーの説明を参照してください。

8.1.5.1.2. 追加の正規表現エントリー

ユーザーグループのように、一致するすべてのエントリーをメンバーとして追加する必要がある場合は、単純な定義で十分です。ただし、他の属性の値によっては、LDAP 検索フィルターに一致するエントリーを異なるグループに追加する必要がある場合があります。たとえば、IP アドレスや物理的な場所に応じて、異なるグループにマシンを追加しないといけない場合があります。ユーザーは、従業員 ID 番号に応じて異なるグループに置かなければならない場合があります。

automember 定義では、正規表現を使用して、グループから含めたり、除外したりするエントリーに追加の条件を提供したり、選択したエントリーを追加する新しい特定のグループを作成したりできます。

たとえば、automember 定義は、汎用ホストグループに追加されるすべてのマシンを設定します。

例8.4 ホストグループの automember 定義

```
dn: cn=Hostgroups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
cn: Hostgroups
autoMemberScope: dc=example,dc=com
autoMemberFilter: objectclass=ipHost
autoMemberDefaultGroup: cn=systems,cn=hostgroups,dc=example,dc=com
autoMemberGroupingAttr: member:dn
```

指定の範囲内に完全修飾ドメイン名を持つマシンが Web サーバーグループに追加されるように、正規表現ルールが追加されます。

例8.5 Web サーバーグループの正規表現条件

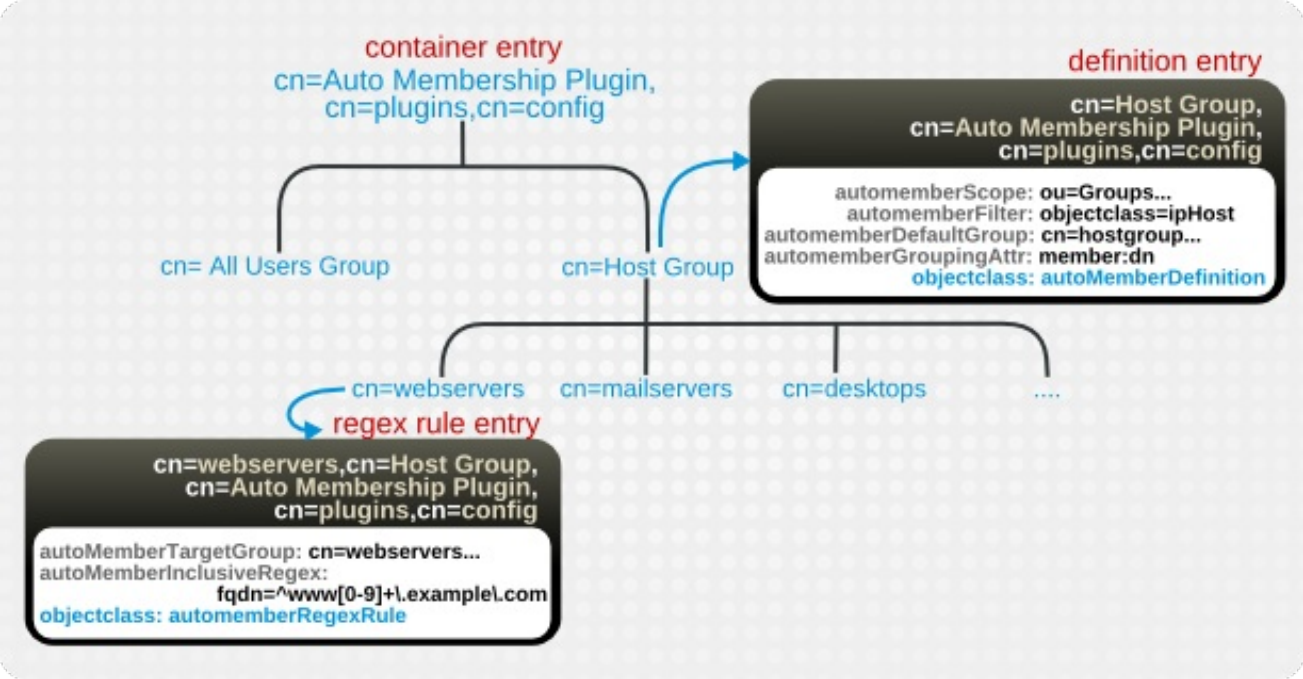
```
dn: cn=webservers,cn=Hostgroups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group for webservers
cn: webservers
autoMemberTargetGroup: cn=webservers,cn=hostgroups,dc=example,dc=com
autoMemberInclusiveRegex: fqdn=^www\.web[0-9]+\.example\.com
```

そのため、**www.web1.example.com** などの式 **^www\.web[0-9]+\.example\.com** に一致する完全修飾ドメイン名で追加されたホストマシンは、正確な正規表現に定義される **cn=webservers** グループに追加されます。LDAP フィルター **objectclass=ipHost** に一致するが別のタイプの完全修飾ドメイン名を持つその他のマシンエントリーは、メインの定義エントリーで定義される一般的なホストグループ **cn=systems** に追加されます。

したがって、定義内のグループは、一般的な定義に一致するが、正規表現ルールの条件を満たさないエントリーのフォールバックです。

正規表現ルールは、自動メンバー定義の子エントリーです。

図8.1 正規表現の条件

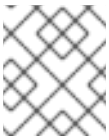


各ルールには、複数の包含および除外の式を含めることができます。(除外は最初に評価されます。) エントリーが包含ルールと一致する場合は、グループに追加されます。

正規表現ルールに指定できるターゲットグループは 1 つだけです。

表8.2 正規表現の条件属性

属性	説明
autoMemberRegexRule (必須オブジェクトクラス)	正規表現ルールとしてエントリーを識別します。このエントリーは、automember 定義 (**objectclass: autoMemberDefinition**) の子である必要があります。
autoMemberInclusiveRegex	含めるエントリーを識別するために使用する正規表現を設定します。一致するエントリーのみがグループに追加されます。複数の正規表現を使用できます。エントリーがこれらの式のいずれかと一致する場合は、グループに含まれます。 式の形式は、Perl と互換性のある正規表現 (PCRE) です。PCRE パターンの詳細は、**pcresyntax**(3) の man ページを参照してください。 これは多値属性です。

属性	説明
autoMemberExclusiveRegex	除外するエントリーを識別するために使用する正規表現を設定します。エントリーが除外条件と一致する場合は、グループに **含まれません**。複数の正規表現を使用できます。エントリーがこれらの式のいずれかと一致する場合は、グループで除外されます。 式の形式は、Perl と互換性のある正規表現 (PCRE) です。PCRE パターンの詳細は、**pcresyntax**(3) の man ページを参照してください。 これは多値属性です。  **注記** 除外条件は最初に評価され、包含条件よりも優先されます。
autoMemberTargetGroup	正規表現の条件を満たす場合に、エントリーをメンバーとして追加するグループを設定します。

8.1.5.2. Auto Membership 定義の設定

Auto Membership プラグインを使用するには、プラグインの定義を作成します。

8.1.5.2.1. コマンドラインを使用した Auto Membership 定義の設定

コマンドラインを使用して **Auto Membership** 定義を作成するには、以下を実行します。

1. **Auto Membership** プラグインを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin automember enable
Enabled Auto Membership Plugin
```

2. **Auto Membership** 定義を作成します。以下に例を示します。

```
#  dsconf -D "cn=Directory Manager" ldap://server.example.com plugin automember
definition definition_name add --default-group "cn=windows-
group,cn=groups,dc=example,dc=com" --scope "ou=People,dc=example,dc=com" --filter
"objectclass=ntUser" --grouping-attr "member:dn"
Automember definition created successfully!
```

3. 必要に応じて、**Auto Membership** 定義に追加のパラメーターを設定して、たとえば正規表現を使用して追加するエントリーを特定します。**ldapmodify** ユーティリティーを使用して、**cn=*definition_name*,cn=Auto Membership Plugin,cn=plugins,cn=config** エントリーにこれらのパラメーターを追加または更新します。設定可能なパラメーターは、Red Hat Directory Server Configuration, Command, and File Reference の cn=Auto Membership Plugin,cn=plugins,cn=config エントリーの説明を参照してください。

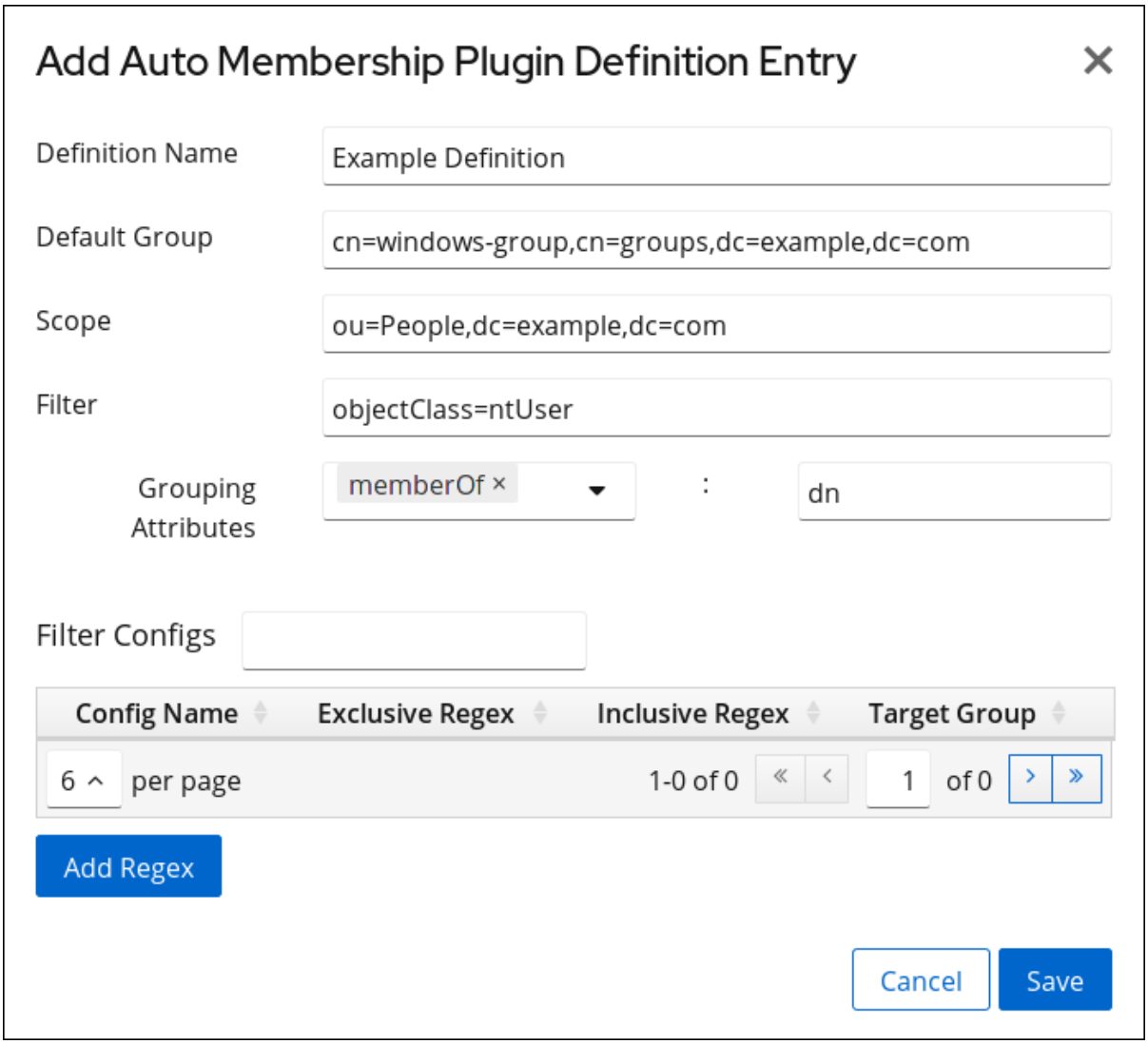
4. インスタンスを再起動します。

```
# dsctl instance_name restart
```

8.1.5.2.2. Web コンソールを使用した Auto Membership 定義の設定

Web コンソールを使用して **Auto Membership** 定義を作成するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを開きます。
4. **Auto Membership** プラグインを選択します。
5. ステータスを **ON** に変更し、プラグインを有効にします。
6. **Add Definition** をクリックします。
7. フィールドに入力します。以下に例を示します。



8. オプションで、正規表現フィルターを追加します。
9. **Save** をクリックします。

10. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

8.1.5.3. 既存エントリーの更新による Auto Membership 定義の適用

デフォルトでは、**cn=Auto Membership Plugin,cn=plugins,cn=config** エントリーの ***autoMemberProcessModifyOps*** パラメーターは有効です。この設定では、Automembership プラグインは、ユーザーエントリーを編集して管理者が別のグループにユーザーを移動する際にグループメンバーシップも更新します。ただし、***autoMemberProcessModifyOps*** を **off** に設定した場合は、ディレクトリーに新しいエントリーを追加したか、既存のエントリーを変更した場合に、手動で修正タスクを実行する必要があります。

タスクエントリーを作成するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin automember fixup -f "filter" -s scope
```

タスクが完了すると、エントリーはディレクトリー設定から削除されます。

8.1.5.4. Automembership ルールの例

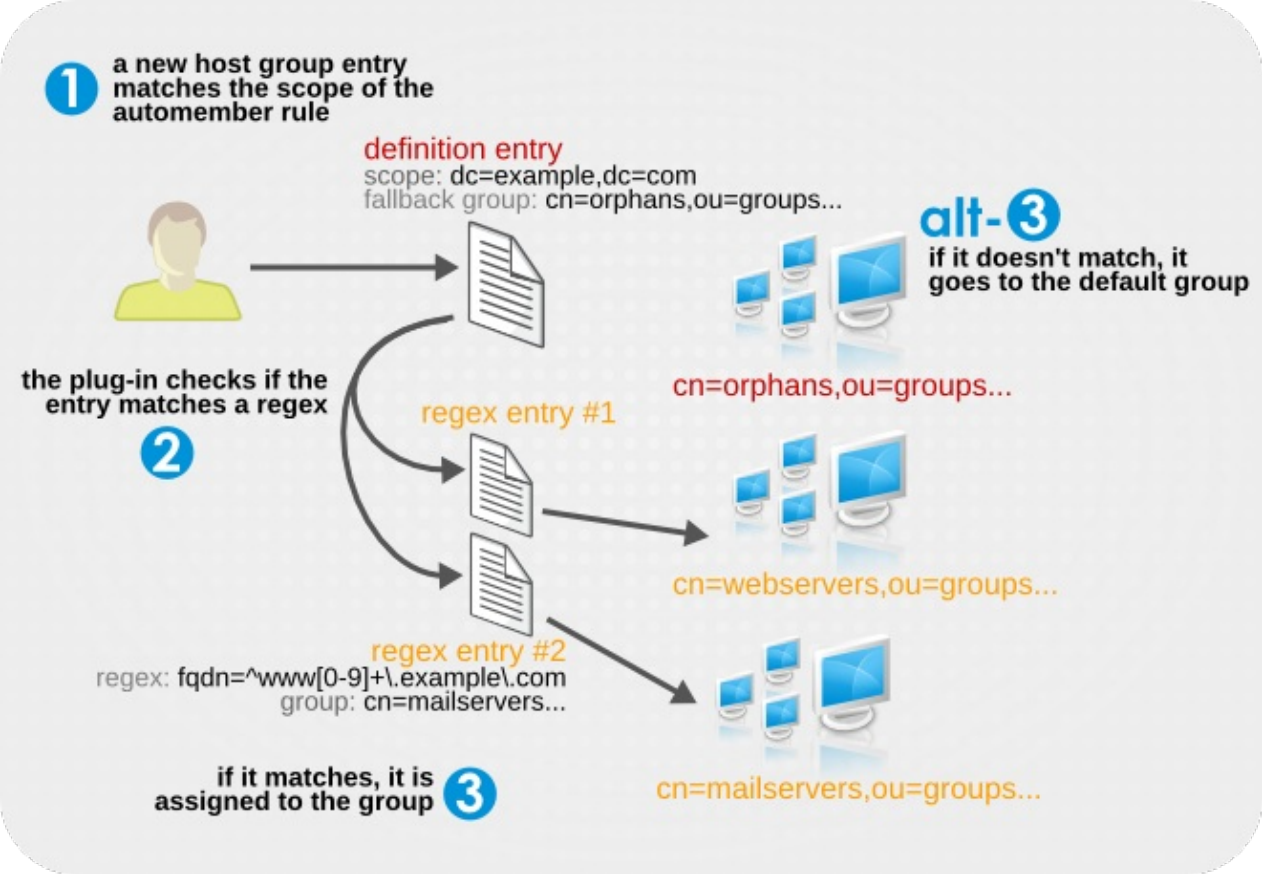
Automembership ルールは通常、ユーザーとマシンに適用されます (ただし、どのタイプのエントリーにも適用することができます)。自動メンバーシップルールの計画に役立つ便利な例がいくつかあります。

- IP アドレスに基づく異なるホストグループ
- Windows ユーザーグループ
- 従業員の ID に基づく異なるユーザーグループ

例8.6 IP アドレス別のホストグループ

automember ルールは、まずルールのスコープとターゲットを定義します。「追加の正規表現エントリー」の例では、設定グループを使用してフォールバックグループと正規表現エントリーを定義し、一致するエントリーをソートします。

スコープは、**全** ホストエントリーの検索に使用されます。その後、プラグインは正規表現エントリーで繰り返し処理します。エントリーが包含する正規表現と一致する場合は、そのホストグループに追加されます。グループに一致しない場合は、デフォルトグループに追加されます。



実際のプラグイン設定エントリーは、定義エントリーに、ホストを Web サーバーグループまたはメールサーバーグループにフィルターを設定する 2 つの正規表現エントリーに対して設定されます。

```
configuration entry
dn: cn=Hostgroups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
cn: Hostgroups
autoMemberScope: dc=example,dc=com
autoMemberFilter: objectclass=bootableDevice
autoMemberDefaultGroup: cn=orphans,cn=hostgroups,dc=example,dc=com
autoMemberGroupingAttr: member:dn

regex entry #1
dn: cn=webservers,cn=Hostgroups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group placement for webservers
cn: webservers
autoMemberTargetGroup: cn=webservers,cn=hostgroups,dc=example,dc=com
autoMemberInclusiveRegex: fqdn=^www[0-9]+\.example\.com
autoMemberInclusiveRegex: fqdn=^web[0-9]+\.example\.com
autoMemberExclusiveRegex: fqdn=^www13\.example\.com
autoMemberExclusiveRegex: fqdn=^web13\.example\.com

regex entry #2
dn: cn=mailservers,cn=Hostgroups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group placement for mailservers
cn: mailservers
autoMemberTargetGroup: cn=mailservers,cn=hostgroups,dc=example,dc=com
```

```
autoMemberInclusiveRegex: fqdn=^mail[0-9]+\.example\.com
autoMemberInclusiveRegex: fqdn=^smtp[0-9]+\.example\.com
autoMemberExclusiveRegex: fqdn=^mail13\.example\.com
autoMemberExclusiveRegex: fqdn=^smtp13\.example\.com
```

例8.7 Windows ユーザーグループ

「Automembership 設定エントリー」 に表示されている基本的なユーザーグループは、***posixAccount*** 属性を使用して新規ユーザーをすべて識別します。Directory Server 内に作成された新規ユーザーはすべて、***posixAccount*** 属性を使用して作成されます。したがって、新しい Directory Server ユーザーにとって安全なキャッチオールです。ただし、ユーザーアカウントを Windows ドメインから Directory Server に同期すると、Windows ユーザーアカウントは ***posixAccount*** 属性 **なし** で作成されます。

Windows ユーザーは ***ntUser*** 属性で識別されます。基本的な all-users グループルールは、特に目的の Windows ユーザーに変更できます。これは、デフォルトの all-users グループまたは Windows 固有のグループに追加できます。

```
dn: cn=Windows Users,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
autoMemberScope: dc=example,dc=com
autoMemberFilter: objectclass=ntUser
autoMemberDefaultGroup: cn=Windows Users,cn=groups,dc=example,dc=com
autoMemberGroupingAttr: member:dn
```

例8.8 従業員タイプによるユーザーグループ

Auto Membership プラグインはカスタム属性で機能します。これは、他のアプリケーションが管理するエントリーに役立ちます。たとえば、人的リソースアプリケーションは、カスタムの ***employeeType*** 属性で従業員タイプをもとにユーザーを作成してから参照できます。

例8.6「IP アドレス別のホストグループ」と同様、ユーザータイプのルールは 2 つの正規表現フィルターを使用して、すべての時間および一時従業員をソートます。この例では、実際の正規表現ではなく明示的な値を使用します。その他の属性については、従業員の ID 番号範囲に基づいてフィルターを作成するなど、正規表現を使用することが推奨されます。

```
configuration entry
dn: cn=Employee groups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberDefinition
cn: Hostgroups
autoMemberScope: ou=employees,ou=people,dc=example,dc=com
autoMemberFilter: objectclass=inetorgperson
autoMemberDefaultGroup: cn=general,cn=employee groups,ou=groups,dc=example,dc=com
autoMemberGroupingAttr: member:dn

regex entry #1
dn: cn=full time,cn=Employee groups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group for full time employees
cn: full time
autoMemberTargetGroup: cn=full time,cn=employee groups,ou=groups,dc=example,dc=com
autoMemberInclusiveRegex: employeeType=full
```

```
regex entry #2
dn: cn=temporary,cn=Employee groups,cn=Auto Membership Plugin,cn=plugins,cn=config
objectclass: autoMemberRegexRule
description: Group placement for interns, contractors, and seasonal employees
cn: temporary
autoMemberTargetGroup: cn=temporary,cn=employee groups,ou=groups,dc=example,dc=com
autoMemberInclusiveRegex: employeeType=intern
autoMemberInclusiveRegex: employeeType=contractor
autoMemberInclusiveRegex: employeeType=seasonal
```

8.1.5.5. 自動メンバー定義のテスト

Auto Member プラグインの各インスタンスは、定義と正規表現に関連してはいますが別々のエントリーのセットであるため、ユーザーがグループにどのようにマップされるかを正確に確認するのは難しい場合があります。これは、ユーザーの異なるサブセットをターゲットとする複数のルールがあると、より困難になります。

ドライランタスクが 2 つあり、すべての Auto Member プラグイン定義が設計通りにグループを適切に割り当てているかどうかを判断するのに役立ちます。

既存のエントリーを使用したテスト

cn=automember export updates が、ディレクトリー内の **既存のエントリー** に対して実行し、ルールに基づいてユーザーの追加結果をエクスポートします。これは、既存のルールをテストして、実際のデプロイメントの実行方法を確認するのに役立ちます。

このタスクには、**cn=automember rebuild membership** タスク (検索、検索フィルター、および検索スコープのベース DN) と同じ情報が必要です。また、推奨されるエントリーの更新を記録するエクスポート LDIF ファイルを指定する追加パラメーターがあります。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=test_export,cn=automember export updates,cn=tasks,cn=config
objectClass: top
objectClass: extensibleObject
cn: test_export
basedn: dc=example,dc=com
filter: (uid=*)
scope: sub
ldif: /tmp/automember-updates.ldif
```

Import LDIF でのテスト

cn=automember map updates タスクは、新規ユーザーの **インポート LDIF** を取得してから、現在の自動メンバールールに対して新規ユーザーを実行します。これは、(実際の) 新規または既存のユーザーエントリーに適用する前に、新しいルールをテストする場合に非常に役立ちます。

これは、提案された新しいエントリーの変更を、既存のルールにマッピングまたは関連付けるため、マップタスクと呼ばれます。

このタスクには、入力 LDIF (少なくとも一部のユーザーエントリーを含む) の場所と、提案されたエントリー更新を書き込む出力 LDIF ファイルの 2 つの属性のみが必要です。入力および出力の LDIF ファイルの両方がローカルマシンの絶対パスです。

たとえば、**ldapmodify** を使用するには、以下を実行します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=test_mapping, cn=automember map updates,cn=tasks,cn=config
objectClass: top
objectClass: extensibleObject
cn: test_mapping
ldif_in: /tmp/entries.ldif
ldif_out: /tmp/automember-updates.ldif
```

8.1.5.6. Auto Membership プラグインタスクのキャンセル

ディレクトリーサーバーの設定が複雑な場合 (大規模なグループ、複雑なルール、他のプラグインとの対話など)、Auto Membership プラグインタスクが原因でサーバーの CPU 使用率が高くなる可能性があります。パフォーマンスの問題を防ぐために、Auto Membership プラグインタスクをキャンセルできます。

手順

- Auto Membership プラグインタスクをキャンセルするには、以下を入力します。

```
# dsconf server.example.com plugin automember abort-fixup
```

検証

- キャンセルされたタスクを含む、すべての Auto Membership プラグインタスクのリストを表示するには、以下を入力します。

```
# dsconf server.example.com plugin automember fixup-status
```

8.2. ロールの使用

ロールは、前述のセクションで説明されている静的および動的グループを統一するエントリーグループ化メカニズムです。ロールは、アプリケーションにより効率的で使いやすいように設計されています。たとえば、アプリケーションはグループを選択し、複数のグループのメンバーリストを参照するのではなく、エントリー自体をクエリーしてメンバーであるロールのリストを取得できます。

8.2.1. ロールの概要

Red Hat には 2 種類のグループがあります。**静的グループ** には、有限で、定義されたメンバーのリストがあります。**動的グループ** は、フィルターを使用してどのエントリーがグループのメンバーかを認識するため、グループメンバーシップはグループフィルターの変更に一致するエントリーとして常に変更されます。(両方の種類グループが、「グループの使用」で説明されています。)

ロール はハイブリッドグループで、静的グループと動的グループの両方として機能します。グループを使用すると、エントリーはメンバーとしてグループエントリーに追加されます。ロールを使用すると、role 属性がエントリーに追加され、その属性はロールエントリー内のメンバーを自動的に識別するために使用されます。

ロール **メンバー** は、ロールを持つエントリーです。メンバーは、明示的に、または動的に指定できます。ロールのメンバーシップの指定方法は、ロールのタイプによって異なります。Directory Server は、以下の 3 種類のロールをサポートします。

- **マネージドロール** には、メンバーの明示的な列挙リストがあります。
- **フィルターされたロール** には、LDAP フィルターで指定される各エントリーに含まれる属性に応じて、エントリーがロールに割り当てられます。フィルターに一致するエントリーはロールを持ちます。
- **ネストされたロール** は、他のロールが含まれるロールです。

マネージドロールは、通常、静的グループで実行可能なものをすべて実行できます。ロールメンバーは、動的グループによるフィルタリングと同様に、フィルターされたロールを使用してフィルタリングできます。ロールはグループよりも使いやすく、実装に柔軟性が高まり、クライアントの複雑さが軽減されます。

ロールの作成時には、ユーザーがロールから追加できるか、ロールから削除できるかどうかを判断します。ロールおよびアクセス制御の詳細は、「セキュアなロールの使用」を参照してください。

注記

サーバーがクライアントアプリケーションに対して機能するため、Directory Server ではロールの評価がグループを評価するよりもリソース集約されます。ロールを使用すると、クライアントアプリケーションは ***nsRole*** 属性を検索してロールのメンバーシップを確認することができます。***nsRole*** 属性は、エントリーが属するロールを識別する計算属性です。***nsRole*** 属性はエントリー自体に保存されません。クライアントアプリケーションの観点からは、メンバーシップを確認する方法は統一されており、サーバー側で実行されます。

ロールの使用に関する考慮事項は、『Red Hat Directory Server デプロイメントガイド』で説明しています。

8.2.2. コマンドラインを使用したロールの管理

コマンドラインを使用して、ロールを表示、作成、および削除できます。

8.2.2.1. 管理ロールの作成

マネージドロールには、メンバーの明示的な列挙リストがあります。エントリーに ***nsRoleDN*** 属性を追加して、管理ロールがエントリーに追加されます。

8.2.2.1.1. コマンドラインでの管理ロールの作成

ロールは、ITU X.509 標準で定義されている **ldapsubentry** オブジェクトクラスから継承されます。さらに、各管理ロールには、**nsRoleDefinition** オブジェクトクラスから継承する 2 つのオブジェクトクラスが必要です。

- nsSimpleRoleDefinition
- nsManagedRoleDefinition

管理ロールでは、任意の ***description*** 属性も許可されます。

管理ロールのメンバーは、エントリーに ***nsRoleDN*** 属性を持ちます。

この例では、マーケティング部門に割り当てることのできるロールを作成します。

1. **-a** オプションで **ldapmodify** を使用して、管理ロールエントリーを追加します。新しいエントリーには、**nsManagedRoleDefinition** オブジェクトクラスが含まれ、その後に

LdapSubEntry、**nsRoleDefinition**、および **nsSimpleRoleDefinition** オブジェクトクラスを継承します。

```
dn: cn=Marketing,ou=people,dc=example,dc=com
objectclass: top
objectclass: LdapSubEntry
objectclass: nsRoleDefinition
objectclass: nsSimpleRoleDefinition
objectclass: nsManagedRoleDefinition
cn: Marketing
description: managed role for marketing staff
```

2. **ldapmodify** を使用して、ロールをマーケティングスタッフメンバーに 1 つずつ割り当てます。

```
dn: cn=Bob,ou=people,dc=example,dc=com
changetype: modify
add: nsRoleDN
nsRoleDN: cn=Marketing,ou=people,dc=example,dc=com
```

エントリーの ***nsRoleDN*** 属性は、エントリーが管理ロール **cn=Marketing,ou=people,dc=example,dc=com** のメンバーであることを示します。

8.2.2.2. フィルター設定されたロールの作成

エントリーは、エントリーがロールに定義されている特定の属性を持つかどうかに応じて、フィルターされたロールに割り当てられます。ロール定義は、ターゲット属性の LDAP フィルターを指定します。フィルターに一致するエントリーは、ロールを持ちます (ロールのメンバーです)。

8.2.2.2.1. コマンドラインでフィルターされたロールの作成

ロールは、ITU X.509 標準で定義されている **ldapsubentry** オブジェクトクラスから継承されます。さらに、フィルターが設定された各ロールには、**nsRoleDefinition** オブジェクトクラスから継承される 2 つのオブジェクトクラスが必要です。

- nsComplexRoleDefinition
- nsFilteredRoleDefinition

フィルタリングされたロールエントリーには、ロールメンバーを判断するために LDAP フィルターを定義する ***nsRoleFilter*** 属性も必要です。任意で、ロールは ***description*** 属性を取ることができます。

フィルタリングされたロールのメンバーは、***nsRoleFilter*** 属性で指定されたフィルターに一致するエントリーです。

この例では、すべての営業マネージャーに適用される、フィルターが設定されたロールを作成します。

1. **-a** オプションを指定して **ldapmodify** を実行して、新規エントリーを追加します。
2. フィルターされたロールエントリーを作成します。

   ロールエントリーには **nsFilteredRoleDefinition** オブジェクトクラスがあり、これは、オブジェクトクラス **LdapSubEntry**、**nsRoleDefinition**、および **nsComplexRoleDefinition** から継承されます。

nsRoleFilter 属性は、**sales managers** の値が含まれる ***o*** (組織) 属性にフィルターを設定します。

```
dn: cn=SalesManagerFilter,ou=people,dc=example,dc=com
changetype: add
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsFilteredRoleDefinition
cn: SalesManagerFilter
nsRoleFilter: o=sales managers
Description: filtered role for sales managers
```

以下のエントリーはフィルター (**sales managers** 値を持つ ***o*** 属性) と一致するため、このフィルターが自動的に設定されたロールのメンバーになります)。

```
dn: cn=Pat Smith,ou=people,dc=example,dc=com
objectclass: person
cn: Pat
sn: Smith
userPassword: secret
o: sales managers
```

8.2.2.3. ネスト化されたロールの作成

ネストされたロールは、他のロールが含まれるロールです。ネストされたロールを作成する前に、別のロールが存在している必要があります。ネストされたロール内でネストされたロールは、***nsRoleDN*** 属性を使用して指定します。

8.2.2.3.1. コマンドラインでのネスト化されたロールの作成

ロールは、ITU X.509 標準で定義されている **ldapsubentry** オブジェクトクラスから継承されます。さらに、ネスト化された各ロールには、**nsRoleDefinition** オブジェクトクラスから継承する 2 つのオブジェクトクラスが必要です。

- nsComplexRoleDefinition
- nsNestedRoleDefinition

ネストされたロールエントリーには、コンテナーロール内でネスト化するロールを識別するための ***nsRoleDN*** 属性も必要です。任意で、ロールは ***description*** 属性を取ることができます。

ネスト化されたロールのメンバーは、ネストされたロール定義エントリーの ***nsRoleDN*** 属性で指定されたロールのメンバーです。

この例では、管理されたマーケティングのロールとフィルター処理されたセールスマネージャーのロールから 1 つのロールを作成します。

1. **-a** オプションを指定して **ldapmodify** を実行して、新規エントリーを追加します。
2. ネストされたロールエントリーを作成します。ネストされたロールには 4 つのオブジェクトクラスがあります。
   - **nsNestedRoleDefinition**

- **LDAPsubentry** (継承)
- **nsRoleDefinition** (継承)
- **nsComplexRoleDefinition** (継承)

nsRoleDN 属性には、マーケティング管理ロールと営業マネージャーのフィルターが設定されたロールの両方の DN が含まれます。

```
dn: cn=MarketingSales,ou=people,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: nsRoleDefinition
objectclass: nsComplexRoleDefinition
objectclass: nsNestedRoleDefinition
cn: MarketingSales
nsRoleDN: cn=SalesManagerFilter,ou=people,dc=example,dc=com
nsRoleDN: cn=Marketing,ou=people,dc=example,dc=com
```

以前の例のユーザー Bob および Pat はどちらも、この新しいネストされたロールのメンバーです。

8.2.2.4. コマンドラインでエントリーのロールの表示

ロール割り当ては、コマンドラインから自動的に返されません。

nsRole 属性は操作の属性です。LDAP では、操作属性を明示的に要求する必要があります。デフォルトでは、エントリーのスキーマに通常の属性が返されません。単一操作属性の一覧を表示することで、明示的に要求するか、**+** を使用して結果オブジェクトの操作属性をすべて出力することができます。たとえば、この **ldapsearch** コマンドは、エントリーの通常の属性に加えて、**uid=*user_name*** がメンバーであるロールのリストを返します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b "dc=example,dc=com"
-s sub -x "(uid=user_name)"" \* nsRole

dn: uid=user_name,ou=people,dc=example,dc=com
...
nsRole: cn=Role for Managers,dc=example,dc=com
nsRole: cn=Role for Accounting,dc=example,dc=com
```

8.2.2.5. ロールの削除の概要

ロールを削除するとロールエントリーが削除されますが、各ロールメンバーの ***nsRoleDN*** 属性は削除されません。各ロールメンバーの ***nsRoleDN*** 属性を削除するには、Referential Integrity プラグインを有効にし、***nsRoleDN*** 属性を管理するように設定します。Referential Integrity プラグインの詳細は、5 章 *参照整合性の維持*を参照してください。

8.2.3. LDAP ブラウザーを使用したディレクトリーサーバーでのロールの管理

ロールは、静的グループと動的グループを統合するグループ化メカニズムです。

8.2.3.1. LDAP ブラウザーでのロールの作成

Web コンソールで **LDAP Browser** ウィザードを使用して、Red Hat ディレクトリーサーバーエントリーのロールを作成できます。

前提条件

- Web コンソールへのアクセス。
- Red Hat Directory Server に存在する親エントリー。

手順

1. Web コンソールにログインし、**Red Hat Directory Server** をクリックします。
2. Web コンソールが **Red Hat Directory Server** インターフェイスをロードしたら、**LDAP browser** をクリックします。
3. LDAP エントリーを選択し、Options メニューをクリックします。
4. ドロップダウンメニューから **New** を選択し、**Create a new role** をクリックします。
5. ウィザードの手順に従い、各手順を完了したら **Next** ボタンをクリックします。
6. ロールを作成するには、**Create Role** の手順でロールの設定を確認し、**Create** ボタンをクリックします。**Back** ボタンをクリックしてロールの設定を変更するか、**Cancel** ボタンをクリックしてロールの作成をキャンセルできます。
7. ウィザードウィンドウを閉じるには、**Finish** ボタンをクリックします。

検証

- LDAP エントリーを展開し、新しいロールがエントリーパラメーターに表示されることを確認します。

8.2.3.2. LDAP ブラウザーでのロールの変更

Web コンソールで **LDAP Browser** を使用して、Red Hat Directory Server エントリーのロールパラメーターを変更できます。

前提条件

- Web コンソールへのアクセス。
- Red Hat Directory Server に存在する親エントリー。

手順

1. Web コンソールにログインし、**Red Hat Directory Server** をクリックします。
2. Web コンソールが **Red Hat Directory Server** インターフェイスをロードしたら、**LDAP browser** をクリックします。
3. LDAP エントリーを展開し、変更するロールを選択します。
4. Options メニューをクリックし、**Edit** を選択してロールのパラメーターを変更するか、**Rename** を選択してロールの名前を変更します。
5. ウィザードウィンドウで必要なパラメーターを変更し、**LDIF Statements** の手順が表示されるまで各手順を完了して **Next** をクリックします。
6. 更新されたパラメーターを確認し、**Modify Entry** または **Change Entry Name** をクリックします。

7. ウィザードウィンドウを閉じるには、**Finish** ボタンをクリックします。

検証

- LDAP エントリーを展開し、更新されたパラメーターがロールにリストされていることを確認します。

8.2.3.3. LDAP ブラウザーでのロールの削除

Web コンソールで **LDAP Browser** を使用して、Red Hat Directory Server エントリーからロールを削除できます。

前提条件

- Web コンソールへのアクセス。
- Red Hat Directory Server に存在する親エントリー。

手順

1. Web コンソールにログインし、**Red Hat Directory Server** をクリックします。
2. Web コンソールが **Red Hat Directory Server** インターフェイスをロードしたら、**LDAP browser** をクリックします。
3. LDAP エントリーを展開し、削除するロールを選択します。
4. Options メニューを開き、**Delete** を選択します。
5. 削除するロールに関するデータを確認し、**Deletion** の手順に到達するまで **Next** ボタンをクリックします。
6. スイッチを **Yes, I'm sure** の位置に切り替え、**Delete** ボタンをクリックします。
7. ウィザードウィンドウを閉じるには、**Finish** ボタンをクリックします。

検証

- LDAP エントリーを展開し、ロールがエントリーの詳細に含まれていないことを確認します。

8.2.4. セキュアなロールの使用

すべてのロールがセキュリティーコンテキストでの使用に適しているわけではありません。新しいロールを作成するときは、そのロールをエントリーに割り当てたり、エントリーから削除したりするのがどれほど簡単かを考慮してください。ユーザーが自身をロールに簡単に追加または削除できることが適切な場合があります。たとえば、**Mountain Biking** と呼ばれるグループロールがある場合は、関心のあるユーザーは自身を追加したり、それ自体を簡単に削除したりできます。

ただし、セキュリティー状況によっては、このようなオープンなロールを持つことは不適切です。潜在的なセキュリティーリスクの 1 つは、ロールを非アクティブ化してユーザー アカウントを非アクティブ化することです。非アクティブなロールには、接尾辞に特別な ACI が定義されます。管理者により、ロールを自由に追加および削除することが許可されている場合、状況によっては、アカウントがロックされないように、非アクティブなロールから自身を削除できる場合があります。

たとえば、ユーザー A には管理ロール **MR** があります。**MR** ロールは、アカウントの非アクティブ化を使用してロックされています。これは、***nsAccountLock*** 属性はそのユーザーの **true** として計算されるため、ユーザー A はサーバーにバインドできないことを意味します。ただし、ユーザー A がすでに

Directory Server にバインドされ、MR ロールでロックされたことに気付くと、ユーザーは自分のエントリーから ***nsRoleDN*** 属性を削除して、自身を妨げる ACI がない場合は自身のロックを解除します。

ユーザーが ***nsRoleDN*** 属性を削除しないようにするには、使用されているロールのタイプに応じて以下の ACI を使用します。

- **マネージドロール。**管理ロールのメンバーであるエントリーの場合は、以下の ACI を使用して適切な ***nsRoleDN*** を削除して、ユーザーが自身をロック解除しないようにします。

  ```
  aci: (targetattr="nsRoleDN") (targattrfilters= add=nsRoleDN:(!
  (nsRoleDN=cn=AdministratorRole,dc=example,dc=com)), del=nsRoleDN:(!
  (nsRoleDN=cn=nsManagedDisabledRole,dc=example,dc=com))) (version3.0;acl "allow mod
  of nsRoleDN by self but not to critical values"; allow(write) userdn=ldap:///self;)
  ```

- **フィルターが設定されたロール。**フィルターに含まれる属性は保護されるべきです。これにより、ユーザーは属性を変更してフィルターされたロールを再取得できません。ユーザーは、フィルター処理されたロールによって使用される属性を追加、削除、または変更することを許可されるべきではありません。フィルター属性の値が計算された場合、フィルター属性の値を変更できるすべての属性を同じ方法で保護する必要があります。

- **ネストされたロール。**ネストされたロールはフィルターされたロールと管理対象のロールで設定されているため、両方の ACI はネストされたロールを設定するロールの属性 (***nsRoleDN*** またはその他) の変更について考慮する必要があります。

アカウントのアクティブ化に関する詳しい情報は、「ユーザーおよびロールの手動による非アクティブ化」を参照してください。

8.3. デュアルエントリーの自動作成

一部のクライアントおよび Red Hat Directory Server と統合には、2 つのエントリーが必要です。たとえば、Posix システムには、通常、各ユーザーにグループがあります。Directory Server の **管理エントリープラグイン** は、適切な作成元のエントリーが作成されるたびに、属性の正確な値と特定の値で新しい管理エントリーが自動的に作成されます。

8.3.1. マネージドエントリー

管理エントリープラグインの背後にある基本的な概念は、エントリー A の作成時に、関連する属性値を含むエントリー B が自動的に配置される必要があることです。たとえば、Posix ユーザー (**posixAccount** エントリー) が作成されると、対応するグループエントリー (**posixGroup** エントリー) も作成する必要があります。管理エントリープラグインのインスタンスは、プラグインが、新しいエントリー (**管理エントリー**) を自動的に生成するエントリー (**作成元のエントリー**) を識別します。

プラグインは、ディレクトリーツリーの定義範囲内で機能し、指定した検索フィルターに一致するエントリーのみが管理エントリー操作をトリガーします。

サービスのクラスを設定するのと同様に、管理エントリーは 2 つのエントリーで設定されます。

- プラグインインスタンスおよび使用するテンプレートの範囲を特定する定義エントリー
- 最終的な管理エントリーをモデル化するテンプレートエントリー

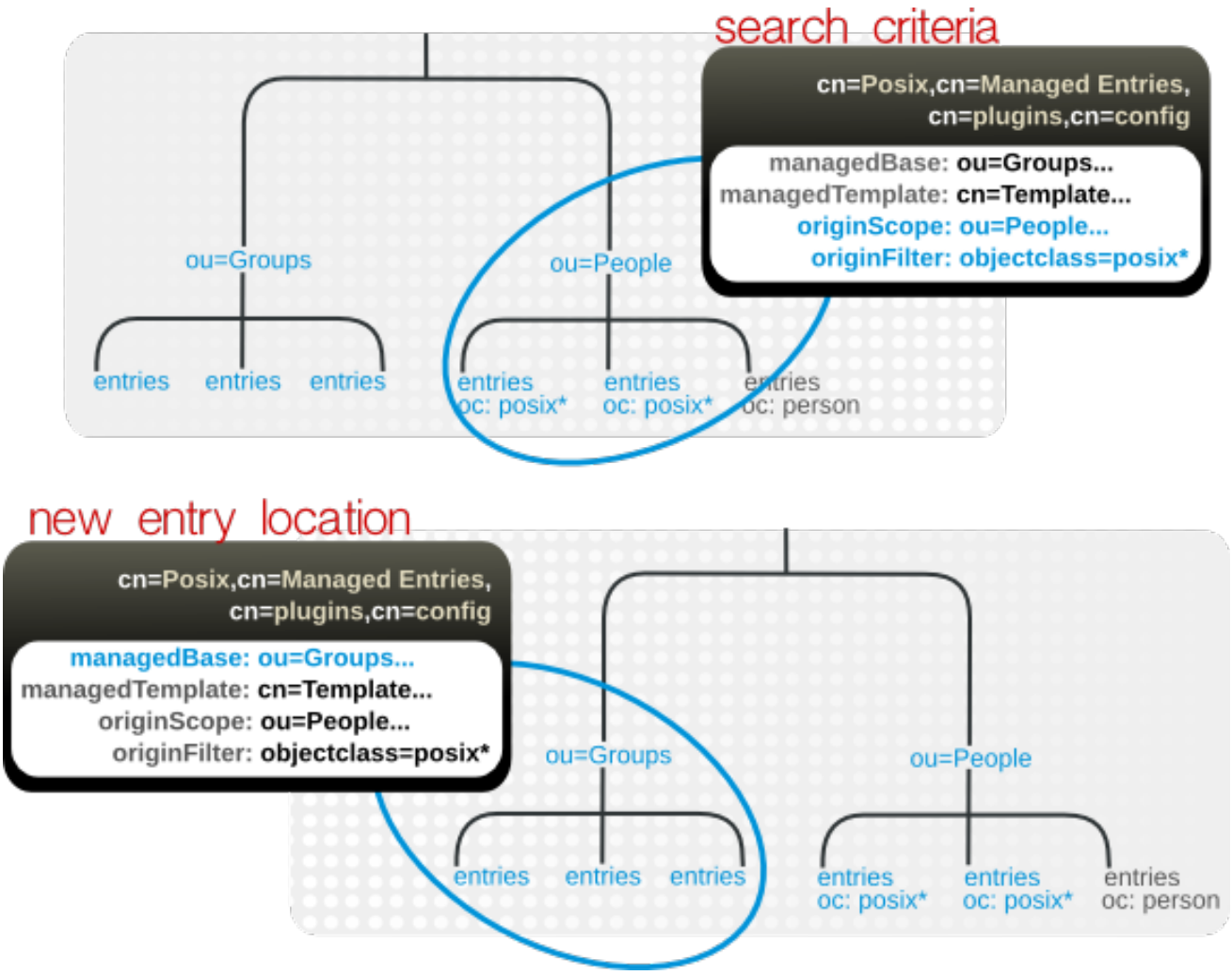
8.3.1.1. インスタンス定義エントリーの概要

リンク先属性および DNA プラグインと同様に、Managed Entries プラグインには **cn=plugins,cn=config** にコンテナーエントリーがあり、プラグインの各固有の設定インスタンスには、そのコンテナーの下に定義エントリーがあります。

管理エントリープラグインのインスタンスは、以下の 3 つを定義します。

- (検索範囲と検索フィルターを使用する) 作成元のエントリーを識別する検索基準
- 管理エントリーを作成するサブツリー (新しいエントリーの場所)
- 管理エントリーに使用するテンプレートエントリー

図8.2 管理エントリーの定義



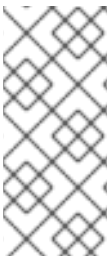
以下に例を示します。

```
dn: cn=Posix User-Group,cn=Managed Entries,cn=plugins,cn=config
objectclass: extensibleObject
cn: Posix User-Group
originScope: ou=people,dc=example,dc=com
originFilter: objectclass=posixAccount
managedBase: ou=groups,dc=example,dc=com
managedTemplate: cn=Posix User-Group Template,ou=Templates,dc=example,dc=com
```

作成元のエントリーには、管理エントリーを作成するために特別な設定または設定は必要ありません。プラグインの範囲内に作成し、指定の検索フィルターと一致させる必要があります。

8.3.1.2. テンプレートエントリーの概要

プラグインの各インスタンスは、管理エントリー設定を定義するテンプレートエントリーを使用します。テンプレートは、オブジェクトクラスからエントリーの値にエントリーを効果的に配列します。



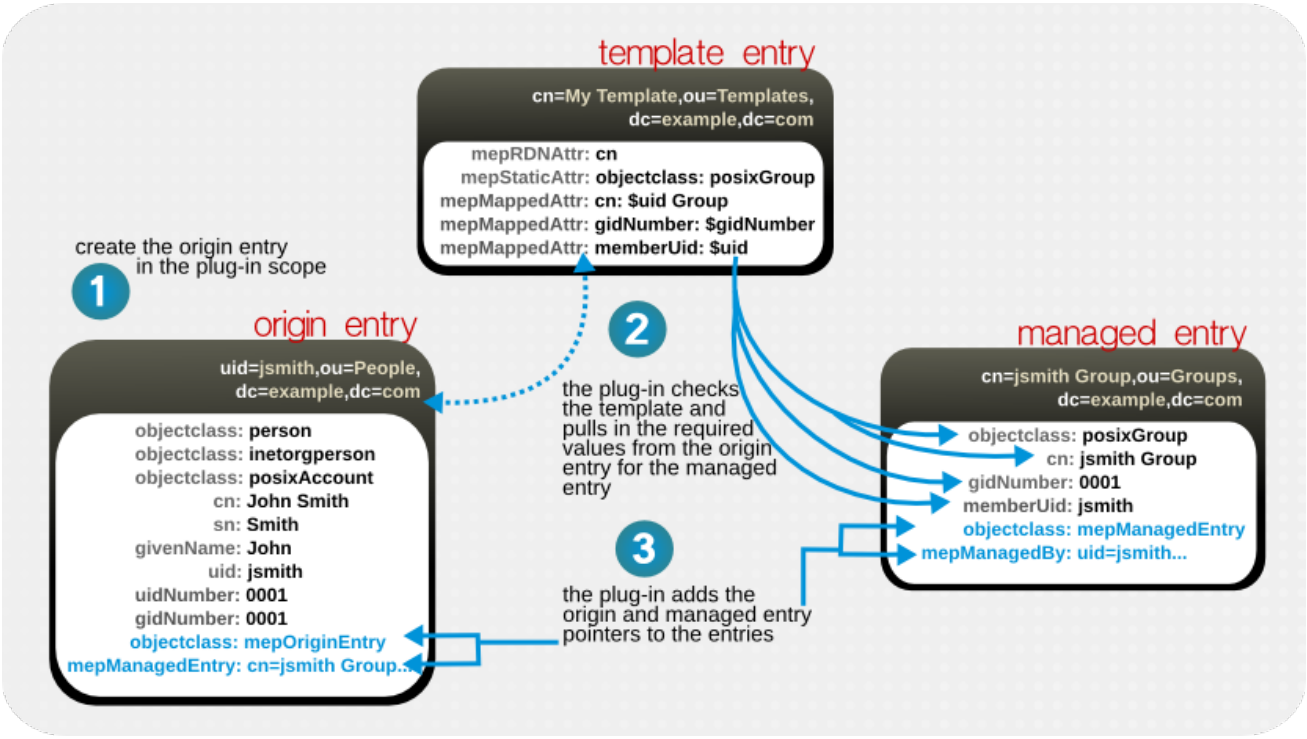
注記

テンプレートは定義エントリーで参照されるため、ディレクトリーのどこにでも配置できます。ただし、テンプレートエントリーを複製された接尾辞の下に置くことが推奨されます。これにより、マルチサプライヤーレプリケーションの他のサプライヤーはすべて、管理エントリープラグインのローカルインスタンスに同じテンプレートを使用します。

テンプレートエントリーの概念は、CoS で使用されるテンプレートに似ていますが、重要な相違点があります。マネージドエントリーテンプレートは、サービスのクラスに使用されるテンプレートのタイプとは若干異なります。サービスのクラスの場合、テンプレートには、その CoS に属するすべてのエントリーに読み込まれる特定の値を持つ 1 つの属性が含まれます。このようなエントリーの CoS 属性は、エントリーに設定した属性ではなく、仮想属性であるため、サービスのクラスへの変更は直ちに関連エントリーに反映されます。

一方、管理エントリープラグインのテンプレートエントリーは、関連するエントリーに値を提供する中央エントリーではありません。これは真のテンプレートです。エントリーの内容をレイアウトします。テンプレートエントリーには、静的属性 (CoS のように事前定義の値を持つもの) とマップされた属性 (作成元のエントリーからの値または値の一部をプルする属性) の両方を含めることができます。テンプレートは、管理エントリーが作成され、作成元のエントリーが変更され、テンプレートがこれらの更新を適用するために再度評価される場合に **のみ** 管理エントリーに適用されます。

図8.3 テンプレート、マネージドエントリー、およびアーティファクトエントリー



テンプレートは、テンプレートに **static** 属性を使用して、管理エントリーの属性に特定の値を指定できます。テンプレートは、作成元のエントリーの一部の属性から派生する値を使用することもできるため、この値はエントリーへのエントリーとは異なる場合があります。これは、値ではなく作成元のエントリーの属性タイプを参照するため、**マップされた** 属性とは異なる場合があります。

マップされた値は、トークン (動的な値) と静的な値の組み合わせを使用しますが、**マップされた属性ごとに 1 つのトークン** しか使用できません。

```
dn: cn=Posix User-Group Template,ou=Templates,dc=example,dc=com
objectclass: mepTemplateEntry
cn: Posix User-Group Template
mepRDNAttr: cn
mepStaticAttr: objectclass: posixGroup
mepMappedAttr: cn: $cn Group Entry
mepMappedAttr: gidNumber: $gidNumber
mepMappedAttr: memberUid: $uid
```

テンプレートのマップされた属性は、先頭にドル記号 ($) を追加して作成元のエントリーから値をプルして管理エントリーで使用するトークンを使用します。ドル記号が管理属性値で実際に定義されている場合は、ドル記号を 2 つ使用してドル記号をエスケープできます。

マッピングされた属性の定義は、**Attr: ${cn}test** など、中括弧で囲んで引用することができます。トークン名の直後にスペースやコンマなど属性名として有効な文字が使用される場合は、トークンの値を引用符で囲む必要はありません。たとえば、**$cn test** は属性名の直後に続くため、属性定義では使用できますが、管理エントリープラグインは作成元のエントリーで **cntest** という名前の属性を検索しようとするため、**$cntest** は有効ではありません。中括弧を使用すると、属性トークン名を識別します。

注記

静的属性およびマップされた属性に値が必要な属性の構文に準拠するようにしてください。

8.3.1.3. 管理エントリープラグインにより書き込まれるエントリー属性

作成元のエントリーと管理エントリーの両方には、管理エントリープラグインのインスタンスによって管理されていることを示す特別な管理エントリー属性があります。作成元のエントリーでは、プラグインは関連付けられた管理エントリーへのリンクを追加します。

```
dn: uid=jsmith,ou=people,dc=example,dc=com
objectclass: mepOriginEntry
objectclass: posixAccount
...
sn: Smith
mail: jsmith@example.com
mepManagedEntry: cn=jsmith Posix Group,ou=groups,dc=example,dc=com
```

管理エントリーでは、プラグインはテンプレートで定義された属性の他に、作成元のエントリーを参照する属性を追加します。

```
dn: cn=jsmith Posix Group,ou=groups,dc=example,dc=com
objectclass: mepManagedEntry
objectclass: posixGroup
...
mepManagedBy: uid=jsmith,ou=people,dc=example,dc=com
```

特別な属性を使用して、管理および作成元のエントリーを示すことで、関連するエントリーを簡単に特定し、管理エントリープラグインによる変更を評価できます。

8.3.1.4. 管理エントリープラグインおよび Directory Server 操作

管理エントリープラグインでは、追加操作および削除操作と同様に、Directory Server が一般的な操作を実行する方法に影響します。

表8.3 管理エントリープラグインおよび Directory Server 操作

演算子	マネージドエントリープラグインによる効果
Add	すべての追加操作では、サーバーは新しいエントリーが管理エントリープラグインインスタンスの範囲内にあるかどうかを確認します。作成元エントリーの基準を満たすと、管理エントリーが作成され、管理エントリー関連の属性が元のエントリーと管理エントリーの両方に追加されます。
Modify	作成元のエントリーが変更すると、管理エントリーを更新するプラグインが誘発されます。ただし、**テンプレート** エントリーを変更しても、自動的に管理エントリーを更新しません。テンプレートエントリーへの変更は、次に作成元エントリーが変更するまで、管理エントリーに反映されません。 管理エントリー **内** でマップされた管理属性は、管理エントリープラグインでのみ手動で変更することができません。マネージドエントリーの他の属性 (管理エントリープラグインで追加された静的属性を含む) は手動で変更できます。
Delete	作成元のエントリーが削除されると、管理エントリープラグインもそのエントリーに関連付けられた管理エントリーを削除します。削除できるエントリーにはいくつかの制限があります。 ● テンプレートエントリーは、プラグインインスタンス定義で現在参照されている場合は削除できません。 ● 管理エントリーは、管理エントリープラグイン以外では削除できません。
Rename	元のエントリーの名前を変更した場合は、プラグインは対応する管理エントリーを更新します。エントリーがプラグインスコープの **外** に移動すると、管理エントリーが削除されますが、エントリーがプラグインスコープの **中** に移動した場合は、追加操作のように処理され、新しい管理エントリーが作成されます。削除操作と同様に、名前変更または移動できるエントリーに制限があります。 ● 設定定義エントリーは、コンテナーエントリーの管理エントリープラグインを外に移動できません。エントリーが削除されると、そのプラグインインスタンスが非アクティブになります。 ● エントリーが管理エントリープラグインのコンテナーエントリー **内** に移動する場合、これは検証され、アクティブな設定定義として処理されます。 ● テンプレートエントリーの名前を変更したり、プラグインインスタンス定義で現在参照している場合は移動したりすることはできません。 ● 管理エントリープラグインの名前を変更または移動することはできません。
レプリケーション	管理エントリープラグイン操作は **レプリケーションの更新によって開始しません**。プラグインスコープのエントリーの追加または修正操作が別のレプリカに複製される場合、その操作はレプリカで管理エントリープラグインインスタンスを発生させず、エントリーを作成または更新しません。管理エントリーの更新を複製する唯一の方法は、最終管理エントリーをレプリカに複製することです。

8.3.2. マネージドエントリーテンプレートエントリーの作成

最初に作成するエントリーはテンプレートエントリーです。テンプレートエントリーには、生成される管理エントリーに必要なすべての設定を含める必要があります。これは、テンプレート内の静的およびマップされた属性に属性値表明を設定して行います。

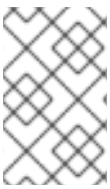
```
mepStaticAttr: attribute: specific_value
mepMappedAttr: attribute: $token_value
```

静的属性は明示的な値を設定し、マップされた属性は、元のエントリーから一部の値をプルし、指定の属性を提供するためにそれを使用します。これらの属性の値は、**attribute: $attr** のトークンになります。属性の拡張トークンの構文が必要な属性構文に違反しない限り、他の用語や文字列を属性で使用できます。以下に例を示します。

```
mepMappedAttr: cn: Managed Group for $cn
```

管理エントリーについて、その後に設定する必要がある構文ルールがいくつかあります。

- マップされた値は、トークン (動的な値) と静的な値の組み合わせを使用しますが、**マップされた属性ごとに 1 つのトークン** しか使用できません。
- テンプレートのマップされた属性は、先頭にドル記号 ($) を追加して作成元のエントリーから値をプルして管理エントリーで使用するトークンを使用します。ドル記号が管理属性値で実際に定義されている場合は、ドル記号を 2 つ使用してドル記号をエスケープできます。
- マッピングされた属性の定義は、**Attr: ${cn}test** など、中括弧で囲んで引用することができます。トークン名の直後にスペースやコンマなど属性名として有効な文字が使用される場合は、トークンの値を引用符で囲む必要はありません。たとえば、**$cn test** は属性名の直後に続くため、属性定義では使用できますが、管理エントリープラグインは作成元のエントリーで **cntest** という名前の属性を検索しようとするため、**$cntest** は有効ではありません。中括弧を使用すると、属性トークン名を識別します。
- 静的属性およびマップされた属性に値が必要な属性の構文に準拠するようにしてください。



注記

静的属性およびマップされた属性に値が必要な属性の構文に準拠するようにしてください。たとえば、マップされた属性のいずれかが ***gidNumber*** の場合、マップされた値は整数にする必要があります。

表8.4 管理エントリーテンプレートの属性

属性	説明
mepTemplateEntry (オブジェクトクラス)	テンプレートとしてエントリーを識別します。
cn	エントリーの共通名を指定します。
mepMappedAttr	プラグインは、元のエントリーから取得した値で管理エントリーの属性を作成するために使用する属性とトークンのペアが含まれます。
mepRDNAttr	管理エントリーで naming 属性として使用する属性を指定します。RDN として使用される属性は、設定のためにマップされた属性である **必要があります**。

属性	説明
mepStaticAttr	管理エントリーに指定された値とともに使用される属性と値のペアが含まれます。

テンプレートエントリーを作成するには、以下を実行します。

dsconf plugin managed-entries template add コマンドを使用して、テンプレートエントリーを追加します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin managed-entries template "cn=Posix User Template,ou=templates,dc=example,dc=com" add --rdn-attr "cn" --static-attr "objectclass: posixGroup" --mapped-attr "cn: $cn Group Entry" "gidNumber: $gidNumber" "memberUid: $uid"
```

8.3.3. マネージドエントリーインスタンス定義の作成

テンプレートエントリーが作成されると、そのテンプレートを参照する定義エントリーを作成できます。定義エントリーは、管理エントリープラグインのインスタンスです。

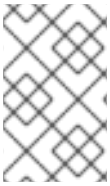
注記

定義が作成されると、サーバーは指定されたテンプレートエントリーが存在するかどうかを確認します。サーバーは、テンプレートが存在しないと、定義設定が無効であるという警告を返します。

定義エントリーは、潜在的な元のエントリーと管理エントリーを作成する情報を認識するためにパラメーターを定義する必要があります。プラグインインスタンスに使用できる属性は、表8.5「管理エントリー定義エントリーの属性」 に記載されています。

表8.5 管理エントリー定義エントリーの属性

属性名	説明
originFilter	検索に使用する検索フィルターで、管理エントリーを必要とするサブツリーのエントリーを特定します。構文は、通常の検索フィルターと同じです。
originScope	監視するプラグインの潜在的な元のエントリーを含むベースサブツリー。
managedTemplate	管理エントリーの作成に使用するテンプレートエントリーを特定します。このエントリーは、ディレクトリーツリーの任意の場所に置くことができます。
managedBase	管理エントリーを作成するサブツリー。



注記

管理エントリープラグインはデフォルトで有効です。このプラグインが無効になっている場合は、「プラグインの有効化および無効化」 で説明されているように、再度有効にします。

インスタンスを作成するには、以下を実行します。

1. **cn=Managed Entries,cn=plugins,cn=config** コンテナーエントリーの下に、新しいプラグインインスタンスを作成します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin managed-entries config "cn=instance,cn=Managed Entries,cn=plugins,cn=config" add --scope="ou=people,dc=example,dc=com" --filter="objectclass=posixAccount" --managed-base="ou=groups,dc=example,dc=com" --managed-template="cn=Posix User-Group Template,ou=Templates,dc=example,dc=com"
```

   このコマンドは、作成元のエントリー検索、新規管理エントリーの場所、使用するテンプレートエントリーの場所を設定します。

2. Directory Server が動的プラグインを有効にするために設定されていない場合は、サーバーを再起動して変更した新しいプラグインインスタンスを読み込みます。

```
# dsctl instance_name restart
```

8.3.4. 複製されたデータベースへの管理エントリープラグイン設定の追加

「マネージドエントリー」強調表示されているように、管理エントリープラグインの異なるインスタンスが、**cn=plugins,cn=com** のコンテナープラグインエントリーの下にある子として作成されます。(これは、複数のインスタンスを許可するプラグインに共通です。 この欠点は、**cn=plugins,cn=com** の設定エントリーが複製されないため、各 Directory Server インスタンスに設定を再作成する必要があります。

管理エントリープラグインエントリーでは、***nsslapd-pluginConfigArea*** 属性が許可されます。この属性は、プラグインインスタンスエントリーが含まれるメインのデータベースエリアにおける、別のコンテナーエントリーへの属性です。このコンテナーエントリーは、複製されたデータベースで使用することができます。これにより、プラグイン設定を複製できます。

1. コンテナーエントリーを作成します。たとえば、コンテナーエントリーを参照するエントリーを作成するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin managed-entries set --config-area="cn=managed entries container,ou=containers,dc=example,dc=com"
```

2. 新規コンテナーエントリーの下にある定義 (「マネージドエントリーインスタンス定義の作成」) エントリーおよびテンプレート (「マネージドエントリーテンプレートエントリーの作成」) エントリーを移動または作成します。

8.4. ビューの使用

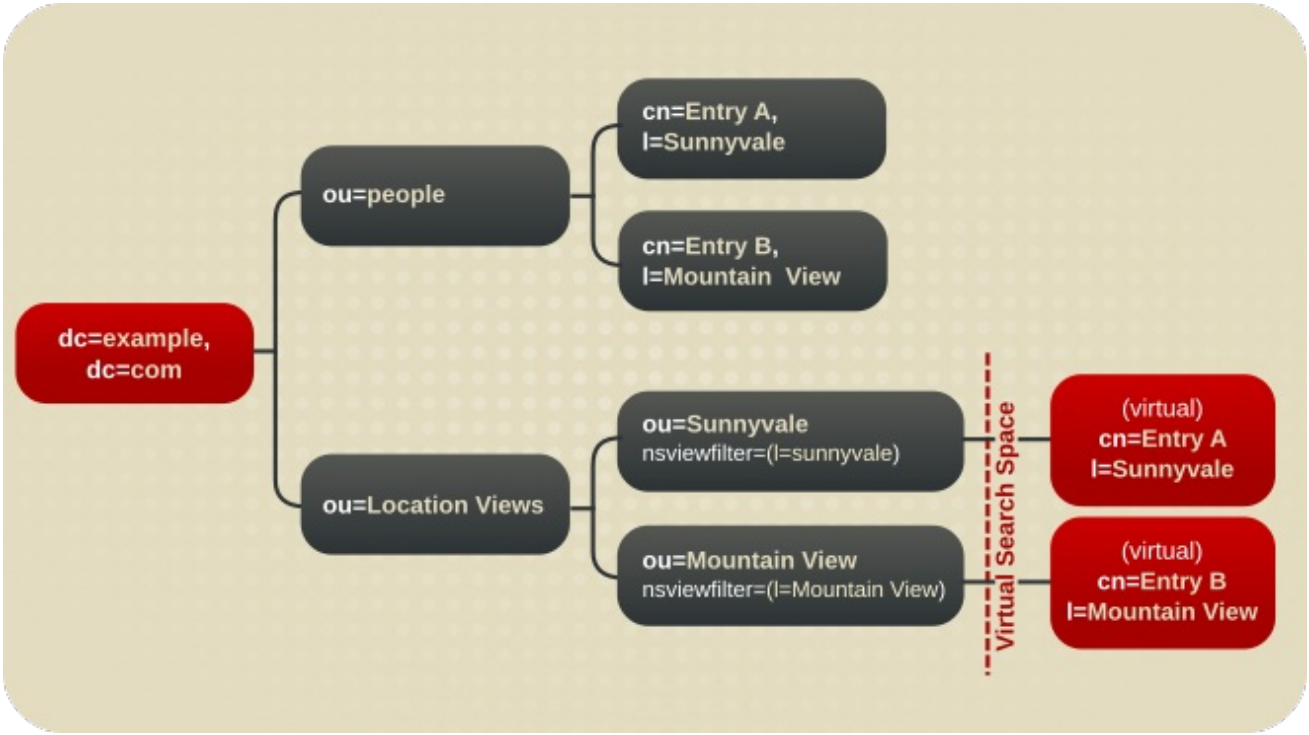
仮想ディレクトリーツリービュー、または **ビュー** は、、仮想ディレクトリー階層を作成するため、これらのエントリーが特定の場所に物理的に存在していることを確認せずに、エントリーの移動が容易になります。ビューは、フィルターされたロールまたは動的グループのメンバーと同様に、エントリーの

情報を使用してビュー階層に置きます。ビューは一連のエントリーに DIT 階層を重ね合わせ、クライアントアプリケーションには、ビューは通常のコンテナー階層として表示されます。

8.4.1. ビューの概要

ビューはサブツリーの組織単位エントリーを使用するなど、通常の階層と同様にディレクトリーツリーを作成しますが、ビューには追加のオブジェクトクラス (**nsview**) およびフィルター属性 (***nsviewfilter***) が含まれており、このビューに属するエントリーのフィルターを設定します。ビューコンテナーエントリーを追加すると、ビューフィルターと一致するすべてのエントリーが即座にビューに入力されます。対象となるエントリーは、ビューの中に存在している **ように見える** だけで、実際の場所は変わりません。たとえば、ビューは **ou=Location Views** として作成され、フィルターが **l=Mountain View** に設定されます。**cn=Jane Smith,l=Mountain View,ou=People,dc=example,dc=com** など、すべてのエントリーは **ou=Location Views** エントリーで直ちに一覧表示されますが、実際の **cn=Jane Smith** エントリーは **ou=People,dc=example,dc=com** サブツリーに残ります。

図8.4 仮想 DIT ビュー階層を含むディレクトリーツリー



仮想 DIT ビューは、サブツリーまたは 1 レベルの検索が、想定された結果で返されることで、通常の DIT と同様に動作します。

注記

Directory Server でインストールされるビューエントリーの例 **Example-views.ldif** を含む LDIF ファイルのサンプルがあります。このファイルは、**/usr/share/dirsrv/data/** ディレクトリーにあります。本章のセクションは、**Example-views.ldif** がサーバーにインポートされることを前提としています。

『Red Hat Directory Server デプロイメントガイド』 には、ディレクトリーツリー階層とビューを統合する方法の詳細が記載されています。

8.4.2. コマンドラインでのビューの作成

1. **ldapmodify** ユーティリティーを使用してサーバーにバインドし、新しいビューエントリーを設定ファイルに追加する準備を行います。

2. **Example-views.ldif** ファイルから、ビューコンテナーの **ou=Location Views,dc=example,dc=com** ファイルを、Directory Server にアサインします。この例では、root 接尾辞 **dc=example,dc=com** の下に新しい views コンテナーエントリーを追加します。このエントリーには、**nsview** オブジェクトクラスおよび ***nsViewFilter*** 属性が必要です。***nsViewFilter*** 属性は、ビューに属するエントリーを識別する属性値を設定します。

```
dn: ou=Mountain View,ou=Location Views,dc=example,dc=com
changetype: add
objectClass: top
objectClass: organizationalUnit
objectClass: nsview
ou: Mountain View
nsViewFilter: l=Mountain View
description: views categorized by location
```

8.4.3. ビューのパフォーマンスの向上

「ビューの概要」の説明通りに、ビューは指定のフィルターに基づいて検索結果から派生します。フィルターの一部は ***nsViewFilter*** 属性で定義される属性です。フィルターの残りの部分はエントリー階層に基づいており、ビューに含まれる実際のエントリーの ***entryid*** と ***parentid*** を探します。

```
(|(parentid=search_base_id)(entryid=search_base_id)
```

searched-for 属性 (***entryid***、***parentid***、または ***nsViewFilter*** に設定された属性) のいずれかがインデックス化されない場合、views 操作は一致するエントリーのツリー全体を検索するため、ビューの検索はインデックスなしの検索になります。

views パフォーマンスを改善するには、*entryid*、*parentid*、および *nsViewFilter* で設定した属性の等価インデックスを作成します。

等価インデックスの作成については「標準インデックスの作成」で説明されています。および既存のインデックスを新しい属性を含めるように更新する方法は、「既存のデータベースへの新規インデックスの作成」で説明されています。

8.5. 組織単位の管理

管理者は、ディレクトリーエントリーのコンテナーとして組織単位 (OU) を使用できます。たとえば、OU を使用して、ユーザーとグループのエントリーを分離できます。Directory Server で OU を管理するには、**dsidm organizationalunit** コマンドを使用します。

- OU を作成するには、次のコマンドを実行します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
organizationalunit create --ou OU_name
```

- エントリー内の OU をリスト表示するには、以下を入力します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
organizationalunit list
People
...
```

- OU の名前を変更するには、次のコマンドを実行します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
organizationalunit rename old_name new_name
```

- OU を削除するには、次のコマンドを実行します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
organizationalunit delete OU_name
```

第9章 セキュアな接続の設定

デフォルトでは、クライアントおよびユーザーは、標準接続で Red Hat Directory Server に接続します。標準接続では暗号化が使用されないため、サーバーとクライアントの間で情報が平文でやり取りされます。

Directory Server は、TLS 接続、**STARTTLS** 接続、および SASL 認証をサポートします。これは、傍受されていても、ディレクトリーデータを保護する暗号化およびセキュリティーの層を提供します。

9.1. セキュアな接続の要求

Directory Server は、暗号化された接続を使用する次の方法を提供します。

LDAPS

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

LDAP 上の STARTTLS

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

警告

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

SASL

Simple Authentication and Security Layer (SASL) を使用すると、Kerberos などの外部認証方法を使用してユーザーを認証できます。詳細については、「SASL Identity マッピングの設定」 を参照してください。

9.2. 最小強度係数の設定

追加のセキュリティーを確保するために、Directory Server は、接続を許可する前に特定の暗号化レベルを必要とするように設定できます。Directory Server は、すべての接続に特定のセキュリティー 強度係数 (SFF) を定義し、要求できます。SSF は、接続または操作に対するキー強度によって定義される最小限の暗号化レベルを設定します。

すべてのディレクトリー操作に最小限の SSF を必要とするには、***nsslapd-minssf*** 設定属性を設定します。最小 SSF を適用する場合、Directory Server は操作で使用可能な各暗号化タイプ (TLS または SASL) を調べ、どちらの SSF 値が高いかを判断し、高い値を最小 SSF と比較します。SASL 認証と TLS は、レプリケーションなどのサーバー間の接続に対して、SASL 認証と TLS の両方を設定できます。

注記

または、***nsslapd-minssf-exclude-rootdse*** 設定属性を使用します。これにより、ルート DSE に対するクエリーを **除き**、Directory Server へのすべての接続の最小 SSF 設定が設定されます。クライアントは、操作を開始する前に、デフォルトの命名コンテキストなどのサーバー設定に関する情報を取得しないといけない場合があります。***nsslapd-minssf-exclude-rootdse*** 属性を使用すると、クライアントは最初にセキュアな接続を確立しなくてもその情報を取得できます。

接続の最初の操作が開始すると、接続の SSF が評価されます。これにより、2 つの接続が通常の接続を開始した場合でも、**STARTTLS** および SASL バインドは成功します。TLS セッションまたは SASL セッションが開かれると、SSF が評価されます。SSF 要件を満たさない接続は、LDAP がエラーを実行することを拒否して閉じられます。

最小の SSF を設定して、セキュアでない接続がディレクトリーへの接続を無効にします。

警告

SASL を使用せずに暗号化されていない LDAP プロトコルを使用してディレクトリーに接続する場合、最初の LDAP メッセージにはバインド要求を含めることができます。この場合、SSF は設定された最小値を満たしていないため、サーバーが接続をキャンセルする前に、認証情報がネットワーク経由で暗号化されずに送信されます。

LDAPS プロトコルまたは SASL バインドを使用して、認証情報を暗号化せず送信しないようにします。

デフォルトの ***nsslapd-minssf*** 属性値は 0 です。これは、サーバー接続の最小 SSF がないことを意味します。値は、適切な正の整数に設定できます。値は、セキュアな接続に必要な鍵強度を表します。

以下の例では、***nsslapd-minssf*** パラメーターを **128** に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-minssf=128
Successfully replaced "nsslapd-minssf"
```

注記

ACI は、「接続に一定レベルのセキュリティーの要求」にあるように、特定タイプの操作に SSF を必要とするように設定できます。

「セキュアなバインドの要求」にあるように、***nsslapd-require-secure-binds*** 属性をオンにすることで、バインド操作にセキュアな接続が必要になる場合があります。

9.3. DIRECTORY SERVER が使用する NSS データベースの管理

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

/etc/dirsrv/slapd-*instance_name*/pwdfile.txt ファイルに保存します。Directory Server はこのファイルを使用しないことに注意してください。**dscreate** ユーティリティーは、管理者にパスワードを提供するためだけにこのファイルを作成しました。パスワードの変更方法は、「NSS データベースのパスワードの変更」を参照してください。

本セクションでは、Directory Server の NSS データベース管理に関する最も一般的なアクションを説明します。

9.3.1. 証明書署名要求の作成

証明書署名要求 (CSR) は、サーバーの鍵を署名するための認証局 (CA) への要求です。このセクションでは、秘密鍵を含む CSR を作成する方法を説明します。

注記

Directory Server は、**certutil** ユーティリティーを使用して NSS データベースへの秘密鍵と証明書および CSR を直接作成することのみをサポートします。

9.3.1.1. コマンドラインを使用した証明書署名要求の作成

キーおよび CSR を作成するには、**dsctl tls generate-server-cert-csr** コマンドを使用します。

```
# dsctl instance_name tls generate-server-cert-csr -s "certificate_subject"
```

dsctl tls generate-server-cert-csr コマンドは、CSR を **/etc/dirsrv/slapd-*instance_name*/Server-Cert.csr** ファイルに、秘密鍵を DDirectory Server のネットワークセキュリティーサービス (NSS) データベースに保存します。

例9.1 単一ホスト名の秘密鍵および CSR の作成

以下のコマンドは、**server.example.com** ホストのビット秘密鍵を生成します。

```
# dsctl instance_name tls generate-server-cert-csr -s
"CN=server.example.com,O=example_organization,OU=IT,ST=North Carolina,C=US"
```

-s パラメーターで指定した文字列は、RFC 1485 に従って有効なサブジェクト名である必要があります。**CN** フィールドが必要で、サーバーの完全修飾ドメイン名 (FQDN) に設定する必要があります。その他のフィールドは任意です。

例9.2 マルチホームホストの秘密鍵および CSR の作成

Directory Server ホストに複数の名前がある場合は、CSR の SAN 拡張で、すべてのホスト名を持つ CSR を作成します。以下のコマンドは、ビットの秘密鍵と、ホスト名 **server.example.com** および **server.example.net** の CSR を作成します。

```
# dsctl instance_name tls generate-server-cert-csr -s
"CN=server.example.com,O=example_organization,OU=IT,ST=North Carolina,C=US"
server.example.com server.example.net
```

最後のパラメーターとしてホスト名を指定した場合、lpmp コマンドは **DNS:server.example.com, DNS:server.example.net** エントリーで SAN 拡張を CSR に追加します。 ***-s*** パラメーターで指定した文字列は、RFC 1485 に従って有効なサブジェクト名である必要があります。**CN** フィールドが必

> 要で、サーバーの FQDN のいずれかに設定する必要があります。その他のフィールドは任意です。

CSR を生成した後、それを CA に送信し、発行された証明書を取得します。詳細は、CA のドキュメントを参照してください。

9.3.2. CA 証明書のインストール

Directory Server が認証局 (CA) を信頼できるようにするには、CA の証明書を Network Security Services (NSS) データベースにインストールする必要があります。このプロセスでは、CA が発行する証明書を信頼すべてきかどうかを設定する必要があります。

表9.1 CA 信頼オプション

Web コンソールオプション	dsconf および certutil オプション	説明
(C) 信頼できる CA	**C,,**	サーバーは、レプリケーションパートナーへの暗号化された接続を確立するために使用する証明書を検証し、信頼できる CA により発行されていることを確認します。
(T) 信頼できる CA クライアント認証	**T,,**	サーバーは、TLS **EXTERNAL** バインドに適したクライアント証明書を発行するためにこの CA 証明書を信頼します。

CA に両方のオプションを設定できます。**certutil** を使用する場合は、***-T "CT,,"*** パラメーターをユーティリティーに渡します。

9.3.2.1. コマンドラインを使用した CA 証明書のインストール

CA 証明書をインストールするには、以下を実行します。

1. CA 証明書をインポートします。たとえば、**/root/ca.crt** ファイルに保存されている CA 証明書をインポートし、**Example CA** のニックネームでデータベースに保存するには、次のコマンドを実行します。

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

2. 信頼オプションを設定します。たとえば、**CT,,** t 信頼フラグを設定するには、以下を実行します。

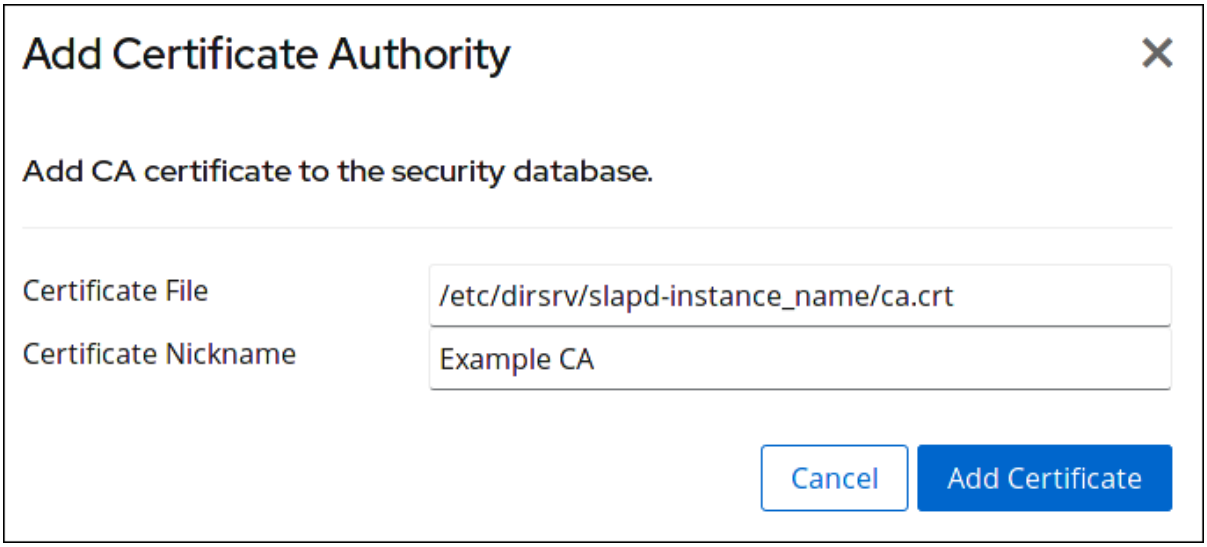
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ca-certificate set-trust-flags "Example CA" --flags "CT,,"
```

9.3.2.2. Web コンソールを使用した CA 証明書のインストール

Web コンソールを使用して CA 証明書をインストールするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** メニューを開き、**Security** エントリーを選択します。
4. **Certificate Management** タブを開き、**Trusted Certificate Authorities** サブタブを選択します。
5. **Add CA Certificate** をクリックします。
6. CA 証明書ファイルへのパスと証明書のニックネームを入力します。

図9.1 CA 証明書の追加

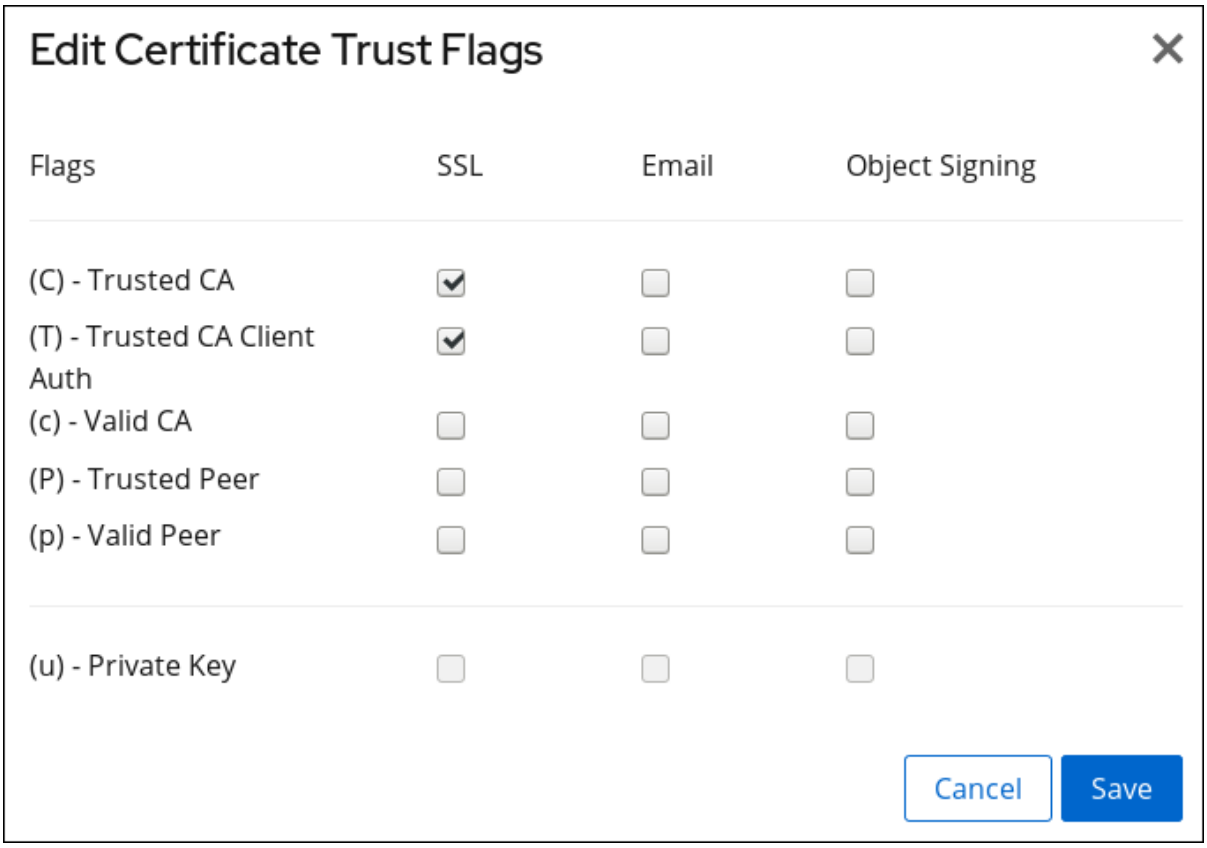


注記

CA 証明書は Directory Server ホストにローカルに保存され、**dirsrv** ユーザーが読み取り可能でなければなりません。

7. **Add Certificate** をクリックします。
8. インポートされた CA 証明書の横にある **Actions** をクリックし、**Edit Trust Flags** を選択します。
9. **SSL** 列で、**(C) - Trusted CA** および **(T)- Trusted CA Client Auth** を選択します。

図9.2 CA 証明書の信頼フラグの追加



9.3.3. 秘密鍵およびサーバー証明書のインポート

このセクションでは、外部ツールを使用して NSS データベースで秘密鍵および秘密鍵と証明書署名要求 (CSR) を作成していない場合に、秘密鍵と証明書署名要求 (CSR) の両方をインポートする方法を説明します。

NSS データベースで秘密鍵および CSR を作成した場合は、「サーバー証明書のインストール」で説明されている手順に従います。

/root/server.crt から証明書をインポートし、**/root/server.key** ファイルから秘密鍵をインポートするには、以下を入力します。

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

dsctl tls import-server-key-cert コマンドには、以下の順序でパスが必要なことに注意してください。

1. サーバー証明書へのパス。
2. 秘密鍵ファイルへのパス。

9.3.4. サーバー証明書のインストール

認証局 (CA) が要求された証明書を発行したら、Network Security Services (NSS) データベースにインストールする必要があります。

NSS データベースにない秘密鍵と証明書署名要求を作成している場合は、「秘密鍵およびサーバー証明書のインポート」で説明されている手順に従ってください。

9.3.4.1. コマンドラインを使用したサーバー証明書のインストール

Directory Server の NSS データベースにサーバー証明書をインストールするには、**certutil** ユーティリティーを使用します。以下に例を示します。

1. CA 証明書をインストールします。「CA 証明書のインストール」を参照してください。
2. サーバー証明書をインポートします。たとえば、**/root/*instance_name*.crt** ファイルに保存されている証明書をインポートし、インスタンスが使用するプライマリー証明書として設定します。

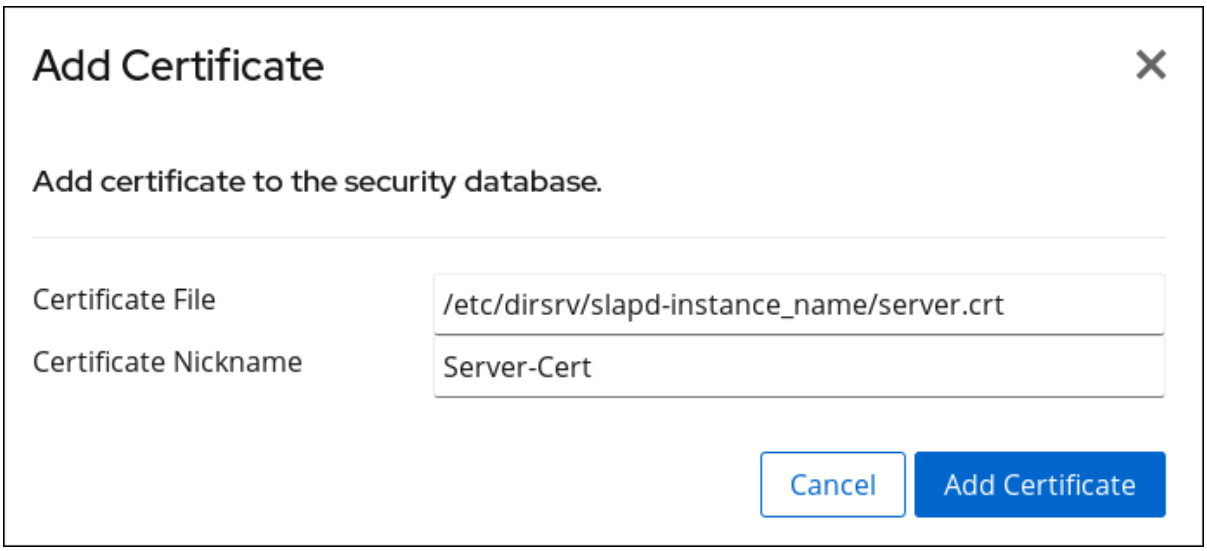
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security certificate add --file
/root/instance_name.crt --name "Server-Cert" --primary-cert
```

9.3.4.2. Web コンソールを使用したサーバー証明書のインストール

Web コンソールを使用してサーバー証明書をインストールするには、以下を実行します。

1. CA 証明書をインストールします。「CA 証明書のインストール」を参照してください。
2. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
3. インスタンスを選択します。
4. **Server Settings** メニューを開き、**Security** エントリーを選択します。
5. **Certificate Management** タブを開き、サブタブの **TLS Certificates** を選択します。
6. **Add Server Certificate** をクリックします。
7. CA 証明書ファイルのパスおよび証明書のニックネームを入力します。

図9.3 サーバー証明書の追加



注記

サーバー証明書は Directory Server ホストにローカルに保存され、**dirsrv** ユーザーが読み取り可能でなければなりません。

8. **Add Certificate** をクリックします。

9.3.5. 自己署名証明書の生成およびインストール

dscreate ユーティリティーを使用して TLS を有効にしたインスタンスを作成すると、**dscreate** が自動的に作成され、自己署名証明書がインストールされます。ただし、インスタンスの作成時に TLS を有効にしなかった場合は、手動で自己署名証明書を作成およびインストールできます。

注記

この操作は、コマンドラインを使用した場合のみ実行できます。

自己署名証明書を作成してインストールするには、以下を実行します。

1. ランダムなデータで関心のあるファイルを生成します。たとえば、サイズが 4096 ビットのあるファイルを生成するには、次のコマンドを実行します。

```
# openssl rand -out /tmp/noise.bin 4096
```

2. 自己署名証明書を作成し、NSS データベースに追加します。

```
# certutil -S -x -d /etc/dirsrv/slapd-instance_name/ -z /tmp/noise.bin \
    -n "Server-Cert" -s "CN=$HOSTNAME" -t "CT,C,C" -m $RANDOM \
    --keyUsage digitalSignature,nonRepudiation,keyEncipherment,dataEncipherment
```

Red Hat Enterprise Linux は、***$HOSTNAME*** 変数を自動的に完全修飾ドメイン名 (FQDN) に置換え、***$RANDOM*** を無作為に生成した番号に置き換えます。先のコマンドで使用したパラメーターの詳細は、certutil(1) の man ページを参照してください。

3. 必要に応じて、生成された証明書が自己署名されていることを確認します。

```
# certutil -L -d /etc/dirsrv/slapd-instance_name/ -n "Server-Cert" | egrep "Issuer|Subject"
        Issuer: "CN=server.example.com"
        Subject: "CN=server.example.com"
```

このコマンドの出力には、証明書の発行者とサブジェクトの両方について Directory Server ホストの FQDN が表示されるはずです。

9.3.6. 証明書の更新

証明書がまもなく期限切れになる場合は、セキュアな接続の確立を継続するのに期間で証明書を更新する必要があります。

9.3.6.1. コマンドラインでの証明書の更新

サーバー証明書を更新するには:

- 属性の暗号化を使用しない場合:

  1. キーサイズ、ホスト名、サブジェクトなど、同じオプションで新しい証明書署名要求 (CSR) を作成します。CSR の作成に関する詳細は、「コマンドラインを使用した証明書署名要求の作成」を参照してください。

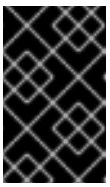
  2. CA から発行した証明書を取得したら、同じニックネームを使用してデータベースにインストールします。「コマンドラインを使用した CA 証明書のインストール」を参照してください。

3. インスタンスを停止します。

```
# dsctl instance_name stop
```

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

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



重要

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

5. インスタンスを起動します。

```
# dsctl instance_name start
```

Directory Server は、新たに発行した証明書を自動的に使用します。

- 属性の暗号化を使用する場合は、「属性暗号化に使用される TLS 証明書の更新」 を参照してください。

9.3.7. 証明書の削除

たとえば、証明書が公開されていないため、証明書が必要なくなった場合は、その証明書をデータベースから削除します。

9.3.7.1. コマンドラインで証明書の削除

コマンドラインで証明書を削除するには、以下を行います。

1. 必要に応じて、データベースの証明書を表示します。

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

Certificate Name: Server-Cert
Subject DN: CN=server.example.com
Issuer DN: CN=Example CA
Expires: 2022-07-29 11:10:14
Trust Flags: ,,
```

2. 証明書を削除します。たとえば、*Server-Cert* ニックネームで証明書を削除するには、次を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security certificate del
Server-Cert
```

9.3.7.2. Web コンソールを使用した証明書の削除

Web コンソールを使用して証明書を削除するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** メニューを開き、**Security** エントリーを選択します。
4. **Certificate Management** タブを開き、サブタブの **TLS Certificates** を選択します。
5. 証明書の横にある **Actions** をクリックし、**Delete Certificate** を選択します。
6. **Yes** をクリックします。

9.3.8. 秘密鍵の削除

たとえば、強力な鍵を作成したなどのため、秘密鍵が必要なくなった場合は、データベースから削除します。

警告

秘密鍵を削除すると、この鍵に基づく証明書は機能しなくなります。

9.3.8.1. コマンドラインでの秘密鍵の削除

秘密鍵を削除するには、次を実行します。

1. 削除する鍵に基づいてすべての証明書を削除します。「証明書の削除」を参照してください。
2. 必要に応じて、データベースのキーを表示します。

```
# certutil -d /etc/dirsrv/slapd-instance_name/ -K
certutil: Checking token "NSS Certificate DB" in slot "NSS User Private Key and Certificate Services"
Enter Password or Pin for "NSS Certificate DB":
< 0> rsa      7a2fb6c269d83c4036eac7e4edb6aaf2ed08bc4a   Server-Cert
< 1> rsa      662b826aa3dd4ca7fd7e6883558cf3866c42f4e2   example-cert
```

3. 秘密鍵を削除します。たとえば、*example-cert* ニックネームで秘密鍵を削除するには、次を実行します。

```
# certutil -d /etc/dirsrv/slapd-instance_name/ -F -n "example-cert"
```

9.3.9. CA 信頼オプションの変更

特定の状況では、認証局 (CA) の trust オプションを更新する必要があります。本セクションでは、この手順を説明します。

9.3.9.1. コマンドラインを使用した CA 信頼オプションの変更

CA の信頼オプションを変更するには、***--flags*** パラメーターの新しいオプションを **dsconf security ca-certificate set-trust-flags** コマンドに渡します。

たとえば、Directory Server が **example-CA** という名前の CA が発行するクライアント証明書のみを信頼するように設定するには、以下を実行します。

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

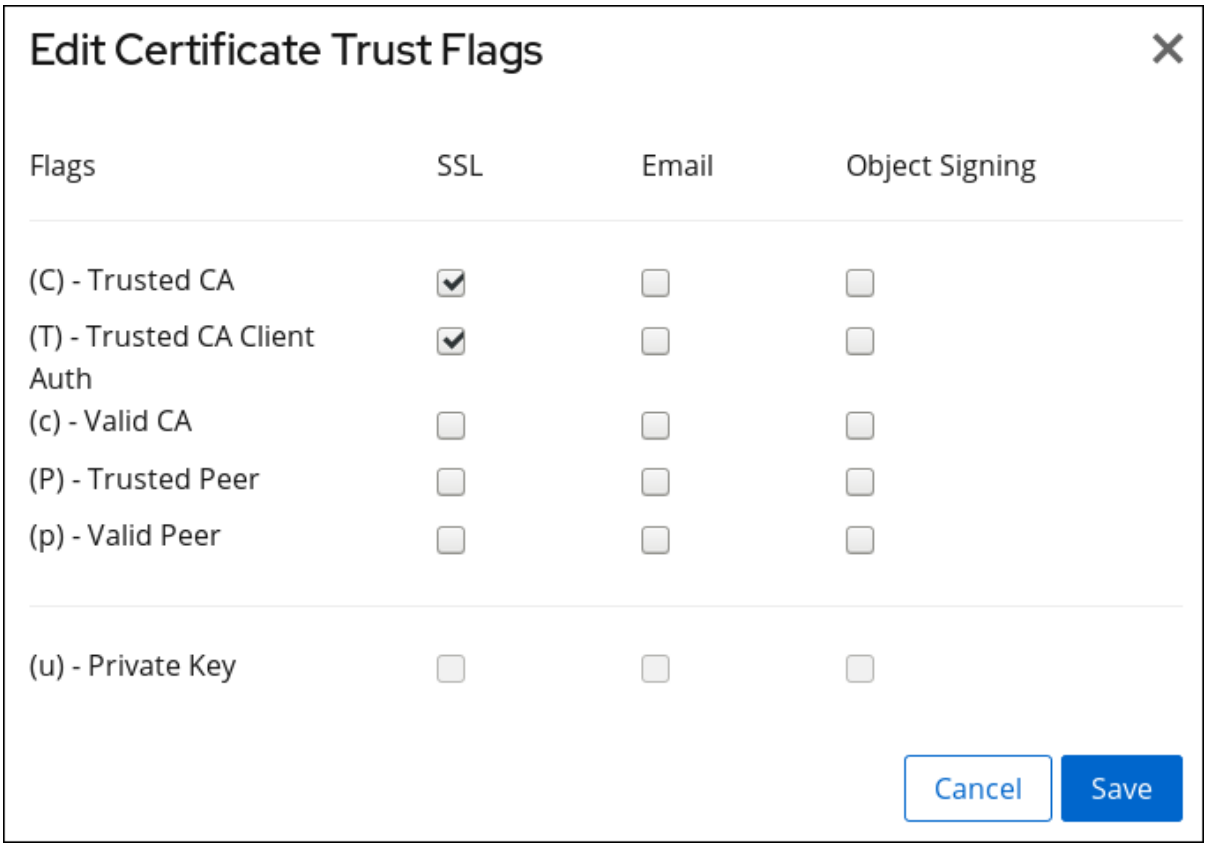
--flags trust_options パラメーターは、CA が発行し、信頼する証明書を設定します。表9.1「CA 信頼オプション」を参照してください。

9.3.9.2. Web コンソールを使用した CA 信頼オプションの変更

Web コンソールを使用して CA の信頼オプションを変更するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** メニューを開き、**Security** エントリーを選択します。
4. **Certificate Management** タブを開きます。
5. **Trusted Certificate Authorities** サブタブで、インポートした CA 証明書の横にある **Actions** をクリックし、**Edit Trust Flags** を選択します。
6. 信頼フラグを選択します。以下に例を示します。

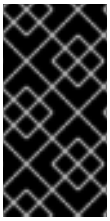
図9.4 CA 証明書の信頼フラグの設定



7. **Save** をクリックします。

9.3.10. NSS データベースのパスワードの変更

特定の状況では、管理者が Network Security Services (NSS) データベースのパスワードを変更します。本セクションでは、この手順を説明します。



重要

パスワードファイルを使用して Directory Server が Network Security Services (NSS) データベースを自動的に開くようにするには、新しいパスワードの設定後にファイルを更新する必要があります。「Directory Server のパスワードファイルの作成」を参照してください。

9.3.10.1. コマンドラインを使用した NSS データベースのパスワードの変更

NSS データベースのパスワードを変更するには、次を実行します。

```
# certutil -d /etc/dirsrv/slapd-instance_name -W
Enter Password or Pin for "NSS Certificate DB":
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:
Re-enter password:
Password changed successfully.
```

9.4. TLS の有効化

Directory Server は、クライアントとサーバーとの間で暗号化された接続と、レプリケーション環境でののサーバー間の暗号化をサポートします。このため、Directory Server は以下に対応します。

- LDAPS プロトコル: TLS 暗号化は接続が確立された後に直接使用されます。
- LDAP プロトコルの **STARTTLS** コマンド: 接続は、クライアントが **STARTTLS** コマンドを送信するまで暗号化されません。

重要

セキュリティー上の理由から、Red Hat は TLS 暗号化を有効にすることを推奨します。

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

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

9.4.1. Directory Server での TLS の有効化

本セクションでは、Directory Server で TLS を有効にする方法を説明します。

9.4.1.1. コマンドラインを使用した Directory Server での TLS の有効化

コマンドラインで TLS を有効にするには、以下を実行します。

1. 証明書を要求してインストールします。
   - 認証局 (CA) が発行する証明書の場合:
     1. Certificate Signing Request (CSR) を生成します。「コマンドラインを使用した証明書署名要求の作成」を参照してください。
     2. CA 証明書をインポートします。「コマンドラインを使用した CA 証明書のインストール」を参照してください。
     3. CA が発行するサーバー証明書をインポートします。「コマンドラインを使用したサーバー証明書のインストール」を参照してください。
   - 自己署名証明書は、「自己署名証明書の生成およびインストール」を参照してください。
2. TLS を有効にし、LDAPS ポートを設定します。

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

3. NSS データベースでサーバー証明書名を表示します。

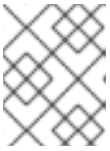
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security certificate list
```

```
Certificate Name: Server-Cert
Subject DN: CN=server.example.com
Issuer DN: CN=Example CA
Expires: 2022-07-29 11:10:14
Trust Flags: ,,
```

次の手順でニックネームが必要です。

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

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



注記

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

5. 必要に応じて、Directory Server がサポートする暗号化のリストを更新します。詳細については、「コマンドラインを使用した Directory Server が使用する暗号の表示および設定」 を参照してください。

6. 必要に応じて、証明書ベースの認証を有効にします。詳細については、「証明書ベースのクライアント認証の使用」 を参照してください。

7. 必要に応じて、パスワードファイルを作成して、NSS データベースのパスワードを要求せずに Directory Server が起動するようにします。詳細については、「Directory Server のパスワードファイルの作成」 を参照してください。

8. Directory Server インスタンスを再起動します。

```
# dsctl instance_name restart
```

NSS データベースにパスワードを設定し、パスワードファイルを作成しないと、Directory Server は NSS データベースのパスワードを要求します。詳細については、「パスワードファイルなしで Directory Server の起動」 を参照してください。

9.4.1.2. Web コンソールを使用した Directory Server での TLS の有効化

Web コンソールを使用して Directory Server で TLS を有効にするには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。

2. インスタンスを選択します。

3. CSR を作成します。「証明書署名要求の作成」を参照してください。

4. 認証局 (CA) 証明書をインポートします。「Web コンソールを使用した CA 証明書のインストール」を参照してください。

5. CA が発行するサーバー証明書をインポートします。「Web コンソールを使用したサーバー証明書のインストール」を参照してください。

6. **Server Settings** メニューを開き、**Security** エントリーを選択します。
7. **Security Configuration** タブで以下を行います。
   a. **Security Enabled** をクリックします。
   b. **Server Certificate Name** フィールドで証明書のニックネームを選択します。
   c. 必要に応じて、サーバーが対応する最小および最大の TLS バージョンの設定を変更します。
   d. 必要に応じて、クライアントが証明書を使用して認証できるように、クライアント認証を設定します。詳細については、「証明書ベースのクライアント認証の使用」 を参照してください。
8. **Save Configuration** をクリックします。
9. 必要に応じて、パスワードファイルを作成して、NSS データベースのパスワードを要求せずに Directory Server が起動するようにします。詳細については、「Directory Server のパスワードファイルの作成」 を参照してください。
10. Directory Server インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

    NSS データベースにパスワードを設定し、パスワードファイルを作成しないと、Directory Server は NSS データベースのパスワードを要求します。詳細については、「パスワードファイルなしで Directory Server の起動」 を参照してください。

9.4.1.3. 暗号化暗号の設定

Directory Server は異なる暗号に対応し、その暗号化を有効または無効にできます。暗号化は、暗号化で使用されるアルゴリズムです。クライアントがサーバーとの TLS 接続を開始すると、クライアントは情報の暗号化を好む暗号をサーバーに指示します。サーバーがこれらの暗号のいずれかに対応する場合は、このアルゴリズムを使用して暗号化された接続を確立できます。

「TLS の有効化」に従って暗号化を有効にすると、Directory Server が使用する暗号化を表示および更新できます。

9.4.1.3.1. デフォルトの暗号の表示

cn=encryption,cn=config エントリーで ***nsSSL3Ciphers*** パラメーターが設定されていない場合、Directory Server は Network Security Service (NSS) のデフォルトの暗号を使用します。デフォルトの暗号を表示するには、次のコマンドを実行します。

```
# /usr/lib64/nss/unsupported-tools/listsuites | grep -B1 --no-group-separator "Enabled"
TLS_AES_128_GCM_SHA256:
  0x1301 TLS 1.3 TLS 1.3 AES-GCM  128 AEAD   Enabled  FIPS Domestic
TLS_CHACHA20_POLY1305_SHA256:
  0x1303 TLS 1.3 TLS 1.3 CHACHA20POLY1305 256 AEAD   Enabled       Domestic
...
```

9.4.1.3.2. コマンドラインを使用した Directory Server が使用する暗号の表示および設定

利用可能なすべての暗号の表示
Directory Server で対応可能なすべての暗号のリストを表示するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ciphers list --supported
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
...
```

これは、有効または無効にできる暗号だけ含まれるリストになります。このリストには、Directory Server が現在使用している暗号は表示されません。

使用する暗号ディレクトリーサーバーの表示

Directory Server が現在使用中の暗号を表示するには、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ciphers list --enabled
TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
...
```

さらに、有効/無効にするように設定された暗号を表示できます。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ciphers list
default
+tls_rsa_aes_128_sha
+tls_rsa_aes_256_sha
...
```

default キーワードは、NSS が提供する推奨のデフォルト暗号を参照します。「デフォルトの暗号の表示」を参照してください。

重要

Directory Server は、***nsSSL3Ciphers*** 属性からの設定を使用して、実際に使用されている暗号の一覧を生成します。ただし、***nsSSL3Ciphers*** で弱い暗号化を有効にし、***allowWeakCiphers*** パラメーターをデフォルトの **off** に設定した場合、Directory Server は強力な暗号化のみを使用し、***nsSSLSupportedCiphers*** 読み取り専用属性に表示します。

有効な暗号リストの更新

有効な暗号のリストを更新するには、次のコマンドを実行します。

1. 現在有効な暗号のリストを表示します。「使用する暗号ディレクトリーサーバーの表示」を参照してください。
2. 特定の暗号のみを有効にするには、***nsSSL3Ciphers*** 属性を更新します。たとえば、**TLS_RSA_WITH_AES_128_GCM_SHA256** 暗号のみを有効にするには、次のコマンドを実行します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com security ciphers set "-all,+TLS_RSA_WITH_AES_128_GCM_SHA256"
   ```

3. Directory Server インスタンスを再起動します。

   ```
   # dsctl instance_name restart
   ```

4. 必要に応じて、有効な暗号のリストを表示して、結果を確認します。「使用する暗号ディレクト

4. 必要に応じて、有効な暗号のリストを表示して、結果を確認します。「使用する暗号アイレクトリーサーバーの表示」を参照してください。

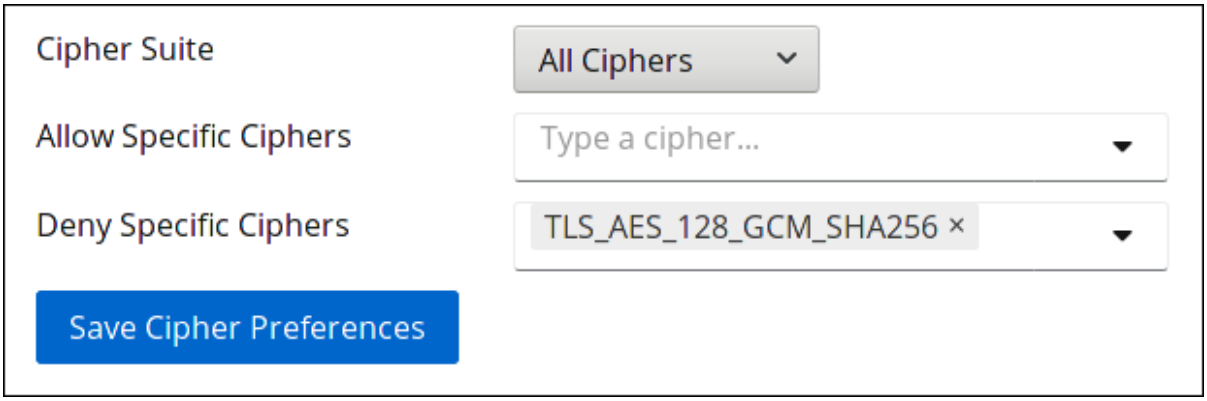
9.4.1.3.3. Web コンソールを使用した Directory Server が使用する暗号の表示および設定

Web コンソールを使用して暗号を選択し、必要に応じて更新するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** メニューを開き、**Security** エントリーを選択します。
4. **Cipher Preferences** タブで、Directory Server では、現在有効な暗号が表示されます。
5. デフォルトとは異なる暗号を使用する場合は、**Ciphers Suite** フィールドで **Default Ciphers** を選択し、デフォルトの暗号化を自動的に有効にします。詳細については、「デフォルトの暗号の表示」 を参照してください。

   または、**暗号スイート** を以下に設定できます。

   - すべての暗号を有効にする **All Ciphers**。必要に応じて、**Deny Specific Ciphers** フィールドで特定の暗号を無効にします。
   - すべての暗号を無効にする場合は **No Ciphers**。必要に応じて、**Allow Specific Ciphers** フィールドで特定の暗号を有効にします。



6. **Save Cipher Preferences** をクリックします。
7. 暗号のリストを更新した場合は、Directory Server インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

9.4.1.4. パスワードファイルなしで Directory Server の起動

暗号化を有効にし、NSS データベースに設定したパスワードを使用して Directory Server を起動する場合は、以下を行います。

- **systemctl** コマンドで **ns-slapd** Directory Server プロセスが起動すると、**systemd** はパスワードを求めるプロンプトを表示し、その入力内容を **systemd-tty-ask-password-agent** ユーティリティーに自動的に渡します。以下に例を示します。

```
# systemctl start dirsrv@instance_name
Enter PIN for Internal (Software) Token:
```

- まれに、**ns-slapd** Directory Server プロセスが **systemctl** ユーティリティーにより開始されず、ターミナルから切り離されていると、**wall** コマンドを使用してすべての端末にメッセージを送信します。以下に例を示します。

  ```
  Broadcast message from root@server (Fri 2017-01-01 06:00:00 CET):

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

  パスワードを入力するには、次を実行します。

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

9.4.1.5. Directory Server のパスワードファイルの作成

暗号化が有効で、NSS データベースに設定したパスワードがあると、サービスの起動時に Directory Server はこのパスワードを要求します。「パスワードファイルなしで Directory Server の起動」 を参照してください。

このプロンプトを省略するには、NSS データベースパスワードを **/etc/dirsrv/slapd-*instance_name*/pin.txt** ファイルに保存できます。これにより、このパスワードを要求せずに Directory Server が自動的に起動できます。

警告

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

パスワードファイルを作成するには、以下を実行します。

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

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

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

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

     ```
     name_of_the_token:password
     ```

2. パーミッションを設定します。

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

9.4.1.6. 証明書の有効期限が切れた場合の Directory Server の動作の管理方法

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

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

例9.3 証明書の有効期限が切れると Directory Server が起動しないようにする

証明書の有効期限が切れている場合は Directory Server が起動しないようにするには、次を実行します。

1. ***nsslapd-validate-cert*** パラメーターを **on** に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-validate-cert=on
Successfully replaced "nsslapd-validate-cert"
```

2. Directory Server インスタンスを再起動します。

```
# dsctl instance_name restart
```

9.4.2. Directory Server が使用する CA 証明書の Red Hat Enterprise Linux のトラストストアへの追加

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

例9.4 クライアントユーティリティーが CA 証明書を使用しない場合の接続エラーの可能性

クライアントユーティリティーが CA 証明書を使用しない場合、ユーティリティーは TLS 暗号化の使用時にサーバーの証明書を検証できません。これにより、サーバーへの接続に失敗します。以下に例を示します。

- **dsconf**

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

- **ldapsearch**

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

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

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

   a. サーバーの NSS データベースの証明書をリスト表示します。

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

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

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

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

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

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

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

3. CA 信頼データベースを再構築します。

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

9.5. DIRECTORY SERVER で有効な暗号化プロトコルの表示

Directory Server で有効な暗号化プロトコルを表示するには、次のコマンドを実行します。

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

sslVersionMin パラメーターおよび ***sslVersionMax*** パラメーターは、Directory Server が使用する暗号化プロトコルを制御します。***sslVersionMin*** のデフォルトは、使用するシステム全体の暗号化ポリシーによって異なります。

9.6. 最小 TLS 暗号化プロトコルバージョンの設定

デフォルトでは、Directory Server は、システム全体の暗号化ポリシーに基づいて ***sslVersionMin*** パラメーターを自動的に設定します。以下の表は、システム全体の暗号化ポリシープロファイルを基に Directory Server が使用する TLS バージョン ***sslVersionMin*** の概要を示しています。

表9.2 定義するシステム全体の暗号化ポリシープロファイルと、これらの最小 TLS バージョンの概要

プロファイル	最小の TLS バージョン
DEFAULT	TLS 1.2
FUTURE	TLS 1.2
FIPS	TLS 1.2
LEGACY	TLS 1.0

システム全体の暗号化ポリシー、プロファイルの変更方法、およびシステム全体の暗号化ポリシーのオプトアウトサービスの詳細は、『RHEL 8 セキュリティーの強化』 の 『システム全体の暗号化ポリシーの使用』 を参照してください。

または、***sslVersionMin*** は、暗号化ポリシープロファイルで定義された値よりも高い値に手動で設定できます。

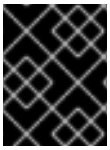
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security set --tls-protocol-min="TLS1.3"
```

9.7. 最も大きい TLS 暗号化プロトコルバージョンの設定

TLS プロトコルの中で Directory Server がサポートする一番高いバージョンを設定するには、以下を入力します。

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

パラメーターを ***sslVersionMin*** 未満の値に設定すると、Directory Server は ***sslVersionMax*** を ***sslVersionMin*** と同じ値に設定します。



重要

sslVersionMax パラメーターでサポートされる最強の暗号化プロトコルバージョンを常に使用するには、このパラメーターを設定しないでください。

9.8. ハードウェアセキュリティーモジュールの使用

セキュリティーモジュールは、Directory Server と TLS レイヤーとの間のメディアとして機能します。モジュールは、暗号化および復号に使用される鍵および証明書を保存します。これらのモジュールを定義する標準は Public Key Cryptography Standard (PKCS) #11 で、これらのモジュールは PKCS#11 モジュールです。

デフォルトでは、Directory Server はビルトインのセキュリティーデータベース **key4.db** と **cert9.db** を使用して、サーバーが使用する鍵と証明書を保存します。

外部のセキュリティーデバイスを使用して Directory Server 証明書および鍵を保存することもできます。Directory Server が外部の PKCS#11 モジュールを使用するには、モジュールのドライバーを Directory Server にインストールする必要があります。

詳細は、ハードウェアのセキュリティーモジュールのドキュメントを参照してください。

9.9. 証明書ベースのクライアント認証の使用

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

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

証明書ベースの認証を使用する利点:

- 効率が改善されました。証明書データベースのパスワードに一度要求されたアプリケーションを使用し、その証明書を後続のバインドまたは認証操作に使用すると、バインド DN およびパスワードを継続的に提供するよりも効率的です。
- セキュリティーが改善されました。証明書ベースの認証は、証明書ベースの認証では公開鍵の暗号化が使用されるため、証明書以外のバインド操作よりも安全です。バインド認証情報はネットワーク全体で傍受することはできません。証明書やデバイスが失われた場合は、PIN なしで使用しないため、フィッシング攻撃などのサードパーティーの干渉の影響を受けません。

9.9.1. 証明書ベースの認証の設定

証明書ベースの認証を有効にするには、以下を行います。

1. 暗号化された接続を有効にします。詳細については、「TLS の有効化」 を参照してください。
2. CA 証明書をインストールし、クライアントとサーバーの接続の信頼オプションを設定します。「CA 証明書のインストール」を参照してください。
3. 必要に応じて、クライアントおよびサーバーの **CT,,** 信頼オプションが CA 証明書に設定されていることを確認します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com security ca-certificate get
"Example-CA"
Certificate Name: Example-CA
Subject DN: CN=server.example.com,ST=Queensland,C=AU
Issuer DN: CN=server.example.com,,ST=Queensland,C=AU
Expires: 2021-05-09 10:57:54
Trust Flags: CT,,
```

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

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

```
default:VerifyCert      on

certmap example         o=Example Inc.,c=US
example:DNComps
```

これは、この発行者には ***DNComps*** パラメーターが空に設定されているため、**o=Example Inc.,c=US** 発行者識別名 (DN) セットを持つ証明書を使用するユーザーを認証するため、Directory Server が証明書のサブジェクトからベース DN を生成しないように設定されています。また、***FilterComps*** および ***VerifyCert*** の設定も、デフォルトのエントリーから継承されます。

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

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

利用可能なパラメーターの詳細と説明は、『Red Hat Directory Server Configuration, Command, and File Reference』の **certmap.conf** ファイルの説明を参照してください。

5. クライアント認証を有効にします。たとえば、クライアント認証を任意に設定するには、以下を実行します。

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

また、***--tls-client-auth*** パラメーターを **required** に設定して、クライアントが認証に使用する必要のある証明書を設定します。

6. **/etc/dirsrv/slapd-*instance_name*/certmap.conf** ファイルで ***alias_name*:VerifyCert on** を設定して、認証証明書がユーザーの ***userCertificate*** 属性に保存されている証明書と一致する必要がある場合は、その証明書をユーザーエントリーに追加します。「ユーザーへの証明書の追加」を参照してください。

9.9.2. ユーザーへの証明書の追加

証明書ベースの認証を設定する際に、認証に使用する証明書が、ユーザーの ***userCertificate*** バイナリー属性に保存されている証明書と一致する必要があるように設定できます。**/etc/dirsrv/slapd-*instance_name*/certmap.conf** ファイルに ***alias_name*:VerifyCert on** を設定してこの機能を有効にした場合は、影響を受けるユーザーの証明書をディレクトリーエントリーに追加する必要があります。

重要

証明書を、***userCertificate*** 属性の識別名エンコーディングルール (DER) 形式で保存する必要があります。

ユーザーの ***userCertificate*** 属性に証明書を保存するには、以下を行います。

1. 証明書が DER 形式ではない場合は、これを変換します。以下に例を示します。

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

2. 証明書をユーザーの ***userCertificate*** 属性に追加します。以下に例を示します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

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

バイナリー属性の使用に関する詳細は、「Binary 属性の使用」を参照してください。

9.9.3. バインドリクエストの EXTERNAL SASL メカニズムの強制

TLS セッションの開始時に、クライアントは証明書をサーバーに送信します。次に、バインド要求を送信します。ほとんどのクライアントは、SASL メカニズム **EXTERNAL** を使用してバインド要求を実行します。これは、バインド要求の認証情報ではなく、バインドの証明書で ID を使用する必要があることを Directory Server に通知します。

ただし、クライアントが簡単な認証または匿名の認証情報を使用する場合は、この情報がありません。この場合は、証明書および証明書のクライアント ID が有効であっても、TLS セッションが無効な認証情報で失敗します。

Directory Server を設定してクライアントが SASL メカニズム **EXTERNAL** を使用し、要求内の他のバインドメソッドを無視するには、以下を行います。

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

9.9.4. 証明書を使用した認証

OpenLDAP クライアントツールを使用して、証明書を使用して認証に対応する Directory Server インスタンスに対して認証します。

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

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

   あるいは、 **~/.ldaprc** ファイルに ***TLS_CACERT*** パラメーター、***TLS_KEY*** パラメーター、および ***TLS_CERT*** パラメーターを設定します。詳細は、ldap.conf(5) の man ページの 『TLS OPTIONS』 セクションを参照してください。

2. サーバーに接続します。以下に例を示します。

```
# ldapwhoami -H ldaps://server.example.com:636
```

別のクライアントを使用する場合は、証明書ベースの認証を使用して接続する方法は、クライアントアプリケーションのドキュメントを参照してください。

9.10. SASL IDENTITY マッピングの設定

Simple Authentication and Security Layer(SASL) は、LDAP などのプロトコルと GSS-API などの認証方法との間の抽象化レイヤーであり、SASL と対話できるプロトコルが SASL と連携できる認証メカニズムを利用できるようにします。簡単に言えば、SASL は、異なるメカニズムを使用して、アプリケーションに対して認証できるようにする仲介者です。SASL は、クライアントとサーバーとの間で暗号化されたセッションを確立するためにも使用できます。

SASL フレームワークでは、クライアントアプリケーションとサーバーアプリケーションの両方で有効になっているメカニズムに応じて、サーバーに対してユーザーを認証するためにさまざまなメカニズムを使用できます。SASL は、暗号化された (セキュアな) セッションのレイヤーも作成します。GSS-API を使用すると、Directory Server は Kerberos チケットを使用してセッションを認証し、データを暗号化します。

9.10.1. SASL Identity マッピングの概要

SASL バインド要求の処理時に、サーバーは、サーバー内に格納されている LDAP エントリーで Directory Server に対して認証するために使用される SASL 認証 ID を照合またはマップします。Kerberos を使用する場合、通常 SASL ユーザー ID の形式は **userid@REALM** になります (例: **scarter@EXAMPLE.COM**)。この ID は、**uid=scarter,ou=people,dc=example,dc=com** など、ユーザーの Directory Server エントリーの DN に変換する必要があります。

認証 ID が個人の LDAP エントリーに明確に対応する場合は、Directory Server が認証 ID を自動的にエントリー DN にマッピングするように設定できます。Directory Server には、最も一般的な設定を処理する事前設定済みのデフォルトマッピングがいくつかあり、カスタマイズされたマップを作成できます。デフォルトでは、バインド試行時に SASL マッピングフォールバックが有効ではない場合は、最初に一致するマッピングルールのみが適用されます。SASL マッピングフォールバックの詳細は、「SASL マッピングフォールバックの有効化」を参照してください。

1 つのマッピングルールのみが認証文字列と一致するように、SASL マップを設定するようにしてください。

SASL マッピングは、コンテナーエントリー下のエントリーによって設定されます。

```
dn: cn=sasl,cn=config
objectClass: top
objectClass: nsContainer
cn: sasl
```

SASL アイデンティティーマッピングエントリーは、以下のエントリーの子です。

```
dn: cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsContainer
cn: mapping
```

マッピングエントリーは以下の属性で定義されます。

- ***nsSaslMapRegexString***: 指定した ***authid*** の要素をマップするために使用される正規表現。
- ***nsSaslMapFilterTemplate***: DN を作成する ***nsSaslMapRegexString*** の要素を適用するテンプレート。
- ***nsSaslMapBaseDNTemplate***: 構築した DN と照合する検索ベースまたは特定のエントリー DN を指定します。

- オプション: ***nsSaslMapPriority***: この SASL マッピングの優先度を設定します。 ***nsslapd-sasl-mapping-fallback*** が ***cn=config*** で有効になっている場合は、優先度値が使用されます。詳細については、「SASL マッピングの優先度の設定」 を参照してください。

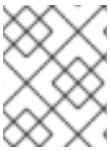
詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の該当するセクションを参照してください。

以下に例を示します。

```
dn: cn=mymap,cn=mapping,cn=sasl,cn=config
objectclass:top
objectclass:nsSaslMapping
cn: mymap
nsSaslMapRegexString: \(.*\)@\(.*\)\.\(.*\)
nsSaslMapFilterTemplate: (objectclass=inetOrgPerson)
nsSaslMapBaseDNTemplate: uid=\1,ou=people,dc=\2,dc=\3
```

nsSaslMapRegexString 属性は、検索中にテンプレート属性に埋め込まれたバインド ID に対し、**\1**、**\2**、**\3** 形式の変数を設定します。この例では、**inetOrgPerson** オブジェクトクラスに属する **ou=People,dc=example,dc=com** サブツリーに含まれるユーザーに対して SASL アイデンティティーマッピングを設定します。

Directory Server が、**mconnors@EXAMPLE.COM** をユーザー ID (***authid***) として使用する SASL バインド要求を受け取ると、正規表現は **uid=mconnors,ou=people,dc=EXAMPLE,dc=COM** をユーザー ID として使用するベース DN テンプレートに入力し、認証がそこから続行します。



注記

dc の値は大文字と小文字を区別しないため、**dc=EXAMPLE** と **dc=example** は同じです。

Directory Server では、以下のようなより包含されたマッピングスキームも使用できます。

```
dn: cn=example map,cn=mapping,cn=sasl,cn=config
objectclass: top
objectclass: nsSaslMapping
cn: example map
nsSaslMapRegexString: \(.*\)
nsSaslMapBaseDNTemplate: ou=People,dc=example,dc=com
nsSaslMapFilterTemplate: (cn=\1)
```

これは任意のユーザー ID に一致し、フィルター **cn=userId** を満たす **ou=People,dc=example,dc=com** サブツリー下でエントリーをマップします。

nsSaslMapRegexString 属性にレルムを指定すると、マッピングを 1 つのレルムに制限することができます。以下に例を示します。

```
dn: cn=example map,cn=mapping,cn=sasl,cn=config
objectclass: top
objectclass: nsSaslMapping
cn: example map
nsSaslMapRegexString: \(.*\)@US.EXAMPLE.COM
nsSaslMapBaseDNTemplate: ou=People,dc=example,dc=com
nsSaslMapFilterTemplate: (cn=\1)
```

このマッピングは以前のマッピングと同じですが、**US.EXAMPLE.COM** レルムから認証されるユーザーにのみ適用されます。(レルムは「プリンシパルおよびレルムについて」で説明されています。)

レプリケーション時やチェーンでなど、別のサーバーに接続する場合、デフォルトのマッピングはアイデンティティーを適切にマッピングしません。これは、あるサーバーのプリンシパル (SASL ID) が、認証が実行するサーバー上のプリンシパルと一致しないため、マッピングエントリーに一致しないためです。

サーバーが SASL を使用したサーバー認証を使用できるようにするには、特定のサーバープリンシパルの特定ユーザーエントリーへのマッピングを作成します。たとえば、このマッピングは **ldap1.example.com** サーバーを **cn=replication manager,cn=config** エントリーに一致します。マッピングエントリー自体は 2 番目のサーバーに作成されます (例: **ldap2.example.com**)。

```
dn: cn=z,cn=mapping,cn=sasl,cn=config
objectclass: top
objectclass: nsSaslMapping
cn: z
nsSaslMapRegexString: ldap/ldap1.example.com@EXAMPLE.COM
nsSaslMapBaseDNTemplate: cn=replication manager,cn=config
nsSaslMapFilterTemplate: (objectclass=*)
```

レルム名は、SASL GSS-API 設定のプリンシパル名に含まれていないことがあります。2 つ目のマッピングは、プリンシパル名にレルムを指定せずに、最初のマッピングと同じ 2 番目のマッピングを作成できます。以下に例を示します。

```
dn: cn=y,cn=mapping,cn=sasl,cn=config
objectclass: top
objectclass: nsSaslMapping
cn: y
nsSaslMapRegexString: ldap/ldap1.example.com
nsSaslMapBaseDNTemplate: cn=replication manager,cn=config
nsSaslMapFilterTemplate: (objectclass=*)
```

レルムが指定されていないため、2 番目のマッピングはより一般的です (つまり、最初のマッピングよりも多くのエントリーと一致する可能性があります)。ベストプラクティスは、より具体的なマッピングを最初に処理し、より一般的なマッピングに徐々に進めていく方法です。

nsSaslMapPriority パラメーターを使用して SASL マッピングに優先度が設定されていない場合は、マッピングが処理される順序を指定する方法はありません。ただし、SASL マッピングの処理方法 (名前) を制御する方法もあります。Directory Server は、ASCII の逆順で SASL マッピングを処理します。過去 2 つの例では、**cn=z** マッピング (最初の例) が最初に処理されます。一致する場合、サーバーは **cn=y** マッピングを処理します (2 番目の例)。

注記

LDIF ファイルでマッピングを指定し、**ConfigFile** ディレクティブで LDIF ファイルを追加すると、サイレントインストール中にインスタンスが作成される時に SASL マッピングを追加できます。サイレントインストールの使用方法は、『インストールガイド』で説明しています。

9.10.2. Directory Server のデフォルトの SASL マッピング

Directory Server には、最も一般的な使用法のいくつかを処理するための事前定義された SASL マッピングルールがあります。

Kerberos UID マッピング

これは、**user@example.com** などの 2 部分レルムを使用して Kerberos プリンシパルと一致します。レルムは、検索ベースの定義に使用され、ユーザー ID (***authid***) はフィルターを定義します。検索ベースは **dc=example,dc=com** と **(uid=user)** のフィルターです。

```
dn: cn=Kerberos uid mapping,cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsSaslMapping
cn: Kerberos uid mapping
nsSaslMapRegexString: \(.*\)@\(.*\)\.\(.*\)
nsSaslMapBaseDNTemplate: dc=\2,dc=\3
nsSaslMapFilterTemplate: (uid=\1)
```

RFC 2829 DN 構文

このマッピングは、**dn:** で始まる有効な DN (RFC 2829 で定義) である ***authid*** と一致します。***authid*** は、指定された DN に直接マッピングします。

```
dn: cn=rfc 2829 dn syntax,cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsSaslMapping
cn: rfc 2829 dn syntax
nsSaslMapRegexString: ^dn:\(.*\)
nsSaslMapBaseDNTemplate: \1
nsSaslMapFilterTemplate: (objectclass=*)
```

RFC 2829 U 構文

このマッピングは、**u:** の接頭辞が付いた UID の ***authid*** と一致します。接頭辞の後に指定された値は **(uid=value)** のフィルターを定義します。検索ベースは、デフォルトの **userRoot** データベースの接尾辞になるようにハードコーディングされます。

```
dn: cn=rfc 2829 u syntax,cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsSaslMapping
cn: rfc 2829 u syntax
nsSaslMapRegexString: ^u:\(.*\)
nsSaslMapBaseDNTemplate: dc=example,dc=com
nsSaslMapFilterTemplate: (uid=\1)
```

UID マッピング

このマッピングは、他のデフォルトのマッピングルールに一致しない平文文字列である ***authid*** と一致します。この値を使用して **(uid=value)** のフィルターを定義します。検索ベースは、デフォルトの **userRoot** データベースの接尾辞になるようにハードコーディングされます。

```
dn: cn=uid mapping,cn=mapping,cn=sasl,cn=config
objectClass: top
objectClass: nsSaslMapping
cn: uid mapping
nsSaslMapRegexString: ^[^:@]+$
nsSaslMapBaseDNTemplate: dc=example,dc=com
nsSaslMapFilterTemplate: (uid=&)
```

9.10.3. SASL Identity マッピングの設定

SASL (Simple Authentication and Security Layer) ID マッピングは、Directory Server またはコマンドラインから設定できます。SASL 認証に対して SASL ID を機能させるには、マッピングが 1 つ返す必要があり、一致するエントリーと Kerberos をホストマシンに設定する必要があります。

9.10.3.1. コマンドラインで SASL ID マッピングの設定

コマンドラインから SASL ID マッピングを設定するには、**dsconf** ユーティリティーを使用して ID マッピングスキームを追加します。

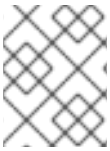
1. ID マッピングスキームを追加します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com sasl create --cn
"example_map" --nsSaslMapRegexString "\(.*\)" --nsSaslMapBaseDNTemplate
"ou=People,dc=example,dc=com" --nsSaslMapFilterTemplate "(cn=\1)" --nsSaslMapPriority
50
Successfully created example_map
```

これは、ユーザーの共通名に一致し、フィルター **cn=userId** に基づいて、ベース **ou=People,dc=example,dc=com** を用いたサブツリー検索の結果にマッピングされます。

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```



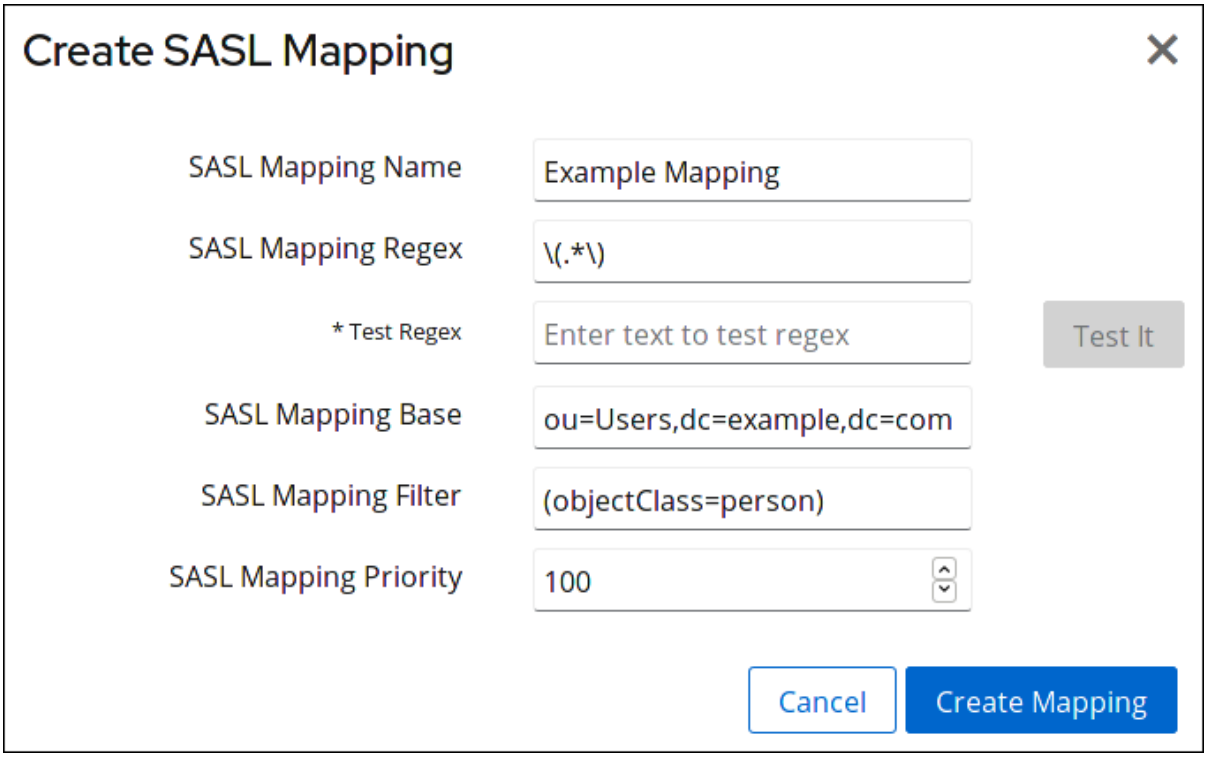
注記

dsconf で SASL マップを追加すると、ASCII 順序に関係なく、リストの最後にマッピングが追加されます。

9.10.3.2. Web コンソールを使用した SASL アイデンティティーマッピングの設定

SASL アイデンティティーマッピングスキームを追加するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** メニューを開き、**SASL Settings & Mappings** を選択します。
4. **Create New Mapping** をクリックします。
5. フォームを入力します。以下に例を示します。



6. **Save** をクリックします。

9.10.4. SASL マッピングフォールバックの有効化

デフォルト設定を使用すると、Directory Server は最初に一致する SASL マッピングのみを検証します。最初に一致するマッピングが失敗すると、バインド操作に失敗し、さらに一致するマッピングは検証されません。

ただし、***nsslapd-sasl-mapping-fallback*** パラメーターを有効にすると、Directory Server が一致するすべてのマッピングを検証するように設定できます。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-sasl-mapping-fallback=on
Successfully replaced "nsslapd-sasl-mapping-fallback"
```

フォールバックが有効であり、1 つのユーザー ID のみが返されると、バインドは成功します。ユーザーがない場合、または複数のユーザーが返されると、バインドは失敗します。

9.10.4.1. SASL マッピングの優先度の設定

nsslapd-sasl-mapping-fallback 属性を使用して SASL マッピングフォールバックを有効にすると、任意でマッピング設定の ***nsSaslMapPriority*** 属性を設定して優先順位を設定できます。***nsSaslMapPriority*** 属性は、**1** (最も高い優先度) から **100** (最も低い優先度) の値をサポートします。デフォルトは **100** です。

たとえば、**cn=Kerberos uid mapping,cn=mapping,cn=sasl,cn=config** マッピングの最も高い優先度を設定するには、以下のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=Kerberos uid mapping,cn=mapping,cn=sasl,cn=config
```

```
changetype: modify
replace: nsSaslMapPriority
nsSaslMapPriority: 1
```

9.11. SASL での KERBEROS GSS-API の使用

Directory Server が SASL 認証に GSS-API メカニズムを使用するには、Kerberos v5 をホストにデプロイする必要があります。Kerberos サービスを活用するには、GSS-API および Kerberos クライアントライブラリーを Directory Server ホストにインストールする必要があります。

9.11.1. Directory Server の SASL の認証メカニズム

Directory Server は、以下の SASL 暗号化メカニズムをサポートします。

- **PLAIN**。PLAIN は、簡単なパスワードベースの認証用にクリアテキストのパスワードを送信します。
- **EXTERNAL**。TLS を使用する EXTERNAL は、証明書ベースの認証を実行します。この方法では、強力な認証に公開鍵を使用します。
- **CRAM-MD5**。**CRAM-MD5** は弱く、単純な challenge-response 認証メソッドです。セキュリティー層を確立しません。

  警告

  Red Hat では、セキュアでない **CRAM-MD5** メカニズムを使用することは推奨されません。

- **DIGEST-MD5**。**DIGEST-MD5** は LDAPv3 サーバーの弱い認証方法です。

  警告

  Red Hat では、セキュアでない **DIGGEST-MD5** メカニズムを使用することは推奨されません。

- **Generic Security Services (GSS-API)**汎用セキュリティーサービス (GSS) は、UNIX ベースのオペレーティングシステムが Kerberos サービスにアクセスして認証するためのネイティブな方法であるセキュリティー API です。GSS-API は TLS と同様にセッション暗号化もサポートします。これにより、Kerberos バージョン 5 の認証情報 (チケット) を使用して LDAP クライアントがサーバーで認証でき、ネットワークセッションの暗号化を使用できます。

  Directory Server が GSS-API を使用するには、Kerberos をホストマシンに設定する必要があります。「SASL での Kerberos GSS-API の使用」を参照してください。

注記

GSS-API および Kerberos は GSS-API サポートのあるプラットフォームでのみサポートされます。GSS-API を使用するには、Kerberos クライアントライブラリーをインストールする必要があります。必要な Kerberos ライブラリーはすべてオペレーティングシステムベンダーから利用できます。

9.11.2. Directory Server の Kerberos の概要

Red Hat Enterprise Linux では、サポートされる Kerberos ライブラリーは MIT Kerberos バージョン 5 です。

Kerberos の概念、ならびに Kerberos の使用および設定については、MIT Kerberos の Web サイト http://web.mit.edu/Kerberos/ を参照してください。

9.11.2.1. プリンシパルおよびレルムについて

プリンシパル は、Kerberos 環境のユーザーまたはサービスです。**レルム** は、誰が何にアクセスできるかに関して、Kerberos が管理する内容を定義します。アクセスするクライアント、KDC、およびホストまたはサービスは同じレルムを使用する必要があります。

注記

Kerberos レルムは GSS-API 認証および暗号化でのみサポートされていますが、DIGEST-MD5 ではサポートされていません。

レルムは、LDAP DN のように、以下の形式でクライアントの DN を関連付けるためにサーバーによって使用されます。

```
uid=user_name/[server_instance],cn=realm,cn=mechanism,cn=auth
```

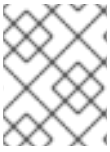
たとえば、**example.com** のヨーロッパの部門の **engineering** レルムの Mike Connors では、以下の関連を使用して、US レルムのサーバーにアクセスします。

```
uid=mconnors/cn=Europe.example.com,cn=engineering,cn=gssapi,cn=auth
```

Babara Jensen は、**US.example.com** の **accounting** レルムから、ローカルサーバーにアクセスする際にレルムを指定する必要はありません。

```
uid=bjensen,cn=accounting,cn=gssapi,cn=auth
```

レルムがメカニズムでサポートされ、デフォルトのレルムがサーバーに対する認証に使用されない場合、**レルム** は Kerberos プリンシパルで指定する必要があります。そうでない場合は、レルムを省略できます。



注記

Kerberos システムは、Kerberos レルムをデフォルトのレルムとして扱います。他のシステムはデフォルトでサーバーになります。

9.11.2.2. KDC サーバーおよびキータブの概要

キー配布センター (KDC) はユーザーを認証し、TGT (Ticket Granting Ticket) を発行します。これによ

り、ユーザーは GSS-API を使用して Directory Server に対して認証が可能になります。Kerberos 操作に応答するには、Directory Server でキータブファイルへのアクセスが必要になります。キータブには、Directory Server が他のサーバーへの認証に使用する暗号鍵が含まれます。

Directory Server は、Kerberos プリンシパルで **ldap** サービス名を使用します。以下に例を示します。

```
ldap/server.example.com@EXAMPLE.COM
```

キータブの作成に関する詳細は、Kerberos ドキュメントを参照してください。

注記

既存のエントリー識別名 (DN) にマッピングする Directory Server Kerberos プリンシパルの Simple Authentication and Security Layer (SASL) マッピングを作成する必要があります。

9.11.3. Directory Server 起動時の SASL 認証の設定

Kerberos チケットを認証に使用できるように、SASL GSS-API 認証は Directory Server でアクティベートする必要があります。これは、キータブファイルの場所を設定する変数を識別する init スクリプト用のシステム設定ファイルを指定することで行います。init スクリプトが Directory Server の起動時に実行すると、SASL 認証はすぐにアクティブになります。

デフォルトの SASL 設定は **/etc/sysconfig/dirsrv** ファイルに保存されます。

複数の Directory Server インスタンスがあり、これらすべてが SASL 認証を使用するわけではありません。その場合は、**dirsrv-instance** という名前の **/etc/sysconfig/** ディレクトリーにインスタンス固有の設定ファイルを作成できます。(例: **dirsrv-example**)。ホストにインスタンスが 1 つある場合は、デフォルトの **dirsrv** ファイルを使用することができます。

SASL 認証を有効にするには、**/etc/sysconfig/dirsrv** (またはインスタンス固有の) ファイルの **KRB5_KTNAME** 行のコメントを解除し、**KRB5_KTNAME** 変数のキータブの場所を設定します。以下に例を示します。

```
# In order to use SASL/GSSAPI the directory
# server needs to know where to find its keytab
# file - uncomment the following line and set
# the path and filename appropriately
KRB5_KTNAME=/etc/dirsrv/krb5.keytab
```

9.12. SASL メカニズムの設定

デフォルトでは、Directory Server は、簡単な認証とセキュリティーレイヤー (SASL) ライブラリーがサポートするすべてのメカニズムを有効にします。これらは、root dse ***supportedSASLMechanisms*** パラメーターに一覧表示されます。特定の SASL メカニズムを有効にするには、**cn=config** エントリーに ***nsslapd-allowed-sasl-mechanisms*** 属性を設定します。たとえば、**GSSAPI** および **DIGEST-MD5** メカニズムのみを有効にするには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-allowed-sasl-mechanisms="GSSAPI, DIGEST-MD5"
Successfully replaced "nsslapd-allowed-sasl-mechanisms"
```

注記

EXTERNAL が ***nsslapd-allowed-sasl-mechanisms*** パラメーターに記載されていない場合でも、このメカニズムは常に有効になります。

詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の該当するセクションを参照してください。

9.13. LDAP クライアントでの SASL の使用

ldapsearch などの LDAP クライアントで SASL を使用するには、 ***-Y SASL_mechanism*** をコマンドに渡します。以下に例を示します。

- LDAP プロトコルで SASL メカニズム **GSSAPI** を使用するには、以下を行います。

```
# ldapsearch -Y GSSAPI -U "dn:uid=user_name,ou=people,dc=example,dc=com" -R
EXAMPLE.COM -H ldap://server.example.com -b "dc=example,dc=com"
```

- LDAPS プロトコルで SASL メカニズム **PLAIN** を使用するには、以下を行います。

```
# ldapsearch -Y PLAIN -D "uid=user_name,ou=people,dc=example,dc=com" -W -H
ldaps://server.example.com -b "dc=example,dc=com"
```

注記

Directory Server では、SASL プロキシーの認証はサポートされません。そのため、Directory Server は、クライアントによって提供される SASL **authzid** 値を無視します。

第10章 属性暗号化の設定

Directory Server は、権限のないユーザーがエントリー内の特定のエントリーや属性を読み取らないようにするアクセス制御ルールや、信頼できないネットワークでのデータの盗聴や改ざんからデータを保護する TLS など、機密データへのアクセスを保護するための多数のメカニズムを提供します。ただし、サーバーのデータベースファイルのコピーが権限のない人の手に渡った場合は、それらのファイルから機密情報を抽出する可能性があります。データベースの情報はプレーンテキストで保存されるため、政府の識別番号やパスワードなど、一部の機密情報が標準アクセス制御手段で保護されない可能性があります。

情報の機密性が高くなると、この情報損失の可能性は重大なセキュリティーリスクをもたらす可能性があります。このセキュリティーリスクを削除するには、Directory Server ではそのデータベースの一部を暗号化することができます。暗号化されると、攻撃者がサーバーのデータベースファイルのコピーがある場合にもデータを安全に実行できます。

データベース暗号化により、属性をデータベースで暗号化できます。暗号化と暗号化暗号の両方は、バックエンドごとの属性ごとに設定できます。これを設定すると、インデックスデータであっても、特定の属性内のすべてのインスタンスは、そのデータベースに保存されているすべてのエントリー用に暗号化されます。

属性の暗号化の利点として、暗号化された値は 1 を超える Security Strength Factor (SSF) を持つクライアントにのみ送信できます。

注記

暗号化されたデータには 1 つの例外があります。エントリーの RDN として使用される値はエントリー DN 内で暗号化されません。たとえば、***uid*** 属性が暗号化されている場合、値はエントリーで暗号化されますが、DN に表示されます。

```
dn: uid=jsmith1234,ou=People,dc=example,dc=com
...
uid:: Sf04P9nJWGU1qiW9JJCGRg==
```

これにより、誰かが暗号化された値を検出できるようになります。

エントリー DN 内で使用される属性は常に DN に表示されるため、実質的には暗号化できません。DN を構築するのに使用される属性に注意し、それに応じて属性暗号化モデルを設計します。

インデックス化された属性は暗号化され、属性の暗号化は **eq** および **pres** のインデックスと完全に互換性があります。通常、属性値から派生するインデックスファイルの内容は、攻撃者がインデックスの分析から暗号化されたデータをすべて復旧することを防ぐためも暗号化されます。

サーバーは暗号化属性のインデックスを検索する前にすべてのインデックスキーを事前に暗号化するため、暗号化されたインデックスを使用する検索にはサーバーパフォーマンスにも影響が及ぶことはありますが、その影響はインデックスを使用する価値がなくなるほど深刻ではありません。

10.1. キーの暗号化

属性暗号化を使用するには、TLS に対してサーバーを設定し、TLS を有効にする必要があります。これは、属性暗号化がサーバーの TLS 暗号化キーと、TLS と同じ PIN 入力メソッドを使用するためです。サーバーの起動時に PIN を手動で入力するか、PIN ファイルを使用する必要があります。

無作為に生成される対称暗号キーは、属性データを暗号化および復号するために使用されます。設定さ

れた暗号には個別のキーが使用されます。これらの鍵は、サーバーの TLS 証明書から公開鍵を使用して **ラップ** され、生成したラップ済みキーがサーバーの設定ファイル内に保存されます。属性暗号化の効果的な強度は、ラップに使用されるサーバーの TLS キーの強度よりも高くなります。サーバーの秘密鍵にアクセスできないと、ラップ済みのコピーから対称キーを復旧することができません。

警告

失われたキーを復元するメカニズムはありません。そのため、サーバーの証明書データベースを安全にバックアップすることが特に重要です。サーバーの証明書が失われた場合は、そのデータベースに保存されている暗号化データを復号することはできません。

警告

TLS 証明書の期限で更新が必要な場合は、更新前に暗号化されたバックエンドインスタンスをエクスポートします。証明書を更新して、エクスポートした LDIF ファイルを再インポートします。

10.2. 暗号化暗号

暗号化暗号は属性ごとに設定でき、属性に対する暗号化時に管理者が選択する必要があります。

以下の暗号がサポートされます。

- AES (Advanced Encryption Standard)
- 3DES (Triple Data Encryption Standard)

注記

強力な暗号化の場合は、AES 暗号のみを使用することが推奨されます。

すべての暗号は Cipher Block Chaining モードで使用されます。

暗号化暗号設定後は、データをエクスポートおよび再インポートせずに変更しないでください。

10.3. 属性暗号化の設定

コマンドラインまたは Web コンソールを使用して、特定の属性の属性暗号化を有効または無効にします。

10.3.1. コマンドラインを使用した属性の暗号化の有効化

(たとえば、AES で暗号化した **userRoot** データベースの ***telephoneNumber*** 属性を) Directory Server が保存するように設定するには、以下を行います。

1. オプションで、既存の ***telephoneNumber*** 属性を暗号化するには、データベースをエクスポートします。「暗号化したデータベースのエクスポート」を参照してください。
2. **userRoot** データベースの ***telephoneNumber*** 属性の AES 暗号化を有効にします。

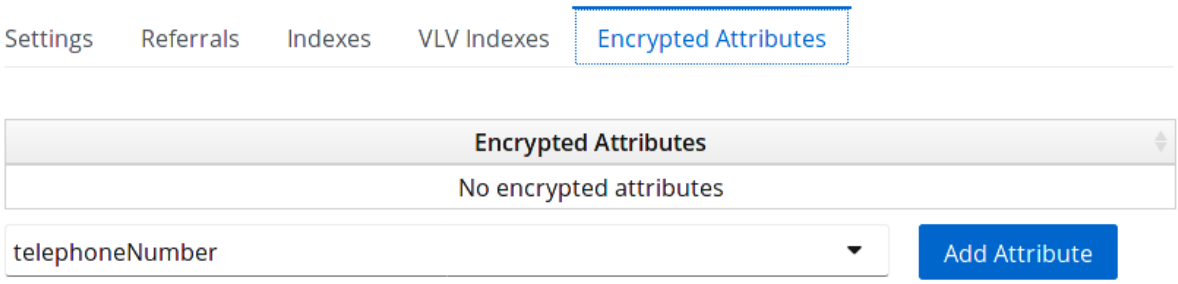
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend attr-encrypt --add-attr telephoneNumber userRoot
```

3. 既存の属性を暗号化するためにデータベースをエクスポートした場合は、データベースを再インポートします。「暗号化されたデータベースへの LDIF ファイルのインポート」を参照してください。

10.3.2. Web コンソールを使用した属性の暗号化の有効化

(たとえば、AES で暗号化したデータベースの ***telephoneNumber*** 属性を) Directory Server が保存するように設定するには、以下を行います。

1. オプションで、既存の ***telephoneNumber*** 属性を暗号化するには、データベースをエクスポートします。「暗号化したデータベースのエクスポート」を参照してください。
2. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
3. インスタンスを選択します。
4. **Database** メニューを開きます。
5. 接尾辞エントリーを選択します。
6. **Encrypted Attributes** タブを開きます。
7. 暗号化する属性の名前を入力します。



8. **Add Attribute** をクリックします。
9. 既存の属性を暗号化するためにデータベースをエクスポートした場合は、データベースを再インポートします。「暗号化されたデータベースへの LDIF ファイルのインポート」を参照してください。

10.3.3. コマンドラインを使用した属性の暗号化の無効化

Directory Server が保存しなくなった属性 (たとえば、**userRoot** データベースに暗号化された ***telephoneNumber*** 属性) を設定するには、以下を実行します。

1. 必要に応じて、既存の ***telephoneNumber*** 属性を複号するには、データベースをエクスポートします。「暗号化したデータベースのエクスポート」を参照してください。

2. **userRoot** データベースの ***telephoneNumber*** 属性の暗号化を無効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend attr-encrypt --del-attr telephoneNumber userRoot
```

3. 既存の属性を復号するためにデータベースをエクスポートした場合は、データベースを再インポートします。「暗号化されたデータベースへの LDIF ファイルのインポート」を参照してください。

10.3.4. Web コンソールを使用した属性の暗号化の無効化

(たとえば、AES で暗号化したデータベースの ***telephoneNumber*** 属性を) Directory Server が保存するように設定するには、以下を行います。

1. オプションで、既存の ***telephoneNumber*** 属性を暗号化するには、データベースをエクスポートします。「暗号化したデータベースのエクスポート」を参照してください。

2. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。

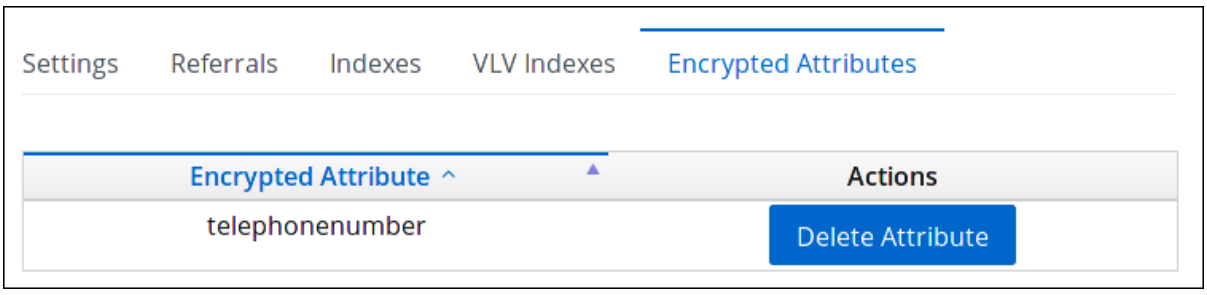
3. インスタンスを選択します。

4. **Database** メニューを開きます。

5. 接尾辞エントリーを選択します。

6. **Encrypted Attributes** タブを開きます。

7. ***telephoneNumber*** 属性の右側にある **Delete Attribute** ボタンをクリックします。



8. **Yes** をクリックして確定します。

9. 既存の属性を復号するためにデータベースをエクスポートした場合は、データベースを再インポートします。「暗号化されたデータベースへの LDIF ファイルのインポート」を参照してください。

10.3.5. 属性暗号化の有効化後の一般的な考慮事項

データベースにすでにあるデータの暗号化を有効にしている場合は、以下を実行します。

- 暗号化されていないデータは、サーバーのデータベースページプールのバッキングファイルで保持できます。このデータを削除するには、以下を実行します。

    1. インスタンスを停止します。

```
# dsctl instance_name stop
```

2. **/var/lib/dirsrv/slapd-*instance_name*/db/guardian** ファイルを削除します。

```
# rm /var/lib/dirsrv/slapd-instance_name/db/guardian
```

3. インスタンスを起動します。

```
# dsctl instance_name start
```

- 暗号化を有効にし、データが正常にインポートされた後に、暗号化されていないデータで LDIF ファイルを削除します。
- 暗号化を有効にしたら、データの再インポート時に Directory Server は新しいデータベースを削除し、作成します。
- レプリケーションログファイルは暗号化されません。このデータを保護するには、暗号化されたディスクに保存します。
- サーバーのメモリー (RAM) のデータは暗号化されず、swap パーティションに一時的に保存できます。このデータを保護するには、暗号化された swap 領域を設定します。

重要

暗号化されていないデータを含むファイルを削除すると、このデータは特定の状況で復元できます。

10.4. 暗号化したデータベースのエクスポートおよびインポート

暗号化されたデータベースのエクスポートおよびインポートは、通常のデータベースのエクスポートおよびインポートに似ています。ただし、暗号化された情報は、データをエクスポートするときに復号し、データベースに再インポートするときに再暗号化する必要があります。

10.4.1. 暗号化したデータベースのエクスポート

暗号化されたデータベースからデータをエクスポートするには、***-E*** パラメーターを **dsconf** コマンドに渡します。

たとえば、復号した属性で完全な **userRoot** データベースをエクスポートするには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend export -E userRoot
```

または、特定のサブツリーのみをエクスポートできます。たとえば、**ou=People,dc=example,dc=com** エントリーからすべてのデータをエクスポートするには、以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend export -E -s
"ou=People,dc=example,dc=com" userRoot
```

dsconf を使用したデータのエクスポートに関する詳細は、「**dsconf backend export** コマンドを使用したデータベースのエクスポート」を参照してください。

10.4.2. 暗号化されたデータベースへの LDIF ファイルのインポート

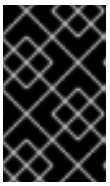
属性の暗号化が有効な場合にデータをデータベースにインポートするには、以下を実行します。

1. Directory Server インスタンスを停止します。

```
# dsctl instance_name stop
```

2. 前回のエクスポートと今回のインポートとの間で証明書データベースを置き換えた場合は、**/etc/dirsrv/slapd-*instance_name*/dse.ldif** ファイルを編集して、その属性を含む以下のエントリーを削除します。

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



重要

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

3. LDIF ファイルをインポートします。たとえば、**/tmp/example.ldif** を **userRoot** データベースにインポートするには、以下を行います。

```
# dsctl instance_name ldif2db --encrypted userRoot /tmp/example.ldif
```

--encrypted パラメーターを使用すると、スクリプトで属性を暗号化して、インポート中に暗号化を設定できます。

4. インスタンスを起動します。

```
# dsctl instance_name start
```

10.5. 属性暗号化に使用される TLS 証明書の更新

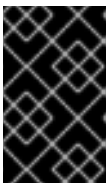
属性の暗号化は TLS 証明書に基づいています。TLS 証明書の更新または置き換え後に属性の暗号化が失敗するのを防ぐには、以下を実行します。

1. 復号化された属性でデータベースをエクスポートします。「暗号化したデータベースのエクスポート」を参照してください。
2. Certificate Signing Request (CSR) を新規作成します。「証明書署名要求の作成」を参照してください。
3. 新しい証明書をインストールします。「サーバー証明書のインストール」を参照してください。
4. Directory Server インスタンスを停止します。

```
# dsctl instance_name stop
```

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

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

   重要

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

6. データベースをインポートします。「暗号化されたデータベースへの LDIF ファイルのインポート」を参照してください。

7. インスタンスを起動します。

```
# dsctl instance_name start
```

第11章 FIPS モードサポートの管理

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

FIPS の一般的な詳細については、『Red Hat Enterprise Linux 8 のセキュリティー強化 』 に関するドキュメントの Federal Information Processing Standard (FIPS) を参照してください。

FIPS モードサポートの有効化

Directory Server の FIPS モードのサポートを有効にするには、以下を実行します。

1. 必要に応じて、Red Hat Enterprise Linux で FIPS モードを有効にします。詳細については、『Red Hat Enterprise Linux 8 セキュリティー強化 』 ドキュメントの対応するセクションを参照してください。
2. ネットワークセキュリティーサービス (NSS) データベースの FIPS モードを有効にします。

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

3. Directory Server インスタンスを再起動します。

```
# dsctl instance_name restart
```

FIPS モードサポートの無効化

Directory Server の FIPS モードのサポートを無効にするには、以下を実行します。

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

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

2. Directory Server インスタンスを再起動します。

```
# dsctl instance_name restart
```

第12章 ディレクトリースキーマの管理

Red Hat Directory Server には、数百のオブジェクトクラスおよび属性を含む標準スキーマが同梱されています。標準のオブジェクトクラスおよび属性はほとんどのデプロイメントの要件を満たす必要がありますが、特定のディレクトリーデータのスキーマを拡張しないといけない場合があります。スキーマの拡張は、新規オブジェクトクラスおよび属性を作成することで行われます。

『Red Hat Directory Server 11 の設定、コマンド、およびファイルリファレンス』 は、ほとんどの標準 Directory Server 属性およびオブジェクトクラスのリファレンスであり、許可された属性および必須属性、どのオブジェクトクラスがどの属性を取得するか、およびどの OID が値情報を取得するかの情報が含まれています。これは、ディレクトリーで有用なスキーマ要素を特定し、作成されるカスタムスキーマを決定するのに適したリソースです。

12.1. スキーマの概要

ディレクトリースキーマは、ディレクトリーへのデータの保存方法を定義する一連のルールです。ディレクトリー情報は個別のエントリーに保存され、各エントリーは属性のセットとその値で設定されます。エントリーで説明されるアイデンティティーの種類は、エントリーのオブジェクトクラスで定義されます。オブジェクトクラスは、オブジェクトクラスの定義された属性セットでエントリーが記述するオブジェクトの種類を指定します。

LDAP では、オブジェクトクラスはエントリーの定義に使用できる属性のセットを定義します。LDAP 標準仕様は、人、グループ、場所、組織、部門、機器など、多くの一般的なエントリーに対するオブジェクトクラスを提供します。ID は、属性とその値を含むディレクトリーエントリーで説明されています。ペアは、**属性値表明** または AVA と呼ばれます。ディレクトリー内の情報には説明的な属性が関連付けられています。一致するルールや LDAP コントロールを含む Directory Server 設定のその他の側面は、スキーマにも定義されます。これらすべてが **スキーマ要素** です。

すべての schema 要素は、一意のドット区切り番号で識別されます。これは **オブジェクト ID** または **OID** と呼ばれます。

12.1.1. デフォルトのスキーマファイル

Directory Server のスキーマは、複数のスキーマファイル (スキーマ要素を定義する LDIF ファイル) で定義されます。Directory Server スキーマファイルは、**/usr/share/dirsrv/schema/** ディレクトリーにあります。このディレクトリーのファイルは、新しい Directory Server インスタンスのテンプレートとして使用されます。このディレクトリーに新しいスキーマを追加すると、新しいインスタンスが利用可能になります。

Directory Server が操作を実行し、エントリーを管理するために使用する属性は、『Red Hat Directory Server 11 の設定、コマンド、およびファイルリファレンス』 で他の設定とともに説明されています。

12.1.2. オブジェクトクラス

LDAP では、オブジェクトクラスはエントリーの定義に使用できる属性のセットを定義します。LDAP 標準仕様は、ユーザー (**person** および **inetOrgPerson**)、グループ (**groupOfNames**)、場所 (**locality**)、組織および部門 (**organization** および **organizationalUnit**)、および機器 (**device**) など、多くの一般的なエントリーに対するオブジェクトクラスを提供します。

スキーマファイルでは、オブジェクトクラスは ***objectclasses*** 行によって識別され、その後 OID、名前、説明、その直接の上位オブジェクトクラス (オブジェクトクラスと使用する必要のあるオブジェクトクラス、およびそのオブジェクトクラスと属性を共有するのに必要なオブジェクトクラス)、および必須属性の一覧 (**MUST**) および許可される属性の一覧 (**MAY**) が続きます。

これは、例12.1「個人のオブジェクトクラススキーマエントリー」 に示されています。

例12.1 個人のオブジェクトクラススキーマエントリー

```
objectClasses: ( 2.5.6.6 NAME 'person' DESC 'Standard LDAP objectclass' SUP top MUST ( sn $
cn ) MAY ( description $ seeAlso $ telephoneNumber $ userPassword ) X-ORIGIN 'RFC 4519' )
```

すべてのオブジェクトクラスは、必須属性 (そのスキーマの **MUST** キーワード) および許可された属性 (そのスキーマの **MAY** キーワード) を定義します。必須属性は、指定されたオブジェクトクラスを使用するエントリーに存在する必要がありますが、許可された属性は許可されており、エントリーで使用できますが、エントリーが有効である必要はありません。

例12.1「個人のオブジェクトクラススキーマエントリー」 のように、**person** オブジェクトクラスには、***cn*** 属性、***sn*** 属性、および ***objectClass*** 属性が必要で、***description*** 属性、***seeAlso*** 属性、***telephoneNumber*** 属性、および ***userPassword*** 属性を許可します。

オブジェクトクラスは、独自の必須属性と許可される属性に加えて、別のクラスから属性を継承できます。2 つ目のオブジェクトクラスは、最初のオブジェクトクラスの superior または parent オブジェクトクラスです。

たとえば、ユーザーのエントリーに **inetOrgPerson** オブジェクトクラスが必要です。その場合、エントリーには、**inetOrgPerson** と **organizationalPerson** の上位オブジェクトクラスと、**organizationalPerson** の上位オブジェクトクラスである **person** も含める必要があります。

```
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
```

オブジェクトクラス定義は、**cn=schema** エントリーの ***objectclasses*** 属性です。***objectclasses*** 属性の形式は以下のとおりです。

```
objectclasses: ( definition )
```

オブジェクトクラス定義には複数のコンポーネントが含まれます。

- OID (通常はドット区切り番号)
- **NAME 名前** 形式の一意の名前
- **DESC 説明** 形式の説明
- **SUP** object_class の形式で、このオブジェクトクラスの上位または親のオブジェクトクラス。関連する親がない場合は、**SUP top** を使用してください。
- **AUXILIARY** という単語で、オブジェクトクラスを適用するエントリーのタイプを指定します。**AUXILIARY** は、任意のエントリーに適用できることを意味します。
- **MUST** の後に続く必要な属性のリスト。複数の属性を含めるには、グループを括弧で囲み、ドル記号 ($) で属性を区切ります。
- **MAY** の後に続く許可される属性のリスト。複数の属性を含めるには、グループを括弧で囲み、ドル記号 ($) で属性を区切ります。

顧客のオブジェクトクラス定義は、コマンドラインまたは Web コンソールで **cn=schema** エントリーを変更する場合に **/etc/dirsrv/slapd-*instance_name*/schema/99user.ldif** に保存されます。

12.1.3. 属性

ディレクトリーエントリーは、属性とその値で設定されます。これらのペアは、**属性値表明** または AVA と呼ばれます。ディレクトリー内の情報には説明的な属性が関連付けられています。たとえば、***cn*** 属性は、**cn: John Smith** などのユーザーの氏名を保存するために使用されます。

追加の属性は、John Smith に関する補足情報を提供できます。

```
givenname: John
surname: Smith
mail: jsmith@example.com
```

スキーマファイルでは、属性が以下によって記述されます。

- OID
- name
- 構文マッチングルール (任意)
- 部分文字列マッチングルール (任意)
- 順序ルール (任意)
- 説明 (任意)
- 構文
- 単値または多値の属性
- 属性が定義されている場所の詳細

これは、例12.2「***uid*属性スキーマエントリー」 に示されています。

例12.2 *uid*属性スキーマエントリー

```
( 0.9.2342.19200300.100.1.1 NAME ( 'uid' 'userid' ) EQUALITY caseIgnoreMatch SUBSTR
caseIgnoreSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORIGIN 'RFC 4519' )
```

12.1.3.1. Directory Server 属性の構文

属性の構文は、属性が許可する値の形式を定義します。他のスキーマ要素と同様に、構文は、スキーマファイルエントリーで構文の OID を使用して属性に対して定義されます。

Directory Server は、属性の構文を使用してエントリーでのソートとパターン一致を実行します。

LDAP 属性の構文に関する詳細は、RFC 4517 を参照してください。

サポートされる LDAP 属性の構文については、Red Hat Directory Server 10 の設定、コマンド、およびファイルリファレンス の 『Directory Server 属性構文』 に記載されています。

12.1.4. スキーマの拡張

新規、カスタム属性、およびオブジェクトクラスを Directory Server インスタンスに追加してスキーマを拡張することができ、スキーマ要素を追加する方法は複数あります。LDAP ツールを使用すると、インスタンスのデフォルトのカスタムスキーマファイルにスキーマ要素が追加されます (例: **99user.ldif**)。新しい個別のスキーマファイルを作成し、デフォルトのスキーマファイルで追加することもできます。

新規スキーマ要素を追加するには、3 点が必要になります。

1. 新規スキーマの OID の計画および定義。スキーマ要素はその OID によってサーバーが認識されるため、OID を一意で、整理することが重要です。Directory Server 自体は OID を管理しませんが、「オブジェクト識別子の管理」で説明するベストプラクティスがいくつかあります。

2. 新しい属性を作成します。属性定義には名前、構文 (許可される値の形式)、OID、および属性をエントリーごとに一度または複数回使用できるかどうかの説明が必要です。

3. 新規属性を含むオブジェクトクラスを作成します。オブジェクトクラスは、そのエントリータイプに必要な属性と許可される (許容) 属性をリスト表示します。デフォルトのスキーマは変更できないため、新規属性を作成している場合は、カスタムオブジェクトクラスに追加する必要があります。

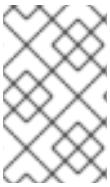
スキーマ要素は事前に計画する必要があります。同じ情報に複数の属性を使用しないでください。可能な場合は、標準の Directory Server スキーマを使用します。Directory Server には、数百の属性があり、デフォルトのスキーマファイルで定義されたオブジェクトクラスが多数あります。『Red Hat Directory Server 11 の設定、コマンド、およびファイルリファレンス』 は、標準の属性およびオブジェクトクラスを一覧表示して説明します。スキーマはすべて **/usr/share/dirsrv/schema/** のスキーマファイルで確認できます。まずは、利用可能なスキーマを確認してください。次に、不足している情報属性と、不足している情報属性を補うためにカスタム属性を使用した最善の方法を計画します。スキーマのプランニングについては、『デプロイメントガイド』 で説明しています。

警告

Directory Server のデフォルトのオブジェクトクラスおよび属性は LDAP および X.500 標準仕様および RFC に基づいています。標準スキーマを使用すると、Directory Server が他のアプリケーションやサーバーとより簡単に統合され、LDAP クライアント、レガシー Directory Server インスタンス、および今後のリリースで相互運用性が可能になります。標準属性を編集したり、オブジェクトクラスを変更したりすることは推奨されません。

Directory Server スキーマをカスタマイズする場合は、以下のルールを念頭に置いてください。

- スキーマはできるだけシンプルに保ちます。
- 可能であれば、既存のスキーマ要素を再利用します。
- 各オブジェクトクラスに定義される必須属性の数を最小限に抑えます。
- 複数のオブジェクトクラスまたは属性を同じ目的で定義しないでください。
- 属性またはオブジェクトクラスの既存の定義は変更しないでください。



注記

標準スキーマを削除または置き換えることは **ありません**。これを行うと、他のディレクトリーやその他の LDAP クライアントアプリケーションとの互換性の問題が発生する可能性があります。

インスタンスが起動すると、スキーマが Directory Server インスタンスに読み込まれます。Directory Server が再起動するか、再読み込みタスクが開始されない限り、新しいスキーマファイルは読み込まれません。インスタンスのデフォルトのカスタムスキーマファイルは、*99user.ldif* が最後のスキーマファイルとして読み込まれます。標準スキーマファイルに定義がすでに含まれる場合、カスタム定義は標準スキーマファイルを上書きします。

12.1.5. スキーマレプリケーション

ディレクトリースキーマが **cn=schema** サブツリーで更新されると、Directory Server は変更状態番号 (CSN) を含むローカルの **/etc/dirsrv/slapd-*instance_name*/schema/99user.ldif** ファイルに変更を保存します。更新されたスキーマは他のレプリカに自動的に複製されません。スキーマレプリケーションは、ディレクトリーのコンテンツが複製されたツリーで更新されると開始します。たとえば、スキーマの変更後にユーザーエントリーまたはグループエントリーを更新すると、***nsSchemaCSN*** 属性に保存されている CSN と、コンシューマーにある CSN が比較されます。リモート CSN がサプライヤー上のものよりも小さい場合、スキーマはコンシューマーに複製されます。レプリケーションに成功すると、サプライヤーにあるすべてのオブジェクトクラスと属性タイプはコンシューマーの定義のスーパーセットである必要があります。

例12.3 スキーマのサブセットとスーパーセット

- **server1** では、**demo** オブジェクトクラスは ***a1*** 属性、***a2*** 属性、および ***a3*** 属性を許可します。
- **server2** では、**demo** オブジェクトクラスは ***a1*** 属性および ***a3*** 属性を許可します。

例12.3「スキーマのサブセットとスーパーセット」では、**server1** にある **demo** オブジェクトクラスのスキーマ定義は、**server2** のオブジェクトクラスのスーパーセットです。検証フェーズで、スキーマが複製または許可されると、Directory Server はスーパーセット定義を取得します。たとえば、ローカルスキーマのオブジェクトクラスがサプライヤースキーマのオブジェクトクラスよりも少ない属性を許可していることをコンシューマーが検出すると、ローカルスキーマが更新されます。

スキーマ定義が正常に複製された場合、***nsSchemaCSN*** 属性は両サーバーで同一であり、レプリケーションセッションの開始時に比較されなくなります。

以下のシナリオでは、スキーマは複製されません。

- あるホストのスキーマが、別のホストのスキーマのサブセットの場合

  たとえば、例12.3「スキーマのサブセットとスーパーセット」では、**server2** にある **demo** オブジェクトクラスのスキーマ定義は **server1** のオブジェクトクラスのサブセットです。サブセットは、属性 (単一値属性は複数値属性のサブセット) および属性の構文 (**IA5** は **Octet_string** のサブセット) に対しても発生する可能性があります。

- サプライヤースキーマとコンシューマースキーマの定義をマージする必要がある場合

  Directory Server はマージスキーマをサポートしません。たとえば、1 台のサーバーのオブジェクトクラスが ***a1*** 属性、***a2*** 属性、および ***a3*** 属性を許可し、別のサーバーのオブジェクトクラスが ***a1*** 属性、***a3*** 属性、および ***a4*** 属性を許可する場合、スキーマはサブセットではなく、マージできません。

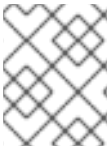
- **/etc/dirsrv/slapd-*instance_name*/schema/99user.ldif** 以外のスキーマファイルが使用されます。

  Directory Server を使用すると、**/etc/dirsrv/slapd-*instance_name*/schema/** ディレクトリーにスキーマファイルを追加できます。ただし、**99user.ldif** ファイルの CSN のみが更新されます。このため、他のスキーマファイルはローカルでのみ使用され、レプリケーションパートナーに自動的に転送されません。更新されたスキーマファイルをコンシューマーに手動でコピーし、スキーマを再読み込みします。詳細については、「スキーマの動的再読み込み」 を参照してください。

  スキーマ定義の重複を回避し、自動レプリケーションを有効にするには、すべてのカスタムスキーマを **/etc/dirsrv/slapd-*instance_name*/schema/99user.ldif** ファイルに保存します。カスタムスキーマファイルの作成方法は、「カスタムスキーマファイルの作成」を参照してください。

12.2. オブジェクト識別子の管理

各 LDAP オブジェクトクラスまたは属性には、一意の名前と **オブジェクト識別子** (OID) を割り当てる必要があります。OID は、サーバーへのスキーマ要素を識別するドットで区切られた番号です。OID は、異なるブランチに対応するために拡張できるベース OID を使用して階層化することができます。たとえば、ベース OID は **1** で、属性のブランチを **1.1** にし、オブジェクトクラスのブランチを **1.2** にすることもできます。



注記

カスタムスキーマを作成するために数字の OID を使用する必要はありませんが、Red Hat では前方互換性とパフォーマンスを向上させることを強く推奨します。

OID は、Internet Assigned Numbers Authority (IANA) を介して組織に割り当てられ、Directory Server は OID を取得するメカニズムを提供しません。OID の取得に関する情報を取得するには、IANA の Web サイト http://www.iana.org/cgi-bin/enterprise.pl にアクセスします。

IANA からベース OID を取得したら、OID をカスタムスキーマ要素に割り当てる方法を計画します。属性とオブジェクトクラスの両方にブランチを定義します。ルールと LDAP コントロールに一致するブランチを使用することもできます。

OID ブランチが定義されたら、OID 割り当てを追跡する OID レジストリーを作成します。OID レジストリーは、ディレクトリースキーマで使用される OID と説明を提供するリストです。これにより、OID が複数の目的に使用されないようにします。カスタムスキーマで OID レジストリーを公開します。

12.3. オブジェクトクラスの作成

本セクションでは、コマンドラインおよび Web コンソールを使用してオブジェクトクラスを作成する方法を説明します。

12.3.1. コマンドラインを使用したオブジェクトクラスの作成

ldapmodify ユーティリティーを使用して、新しいオブジェクトクラスエントリーを作成します。以下に例を示します。

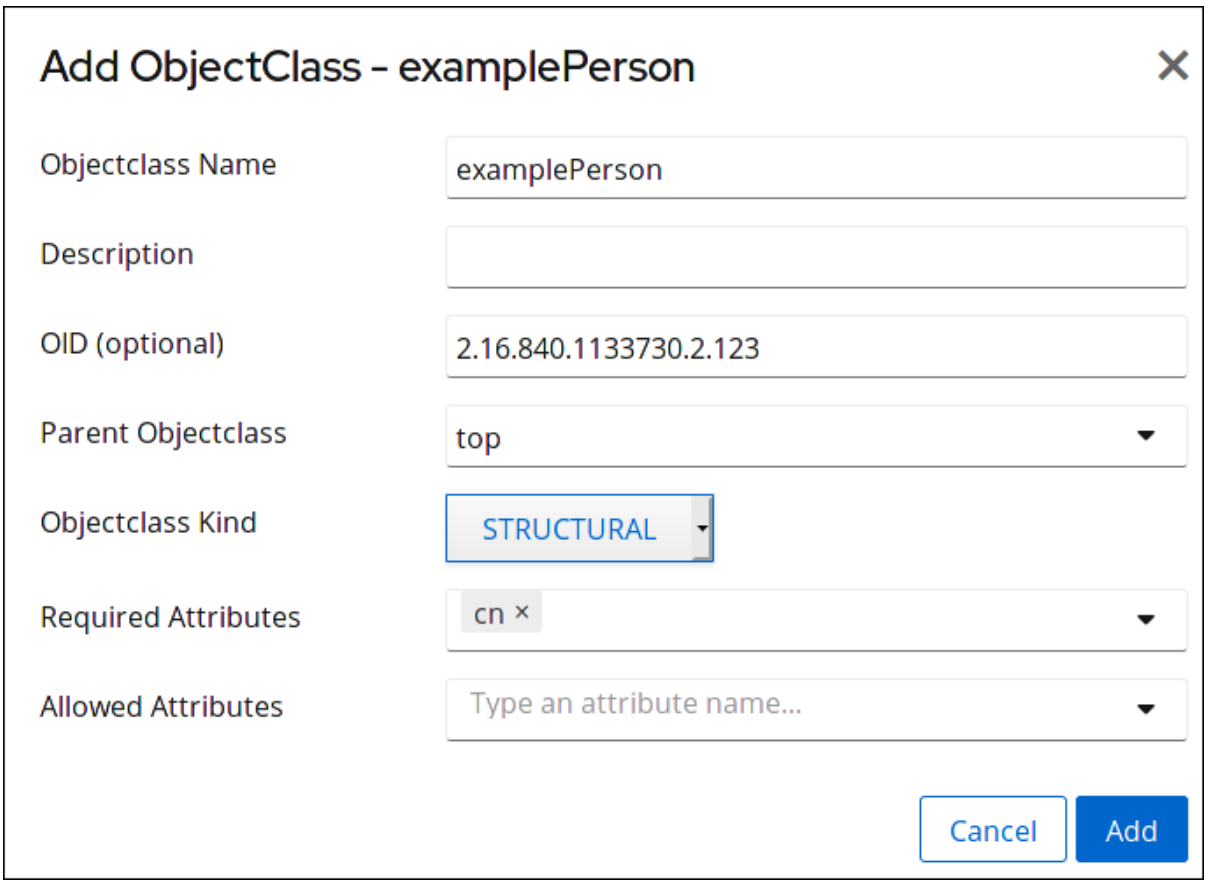
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com schema objectclasses add
examplePerson --oid="2.16.840.1133730.2.123" --desc="Example Person Object Class" --
sup="inetOrgPerson" --kind="AUXILIARY" --must="cn" --may exampleDateOfBirth
examplePreferredOS
```

オブジェクトクラス定義の詳細は、「オブジェクトクラス」を参照してください。

12.3.2. Web コンソールを使用したオブジェクトクラスの作成

Web コンソールを使用してオブジェクトクラスの作成するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Schema → Objectclasses** を選択します。
4. **Add ObjectClass** をクリックします。
5. 親オブジェクトクラスを選択します。
6. オブジェクトクラス名を入力し、オプションでオブジェクト識別子 (OID) を設定します。
7. オブジェクトクラスの種類を選択します。
8. 対応するフィールドに必須属性および許可属性を入力します。



9. **Add** をクリックします。

12.4. オブジェクトクラスの更新

本セクションでは、コマンドラインおよび Web コンソールを使用してオブジェクトクラスを更新する方法を説明します。

12.4.1. コマンドラインを使用したオブジェクトクラスの更新

dsconf ユーティリティーを使用して、オブジェクトクラスエントリーを更新します。以下に例を示します。

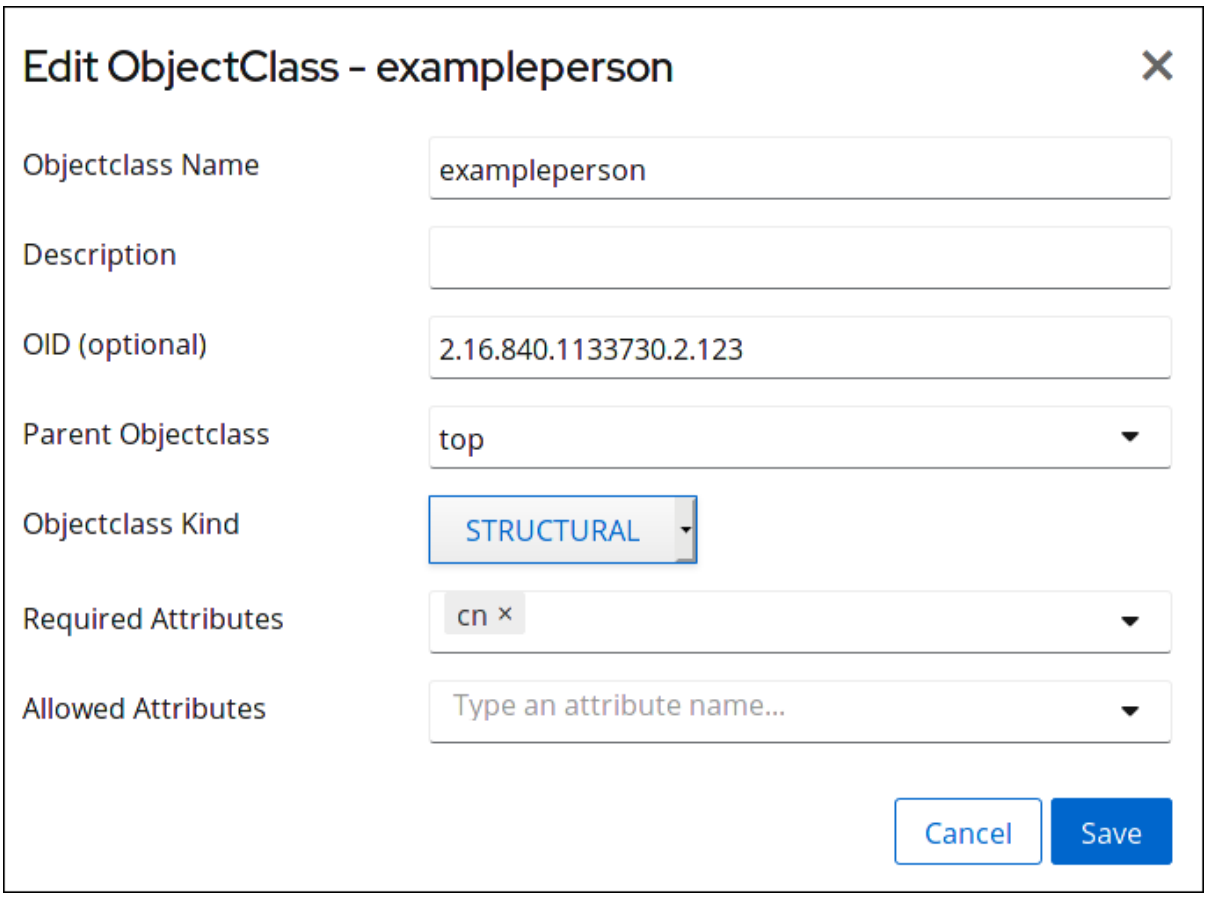
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com schema objectclasses replace
examplePerson --oid="2.16.840.1133730.2.123" --desc="Example Person Object Class" --
sup="inetOrgPerson" --kind="AUXILIARY" --must="cn" --may exampleDisplayName exampleAlias
```

オブジェクトクラス定義の詳細は、「オブジェクトクラス」を参照してください。

12.4.2. Web コンソールを使用したオブジェクトクラスの更新

Web コンソールを使用してオブジェクトクラスの更新するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Schema → Objectclasses** を選択します。
4. 編集するオブジェクトクラスエントリーの右側にある **Choose Action** ボタンをクリックします。
5. **Edit Object Class** を選択します。
6. パラメーターを更新します。



7. **Save** をクリックします。

12.5. オブジェクトクラスの削除

本セクションでは、コマンドラインおよび Web コンソールを使用してオブジェクトクラスを削除する方法を説明します。

警告

デフォルトのスキーマ要素は削除しないでください。これらは Directory Server に必要です。

12.5.1. コマンドラインを使用したオブジェクトクラスの削除

ldapmodify ユーティリティーを使用してオブジェクトクラスエントリーを削除します。たとえば、**examplePerson** オブジェクトクラスを削除するには、以下を実行します。

1. 不要な属性を使用するエントリーから、その属性を削除します。詳細については、「属性の削除」 を参照してください。
2. オブジェクトクラスエントリーを削除します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com schema objectclasses delete
examplePerson
```

オブジェクトクラス定義の詳細は、「オブジェクトクラス」を参照してください。

12.5.2. Web コンソールを使用したオブジェクトクラスの削除

Web コンソールを使用してオブジェクトクラスの削除するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Schema → Objectclasses** を選択します。
4. 削除するオブジェクトクラスエントリーの横にある **Choose Action** ボタンをクリックします。
5. **Delete Objectclass** を選択します。
6. **Yes** をクリックして確定します。

12.6. 属性の作成

本セクションでは、コマンドラインおよび Web コンソールを使用して属性を作成する方法を説明します。

12.6.1. コマンドラインで属性の作成

ldapmodify ユーティリティーを使用して新しい属性を作成します。以下に例を示します。

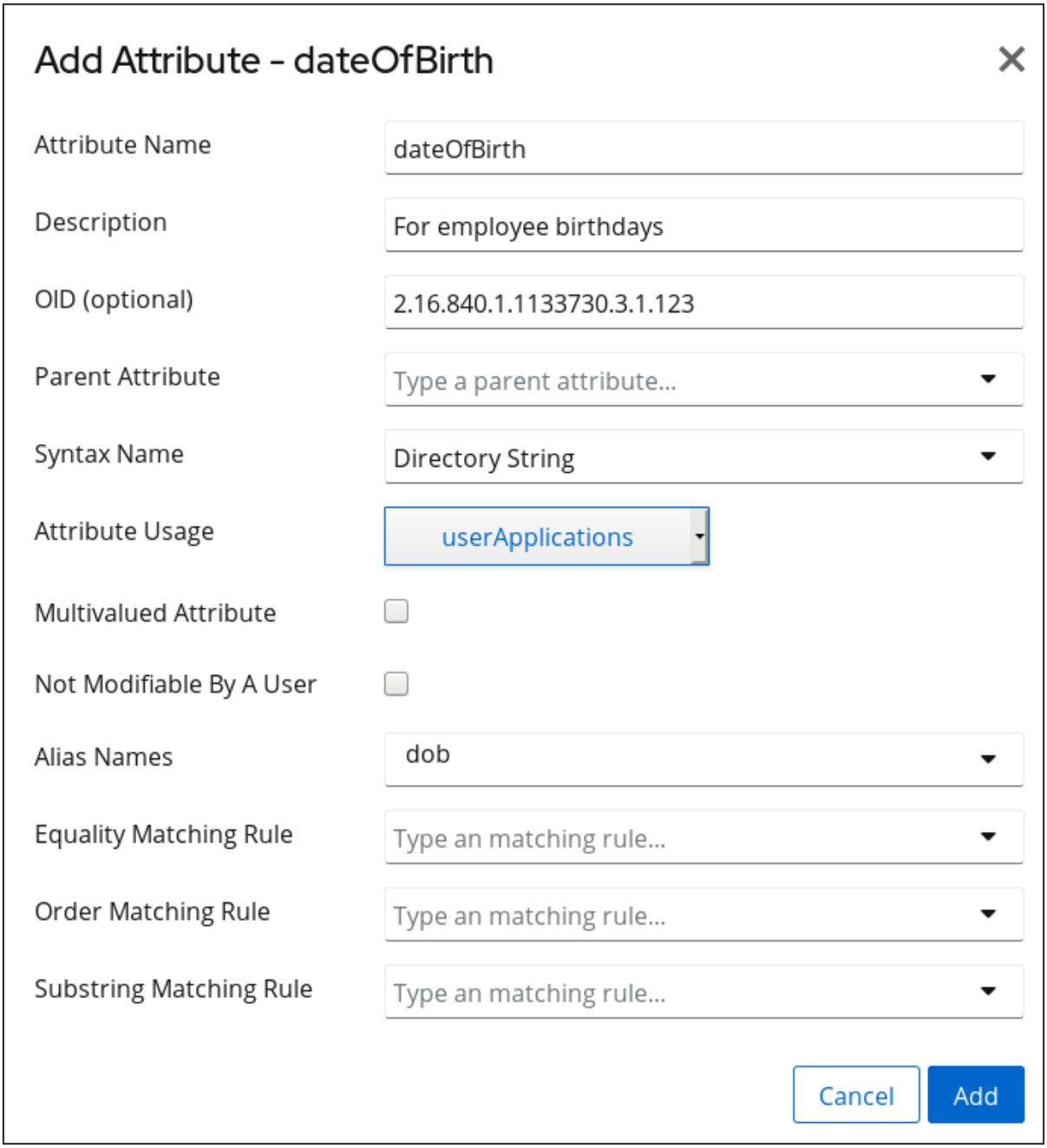
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com schema attributetypes add
dateofbirth --desc="For employee birthdays" --syntax="1.3.6.1.4.1.1466.115.121.1.15" --single-value
--x-origin="Example defined"
```

属性定義の詳細は、「属性」を参照してください。

12.6.2. Web コンソールを使用した属性の作成

Web コンソールを使用して属性を作成するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Schema → Attributes** を選択します。
4. **Add Attribute** をクリックします。
5. パラメーターを入力します。



6. **Add** をクリックします。

12.7. 属性の更新

本セクションでは、コマンドラインおよび Web コンソールを使用して属性を更新する方法を説明します。

12.7.1. コマンドラインを使用した属性の更新

dsconf ユーティリティーを使用して、属性エントリーを更新します。以下に例を示します。

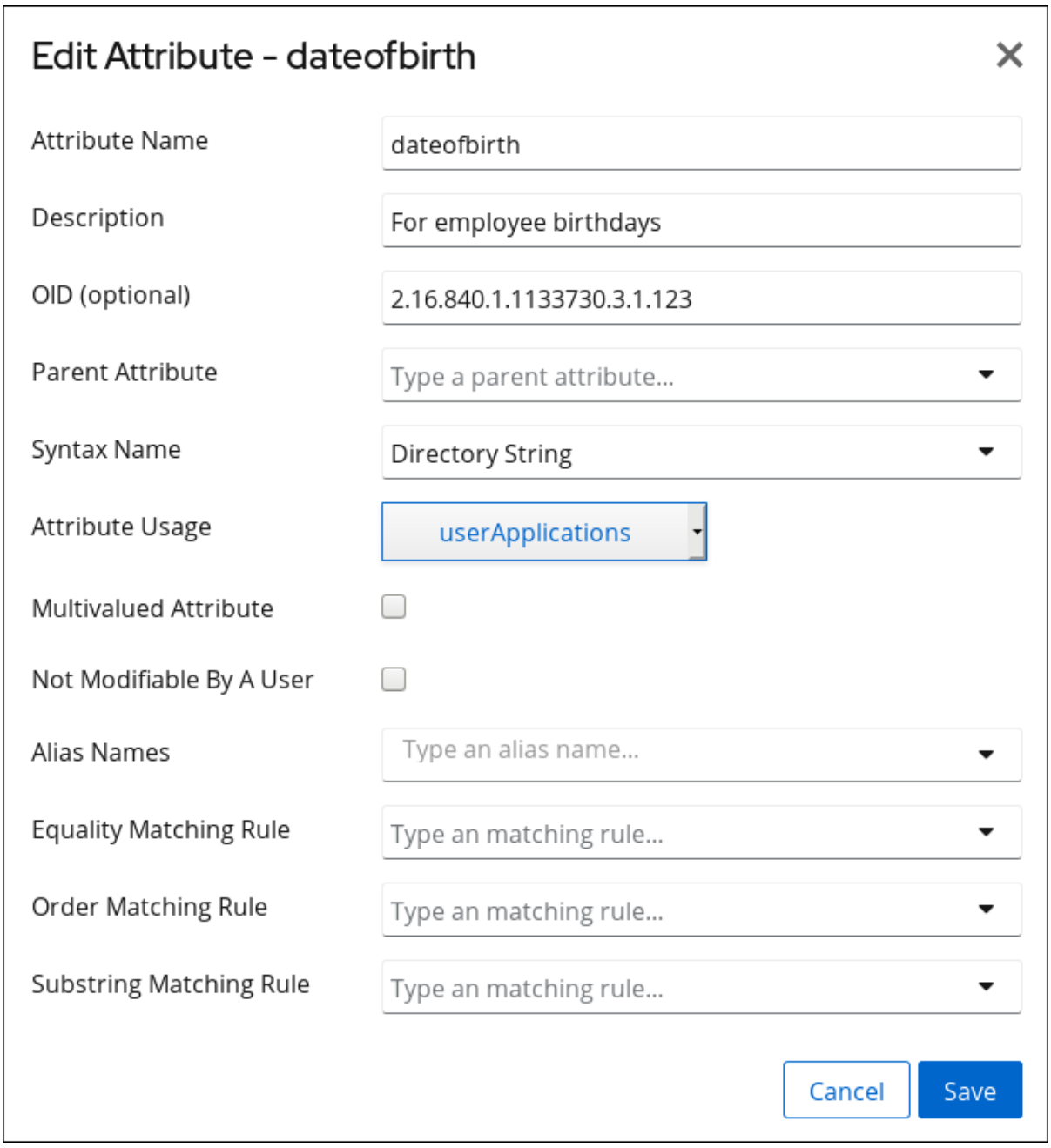
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com schema attributetypes replace
dateofbirth --desc="Employee birthday" --syntax="1.3.6.1.4.1.1466.115.121.1.15" --single-value --x-
origin="Example defined"
```

オブジェクトクラス定義の詳細は、「オブジェクトクラス」を参照してください。

12.7.2. Web コンソールを使用した属性の更新

Web コンソールを使用して属性を更新するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Schema → Attributes** を選択します。
4. 編集する属性の横にある **Choose Action** ボタンをクリックします。
5. **Edit Attribute** を選択します。
6. パラメーターを更新します。



7. **Save** をクリックします。

12.8. 属性の削除

本セクションでは、コマンドラインと Web コンソールを使用して属性を削除する方法を説明します。

12.8.1. コマンドラインを使用した属性の削除

ldapmodify ユーティリティーを使用して属性を削除します。以下に例を示します。

1. 不要な属性を使用するエントリーから、その属性を削除します。
2. 属性を削除します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com schema attributetypes
remove dateofbirth
```

オブジェクトクラス定義の詳細は、「オブジェクトクラス」を参照してください。

12.8.2. Web コンソールを使用した属性の削除

Web コンソールを使用して属性を削除するには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Schema → Objectclasses** を選択します。
4. 削除する属性の横にある **Choose Action** ボタンをクリックします。
5. **Delete Attribute** を選択します。
6. **Yes** をクリックして確定します。

12.9. カスタムスキーマファイルの作成

スキーマファイルは、**cn=schema** エントリーを定義する単純な LDIF ファイルです。各属性とオブジェクトクラスは、そのエントリーの属性として追加されます。スキーマファイルの作成要件を以下に示します。

- 最初の行は **dn: cn=schema** である必要があります。
- スキーマファイルには、属性とオブジェクトクラスの両方を含めることができますが、どちらか一方のみを含めることもできます。
- スタイルに属性とオブジェクトクラスの両方が定義されている場合は、最初にすべての属性がファイルに記載し、次にオブジェクトクラスを記載する必要があります。
- オブジェクトクラスは、他のスキーマファイルで定義された属性を使用できます。
- このファイルは、**[1-9][0-9]text.ldif** の形式で指定する必要があります。

  このファイルは、常に 2 つの数字で開始する必要があります。数値的には、コア設定スキーマ (**00** および **01**) の前にスキーマファイルを読み込ませることができません。

また、Directory Server は、常に、そのカスタムスキーマをスキーマディレクトリー内の数値およびアルファベット順で最も高い名前のスキーマファイルに書き込みます。このファイルは、**99user.ldif** であることを想定しています。このファイルが **99user.ldif** ではない場合には、サーバーで問題が発生する可能性があります。そのため、常に、カスタムスキーマファイルが、少なくともアルファベット順で **99user.ldif** よりも低くなることを確認します。名前 **99alpha.ldif** は問題ではありませんが、名前 **99zzz.ldif** は問題です。

スキーマファイル作成のプラクティスは、『デプロイメントガイド』 を参照してください。

属性は、スキーマへの ***attributetypes*** 属性として、5 つのコンポーネントがあるスキーマファイルで定義されます。

- OID (通常はドット区切り番号)
- **NAME 名前** 形式の一意の名前
- **DESC 説明** 形式の説明
- 属性値の構文の OID。(**SYNTAX OID** 形式については「Directory Server 属性の構文」 で説明)
- 任意で、属性が定義されているソース

以下に例を示します。

```
attributetypes: ( 1.2.3.4.5.6.1 NAME 'dateofbirth' DESC 'For employee birthdays' SYNTAX
1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUED X-ORIGIN 'Example defined')
```

同様に、オブジェクトクラスは ***objectclasses*** 属性の値として定義されますが、オブジェクトクラスの定義方法には若干柔軟性があります。必要な設定は、オブジェクトクラスの名前と OID のみになります。他のすべての設定は、オブジェクトクラスのニーズに依存します。

- OID (通常はドット区切り番号)
- **NAME 名前** 形式の一意の名前
- **DESC 説明** 形式の説明
- **SUP object_class** の形式で、このオブジェクトクラスの上位または親のオブジェクトクラス。関連する親がない場合は、**SUP top** を使用してください。
- **AUXILIARY** という単語で、オブジェクトクラスを適用するエントリーのタイプを指定します。**AUXILIARY** は、任意のエントリーに適用できることを意味します。
- **MUST** の後に続く必要な属性のリスト。複数の属性を含めるには、グループを括弧で囲み、ドル記号 ($) で属性を区切ります。
- **MAY** の後に続く許可される属性のリスト。複数の属性を含めるには、グループを括弧で囲み、ドル記号 ($) で属性を区切ります。

以下に例を示します。

```
objectclasses: ( 2.16.840.1133730.2.123 NAME 'examplePerson' DESC 'Example Person Object
Class' SUP inetOrgPerson AUXILIARY  MUST cn MAY (exampleDateOfBirth $
examplePreferredOS) )
```

例12.4「スキーマファイルの例」は、簡潔なスキーマファイルを示しています。

-

例12.4 スキーマファイルの例

```
dn: cn=schema
attributetypes: ( 2.16.840.1133730.1.123 NAME 'dateofbirth' DESC 'For employee birthdays'
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 X-ORIGIN 'Example defined')
objectclasses: ( 2.16.840.1133730.2.123 NAME 'examplePerson' DESC 'Example Person Object
Class' SUP inetOrgPerson AUXILIARY MAY (dateofbirth) )
```

カスタムスキーマファイルは Directory Server インスタンスのスキーマディレクトリー **/etc/dirsrv/slapd-*instance*/schema** に追加する必要があります。サーバーが再起動するか、動的に再読み込みされたタスクが実行されない限り、これらのファイルのスキーマは読み込まれず、サーバーで使用できなくなります。

重要

/usr/share/data/ ディレクトリーから標準スキーマを使用する場合は、スキーマファイルを **/usr/share/dirsrv/schema/** ディレクトリーにコピーします。標準スキーマが特定のインスタンスでのみ利用できるようにする必要がある場合は、スキーマファイルを **/etc/dirsrv/slapd-*instance_name*/schema/** ディレクトリーにコピーしますが、宛先ディレクトリーで別のファイル名を使用します。それ以外の場合は、Directory Server はアップグレード中にファイルの名前を変更し、**.bak** 接尾辞を追加します。

12.10. スキーマの動的再読み込み

デフォルトでは、Directory Server インスタンスが使用するスキーマファイルが、起動時にディレクトリーに読み込まれます。つまり、サーバーが再起動しない限り、スキーマディレクトリーに追加された新しいスキーマファイルが使用できません。Directory Server には、サーバーの再起動を必要とせずに、カスタムファイルを含む Directory Server インスタンスの完全なスキーマを手動で再読み込みするタスクがあります。

以下の方法の 1 つを使用すると、スキーマを再読み込みできます。

- **dsconf schema reload** コマンド。「**dsconf schema reload** コマンドを使用したスキーマの動的再読み込み」を参照してください。
- **cn=tasks** エントリー。「**cn=tasks** エントリーを使用したスキーマの動的再読み込み」を参照してください。

注記

Directory Server は常に、**/usr/share/dirsrv/schema/** ディレクトリーに保存されたすべてのスキーマファイルを再読み込みします。

12.10.1. dsconf schema reload コマンドを使用したスキーマの動的再読み込み

dsconf schema reload コマンドを使用してスキーマを再読み込みします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com schema reload
Attempting to add task entry... This will fail if Schema Reload plug-in is not enabled.
Successfully added task entry cn=schema_reload_2018-08-28T09:45:48.027962,cn=schema reload
task,cn=tasks,cn=config
To verify that the schema reload operation was successful, please check the error logs
```

デフォルトでは、Directory Server は、***nsslapd-schemadir*** パラメーターに設定したディレクトリーに保存されているスキーマファイルを再読み込みします。必要に応じて、**-d *directory*** オプションをコマンドに指定して、別のディレクトリーに保存されているスキーマを再読み込みします。

12.10.2. cn=tasks エントリーを使用したスキーマの動的再読み込み

Directory Server 設定の **cn=tasks,cn=config** エントリーは、サーバーがタスクの管理に使用する一時的なエントリー用のコンテナーエントリーです。スキーマ再読み込み操作を開始するには、**cn=schema reload task,cn=tasks,cn=config** エントリーでタスクを作成します。

デフォルトでは、Directory Server は、***nsslapd-schemadir*** パラメーターに設定したディレクトリーに保存されているスキーマファイルを再読み込みします。たとえば、このパラメーターで設定したディレクトリーに保存されたスキーマファイルを再読み込みするには、次のコマンドを実行します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=example_schema_reload,cn=schema reload task,cn=tasks,cn=config
objectclass: extensibleObject
cn: cn=example_schema_reload
```

オプションで、***schemadir*** パラメーターを指定して、別のディレクトリーに保存されているスキーマを再読み込みします。以下に例を示します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=example_schema_reload,cn=schema reload task,cn=tasks,cn=config
objectclass: extensibleObject
cn: cn=example_schema_reload
schemadir: /example/schema/
```

タスクが完了すると、エントリーはディレクトリー設定から削除されます。

cn=schema reload task,cn=tasks,cn=config エントリーの詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の 『cn=schema 再読み込みタスク』 を参照してください。

12.10.3. レプリケーショントポロジーでのスキーマの再読み込み

スキーマの再読み込みタスクはローカル操作であるため、スキーマが 1 つのサプライヤーに追加され、他のサプライヤーに追加されない場合は、スキーマの変更がマルチサプライヤー環境で複製されません。全サプライヤーサーバーに新しいスキーマファイルを読み込むには、次のコマンドを実行します。

1. レプリケーションを停止します。「レプリケーションの無効化および再有効化」を参照してください。
2. 新しいスキーマファイルをコピーし、各サプライヤーおよびレプリカサーバーに対してスキーマ再読み込みタスクを実行します。参照:
   - 「**dsconf schema reload** コマンドを使用したスキーマの動的再読み込み」
   - 「**cn=tasks** エントリーを使用したスキーマの動的再読み込み」
3. レプリケーションを再起動します。「レプリケーションの無効化および再有効化」を参照してください。

12.10.4. スキーマの再読み込みエラー

スキーマ再読み込みタスクが実行しても、サーバーは正常に完了するかどうかを返しません。スキーマ再読み込み操作が正常に行われたことを確認するには、エラーログを確認します。スキーマの再読み込みには、最初にスキーマファイルを検証して読み込む 2 つのタスクがあります。

成功メッセージは、検証が渡され、タスクが完了したことを示しています。

```
[06/Jan/2009:17:52:04.001214874 -0500] schemareload - Schema reload task starts (schema dir:
default) ...
[06/Jan/2009:17:52:04.754335904 -0500] schemareload - Schema validation passed.
[06/Jan/2009:17:52:04.894255328 -0500] schemareload - Schema reload task finished.
```

失敗したステップがある場合は、失敗したステップおよびその理由がログに表示されます。

```
[..] schemareload - Schema reload task starts (schema dir: /bogus) ...
[..] schema - No schema files were found in the directory /bogus
[..] schema_reload - schema file validation failed
[..] schemareload - Schema validation failed.
```

12.11. スキーマチェックのオンとオフを切り替える

スキーマチェックがオンの場合、Directory Server では以下の 3 つの内容が確保されます。

- 使用するオブジェクトクラスおよび属性はディレクトリースキーマで定義されます。
- オブジェクトクラスに必要な属性はエントリーに含まれます。
- オブジェクトクラスで使用できる属性のみがエントリーに含まれます。

重要

Red Hat は、スキーマチェックを無効にしないことを推奨します。

Directory Server では、スキーマチェックはデフォルトで有効になっており、Directory Server は常にスキーマチェックが有効な状態で実行します。スキーマの確認をオフにしておくことは LDAP インポート操作を迅速化するのが唯一の状況です。ただし、スキーマに準拠していないエントリーをインポートするリスクがあります。したがって、このエントリーを更新することはできません。

12.11.1. コマンドラインでスキーマチェックのオンおよびオフを切り替え

スキーマチェックのオンとオフを切り替えるには、***nsslapd-schemacheck*** パラメーターの値を設定します。スキーマの確認を無効にするには、次のコマンドを実行します。

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

nsslapd-schemacheck パラメーターの詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 で、パラメーターの説明を参照してください。

12.11.2. Web コンソールを使用したスキーマチェックのオンおよびオフの切り替え

Web コンソールを使用してスキーマチェックを有効または無効にするには、以下を行います。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** を開き、**Server Settings** エントリーを選択します。
4. **Advanced Settings** タブを開きます。
5. スキーマチェックを有効にするには、**Enable Schema Checking** チェックボックスを選択します。この機能を無効にするには、チェックボックスの選択を解除します。
6. **Save** をクリックします。

12.12. 構文の検証の使用

構文の検証では、Directory Server は、属性の値が、その属性の定義に指定された構文のルールに従うことを確認します。たとえば、構文の検証では、新しい ***telephoneNumber*** 属性に、実際にその値に有効な電話番号が指定されていることを確認します。

重要

Red Hat は、構文の検証を無効にしないことを推奨します。

12.12.1. 構文の検証の概要

スキーマチェックと同様に、検証によりディレクトリーの変更がレビューされ、構文に違反する変更を拒否します。オプションとして追加の設定を行い、構文検証により構文違反に関する警告メッセージをログに記録し、変更を拒否したり、変更プロセスを正常に実行できるようにしたりすることもできます。

この機能は、バイナリー構文 (検証できない) および標準以外の構文 (定義された必要な形式がない) を除き、すべての属性構文を検証します。構文は RFC 4514 に対して検証されます。

12.12.2. 構文の検証およびその他の Directory Server 操作

構文の検証は、エントリーの作成 (add) や属性の編集 (modify) などの標準の LDAP 操作に主に関係します。ただし、属性の構文の検証は他の Directory Server 操作に影響を及ぼす可能性があります。

データベース暗号化

通常の LDAP 操作では、値がデータベースに書き込まれる直前に属性は暗号化されます。これは、属性構文の検証 **後に** 暗号化が実行されることを意味します。

暗号化されたデータベース (*10章 属性暗号化の設定* で説明されているように) をエクスポートおよびインポートすることができます。通常、これらのエクスポート操作およびインポート操作は **db2ldif** および **ldif2db** とともに **-E** フラグを使用して行うことが強く推奨されます。これにより、インポート操作で構文の検証が問題になる可能性もあります。ただし、**-E** フラグを使用せずに暗号化されたデータベースをエクスポートする場合は (サポートされていない)、暗号化された値で LDIF が作成されます。この LDIF をインポートすると、暗号化された属性を検証できず、警告がログに記録され、インポートされたエントリーで属性検証はスキップされます。

同期

Windows Active Directory エントリーと Red Hat Directory Server エントリーでは、属性の許容構文または強制構文に違いがある場合があります。この場合、構文の検証により Directory Server エントリーの RFC 標準が強制されるため、Active Directory の値を適切に同期できませんでした。

レプリケーション

Directory Server 11 インスタンスがコンシューマーに変更を複製するサプライヤーである場合は、構文検証の使用に問題はありません。ただし、レプリケーションのサプライヤーが古いバージョンの Directory Server であったり、構文の検証が無効になっていたりする場合は、Directory Server 11 コンシューマーは、サプライヤーが許可する属性値を拒否する可能性があるため、構文の検証をコンシューマーで使用しないでください。

12.12.2.1. コマンドラインを使用した構文検証のオンおよびオフを切り替え

構文の検証をオンまたはオフにするには、***nsslapd-syntaxcheck*** パラメーターの値を設定します。構文の検証を無効にするには、次のコマンドを実行します。

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

nsslapd-syntaxcheck パラメーターの詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 で、パラメーターの説明を参照してください。

12.12.2.2. Web コンソールを使用した構文検証のオンおよびオフの切り替え

Web コンソールを使用して構文の検証を有効または無効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** を開き、**Server Settings** エントリーを選択します。
4. **Advanced Settings** タブを開きます。
5. 属性構文の確認を有効にするには、**Enable Attribute Syntax Checking** チェックボックスを選択します。この機能を無効にするには、チェックボックスの選択を解除します。
6. **Save** をクリックします。

12.12.3. DN の厳格な構文検証の有効化または無効化

構文の検証が有効な場合は、RFC 4514 のセクション 3 で説明されているように、識別名 (DN) が検証されます。DN 構文の検証は、後の標準の厳格さが異なる構文を持つ古い DN、つまりディレクトリーツリーを無効にする可能性があるため、個別に有効になります。

注記

厳格な DN 検証が有効になり、DN 値が必要な構文に準拠しない場合は、LDAP の結果コード **34**、**INVALID_DN_SYNTAX** で操作は失敗します。

12.12.3.1. コマンドラインを使用した DN の厳格な構文の検証の有効化または無効化

DN のオンおよびオフの構文検証を厳格にするには、***nsslapd-dn-validate-strict*** パラメーターの値を設定します。たとえば、DN の厳格な構文検証を無効にするには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-dn-validate-strict=off
Successfully replaced "nsslapd-dn-validate-strict"
```

nsslapd-syntaxcheck パラメーターの詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 で、パラメーターの説明を参照してください。

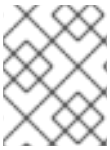
12.12.3.2. Web コンソールを使用した DN の厳格な構文検証の有効化または無効化

Web コンソールを使用した DN の厳格な構文検証を有効または無効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** を開き、**Server Settings** エントリーを選択します。
4. **Advanced Settings** タブを開きます。
5. 機能を有効または無効にするかどうかに応じて、**Strict DN Syntax Validation** オプションを選択または選択解除します。
6. **Save** をクリックします。

12.12.4. 構文検証ロギングの有効化

デフォルトでは、構文の検証は、属性値が必要な構文に違反する追加または変更の操作を拒否します。ただし、違反自体は、デフォルトでは errors ログに記録されません。***nsslapd-syntaxlogging*** 属性は、構文違反のエラーロギングを有効にします。



注記

構文検証スクリプトおよびタスクによって検出された構文違反は、Directory Server エラーログに記録されます。

nsslapd-syntaxlogging パラメーターおよび ***nsslapd-syntaxcheck*** パラメーターの両方が有効な場合には、無効な属性の変更が拒否され、メッセージがログに書き込まれます。***nsslapd-syntaxlogging*** が有効、***nsslapd-syntaxcheck*** が無効の場合、操作は成功できますが、警告メッセージがエラーログに書き込まれます。

12.12.4.1. コマンドラインを使用した構文検証ロギングの有効化

構文検証ログを有効にするには、***nsslapd-syntaxlogging*** パラメーターの値を **on** に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-syntaxlogging=on
Successfully replaced "nsslapd-syntaxlogging"
```

nsslapd-syntaxlogging パラメーターの詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 で、パラメーターの説明を参照してください。

12.12.4.2. Web コンソールを使用した構文検証ロギングの有効化

Web コンソールを使用して検証ロギングを有効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** を開き、**Server Settings** エントリーを選択します。
4. **Advanced Settings** タブを開きます。
5. **Enable Attribute Syntax Logging** オプションを選択します。
6. **Save** をクリックします。

12.12.5. 既存の属性値の構文の検証

特定の状況では、既存の値の構文を手動で検証したい場合があります。以下に例を示します。

- ***nsslapd-syntaxcheck*** パラメーターで構文の検証が無効になっている場合。詳細については、「構文の検証およびその他の Directory Server 操作」 を参照してください。

重要

Red Hat は、構文の検証を無効にしないことを推奨します。

- 構文検証なしまたは無効化されたサーバーからデータを移行する場合。

Directory Server は、構文検証タスクの結果を **/var/log/dirsrv/slapd-*instance_name*/errors** ファイルに記録します。以下に例を示します。

- 検証済みの値がすべて有効であれば、以下を実行します。

```
[28/Jun/2017:12:52:43.669867966 +0200] - ERR - syntax-plugin -
syntax_validate_task_thread - Starting (base: "dc=example,dc=com", filter: "(objectclass=*)")
...
[28/Jun/2017:12:52:43.696850129 +0200] - ERR - syntax-plugin -
syntax_validate_task_thread - Complete.  Found 0 invalid entries.
```

- 無効なエントリーが見つかった場合は、以下を行います。

```
[28/Jun/2017:12:54:05.736087520 +0200] - ERR - syntax-plugin -
syntax_validate_task_thread - Starting (base: "dc=example,dc=com", filter: "(objectclass=*)")
...
[28/Jun/2017:12:54:05.754195607 +0200] - ERR - syntax-plugin -
syntax_validate_task_callback - Entry "cn=user,ou=People,dc=example,dc=com" violates
syntax.
description: value #0 invalid per syntax
[28/Jun/2017:12:54:05.759905671 +0200] - ERR - syntax-plugin -
syntax_validate_task_thread - Complete.  Found 1 invalid entries.
```

注記

構文検証タスクは、構文違反のみを識別します。誤った値を手動で修正する必要があります。

12.12.5.1. dsconf スキーマの validate-syntax コマンドを使用した構文検証タスクの作成

dsconf schema validate-syntax コマンドを使用して、構文検証タスクを作成します。たとえば、**(objectclass=inetorgperson)** フィルターに一致する **ou=People,dc=example,dc=com** サブツリーのすべての値の構文を検証するタスクを作成するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com schema validate-syntax -f
'(objectclass=inetorgperson)' ou=People,dc=example,dc=com
```

12.12.5.2. cn=tasks エントリーを使用した構文検証タスクの作成

Directory Server 設定の **cn=tasks,cn=config** エントリーは、サーバーがタスクの管理に使用する一時的なエントリー用のコンテナーエントリーです。構文の検証操作を開始するには、**cn=syntax validate,cn=tasks,cn=config** エントリーにタスクを作成します。

たとえば、**(objectclass=inetorgperson)** フィルターに一致する **ou=People,dc=example,dc=com** サブツリーのすべての値の構文を検証するタスクを作成するには、次のコマンドを実行します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=example_syntax_validate,cn=syntax validate,cn=tasks,cn=config
objectclass: extensibleObject
cn: cn=example_syntax_validate
basedn: ou=People,dc=example,dc=com
filter: (objectclass=inetorgperson)
```

タスクが完了すると、エントリーはディレクトリー設定から削除されます。

cn=syntax validate,cn=tasks,cn=config エントリーの詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の 『cn=schema 再読み込みタスク』 を参照してください。

第13章 インデックスの管理

インデックス付けは、属性または値を分類および整理することにより、情報の検索と取得を容易にします。本章では、検索アルゴリズム自体、コンテキストにインデックスのメカニズムを配置し、インデックスを作成、削除、および管理する方法を説明します。

13.1. インデックスの概要

本セクションでは、Directory Server でのインデックスの概要を説明します。これには、以下のトピックが含まれます。

- 「インデックスタイプの概要」
- 「デフォルトインデックスおよびデータベースインデックスの概要」
- 「検索アルゴリズムの概要」
- 「インデックスのメリットとのバランス」

13.1.1. インデックスタイプの概要

インデックスはディレクトリーのデータベースにあるファイルに保存されます。ファイルの名前は、インデックス化された属性に基づいて、ファイルに含まれるインデックスの型は生成されません。各インデックスファイルには、特定の属性に対して複数のインデックスが保持されると、複数のインデックスが含まれる場合があります。たとえば、共通の name 属性用に保持されるすべてのインデックスは **cn.db** ファイルに含まれます。

Directory Server は、以下のタイプのインデックスをサポートします。

- **Presence index (pres)** には、特定の属性を含むエントリーのリストが含まれており、検索には非常に便利です。たとえば、アクセス制御情報を含むエントリーを簡単に検証できます。プレゼンスインデックスを含む **aci.db** ファイルを生成すると、**ACI=*** の検索を効率的に実行して、サーバーのアクセス制御リストを生成します。

- **Equality index (eq)** により、特定の属性値を含むエントリーの検索が改善されます。たとえば、***cn*** 属性の等価インデックスを使用すると、ユーザーは **cn=Babs Jensen** の検索をより効率的に実行できます。

- **Approximate index (approx)** は、効率的な近似検索や sounds-like 検索に使われます。たとえば、エントリーには属性値 **cn=Firstname M Lastname** を含めることができます。概算検索では、この値を **cn~=Firstname Lastname**、**cn~=Firstname**、または **cn~=Lastname** に対する検索に対して返します。同様に、**l~=San Fransisco** (スペルミスに注意) を検索すると、**l=San Francisco** を含むエントリーが返されます。

- **Substring index (sub)** は、維持するコストのかかるインデックスですが、エントリー内の部分文字列に対して効率的な検索が可能になります。部分文字列のインデックスは、各エントリーの最小 3 文字に制限されます。

  たとえば、**cn=*derson** の形式で検索すると、**Bill Anderson**、**Jill Henderson**、または **Steve Sanderson** といった文字列が含まれる共通名と一致します。同様に、**telephoneNumber=*555*** の検索は、**555** が含まれる電話番号を持つディレクトリー内の全エントリーを返します。

- **国際インデックス** は、国際ディレクトリー内の情報の検索を迅速化します。国際インデックスの作成プロセスは、通常のインデックスを作成するプロセスと似ています。ただし、**オブジェクト識別子** (OID) をインデックス化する属性に関連付けることで **一致するルール** を適用する点

が異なります。

サポートされるロケールおよび関連付けられた OID が付録D *国際化*にリスト表示されています。追加のマッチングルールを受け入れるように Directory Server を設定する必要がある場合は、Red Hat コンサルティングにお問い合わせください。

13.1.2. デフォルトインデックスおよびデータベースインデックスの概要

Directory Server には、一連のデフォルトインデックスが含まれます。新規データベースの作成時に、Directory Server はこれらのデフォルトインデックスを **cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config** から新規データベースにコピーします。次に、データベースはこれらのインデックスのコピーのみを使用します。このインデックスは **cn=index,cn=*database_name*,cn=ldbm database,cn=plugins,cn=config** に保存されます。

注記

Directory Server は cn=config エントリーの設定を複製しません。したがって、レプリケーショントポロジーの一部であるサーバーでは、インデックスを異なる方法で設定できます。たとえば、レプリケーションがカスケードする環境では、クライアントがハブからデータを読み取らない場合は、ハブにカスタムインデックスを作成する必要はありません。

Directory Server のデフォルトインデックスを表示するには、以下を実行します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com \
    -b "cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config" \
    '(objectClass=nsindex)'
```

注記

cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config に保存されているデフォルトのインデックス設定を更新しても、変更は **cn=index,cn=*database_name*,cn=ldbm database,cn=plugins,cn=config** の個々のデータベースには適用されません。

個別のデータベースのインデックスを表示するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend index list database_name
```

13.1.3. 検索アルゴリズムの概要

インデックスを使用して検索を迅速化します。ディレクトリーがインデックスをどのように使用するかを理解するには、検索アルゴリズムを理解するのに役立ちます。各インデックスには、(***cn***、共通名、属性など) 属性の一覧と、インデックス化された属性値が含まれるエントリーの ID の一覧が含まれます。

1. LDAP クライアントアプリケーションは、検索要求をディレクトリーに送信します。
2. ディレクトリーは、受信要求を調べて、指定したベース DN が、1 つ以上のデータベースまたはデータベースリンクに含まれる接尾辞と一致することを確認します。
   - 一致する場合には、ディレクトリーはリクエストを処理します。

- 一致しない場合、ディレクトリーは接尾辞と一致しないことを示すエラーをクライアントに返します。**cn=config** 下の **nsslapd-referral** 属性で参照を指定している場合、ディレクトリーは、クライアントがリクエストを購入できる LDAP URL も返します。

- Directory Server は、どのインデックスが適用されるかを確認するための検索フィルターを調べ、フィルターを満たす各インデックスからエントリー ID のリストを読み込もうとします。ID リストは、フィルターで使用されている AND または OR に参加するかによって組み合わせられます。

  各フィルターコンポーネントは個別に処理され、ID リストを返します。

- エントリー ID の一覧が設定された ID リストのスキャン制限よりも大きい場合や、属性にインデックスが定義されていない場合、Directory Server は、この **filtercomponent** の結果を **allids** に設定します。個々の検索コンポーネントの結果に論理操作を適用すると、その一覧は引き続き **ALLIDs** のままになる場合は、データベース内のすべてのエントリーを検索します。これは **インデックスのない** 検索です。

3. Directory Server は、ID リストのすべてのエントリー ID について、**id2entry.db** データベースまたはエントリーキャッシュから (またはインデックスなしの検索の場合はデータベース全体から) すべてのエントリーを読み取ります。その後、サーバーはエントリーをチェックして、検索フィルターと一致するかどうかを確認します。それぞれの一致が見つかったら、それが返されます。

   サーバーは、すべての候補エントリーを検索するか、設定されたリソース制限に達するまで、ID のリストを検索し続けます。(リソース制限は「コマンドラインを使用したユーザーおよびグローバルリソース制限の設定」にリスト表示されます。)

注記

簡単なページ化された結果制御を使用して、検索に対して別のリソース制限を設定できます。たとえば、管理者は、高いサイズまたは無制限サイズを設定し、ページ化された検索で制限を検索しますが、ページのない検索には低いデフォルト制限を使用します。

13.1.4. おおよその検索

また、このディレクトリーでは、metaphone 表音アルゴリズムのバリエーションを用いて、近似的なインデックスで検索を行っています。各値は単語のシーケンスとして処理され、各単語について電話番号が生成されます。

注記

Directory Server の metaphone 表音アルゴリズムは US-ASCII 文字のみをサポートします。したがって、インデックスは、英語の値でのみ使用してください。

概算検索に入力した値は、表音コードシーケンスに変換されます。以下の両方が当てはまる場合、エントリーはクエリーに一致すると考えられます。

- すべてのクエリー文字列コードは、エントリー文字列に生成されたコードと一致します。
- クエリー文字列コードはすべて、エントリー文字列コードと同じ順序で実行されます。

ディレクトリーの名前 (フォネティックコード)	クエリー文字列 (フォネティックコード)	一致のコメント
Alice B Sarette (ALS B SRT)	Alice Sarette (ALS SRT)	一致。コードが正しい順序で指定されます。
	Alice Sarrette (ALS SRT)	一致。コードは、Sarette が間違っているにもかかわらず、正しい順序で指定されます。
	Surette (SRT)	一致。生成されたコードは、Sarette のスペルが間違っているにもかかわらず、元の名前で存在しています。
	Bertha Sarette (BR0 SRT)	一致するものはありません。コード BR0 は元の名前に存在しません。
	Sarette, Alice (SRT ALS)	一致するものはありません。コードが正しい順序で指定されていません。

13.1.5. インデックスのメリットとのバランス

新しいインデックスを作成する前に、インデックスを維持することのメリットとコストのバランスを考えてください。

- 概算インデックスは、通常、数字を含む属性 (電話番号など) には効率的ではありません。
- 部分文字列のインデックスはバイナリー属性では機能しません。
- 等価インデックスは、値が大きい場合に使用しないようにしてください (例: 暗号化データを含む写真やパスワードを含む属性など)。
- 検索であまり使用されない属性のインデックスを維持することは、グローバル検索のパフォーマンスを向上させることなく、オーバーヘッドを増加させます。
- インデックス化されていない属性は、検索要求で依然として指定できますが、検索のタイプによっては検索パフォーマンスが大幅に低下する可能性があります。
- 保守するインデックスが多いほど、必要なディスク領域が多くなります。

インデックスは、非常に時間がかかります。以下に例を示します。

1. Directory Server は add 操作または modify 操作を受け取ります。
2. Directory Server は indexing 属性を調べ、属性値に対してインデックスが維持されているかどうかを判断します。
3. 作成した属性値がインデックス化されると、Directory Server はインデックスから新しい属性値を追加または削除します。

4. 実際の属性値はエントリーに作成されます。

たとえば、Directory Server はエントリーを追加します。

```
dn: cn=John Doe,ou=People,dc=example,dc=com
objectclass: top
objectClass: person
objectClass: orgperson
objectClass: inetorgperson
cn: John Doe
cn: John
sn: Doe
ou: Manufacturing
ou: people
telephoneNumber: 408 555 8834
description: Manufacturing lead for the Z238 line of widgets.
```

Directory Server は以下のインデックスを維持します。

- ***cn*** (通称) および ***sn*** (姓) 属性の等価、近似、および部分文字列インデックス。
- 電話番号属性の等価および部分文字列のインデックス。
- 説明属性の部分文字列インデックス。

そのエントリーをディレクトリーに追加する場合は、Directory Server で以下の手順を実行する必要があります。

1. **John** と **John Doe** の ***cn*** 等価インデックスエントリーを作成します。
2. **John** と **John Doe** の適切な ***cn*** 近似インデックスエントリーを作成します。
3. **John** と **John Doe** の適切な ***cn*** 部分文字列インデックスエントリーを作成します。
4. **Doe** の ***sn*** 等価インデックスエントリーを作成します。
5. **Doe** の適切な ***sn*** 近似インデックスエントリーを作成します。
6. **Doe** の適切な ***sn*** 部分文字列インデックスエントリーを作成します。
7. **408 555 8834** の電話番号等価インデックスエントリーを作成します。
8. **408 555 8834** の適切な電話番号部分文字列インデックスエントリーを作成します。
9. **Manufacturing lead for the Z238 line of widgets** の適切な説明部分文字列インデックスエントリーを作成します。この文字列に対して多数の部分文字列エントリーが生成されます。

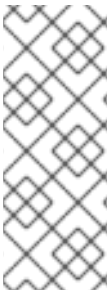
この例が示すように、大規模なディレクトリーのデータベースの作成および維持に必要なアクションの数は、リソースを必要とします。

13.1.6. インデックスの制限

nsrole や ***cos_attribute*** などの仮想属性をインデックス化できません。仮想属性には計算値が含まれます。これらの属性をインデックス化すると、Directory Server は無効なエントリーセットを返して直接的かつ内部検索を行うことができます。

13.2. 標準インデックスの作成

本セクションでは、コマンドラインと Web コンソールを使用した特定の属性について、存在、等価、概算、部分文字列、および国際指標を作成する方法を説明します。

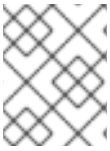


注記

新しいインデックスタイプを作成すると、Directory Server はこのデフォルトインデックスを、今後作成する新規データベースのテンプレートとして使用します。デフォルトのインデックスを更新すると、更新された設定は既存のデータベースに適用されません。新しいインデックスを既存のデータベースに適用するには、「既存のデータベースへの新規インデックスの作成」 で説明されているように、**dsctl db2index** コマンドまたは **cn=index,cn=tasks** タスクを使用します。

- 「Web コンソールを使用したインデックスの作成」
- 「コマンドラインを使用したインデックスの作成」

13.2.1. コマンドラインを使用したインデックスの作成

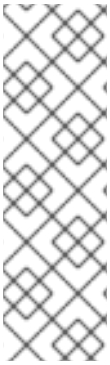


注記

システムインデックスは Directory Server にハードコーディングされるため、新しいシステムインデックスを作成できません。

ldapmodify を使用して、新しいインデックス属性をディレクトリーに追加します。

- デフォルトインデックスのいずれかになる新しいインデックスを作成するには、新しいインデックス属性を **cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config** エントリーに追加します。
- 特定のデータベースに新しいインデックスを作成するには、作成するインデックスを **cn=index,cn=database_name,cn=ldbm database,cn=plugins,cn=config** エントリーに追加します。ここで、**cn=database_name** はデータベースの名前に対応します。



注記

dse.ldif ファイルの **cn=config** の下にエントリーを作成しないでください。**dse.ldif** 設定ファイルの **cn=config** エントリーは、通常のエントリーと同じ拡張性の高いデータベースには保存されません。その結果、多くのエントリー (特に頻繁に更新される可能性のあるエントリー) を **cn=config** に保存すると、パフォーマンスが低下する可能性があります。パフォーマンスの理由から、**cn=config** に単純なユーザーエントリーを保存することは推奨していませんが、設定情報が一元化されるため、**cn=config** に Directory Manager エントリーやレプリケーションマネージャー (サプライヤーのバインド DN) エントリーなどの特別なユーザーエントリーを保存すると便利です。

エントリーの追加に必要な LDIF 更新ステートメントの詳細は、「ディレクトリーエントリーの更新」を参照してください。

たとえば、**Example1** データベースに ***sn*** (姓 (surname)) 属性の有無、等価、および部分文字列インデックスを作成するには、以下を実行します。

1. **ldapmodify** を実行して、新しいインデックスの LDIF エントリーを追加します。

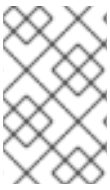
```
# ldapmodify -a -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=sn,cn=index,cn=Example1,cn=ldbm database,cn=plugins,cn=config
changetype: add
objectClass:top
objectClass:nsIndex
cn:sn
nsSystemIndex:false
nsIndexType:pres
nsIndexType:eq
nsIndexType:sub
nsMatchingRule:2.16.840.1.113730.3.3.2.3.1
```

cn 属性には、インデックスの属性の名前 (この例では ***sn*** 属性) が含まれます。エントリーは **nsIndex** オブジェクトクラスのメンバーです。**nsSystemIndex** 属性は **false** で、インデックスが Directory Server 操作に不可欠ではないことを示します。複数値の **nsIndexType** 属性は、存在 (**pres**)、等価 (**eq**)、および部分文字列 (**sub**) のインデックスを指定します。各キーワードは別々の行で入力する必要があります。この例の **nsMatchingRule** 属性は、ブルガリア語の照合順序の OID を指定しています。マッチングルールは、言語や、日付や整数などの他のフォーマットなど、値の一致の可能性を示すことができます。

nsIndexType 属性の **none** キーワードを使用して、インデックスが属性に対して維持されないように指定できます。この例では、**nsIndexType** を **none** に変更して、**Example1** データベースの ***sn*** インデックスを一時的に無効にします。

```
dn: cn=sn,cn=index,cn=Example1,cn=ldbm database,cn=plugins,cn=config
objectClass:top
objectClass:nsIndex
cn:sn
nsSystemIndex:false
nsIndexType:none
```

マッチングルールとその OID の完全リストは、「一致するルールの使用」 を参照してください。インデックス設定属性は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 を参照してください。



注記

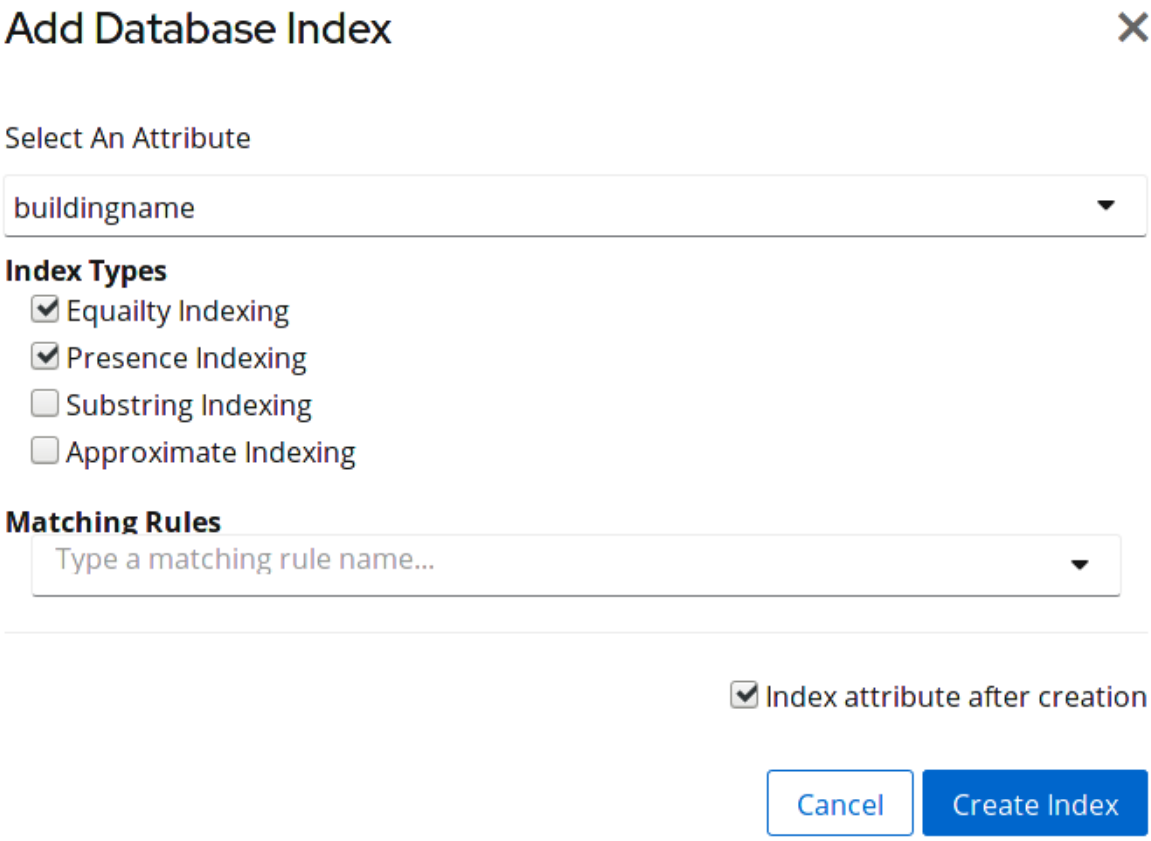
インデックスの作成時に、(属性のエイリアスではなく) 属性のプライマリー名を常に使用します。属性の主な名前は、スキーマの属性に対して最初に一覧表示される名前です。たとえば、ユーザー ID 属性の **uid** です。

13.2.2. Web コンソールを使用したインデックスの作成

存在、等価性、概算、部分文字列、または国際インデックスを作成するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. 接尾辞エントリーを選択します。

5. **Indexes** タブを開きます。
6. **Add Index** ボタンをクリックします。
7. インデックスの属性、インデックスのタイプ、および任意でマッチングルールを選択します。



8. **Create Index** をクリックします。

13.3. 既存のデータベースへの新規インデックスの作成

Directory Server でインデックス操作を開始する方法を説明します。Directory Server はデータベースを自動的にインデックスしないため、インデックスを手動で作成する必要があります。

重要

インデックスを再生成する前に検索を続行しますが、誤った結果または一貫性のない結果が返される場合があります。

13.3.1. インスタンスの実行中にインデックスの作成

13.3.1.1. dsconf backend index reindex コマンドを使用したインデックスの作成

インスタンスの実行中にデータベースのインデックスを再作成するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend index reindex
database_name
```

13.3.1.2. cn=tasks エントリーを使用したインデックスの作成

Directory Server 設定の **cn=tasks,cn=config** エントリーは、サーバーがタスクの管理に使用する一時的なエントリー用のコンテナーエントリーです。インデックス操作を開始するには、**cn=index,cn=tasks,cn=config** エントリーでタスクを作成します。

ldapadd ユーティリティーを使用して、新しいインデックスタスクを追加します。たとえば、**userRoot** データベースに ***cn*** 属性の presence インデックスを作成するタスクを追加するには、以下を実行します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=example_presence_index,cn=index,cn=tasks,cn=config
objectclass: top
objectclass: extensibleObject
cn: example presence index
nsInstance: userRoot
nsIndexAttribute: "cn:pres"
```

タスクが完了すると、エントリーはディレクトリー設定から削除されます。

cn=index,cn=tasks,cn=config エントリーの詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』の『cn=index』セクションを参照してください。

13.3.2. インスタンスのオフライン時におけるインデックスの作成

インデックスエントリーの作成、または既存のインデックスエントリーにインデックスタイプを追加したら、**dsconf db2index** コマンドを使用します。

1. インスタンスをシャットダウンします。

```
# dsctl instance_name stop
```

2. インデックスを再作成します。

```
# dsctl instance_name db2index userRoot
[13/Aug/2019:15:25:37.277426483 +0200] - INFO -
ldbm_instance_config_cachememsize_set - force a minimal value 512000
[13/Aug/2019:15:25:37.289257996 +0200] - INFO - check_and_set_import_cache -
pagesize: 4096, available bytes 1704378368, process usage 22212608
[13/Aug/2019:15:25:37.291738104 +0200] - INFO - check_and_set_import_cache - Import
allocates 665772KB import cache.
...
db2index successful
```

3. インスタンスを起動します。

```
# dsctl instance_name start
```

13.4. 仮想リストビューコントロールを使用して、大規模な検索結果の連続したサブセットを要求する

Directory Server は、LDAP 仮想リストビューコントロールをサポートしています。この制御により、LDAP クライアントは大規模な検索結果の連続したサブセットを要求できます。

たとえば、Directory Server に 100.000 エントリーのアドレス帳を保存したとします。デフォルトでは、すべてのエントリーのクエリーはすべてのエントリーを一度に返します。これはリソースと時間のかかる操作であり、ユーザーが結果をスクロールすると一部のセットしか表示されないため、クライアントはデータセット全体を必要としないことがよくあります。

ただし、クライアントが VLV コントロールを使用する場合、サーバーはサブセットのみを返します。たとえば、ユーザーがクライアントアプリケーションでスクロールすると、サーバーはさらに多くのエントリーを返します。これにより、サーバーの負荷が軽減され、クライアントはすべてのデータを一度に保存して処理する必要がなくなります。

すべての検索パラメーターが固定されている場合、VLV はサーバーでソートされた検索のパフォーマンスも向上させます。Directory Server は、VLV インデックス内の検索結果を事前に計算します。したがって、VLV インデックスは、結果を取得してから後で並べ替えるよりもはるかに効率的です。

Directory Server では、VLV コントロールは常に利用可能です。ただし、大きなディレクトリーで使用する場合は、ブラウジングインデックスとも呼ばれる VLV インデックスを使用すると、速度が大幅に向上します。

Directory Server は、標準インデックスなどの属性の VLV インデックスを維持しません。サーバーは、エントリーに設定された属性とディレクトリーツリー内のそれらのエントリーの場所に基づいて、VLV インデックスを動的に生成します。標準エントリーとは異なり、VLV エントリーはデータベース内の特別なエントリーです。

13.4.1. ldapsearch コマンドでの VLV コントロールの動作

通常、LDAP クライアントアプリケーションでは仮想リストビュー (VLV) 機能を使用します。ただし、たとえばテスト目的で、**ldapsearch** ユーティリティーを使用して部分的な結果のみを要求できます。

ldapsearch コマンドで VLV 機能を使用するには、**sss** (サーバー側の並べ替え) と **vlv** 検索拡張機能の両方に **-E** オプションを指定します。

```
# ldapsearch ... -E 'sss=attribute_list' -E 'vlv=query_options'
```

sss 検索拡張機能の構文は次のとおりです。

```
[!]sss=[-]<attr[:OID]>[/[-]<attr[:OID]>...]
```

vlv 検索拡張機能の構文は次のとおりです。

```
[!]vlv=<before>/<after>(/<offset>/<count>|:<value>)
```

- **before** は、対象のエントリーの前に返されるエントリーの数を設定します。
- **after** は、対象のエントリーの後に返されるエントリーの数を設定します。
- **index**、**count**、および **value** は、ターゲットエントリーの決定に役立ちます。**value** を設定すると、ターゲットエントリーは、その値で始まる最初の並べ替え属性を持つ最初のエントリーになります。それ以外の場合は、**count** を **0** に設定し、ターゲットエントリーは **index** 値 (1 から開始) によって決定されます。**count** 値が **0** より大きい場合、ターゲットエントリーは **ratio index * number of entries / count** によって決定されます。

例13.1 VLV 検索拡張機能を使用した ldapsearch コマンドの出力

次のコマンドは、**ou=People,dc=example,dc=com** を検索します。次に、サーバーは結果を ***cn*** 属性でソートし、70 番目のエントリーの ***uid*** 属性をオフセットの前の 1 つのエントリーと後の 2 つのエントリーとともに返します。

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -b
"ou=People,dc=example,dc=com" -s one -x -E 'sss=cn' -E 'vlv=1/2/70/0' uid
# user069, People, example.com
dn: uid=user069,ou=People,dc=example,dc=com
uid: user069

# user070, People, example.com
dn: uid=user070,ou=People,dc=example,dc=com
uid: user070

# user071, People, example.com
dn: uid=user071,ou=People,dc=example,dc=com
uid: user071

# user072, People, example.com
dn: uid=user072,ou=People,dc=example,dc=com
uid: user072

# search result
search: 2
result: 0 Success
control: 1.2.840.113556.1.4.474 false MIQAAAADCgEA
sortResult: (0) Success
control: 2.16.840.1.113730.3.4.10 false MIQAAAALAgFGAgMAnaQKAQA=
vlvResult: pos=70 count=40356 context= (0) Success

# numResponses: 5
# numEntries: 4
Press [before/after(/offset/count|:value)] Enter for the next window.
```

詳細については、ldapsearch(1) man ページの ***-E***パラメーターの説明を参照してください。

13.4.2. 認証されていないユーザーが VLV コントロールを使用できるようにする

デフォルトでは、**oid=2.16.840.1.113730.3.4.9,cn=features,cn=config** エントリーのアクセス制御命令 (ACI) により、認証されたユーザーのみが VLV コントロールを使用できるようになります。認証されていないユーザーも VLV コントロールを使用できるようにするには、**userdn = "ldap:///all"** を **userdn = "ldap:///anyone"** に変更して ACI を更新します。

手順

- **oid=2.16.840.1.113730.3.4.9,cn=features,cn=config** の ACI を更新します。

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

dn: oid=2.16.840.1.113730.3.4.9,cn=features,cn=config
changetype: modify
replace: aci
aci: (targetattr != "aci")(version 3.0; acl "VLV Request Control"; allow( read, search, compare,
proxy ) userdn = "ldap:///anyone";)
```

検証

- バインドユーザーを指定せずに、VLV コントロールでクエリーを実行します。

```
# ldapsearch -H ldap://server.example.com -b "ou=People,dc=example,dc=com" -s one -x -E 'sss=cn' -E 'vlv=1/2/70/0' uid
```

このコマンドでは、サーバーが匿名バインドを許可する必要があります。

コマンドが成功してもエントリーが返されない場合は、バインドユーザーを使用してクエリーを再度実行し、認証の使用時にクエリーが機能することを確認します。

13.4.3. コマンドラインを使用して VLV インデックスを作成し、VLV クエリーの速度を向上させる

次の手順に従って、**ou=People,dc=example,dc=com** 内のエントリーに対して、ブラウジングインデックスとも呼ばれる仮想リストビュー (VLV) インデックスを作成します。***mail***属性を持ち、**person** に設定された ***objectClass*** 属性があります。

前提条件

- クライアントアプリケーションは VLV コントロールを使用している。
- クライアントアプリケーションでは、大規模な検索結果の連続したサブセットに対してクエリーを実行する必要がある。
- ディレクトリーには多数のエントリーが含まれている。

手順

1. VLV 検索エントリーを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend vlv-index add-search --name "VLV People" --search-base "ou=People,dc=example,dc=com" --search-filter "(&(objectClass=person)(mail=*))" --search-scope 2 userRoot
```

コマンドは、以下のオプションを使用します。

- ***--name*** は検索エントリーの名前を設定します。これは任意の名前にすることができます。
- ***--search-base*** VLV インデックスのベース DN を設定します。Directory Server は、このエントリーに VLV インデックスを作成します。
- ***--search-scope*** は VLV インデックス内のエントリーに対して実行する検索の範囲を設定します。このオプションは、**0** (ベース検索)、**1** (1 レベル検索)、または **2** (サブツリー検索) に設定できます。
- ***--search-filter*** は VLV インデックスの作成時に Directory Server が適用するフィルターを設定します。このフィルターに一致するエントリーのみがインデックスの一部になります。
- ***userRoot*** は、エントリーを作成するデータベースの名前です。

2. インデックスエントリーを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend vlv-index add-index
--index-name "VLV People - cn sn" --parent-name "VLV People" --sort "cn sn" --index-it
dc=example,dc=com
```

コマンドは、以下のオプションを使用します。

- ***--index-name*** は、インデックスエントリーの名前を設定します。これは任意の名前にすることができます。
- ***--parent-name*** は、VLV 検索エントリーの名前を設定し、前の手順で設定した名前と一致する必要があります。
- ***--sort*** は属性名とソート順序を設定します。属性をスペースで区切ります。
- ***--index-it*** は、エントリーの作成後に、Directory Server がインデックスタスクを自動的に開始します。
- ***dc=example,dc=com*** は、エントリーを作成するデータベースの接尾辞です。

検証

1. **/var/log/dirsrv/slapd-instance_name/errors** ファイルで VLV インデックスが正常に作成されたことを確認します。

```
[26/Nov/2021:11:32:59.001988040 +0100] - INFO - bdb_db2index - userroot: Indexing VLV:
VLV People - cn sn
[26/Nov/2021:11:32:59.507092414 +0100] - INFO - bdb_db2index - userroot: Indexed 1000
entries (2%).
...
[26/Nov/2021:11:33:21.450916820 +0100] - INFO - bdb_db2index - userroot: Indexed 40000
entries (98%).
[26/Nov/2021:11:33:21.671564324 +0100] - INFO - bdb_db2index - userroot: Finished
indexing.
```

2. **ldapsearch** コマンドで VLV コントロールを使用して、ディレクトリーから特定のレコードのみをクエリーします。

```
# ldapsearch -D "cn=Directory Manager" -W -H ldap://server.example.com -b
"ou=People,dc=example,dc=com" -s one -x -E 'sss=cn' -E 'vlv=1/2/70/0' uid
# user069, People, example.com
dn: uid=user069,ou=People,dc=example,dc=com
cn: user069

# user070, People, example.com
dn: uid=user070,ou=People,dc=example,dc=com
cn: user070

# user071, People, example.com
dn: uid=user071,ou=People,dc=example,dc=com
cn: user071

# user072, People, example.com
dn: uid=user072,ou=People,dc=example,dc=com
cn: user072
```

この例では、**ou=People,dc=example,dc=com** に**uid=user001** から少なくとも **uid=user072** まで連続して名前が付けられたエントリーがあることを前提としています。

詳細については、ldapsearch(1) man ページの ***-E***パラメーターの説明を参照してください。

13.4.4. Web コンソールを使用して VLV インデックスを作成し、VLV クエリーの速度を向上させる

次の手順に従って、**ou=People,dc=example,dc=com** 内のエントリーに対して、ブラウジングインデックスとも呼ばれる仮想リストビュー (VLV) インデックスを作成します。***mail***属性を持ち、**person** に設定された ***objectClass*** 属性があります。

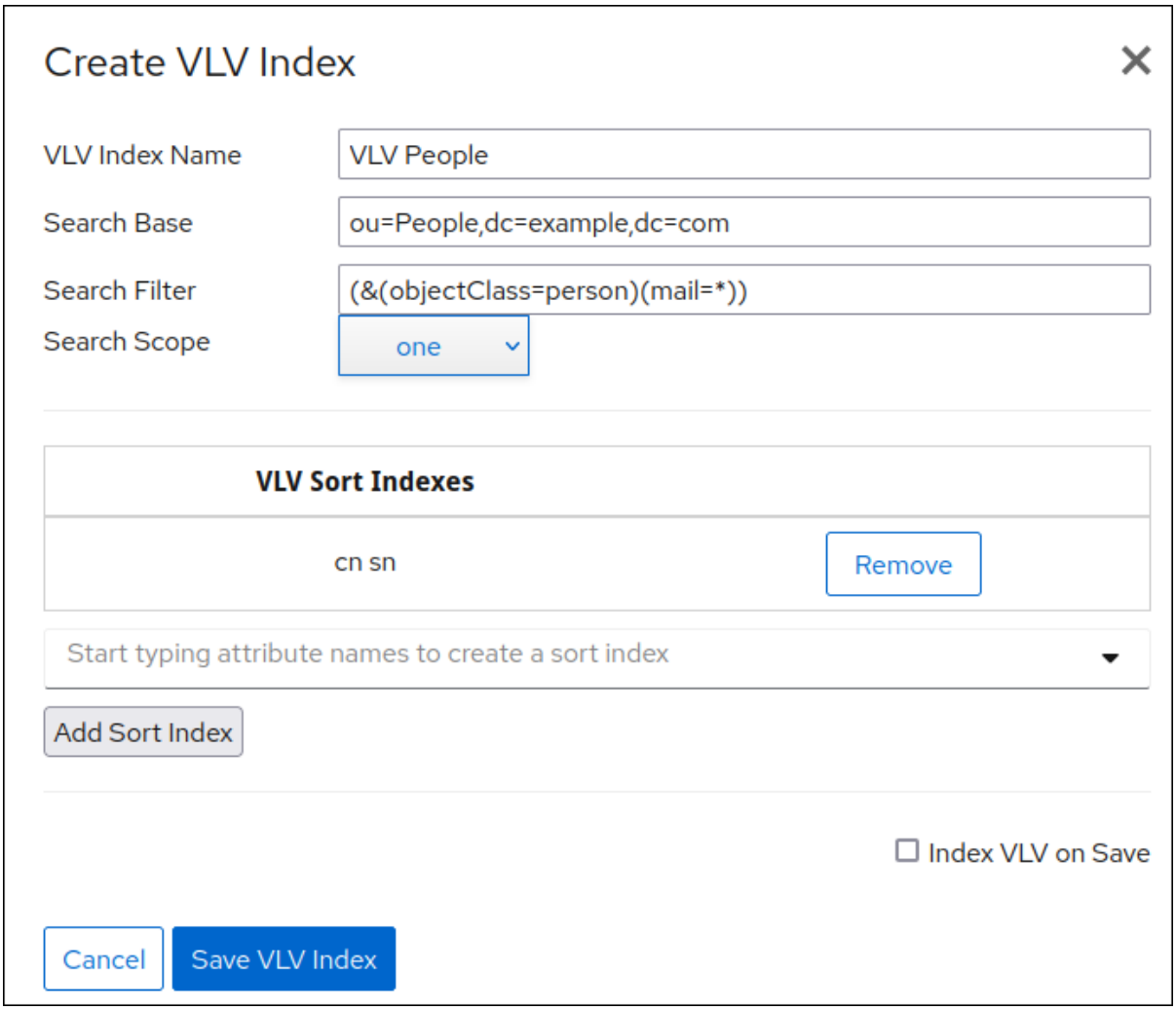
前提条件

- クライアントアプリケーションは VLV コントロールを使用している。
- クライアントアプリケーションでは、大規模な検索結果の連続したサブセットに対してクエリーを実行する必要がある。
- ディレクトリーには多数のエントリーが含まれている。

手順

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. **Database** → **Suffixes** → *dc=example,dc=com* → **VLV Indexes** に移動します。
3. **Create VLV Index** をクリックして、フィールドに入力します。

図13.1 Web コンソールを使用した VLV インデックスの作成



4. 属性名を入力し、**Add Sort Index** をクリックします。
5. **Index VLV on Save** を選択します。
6. **Save VLV Index** をクリックします。

検証

1. **Monitoring → Logging →** ***Errors Log*** に移動します。

```
[26/Nov/2021:11:32:59.001988040 +0100] - INFO - bdb_db2index - userroot: Indexing VLV: VLV People - cn sn
[26/Nov/2021:11:32:59.507092414 +0100] - INFO - bdb_db2index - userroot: Indexed 1000 entries (2%).
...
[26/Nov/2021:11:33:21.450916820 +0100] - INFO - bdb_db2index - userroot: Indexed 40000 entries (98%).
[26/Nov/2021:11:33:21.671564324 +0100] - INFO - bdb_db2index - userroot: Finished indexing.
```

2. **ldapsearch** コマンドで VLV コントロールを使用して、ディレクトリーから特定のレコードのみをクエリーします。

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

```
"ou=People,dc=example,dc=com" -s one -x -E 'sss=cn' -E 'vlv=1/2/70/0' uid
# user069, People, example.com
dn: uid=user069,ou=People,dc=example,dc=com
cn: user069

# user070, People, example.com
dn: uid=user070,ou=People,dc=example,dc=com
cn: user070

# user071, People, example.com
dn: uid=user071,ou=People,dc=example,dc=com
cn: user071

# user072, People, example.com
dn: uid=user072,ou=People,dc=example,dc=com
cn: user072
```

この例では、**ou=People,dc=example,dc=com** に**uid=user001** から少なくとも **uid=user072** まで連続して名前が付けられたエントリーがあることを前提としています。

詳細については、ldapsearch(1) man ページの ***-E***パラメーターの説明を参照してください。

13.5. インデックスのソート順序の変更

デフォルトでは、インデックスは ASCII 降順でアルファベット順にソートられます。これは、整数または電話番号などの数値属性値がある属性であっても、すべての属性に対して当てはまります。属性のマッチングルールセットを変更することで、sort メソッドを変更できます。

13.5.1. コマンドラインを使用したソート順序の変更

コマンドラインを使用してソート順序を変更するには、属性インデックスの ***nsMatchingRule*** を変更します。以下に例を示します。

```
# ldapmodify -D "cn=Directory Manager" -W -x

dn: cn=sn,cn=index,cn=Example1,cn=ldbm database,cn=plugins,cn=config
changetype:modify
replace:nsMatchingRule
nsMatchingRule: integerOrderingMatch
```

13.6. INDEXED SUBSTRING SEARCH の WIDTH の変更

デフォルトでは、検索がインデックス化されるようにするには、検索文字列はワイルドカード文字をカウントせずに 3 文字以上である必要があります。たとえば、**abc** という文字列はインデックス検索になりますが、**ab*** はインデックス検索になりません。インデックス化された検索は、インデックスなし検索よりもはるかに高速であるため、検索キーの最小長を変更すると、インデックス化された検索の数を増やすと便利です。

検索パフォーマンスを改善するために、特に多くのワイルドカード検索を持つサイトの場合は、インデックス化された検索の検索文字列の長さを変更できます。Directory Server には、インデックス化された検索に必要な最小文字数を変更できる属性が 3 つあります。

- ***nsSubStrBegin*** 属性は、ワイルドカードの前に検索文字列の最初にインデックス化された検索に必要な文字数を設定します。

```
abc*
```

- ***nsSubStrMiddle*** 属性は、検索文字列の途中でワイルドカードが使用される、インデックス化された検索に必要な文字数を設定します。以下に例を示します。

```
*abc*
```

- ***nsSubStrEnd*** 属性は、ワイルドカードの後に検索文字列の最後にインデックス化された検索に必要な文字数を設定します。以下に例を示します。

```
*xyz
```

文字列トリプレット (before、middle、end) のデフォルトの部分文字列検索の長さは 3、3、および 3 であり、すべての検索でワイルドカードの位置に最低 3 文字を必要とします。

別の文字列の長さを持つ属性インデックスは、**extensibleObject** オブジェクトクラスをエントリーに追加してから、部分文字列の検索の長さを設定します。

- 特定の属性インデックスの新しいキーの長さを設定します。これには、**extensibleObject** オブジェクトクラスを追加してから、***nsSubStrBegin*** 属性、***nsSubStrEnd*** 属性、または ***nsSubStrMiddle*** 属性を適宜追加する必要があります。以下に例を示します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: attribute_name,cn=index,cn=database_name,cn=ldbm database,cn=plugins,cn=config
changetype: modify
add: objectclass
objectclass: extensibleObject
-
add: nsSubStrBegin
nsSubStrBegin: 2
-
add: nsSubStrMiddle
nsSubStrMiddle: 2
-
add: nsSubStrEnd
nsSubStrEnd: 2
```

13.7. インデックスの削除

本セクションでは、インデックスから属性とインデックスタイプを削除する方法を説明します。

13.7.1. デフォルトインデックスエントリーからの属性の削除

Directory Server のデフォルト設定を使用する場合は、***sn*** などのデフォルトのインデックスエントリーに一覧表示される複数の属性がインデックス化されます。以下の属性はデフォルトインデックスの一部です。

表13.1 デフォルトのインデックス属性

aci	cn	entryusn

givenName	mail	mailAlternateAddress
mailHost	member	memberOf
nsUniqueId	ntUniqueId	ntUserDomainId
numsubordinates	objectclass	owner
parentid	seeAlso	sn
telephoneNumber	uid	uniquemember

警告

システムインデックスを削除すると、Directory Server のパフォーマンスが大幅に影響を受ける可能性があります。

たとえば、デフォルトのインデックスから ***sn*** 属性を削除するには、次のコマンドを実行します。

1. ***cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config*** エントリーから属性を削除します。

```
# ldapdelete -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
cn=sn,cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config
```

   このエントリーから属性を削除しない場合は、サーバーの再起動後に ***sn*** 属性のインデックスが自動的に再作成され、破損します。

2. ***cn=attribute_name,cn=index,cn=userRoot,cn=ldbm database,cn=plugins,cn=config*** エントリーを削除します。詳細については、「インデックスからの属性の削除」 を参照してください。

13.7.2. インデックスからの属性の削除

特定の状況では、インデックスから属性を削除します。本セクションでは、コマンドラインを使用する手順と、Web コンソールの使用方法を説明します。

13.7.2.1. コマンドラインを使用したインデックスから属性の削除

インデックスから属性を削除するには、以下を実行します。

1. 削除する属性が ***cn=default indexes,cn=config,cn=ldbm database,cn=plugins,cn=config*** デフォルトインデックスエントリーに一覧表示されている場合は、最初にこのエントリーから削除します。詳細については、「デフォルトインデックスエントリーからの属性の削除」 を参照してください。

2. インデックスから属性を削除します。以下に例を示します。

```
# ldapdelete -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
cn=sn,cn=index,cn=database_name,cn=ldbm database,cn=plugins,cn=config
```

エントリーを削除した後に、Directory Server は属性のインデックスを維持しません。

3. 属性インデックスを再作成します。「既存のデータベースへの新規インデックスの作成」を参照してください。

13.7.2.2. Web コンソールを使用したインデックスからの属性の削除

インデックスから属性を削除するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. 接尾辞エントリーを選択します。
5. **Indexes** タブを開きます。
6. インデックスを削除する属性の横にある **Actions** ボタンをクリックして、**Delete Index** を選択します。
7. **Yes** をクリックして確定します。

13.7.3. コマンドラインを使用したインデックスタイプの削除

たとえば、インデックスから ***sn*** 属性の **sub** インデックスタイプを削除するには、以下を実行します。

1. インデックスタイプを削除します。

```
# ldapmodify -D "cn=Directory Manager" -W -x

dn: cn=sn,cn=index,cn=database_name,cn=ldbm database,cn=plugins,cn=config
changetype: modify
delete: nsIndexType
nsIndexType: sub
```

インデックスエントリーを削除した後に、Directory Server は属性の部分文字列インデックスを維持しなくなります。

2. 属性インデックスを再作成します。「既存のデータベースへの新規インデックスの作成」を参照してください。

13.7.4. 参照インデックスの削除

このセクションでは、データベースからエントリーの閲覧項目を削除する方法を説明します。

13.7.4.1. コマンドラインを使用したインデックスの削除

アルファベット順の参照インデックスと仮想リストビュー (VLV) のエントリーは同じです。ここでは、参照インデックスを削除する手順を説明します。

コマンドラインで閲覧用インデックスや仮想リストビューインデックスを削除するには、以下を行います。

1. **cn=index,cn=*database_name*,cn=ldbm database,cn=plugins,cn=config** エントリーから、参照しているインデックスエントリーを削除します。以下に例を示します。

```
# ldapdelete -D "cn=Directory Manager" -W -p 389 -h server.example.com -x "cn=MCC
ou=People dc=example dc=com,cn=userRoot,cn=ldbm database,cn=plugins,cn=config"
"cn=by MCC ou=People dc=example dc=com,cn=MCC ou=People dc=example
dc=com,cn=userRoot,cn=ldbm database,cn=plugins,cn=config"
```

   2 つの参照インデックスエントリーを削除すると、Directory Server ではこれらのインデックス化は維持されなくなります。

2. 属性インデックスを再作成します。「既存のデータベースへの新規インデックスの作成」を参照してください。

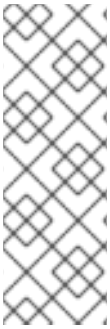
第14章 ディレクトリーエントリーの検索

ディレクトリー内のエントリーは、コマンドラインまたは Web コンソールを使用して検索および検出できます。

14.1. コマンドラインを使用したディレクトリーエントリーの検索

ldapsearch コマンドラインユーティリティーを使用して、ディレクトリーエントリーを検索できます。このユーティリティーは、指定した ID および認証情報を使用して指定のサーバーへの接続を開き、指定の検索フィルターに基づいてエントリーを見つけます。検索範囲には以下を含めることができます。

- 1 つのエントリー (**-s base**)
- エントリーの直接のサブエントリー (**-s one**)
- ツリー全体またはサブツリー全体 (**-s sub**)



注記

一般的な間違いは、識別名で使用される属性に基づいてディレクトリーを検索することを仮定することです。識別名はディレクトリーエントリーの一意の識別子であり、検索キーとして使用できません。代わりに、エントリー自体に保存されている属性とデータのペアに基づいてエントリーを検索します。したがって、エントリーの識別名が **uid=bjensen,ou=People,dc=example,dc=com** の場合、**dc=example** の検索は、そのエントリーの属性として **dc:example** が明示的に追加されない限り、そのエントリーと一致しません。

検索結果は LDIF 形式で返されます。LDIF は RFC 2849 に定義されており、付録B *LDAP データ交換形式*で詳細に説明されています。

このセクションには、以下のトピックに関する情報が含まれます。

- 「ldapsearch コマンドライン形式」
- 「一般的に使用される ldapsearch オプション」
- 「特殊文字の使用」

14.1.1. ldapsearch コマンドライン形式

ldapsearch コマンドは、次の形式を使用する必要があります。

```
# ldapsearch [-x | -Y mechanism] [options] [search_filter] [list_of_attributes]
```

- **-x** (簡単なバインドを使用するため) または **-Y** (SASL メカニズムを設定するため) は、接続の種類を設定するために使用されます。
- **オプション** は、一連のコマンドラインオプションです。これが存在する場合は、検索フィルターの前に指定する必要があります。
- **search_filter** は、「LDAP 検索フィルター」で説明されているように、LDAP 検索フィルターです。**-f** オプションを使用して、検索フィルターがファイルで指定されている場合には、別の検索フィルターを指定しないでください。

- **list_of_attributes** は、スペースで区切られた属性のリストです。属性のリストを指定すると、検索結果で返される属性の数を減らすことができます。この属性のリストは、検索フィルターの後に表示されなければなりません。例は、「属性のサブセットの表示」を参照してください。属性の一覧が指定されていない場合、検索はディレクトリーで設定されているアクセスコントロールで許可されているすべての属性の値を返しますが、操作属性は例外となります。

  操作属性を検索結果として返すためには、検索コマンドの中で明示的に指定する必要があります。オブジェクトのすべての操作属性を返すには、**+** を指定します。明示的に指定した操作属性に加えて通常の属性を取得するには、**ldapsearch** コマンドの属性一覧でアスタリスク (*) を使用します。

  一致する DN の一覧のみを取得するには、特別な属性 **1.1** を使用します。以下に例を示します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com \
    -b "dc=example,dc=com" -x "(objectclass=inetorgperson)" 1.1
```

14.1.2. 一般的に使用される ldapsearch オプション

以下の表は、最も一般的に使用される **ldapsearch** コマンドラインオプションを示しています。指定した値に空白 ( ) が含まれる場合、値は **-b "cn=My Special Group,ou=groups,dc=example,dc=com"** などの単一引用符または二重引用符で囲む必要があります。

重要

OpenLDAP の **ldapsearch** ユーティリティーは、デフォルトで SASL 接続を使用します。簡単なバインドを実行するか、TLS を使用するには、**-x** 引数を使用して SASL を無効にし、他の接続方法を許可します。

オプション	説明
-b	検索の開始点を指定します。ここで指定する値は、現在データベースに存在する識別名である必要があります。これは、**LDAP_BASEDN** 環境変数がベース DN に設定されている場合は任意です。このオプションで指定する値は、単一引用符または二重引用符で指定する必要があります。以下に例を示します。 -b "cn=*user*,ou=Product Development,dc=example,dc=com" ルート DSE エントリーを検索するには、ここで **-b ""** などの空の文字列を指定します。
-D	サーバーへの認証に使用する識別名を指定します。これは、サーバーによって匿名アクセスに対応している場合はオプションです。指定している場合、この値は Directory Server が認識する DN である必要があります。また、エントリーを検索する権限も必要です、例: **-D "uid=*user_name*,dc=example,dc=com"**

オプション	説明
-H	サーバーへの接続に使用する LDAP URL を指定します。従来の LDAP URL の場合は、以下の形式になります。 ldap[s]://*hostname*[:*port*] **ポート** は任意です。ポートを使用しないと、LDAP ポートの場合は 389、LDAPS ポートの場合は 636 がデフォルトで使用されます。 これは LDAPI URL を使用することもできます。各要素は、スラッシュ (/) ではなく、HTML の 16 進コード **%2F** で区切られています。 ldapi://%2F*full*%2F*path*%2F*to*%2Fslapd-example.socket LDAPI の場合は、サーバーがリッスンする LDAPI ソケットの完全パスおよびファイル名を指定します。この値は LDAP URL として解釈されるため、パスおよびファイル名のスラッシュ (/) をエスケープ処理して、URL エスケープ値 **%2F** としてエスケープする必要があります。 **-h** および **-p** の代わりに **-H** オプションが使用されます。
-h	Directory Server がインストールされているマシンのホスト名または IP アドレスを指定します。たとえば、**-h server.example.com** です。ホストが指定されない場合、**ldapsearch** は localhost を使用します。  **注記** Directory Server は、IPv4 アドレスと IPv6 アドレスの両方に対応します。
-l	検索要求が完了するまで待つ最大秒数を指定します。たとえば、**-l 300** です。***nsslapd-timelimit*** 属性のデフォルト値は **3600** 秒です。指定された値に関係なく、**ldapsearch** はサーバーの ***nsslapd-timelimit*** 属性によって許可されるよりも長く待機します。
-p	Directory Server が使用する TCP ポート番号を指定します。たとえば、**-p 1049** です。デフォルトは **389** です。 **-h** が指定されている場合は、デフォルト値が指定されていても **-p** も指定する必要があります。
-s *scope*	検索の範囲を指定します。範囲は以下のいずれかになります。 **base**: **-b** オプションで指定されたエントリー、または **LDAP_BASEDN** 環境変数により定義されたエントリーだけを検索します。 **one**: **-b** オプションで指定したエントリーの即時の子のみを検索します。子のみが検索されます。**-b** オプションに指定された実際のエントリーは検索されません。 **sub**: **-b** オプションで指定されたエントリーおよびその子すべてを検索します。つまり、**-b** オプションで識別された時点から、サブツリー検索を実行します。これはデフォルトになります。

オプション	説明
-W	パスワードの入力を要求します。このオプションが設定されていない場合は、匿名アクセスが使用されます。 あるいは、**-w** オプションを使用して、パスワードをそのユーティリティーに渡します。他のユーザーのプロセスリストにパスワードが表示され、シェルの履歴に保存されていることに注意してください。
-x	単純なバインドを許可するためにデフォルトの SASL 接続を無効にします。
-Y *SASL_mechanism*	認証に使用する SASL メカニズムを設定します。メカニズムが設定されていない場合、**ldapsearch** はサーバーがサポートする最適なメカニズムを選択します。 **-x** を使用しない場合は、**-Y** オプションを使用する必要があります。
-z *number*	検索要求への応答で返すエントリーの最大数を設定します。この値は、ルート DN を使用してバインディングする際にサーバー側の ***nsslapd-sizelimit*** パラメーターを上書きします。

14.1.3. 特殊文字の使用

ldapsearch コマンドラインユーティリティーを使用する場合は、スペース ( )、アスタリスク (*)、バックスラッシュ (\) などのコマンドラインインタープリターに対して特別な意味を持つ文字を含む値を指定する必要がある場合があります。引用符 ("") で特殊文字が含まれる値を囲みます。以下に例を示します。

```
-D "cn=user_name,ou=Product Development,dc=example,dc=com"
```

コマンドラインインタープリターに応じて、一重引用符または二重引用符を使用します。通常、単一引用符 (') を使用して値を囲みます。シェル変数がある場合は、二重引用符 (") を使用して変数挿入を許可します。詳細は、オペレーティングシステムのドキュメントを参照してください。

14.2. WEB コンソールを使用したエントリーの検索

Web コンソールで LDAP ブラウザーを使用して、ディレクトリーサーバーデータベースのエントリーを検索できます。

ディレクトリーサーバーは、エントリーの識別名 (DN) で使用される属性ではなく、エントリーに格納されている属性と値のペアに基づいてエントリーを検索します。たとえば、エントリーの DN が***uid=user_name,ou=People,dc=example,dc=com*** の場合、このエントリーに ***dc:example*** 属性が存在する場合にのみエントリーに一致する ***dc=example*** を検索します。

前提条件

- ディレクトリーサーバー Web コンソールにログインしている。
- root 権限がある。

手順

1. Web コンソールで、**LDAP Browser → Search** に移動します。
2. 検索基準を展開して選択し、エントリーをフィルタリングします。

表14.1 デフォルトのインデックス属性

検索パラメーター	説明
検索ベース	検索の開始点を指定します。現在データベースに存在する識別名 (DN) です。 **注記** Search タブが開き、定義済みの検索ベースが表示されます。**Tree View** または **Table View** でエントリーの詳細を開いたら、**Options メニュー (⋮)** をクリックし、**Search** を選択します。
検索範囲	***Subtree*** を選択すると、検索ベースから開始して子エントリーすべてを含むサブツリー全体のエントリーを検索できます。 ***One Level*** を選択すると、検索ベースから開始して第 1 レベルの子エントリーのみを含むエントリーを検索できます。 ***Base*** を選択すると、検索ベースとして指定されたエントリーでのみ属性値を検索します。
サイズ制限	検索操作で返されるエントリーの最大数を設定します。
時間制限	検索エンジンがエントリーを検索する時間を秒単位で設定します。
ロックを表示	スイッチをオンに切り替えて、見つかったエントリーのロックステータスを確認します。
検索属性	検索にかかわる属性を選択します。事前定義された属性から選択するか、カスタム属性を追加できます。

3. 検索テキストフィールドに属性値を入力し、**Enter** キーを押します。

注記

ディレクトリーサーバーは、すべての検索要求をアクセスログファイルに記録します。このファイルは、**Monitoring → Logging → Access Log** で表示できます。

4. オプション: 検索をさらに絞り込むには、**Filter** タブの検索フィルターを使用して、エントリーを検索します。

14.3. LDAP 検索フィルター

検索フィルターでは、返されるエントリーを選択します。これらは **ldapsearch** コマンドラインユーティリティーで最もよく使用されます。**ldapsearch** を使用する場合は、ファイルに複数の検索フィルターがあり、各フィルターがファイルの別々の行にあるか、検索フィルターをコマンドラインに直接指定することができます。

検索フィルターの基本的な構文は、以下のとおりです。

```
attribute operator value
```

以下に例を示します。

```
buildingname>=alpha
```

この例では、***buildingname*** が属性、**>=** が演算子、および **alpha** が値となります。フィルターは、異なる属性をブール値演算子とともに使用するように定義することもできます。

注記

一致するルールフィルターを使用して部分文字列検索を実行する場合は、アスタリスク (*) 文字をワイルドカードとして使用し、ゼロ以上の文字を表します。

たとえば、文字 **l** で始まり文字 **n** で終わる属性値を検索するには、検索フィルターの値の部分に **l*n** を入力します。同様に、文字 **u** で始まるすべての属性値を検索するには、検索フィルターの値の部分に **u*** の値を入力します。

アスタリスク (*) 文字を含む値を検索するには、アスタリスクを指定のエスケープシーケンス **\5c2a** でエスケープする必要があります。たとえば、***businessCategory*** 属性の値が **Example*Net product line** の社員をすべて検索するには、検索フィルターに以下の値を入力します。

```
Example\5c2a*Net product line
```

注記

一般的な間違いは、識別名で使用される属性に基づいてディレクトリーを検索することを仮定することです。識別名はディレクトリーエントリーの一意の識別子であり、検索キーとして使用できません。代わりに、エントリー自体に保存されている属性とデータのペアに基づいてエントリーを検索します。したがって、エントリーの識別名が **uid=*user_name*,ou=People,dc=example,dc=com** の場合、**dc=example** の検索は、このエントリー内で ***dc*** 属性が存在し **example** に設定していなければ、そのエントリーと一致しません。

14.3.1. 検索フィルターの属性の使用

検索の最も基本的なソートでは、エントリーに属性や特定の値があるかどうかを調べます。エントリーで属性を検索する **方法** は多数あります。属性が存在するかどうかを確認したり、完全値と一致するか、部分的な値に対して一致をリスト表示したりすることができます。

存在 の検索では、値に関係なく、ワイルドカード (アスタリスク) を使用して、属性が設定されているすべてのエントリーを返します。たとえば、以下は、***manager*** 属性を持つすべてのエントリーを返します。

```
"(manager=*)"
```

特定の値を持つ属性を検索することもできます。これは **等価** 検索と呼ばれます。以下に例を示します。

```
"(cn=example)"
```

この検索フィルターは、***example*** に設定された一般名が含まれるすべてのエントリーを返します。多くの場合、等価検索では大文字と小文字が区別されません。

属性が言語タグに関連付けられた値を持つ場合は、すべての値が返されます。そのため、以下の 2 つの属性値は両方とも **"(cn=*example*)"** フィルターと一致します。

```
cn: example
cn;lang-fr: example
```

属性値 (**部分文字列** インデックス) で部分的な一致を検索することもできます。以下に例を示します。

```
"(description=*X.500*)"
"(sn=*nderson)"
"(givenname=car*)"
```

部分文字列検索の長さは、「Indexed Substring Search の Width の変更」 で説明されているように、部分文字列のインデックス自体で設定されます。

14.3.2. 検索フィルターでの演算子の使用

検索フィルターの演算子は、属性と指定の検索値間の関係を設定します。人名検索では、演算子を使用して範囲を設定し、アルファベットのサブセット内の名字を返したり、ある数字以降の社員番号を返したりすることができます。

```
"(employeeNumber>=500)"
"(sn~=suret)"
"(salary<=150000)"
```

演算子は、音声検索や近似検索も可能で、不完全な情報でも効果的な検索ができ、特に国際化されたディレクトリーでは有効です。

検索フィルターで使用できる演算子が 表14.2「検索フィルター演算子」に一覧表示されています。これらの検索フィルターに加えて、特別なフィルターを指定して、望ましい言語の照合順序で動作させることができます。国際文字セットを持つディレクトリーを検索する方法は、「国際化されたディレクトリーの検索」を参照してください。

表14.2 検索フィルター演算子

検索タイプ	演算子	説明
等号	=	指定された値と完全に一致する属性値が含まれるエントリーを返します。たとえば、**cn=*example*** です。
部分文字列	=string* string	指定の部分文字列が含まれる属性が含まれるエントリーを返します。たとえば、**cn=*exa*l*** です。アスタリスク (*) はゼロ (0) 以上の文字を示します。

検索タイプ	演算子	説明
以上	>=	指定された値以上の属性を含むエントリーを返します。たとえば、**uidNumber >= 5000** です。
より小か等しい	<=	指定された値以下の属性が含まれるエントリーを返します。たとえば、**uidNumber <= 5000** です。
存在	=*	指定属性の 1 つ以上の値が含まれるエントリーを返します。たとえば、**cn=*** です。
概算値	~=	指定した属性を含むエントリーを、検索フィルターで指定された値とほぼ等しい値で返します。たとえば、**l~=san fransico** は **l=san francisco** を返すことができます。

14.3.3. 複合検索フィルターの使用

複数の検索フィルターコンポーネントは、以下のように接頭辞表記で表現されるブール値演算子を使用して組み合わせることができます。

```
(Boolean-operator(filter)(filter)(filter)...)
```

boolean-operator は、表14.3「検索フィルターのブール値演算子」に一覧表示されているブール値演算子の 1 つになります。

たとえば、このフィルターは、指定された値を含まないすべてのエントリーを返します。

```
(!(objectClass=person))
```

当然のことながら、複合検索フィルターは、入れ子にして完成された表現にしたときに最も有効です。

```
(Boolean-operator(filter)((Boolean-operator(filter)(filter)))
```

これらの複合フィルターは、他のタイプの検索 (近似、部分文字列、その他の演算子) と組み合わせることで、非常に詳細な結果を得ることができます。たとえば、このフィルターは組織単位が **Marketing** で、その ***description*** 属性に部分文字列 **X.500** が含まれていないすべてのエントリーを返します。

```
(&(ou=Marketing)(!(description=*X.500*)))
```

このフィルターは、組織単位が **Marketing** で、部分文字列 **X.500** が含まれず、**example** または **demo** が ***manager*** として設定されているエントリーを返すように拡張できます。

```
(&(ou=Marketing)(!(description=*X.500*))(|
(manager=cn=example,ou=Marketing,dc=example,dc=com)
(manager=cn=demo,ou=Marketing,dc=example,dc=com)))
```

このフィルターは、人を表すことなく、共通名が **printer3b** と似たすべてのエントリーを返します。

```
(&(!(objectClass=person))(cn~=printer3b))
```

表14.3 検索フィルターのブール値演算子

演算子	記号	説明
AND	&	文が true になるには、指定したフィルターはすべて true である必要があります。例: **(&(filter)(filter)(filter)...)**
OR	|	文が true になるには、少なくとも 1 つのフィルターを true にする必要があります。例: **(|(filter)(filter)(filter)...)**
NOT	!	文が true になるには、指定の文が true にならないようにする必要があります。**NOT** 演算子の影響を受けるフィルターは 1 つだけです。例:**(!(filter))**

ブール値は、以下の順番で評価されます。

- 一番内側から外側に向かって、親表現が優先されます。
- 左から右へのすべての式。

14.3.4. 一致するルールの使用

マッチングルール は、Directory Server に対して、2 つの値 (属性に保存されている値および検索フィルターの値) を比較する方法を説明します。マッチングルールは、インデックスキーの生成方法も定義します。マッチングルールは、属性構文に関連するものです。構文は属性値の **形式** を定義します。マッチングルールは、形式が比較およびインデックス化される方法を定義します。

マッチングルールは 3 種類あります。

- **EQUALITY** は、同じ一致の 2 つの値を比較する方法を指定します。たとえば、Fred および FRED などの文字列の処理方法です。等価をテストする検索フィルター (**attribute=value** など) は EQUALITY ルールを使用します。等価 (eq) インデックスは EQUALITY ルールを使用してインデックスキーを生成します。更新操作は EQUALITY ルールを使用して、値を比較して、エントリーにすでにある値と比較します。
- **ORDERING** は、2 つの値を比較して、ある値が別の値以上であるかを確認できます。範囲 (例: **attribute<=value** または **attribute>=value**) を設定する検索フィルターは、ORDERING ルールを使用します。ORDERING ルールを持つ属性のインデックスは等価値の順序です。
- **SUBSTR** は、部分文字列照合を行う方法を指定します。部分文字列検索フィルター (例: **attribute=*partial_string*** または **attribute=*end_string**) は SUBSTR ルールを使用します。部分文字列 (サブ) インデックスは SUBSTR ルールを使用してインデックスを生成します。

重要

対応する検索フィルターまたはインデックスタイプの検索またはインデックスをサポートするには、マッチングルールが必要です。たとえば、ある属性の等価検索フィルターや eq インデックスをサポートするには、その属性に EQUALITY マッチングルールが必要です。範囲検索フィルターとインデックス化の範囲検索に対応するために、属性に ORDERING マッチングルールと EQUALITY マッチングルールの両方が必要です。

マッチングルールのない属性の検索フィルターの使用を試みた場合は、PROTOCOL_ERROR または UNWILLING_TO_PERFORM で検索操作は拒否されます。

例14.1 マッチングルールおよびカスタム属性

Example Corp. 管理者は、IA5 文字列 (7 ビット ASCII) 構文で ***MyFirstName*** という名前のカスタム属性タイプと、caseExactIA5Match の EQUALITY マッチングルールを作成します。***MyFirstName*** の値が **Fred** のエントリーは、**(MyFirstName=Fred)** のフィルターを使用した検索で返されますが、**(MyFirstName=FRED)** および **(MyFirstName=fred)** のフィルターでは返されません。**Fred**、**FRED**、および **fred** はすべて有効な IA5 文字列値ですが、caseExactIA5Match ルールを使用した場合一致しません。

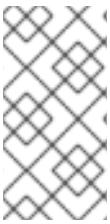
検索で返される Fred の 3 つのすべてのバリアントについては、caseIgnoreIA5Match マッチングルールを使用するように ***MyFirstName*** を定義する必要があります。

拡張されたマッチングルール検索フィルターを使用すると、属性に定義されたルールとは異なるマッチングルールを持つ属性値を検索できます。マッチングルールは、検索される属性の構文と互換性がある必要があります。たとえば、大文字と小文字を区別するマッチングルールが定義されている属性に対して大文字と小文字を区別しない検索を行うには、検索フィルターに大文字と小文字を区別しないマッチングルールを指定します。

```
(MyFirstName:caseIgnoreIA5Match:=fred)
```

注記

マッチングルールは、国際化されたディレクトリーの検索に使用され、結果に使用する言語タイプを指定します。詳細は、「国際化されたディレクトリーの検索」を参照してください。



注記

属性のインデックスは、その属性のスキーマ定義で定義されているマッチングルールを使用します。インデックスに使用する追加のマッチングルールは、「コマンドラインを使用したインデックスの作成」にあるように ***nsMatchingRule*** 属性を使用して設定できます。

マッチングルールフィルターの構文では、一致するルール名または OID が検索フィルターに挿入されます。

```
attr:matchingRule:=value
```

- **attr** は、***cn*** や ***mail*** など、検索されるエントリーに属する属性です。
- **matchingRule** は、必要な構文に従って属性値と一致するために使用するルールの名前または OID を含む文字列です。
- **value** は、検索する属性値か、比較演算子および検索する属性値のいずれかです。フィルターの値の構文は、使用されるマッチングルール形式によって異なります。

マッチングルールは実際にはスキーマ要素であり、他のスキーマ要素と同様に、オブジェクト識別子 (OID) によって一意に識別されます。

Red Hat Directory Server 向けに定義されたマッチングルールの多くは言語コードに関連し、Directory Server によってサポートされる国際化された照合順序が設定されます。たとえば、OID **2.16.840.1.113730.3.3.2.17.1** はフィンランドの照合順序を識別します。

注記

その他のスキーマ要素とは異なり、Directory Server の設定には、追加のマッチングルールを追加できません。

以下のリストにおけるマッチングルールリストのほとんどは、等価インデックスに使用されます。名前に **順序** を含むマッチングルールは順序インデックスに、名前に **部分文字列** を含むマッチングルールは部分文字列 (SUBSTR) インデックスに使用されます。(国際的な一致や照合順序に用いられるマッチングルールは、別の命名法を用いています。)

ビット単位の AND 一致

ビット単位の **AND** 一致を実行します。

OID: 1.2.840.113556.1.4.803

互換性のある構文: 通常、**Integer** および数値の文字列で使用されます。Directory Server は自動的に数値の文字列を整数に変換します。

ビット単位の OR 一致

ビット単位の **OR** 一致を実行します。

OID: 1.2.840.113556.1.4.804

互換性のある構文: 通常、**Integer** および数値の文字列で使用されます。Directory Server は自動的に数値の文字列を整数に変換します。

booleanMatch

照合する値が **TRUE** または **FALSE** かを評価します。

OID: 2.5.13.13

互換性のある構文: ブール値

caseExactIA5Match

値の大文字と小文字を区別する比較を行います。

OID: 1.3.6.1.4.1.1466.109.114.1

互換性のある構文: **IA5** 構文、URI

caseExactMatch

値の大文字と小文字を区別する比較を行います。

OID: 2.5.13.5

互換性のある構文: Directory String、Printable String、OID

caseExactOrderingMatch

大文字と小文字を区別する範囲検索が可能になります (より小さいおよびより大きい)。

OID: 2.5.13.6

互換性のある構文: Directory String、Printable String、OID

caseExactSubstringsMatch

大文字と小文字を区別した部分文字列とインデックスの検索を実行します。

OID: 2.5.13.7

互換性のある構文: Directory String、Printable String、OID

caseIgnoreIA5Match

値に対して大文字と小文字を区別しない比較を実行します。

OID: 1.3.6.1.4.1.1466.109.114.2

互換性のある構文: **IA5** 構文、URI

caseIgnoreIA5SubstringsMatch

部分文字列およびインデックスで大文字と小文字を区別しない検索を実行します。

OID: 1.3.6.1.4.1.1466.109.114.3

互換性のある構文: **IA5** 構文、URI

caseIgnoreListMatch

値に対して大文字と小文字を区別しない比較を実行します。

OID: 2.5.13.11

互換性のある構文: 住所

caseIgnoreListSubstringsMatch

部分文字列およびインデックスで大文字と小文字を区別しない検索を実行します。

OID: 2.5.13.12

互換性のある構文: 住所

caseIgnoreMatch

値に対して大文字と小文字を区別しない比較を実行します。

OID: 2.5.13.2

互換性のある構文: Directory String、Printable String、OID

caseIgnoreOrderingMatch

大文字と小文字を区別しない範囲検索が可能になります (より小さいおよびより大きい)。

OID: 2.5.13.3

互換性のある構文: Directory String、Printable String、OID

caseIgnoreSubstringsMatch

部分文字列およびインデックスで大文字と小文字を区別しない検索を実行します。

OID: 2.5.13.4

互換性のある構文: Directory String、Printable String、OID

distinguishedNameMatch
識別名の値を比較します。

OID: 2.5.13.1

互換性のある構文: 識別名 (DN)

generalizedTimeMatch
一般化された時間形式の値を比較します。

OID: 2.5.13.27

互換性のある構文: 一般化時刻

generalizedTimeOrderingMatch
一般化された時間形式の値の範囲検索 (より小さいおよびより大きい) が可能になります。

OID: 2.5.13.28

互換性のある構文: 一般化時刻

integerMatch
整数値を評価します。

OID: 2.5.13.14

互換性のある構文: 整数

integerOrderingMatch
整数値に範囲化された検索が可能になります (より小さいおよびより大きい)。

OID: 2.5.13.15

互換性のある構文: 整数

keywordMatch
指定した検索値を、属性値の文字列と比較します。

OID: 2.5.13.33

互換性のある構文: ディレクトリー文字列

numericStringMatch
より一般的な数値を比較します。

OID: 2.5.13.8

互換性のある構文: 数値文字列

numericStringOrderingMatch
複数の一般的な値に対する範囲検索 (より小さいおよびより大きい) が可能になります。

OID: 2.5.13.9

互換性のある構文: 数値文字列

numericStringSubstringMatch
より一般的な数値を比較します。

OID: 2.5.13.10

互換性のある構文: 数値文字列

objectIdentifierMatch
オブジェクト識別子 (OID) 値を比較します。

OID: 2.5.13.0

互換性のある構文: OID

octetStringMatch
octet 文字列の値を評価します。

OID: 2.5.13.17

互換性のある構文: オクテット文字列

octetStringOrderingMatch
一連のオクテット文字列値で範囲検索 (より小さいおよびより大きい) をサポートします。

OID: 2.5.13.18

互換性のある構文: オクテット文字列

telephoneNumberMatch
電話番号の値を評価します。

OID: 2.5.13.20

互換性のある構文: 電話番号

telephoneNumberSubstringsMatch
電話番号の値に対して部分文字列とインデックス検索を行います。

OID: 2.5.13.21

互換性のある構文: 電話番号

uniqueMemberMatch
名前と UID の値を比較します。

OID: 2.5.13.23

互換性のある構文: 名前および任意の UID

wordMatch

指定した検索値を、属性値の文字列と比較します。このマッチングルールは大文字と小文字を区別しません。

OID: 2.5.13.32

互換性のある構文: ディレクトリー文字列

表14.4 言語順序のマッチングルール

マッチングルール	オブジェクト識別子 (OID)
英語 (大文字と小文字を区別する順序の一致)	2.16.840.1.113730.3.3.2.11.3
アルバニア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.44.1
アラビア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.1.1
ベラルーシ語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.2.1
ブルガリア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.3.1
カタロニア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.4.1
中国語: 簡体字 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.49.1
中国語: 繁体字 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.50.1
クロアチア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.22.1
チェコ語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.5.1
デンマーク語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.6.1
オランダ語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.33.1

マッチングルール	オブジェクト識別子 (OID)
オランダ語: ベルギー (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.34.1
英語 - アメリカ (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.11.1
英語 - カナダ語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.12.1
英語: アイルランド (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.14.1
エストニア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.16.1
フィンランド語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.17.1
フランス語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.18.1
フランス語: ベルギー (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.19.1
フランス語 - カナダ語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.20.1
フランス語 - スイス (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.21.1
ドイツ語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.7.1
ドイツ語 - オーストリア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.8.1
ドイツ語: スイス (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.9.1
ギリシャ語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.10.1
ヘブライ語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.27.1
ハンガリー語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.23.1
アイスランド語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.24.1

マッチングルール	オブジェクト識別子 (OID)
イタリア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.25.1
イタリア語: スイス (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.26.1
日本語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.28.1
韓国語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.29.1
ラトビア語、レット語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.31.1
リトアニア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.30.1
マケドニア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.32.1
ノルウェー語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.35.1
ノルウェー語 - ブークモール (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.36.1
ノルウェー語 - ニーノルスク (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.37.1
ポーランド語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.38.1
ルーマニア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.39.1
ロシア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.40.1
セルビア語 - キリル文字 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.45.1
セルビア語 - ラテン語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.41.1
スロバキア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.42.1
スロベニア語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.43.1

マッチングルール	オブジェクト識別子 (OID)
スペイン語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.15.1
スウェーデン語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.46.1
トルコ語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.47.1
ウクライナ語 (大文字と小文字を区別しない順序一致)	2.16.840.1.113730.3.3.2.48.1

表14.5 言語部分文字列マッチングルール

マッチングルール	オブジェクト識別子 (OID)
英語 (大文字と小文字を区別する部分文字列の一致)	2.16.840.1.113730.3.3.2.11.3.6
アルバニア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.44.1.6
アラビア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.1.1.6
ベラルーシ語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.2.1.6
ブルガリア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.3.1.6
カタロニア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.4.1.6
中国語 - 簡体字 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.49.1.6
中国語 - 繁体字 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.50.1.6
クロアチア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.22.1.6
チェコ語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.5.1.6
デンマーク語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.6.1.6

マッチングルール	オブジェクト識別子 (OID)
オランダ語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.33.1.6
オランダ語: ベルギー (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.34.1.6
英語 - アメリカ (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.11.1.6
英語: カナダ (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.12.1.6
英語: アイルランド (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.14.1.6
エストニア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.16.1.6
フィンランド語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.17.1.6
フランス語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.18.1.6
フランス語: ベルギー (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.19.1.6
フランス語: カナダ (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.20.1.6
フランス語: スイス (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.21.1.6
ドイツ語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.7.1.6
ドイツ語 - オーストリア (大文字小文字を区別しない文字列一致)	2.16.840.1.113730.3.3.2.8.1.6
ドイツ語: スイス (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.9.1.6
ギリシャ語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.10.1.6
ヘブライ語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.27.1.6

マッチングルール	オブジェクト識別子 (OID)
ハンガリー語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.23.1.6
アイスランド語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.24.1.6
イタリア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.25.1.6
イタリア語: スイス (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.26.1.6
日本語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.28.1.6
韓国語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.29.1.6
ラトビア語、レット語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.31.1.6
リトアニア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.30.1.6
マケドニア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.32.1.6
ノルウェー語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.35.1.6
ノルウェー語 - ブークモール (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.36.1.6
ノルウェー語 - ニーノルスク (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.37.1.6
ポーランド語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.38.1.6
ルーマニア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.39.1.6
ロシア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.40.1.6
セルビア語 - キリル文字 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.45.1.6

マッチングルール	オブジェクト識別子 (OID)
セルビア語 - ラテン語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.41.1.6
スロバキア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.42.1.6
ストベニア語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.43.1.6
スペイン語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.15.1.6
スウェーデン語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.46.1.6
トルコ語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.47.1.6
ウクライナ語 (大文字と小文字を区別しない部分文字列一致)	2.16.840.1.113730.3.3.2.48.1.6

14.4. 一般的な LDAPSEARCH の例

次の例セットは、以下を想定しています。

- 検索は、ディレクトリー内のすべてのエントリーに対するものです。
- このディレクトリーは、検索および読み取りの匿名アクセスをサポートするように設定されます。つまり、検索を実行するためにバインド情報を提供する必要はありません。匿名アクセスの詳細は、「匿名アクセスの付与」を参照してください。
- サーバーは、**server.example.com** という名前のホストにあります。
- サーバーはポート番号 **389** を使用します。これはデフォルトのポートであるため、ポート番号は検索リクエストで送信する必要がありません。
- TLS は、ポート **636** のサーバー (デフォルトの LDAPS ポート番号) に対して有効になります。
- すべてのデータが格納される接尾辞は **dc=example,dc=com** です。

14.4.1. すべてのエントリーの返信

以前の情報を指定すると、以下の呼び出しはディレクトリー内のすべてのエントリーを返します (設定されるサイズおよび時間制限に従います)。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b "dc=example,dc=com"
-s sub -x "(objectclass=*)"
```

"objectclass=*" は、このディレクトリー内のエントリーに一致する検索フィルターです。すべてのエントリーにオブジェクトクラスが含まれる必要があり、***objectclass*** 属性は常にインデックス化されるため、すべてのエントリーを返すための便利な検索フィルターになります。

14.4.2. コマンドラインでの検索フィルターの指定

フィルターが引用符 ("filter") で囲まれている場合、検索フィルターはコマンドラインで直接指定することができます。フィルターがコマンドで提供されている場合は、**-f** オプションを指定しないでください。以下に例を示します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b "dc=example,dc=com"
-s sub -x "cn=babs jensen"
```

14.4.3. ルート DSE エントリーの検索

ルート DSE は、ローカルの Directory Server でサポートされるすべての接尾辞を含む、ディレクトリーサーバーのインスタンスに関する情報が含まれる特別なエントリーです。このエントリーは、"" の検索ベース、**base** の検索範囲、および **"objectclass=*"** のフィルターを指定して検索できます。以下に例を示します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -x -b "" -s base
"objectclass=*"
```

14.4.4. スキーマエントリーの検索

cn=schema エントリーは、オブジェクトクラスや属性タイプなどのディレクトリースキーマに関する情報が含まれる特別なエントリーです。

以下のコマンドは、**cn=schema** エントリーの内容を一覧表示します。

```
# ldapsearch -o ldif-wrap=no -D "cn=Directory Manager" -W -b "cn=schema" \
    '(objectClass=subSchema)' -s sub objectClasses attributeTypes matchingRules \
    matchingRuleUse  dITStructureRules nameForms ITContentRules ldapSyntaxes
```

14.4.5. LDAP_BASEDN の使用

検索を容易にするには、**LDAP_BASEDN** 環境変数を使用して検索ベースを設定できます。これを行うと、検索ベースは **-b** オプションで設定する必要があります。環境変数の設定方法については、オペレーティングシステムのドキュメントを参照してください。

通常、**LDAP_BASEDN** をディレクトリーの接尾辞の値に設定します。ディレクトリーの接尾辞は、ディレクトリーのルート (最上位の) エントリーと等しいため、これにより、すべての検索はディレクトリーのルートエントリーから始まることになります。

たとえば、**LDAP_BASEDN** を **dc=example,dc=com** に設定し、ディレクトリー内の **cn=babs jensen** を検索するには、以下のコマンドライン呼び出しを使用します。

```
# export LDAP_BASEDN="dc=example,dc=com"

# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -x "cn=babs jensen"
```

この例では、**-s** オプションがスコープの指定に使用されないため、**sub** のデフォルトスコープが使用されます。

14.4.6. 属性のサブセットの表示

ldapsearch コマンドは、すべての検索結果を LDIF 形式で返します。デフォルトでは、**ldapsearch** はエントリーの識別名と、ユーザーが読み取りできるすべての属性を返します。ディレクトリーアクセス制御は、指定されたディレクトリーエントリーの属性のサブセットのみをユーザーが読み取りできるように設定できます。操作属性は返されません。検索操作の結果として操作属性が返される場合は、search コマンドで明示的に指定するか、**+** を使用してすべての操作属性を返します。

検索結果で返されるエントリーのすべての属性を作成する必要はない場合があります。返された属性は、検索フィルターの直後にコマンドラインに必要な属性を指定することにより、いくつかの特定の属性のみに限定できます。たとえば、ディレクトリー内のすべてのエントリーの ***cn*** 属性および ***sn*** 属性を表示するには、コマンドライン呼び出しを使用します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b "dc=example,dc=com"
-s sub -x "(objectclass=*)" sn cn
```

14.4.7. 操作属性の検索

操作属性は、アクセス制御命令を処理するなどのメンテナンスタスクを実行するためにサーバーによって使用される Directory Server 自体によって設定される特別な属性です。また、最初に作成された時間や、作成したユーザーの名前など、エントリーに関する特定の情報も表示します。操作属性は、属性がエントリーのオブジェクトクラスに対して属性が特別に定義されているかどうかにかかわらず、ディレクトリー内のすべてのエントリーで使用することができます。

操作属性は通常の **ldapsearch** では返されません。RFC3673 に従って、**+** を使用して検索要求の操作属性をすべて返します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b "dc=example,dc=com"
-s sub -x "(objectclass=*)" '+'
```

定義された操作属性のみを返すには、**ldapsearch** リクエストに明示的に指定します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b "dc=example,dc=com"
-s sub -x "(objectclass=*)" creatorsName createTimestamp modifiersName modifyTimestamp
```

操作属性の完全なリストは、『Red Hat Directory Server 11 の設定、コマンド、およびファイルリファレンス』の操作属性およびオブジェクトクラスの章にあります。

注記

指定した操作属性とともにすべての通常のエントリー属性を返すには、記載されている操作属性に加えて、特別な search 属性 **"*"** を使用します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b
"dc=example,dc=com" -s sub -x "(objectclass=*)" "*" aci
```

シェルで解釈されないように、アスタリスクを引用符で囲む必要があります。

14.4.8. ファイルを使用した検索フィルターの指定

検索フィルターは、コマンドラインで入力するのではなく、ファイルに入力できます。この場合は、ファイル内の個別の行に各検索フィルターを指定します。**ldapsearch** コマンドは、ファイルに表示される順序で各検索を実行します。

以下に例を示します。

```
sn=example
givenname=user
```

ldapsearch は、最初に、姓が **example** に設定されたエントリーをすべて検索し、次に ***givenname*** が **user** に設定されたすべてのエントリーを検索します。両方の検索条件に一致するエントリーが見つかると、エントリーは 2 回返されます。

たとえば、この検索では、**searchdb** という名前のファイルにフィルターを指定します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -x -f searchdb
```

ここで返される属性のセットは、検索行の最後に属性名を指定すると制限されます。たとえば、以下の **ldapsearch** コマンドは両方の検索を実行しますが、各エントリーの DN、***givenname*** 属性、および ***sn*** 属性のみが返されます。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -x -f searchdb sn
givenname
```

14.4.9. 検索フィルターでコンマを含む DN の指定

検索フィルター内の DN に値の一部としてコンマが含まれている場合は、バックスラッシュ (\) でエスケープする必要があります。たとえば、**example.com Bolivia, S.A.** サブツリーですべてのユーザーが検索するには、以下のコマンドを使用します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -x -s base -b
"l=Bolivia\,S.A.,dc=example,dc=com" "objectclass=*"
```

14.4.10. クライアント証明書の Directory Server へのバインド

「証明書を使用した認証」を参照してください。

14.4.11. 言語マッチングルールでの検索

検索フィルターでマッチングルールを明示的に送信するには、属性の後にマッチングルールを挿入します。

```
attr:matchingRule:=value
```

マッチングルールは、国際化されたディレクトリーの検索に頻繁に使用されます。たとえば、これにより、スウェーデン語 (**2.16.840.1.113730.3.3.2.46.1**) のマッチングルールの N4709 以降の部署番号を検索します。

```
departmentNumber:2.16.840.1.113730.3.3.2.46.1:=>= N4709
```

国際化された検索を行うためのその他の例は、「国際化されたディレクトリーの検索」 に記載されています。

14.4.12. Bit Field の値での属性の検索

ビット単位は、ビットフィールドの値を持つ属性に対して、ビット単位の AND またはビット単位の OR マッチングルールを使用してビット単位の検索操作を行います。

注記

ビットフィールドの値が含まれる属性は LDAP で一般的ではありません。(デフォルトの Directory Server スキーマは、ビットフィールドを属性構文として使用しません。) ただし、複数の LDAP 構文は整数形式の値をサポートします。カスタム属性はビットフィールド値を使用して定義でき、アプリケーションはこれらのカスタム属性を使用してビットフィールドの値に対してビット単位の操作を実行できます。

ビット単位 AND マッチングルール (**1.2.840.113556.1.4.803**) は、アサーション値に指定されたビットがビットフィールド属性値に設定されていることを確認します。(これは等価検索に類似しています) この例では、userAccountControl の値は、2 を表すビットに設定する必要があります。

```
"(UserAccountControl:1.2.840.113556.1.4.803:=2)"
```

この例では、userAccountControl の値には、値 6 (ビット 2 および 4) に設定されるすべてのビットセットが必要になります。

```
"(UserAccountControl:1.2.840.113556.1.4.803:=6)”
```

ビット単位 OR マッチングルール (**1.2.840.113556.1.4.804**) は、アサーション文字列のビットの **いずれか** が属性値で表されるかどうかを確認します。(これは部分文字列検索に似ています) この例では、userAccountControl の値には、6 のビットフィールドに設定されるビットのいずれかが必要です。つまり、属性値は 2、4、または 6 のいずれかになります。

```
"(UserAccountControl:1.2.840.113556.1.4.804:=6)"
```

ビット単位検索は、Samba ファイルサーバーの使用など、Windows と Red Hat Enterprise Linux の統合で使用することができます。

14.5. リソース制限による検索パフォーマンスの改善

ディレクトリーが大きいと、データベース内のすべてのエントリーを検索すると、サーバーのパフォーマンスに影響を及ぼす可能性があります。効果的なインデックス化は、特定のシナリオでパフォーマンスを向上できます。ただし、大規模なデータベースでは、パフォーマンスを向上するのに十分な検索範囲が減っていない場合があります。

ユーザーおよびクライアントアカウントで妥当な制限を設定して、エントリーの合計数または個々の検索で費やした合計時間を節約できます。両方により、応答性が向上し、サーバー全体のパフォーマンスが向上します。

検索操作のサーバー制限は、ディレクトリーへのクライアントアプリケーションバインディングの特別な操作属性値を使用して制御されます。以下の検索操作制限を設定できます。

- **ルックスルー制限。** 検索操作で確認できるエントリーの数を指定します。
- **サイズ制限。** 検索操作にしたがって、サーバーがクライアントアプリケーションに返すエントリーの最大数を指定します。
- **時間制限。** サーバーが検索操作の処理に費やす最大時間を指定します。

- **アイドルタイムアウト。**接続が切断される前に、サーバーへの接続がアイドル状態でいられる時間を指定します。
- **範囲のタイムアウト。**範囲を使用した検索のために、別のルックスルー制限を指定します。

クライアントアプリケーションに設定されたリソース制限は、グローバルサーバー設定で設定されるデフォルトのリソース制限よりも優先されます。

注記

Directory Manager は、範囲検索を除き、デフォルトで無制限のリソースを受け取ります。

14.5.1. パフォーマンスおよびリソース制限の検索

詳細は、『Red Hat Directory Server パフォーマンスチューニングガイド』の該当するセクションを参照してください。

14.5.2. 粒度の細かい ID リストサイズ

詳細は、『Red Hat Directory Server パフォーマンスチューニングガイド』の該当するセクションを参照してください。

14.5.3. コマンドラインを使用したユーザーおよびグローバルリソース制限の設定

コマンドラインで、管理者は、簡単なページや範囲検索など、ユーザーレベルのリソース制限、グローバルリソース制限、および特定の種類の検索を設定できます。「検索アルゴリズムの概要」は、これらのリソース制限が Directory Server 検索パフォーマンスにどのように影響するかについて詳しく説明しています。

「コマンドラインを使用したユーザーおよびグローバルリソース制限の設定」は、コマンドラインを使用して各エントリーに設定できる操作属性をリスト表示します。**ldapmodify** を使用して、エントリーに属性を追加します。

ユーザーレベルの属性は各エントリーに設定されますが、グローバル設定属性は適切なサーバー設定エリアに設定されます。

シークスルー制限

検索操作に対して検査するエントリーの数を指定します。この属性を **-1** の値に指定すると、制限がないことを意味します。

- ユーザーレベルの属性: ***nsLookThroughLimit***
- グローバル設定:
  - 属性: ***nsslapd-lookthroughlimit***
  - エントリー: **cn=config,cn=ldbm database,cn=plugins,cn=config**

ページルックアップの制限

ルックスルー制限と同様に、検査するエントリーの数を指定しますが、単純なページング検索操作に特化しています。この属性を **-1** の値に指定すると、制限がないことを意味します。

- ユーザーレベルの属性: ***nsPagedLookThroughLimit***

- グローバル設定:
  - 属性: ***nsslapd-pagedlookthroughlimit***
  - エントリー: **cn=config,cn=ldbm database,cn=plugins,cn=config**

サイズ制限

検索操作にしたがって、サーバーがクライアントアプリケーションに返すエントリーの最大数を指定します。この属性を **-1** の値に指定すると、制限がないことを意味します。

- ユーザーレベルの属性: ***nsSizeLimit***
- グローバル設定:
  - 属性: ***nsslapd-sizelimit***
  - エントリー: **cn=config**

ページサイズ制限

サイズの制限と同様、サーバーがクライアントアプリケーションに戻る最大エントリー数を指定しますが、単純なページング検索操作の場合に限ります。この属性を **-1** の値に指定すると、制限がないことを意味します。

- ユーザーレベルの属性: ***nsPagedSizeLimit***
- グローバル設定:
  - 属性: ***nsslapd-pagedsizelimit***
  - エントリー: **cn=config**

時間制限

サーバーが検索操作の処理に費やす最大時間を指定します。この属性を **-1** の値に指定すると、制限がないことを意味します。

- ユーザーレベルの属性: ***nsTimeLimit***
- グローバル設定:
  - 属性: ***nsslapd-timelimit***
  - エントリー: **cn=config**

アイドルタイムアウト

接続が切断される前に、サーバーへの接続がアイドル状態でいられる時間を指定します。値は秒単位で指定されます。この属性を **-1** の値に指定すると、制限がないことを意味します。

- ユーザーレベルの属性: ***nsidletimeout***
- グローバル設定:
  - 属性: ***nsslapd-idletimeout***
  - エントリー: **cn=config**

ID リストのスキャン制限

検索結果のインデックスファイルから読み込まれるエントリー ID の最大数を指定します。ID リストのサイズがこの値よりも大きい場合、検索はインデックスリストを使用せず、インデックスなしの検索として扱われ、データベース全体を検索します。

- ユーザーレベルの属性: ***nsIDListScanLimit***
- グローバル設定:
  - 属性: ***nsslapd-idlistscanlimit***
  - エントリー: **cn=config,cn=ldbm database,cn=plugins,cn=config**

ページ ID リストスキャンの制限

ID リストスキャンの制限と同様、検索結果のインデックスファイルから読み込まれるエントリー ID の最大数を指定しますが、ページングされた検索操作に対して指定します。

- ユーザーレベルの属性: ***nsPagedIDListScanLimit***
- グローバル設定:
  - 属性: ***nsslapd-pagedidlistscanlimit***
  - エントリー: **cn=config,cn=ldbm database,cn=plugins,cn=config**

範囲のルックアップの制限

範囲検索操作に関するエントリー数を指定します (greater-than、equal-to-or-greater-than、less-than、または equal-to-less-than 演算子を使用した検索)。この属性を **-1** の値に指定すると、制限がないことを意味します。

- ユーザーレベルの属性: 利用できません。
- グローバル設定:
  - 属性: ***nsslapd-rangelookthroughlimit***
  - エントリー: **cn=config,cn=ldbm database,cn=plugins,cn=config**

上記のパラメーターの詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の説明を参照してください。

たとえば、以下はユーザーのサイズ制限を設定します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=user_name,ou=People,dc=example,dc=com
changetype: modify
add: nsSizeLimit
nsSizeLimit: 500
```

ldapmodify 文は、ユーザーのエントリーに ***nsSizeLimit*** 属性を追加して、検索結果のサイズ制限を 500 エントリーにします。

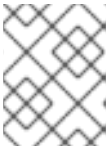
注記

ユーザーが設定を変更できないように、アクセス制御リスト (ACL) を設定します。ACL の詳細は、18章 *アクセス制御の管理*を参照してください。

14.5.4. 匿名バインドでのリソース制限の設定

リソース制限はユーザーエントリーに設定されます。匿名のバインディングは、当然ながら、ユーザーエントリーとは関係ありません。これは、通常グローバルリソース制限が匿名操作に適用されることを意味します。ただし、リソース制限のあるテンプレートユーザーエントリーを作成し、そのテンプレートを匿名バインドに適用することで、匿名バインド専用のリソース制限を設定することができます。

1. テンプレートエントリーを作成し、匿名バインドに適用するリソース制限を設定します。

   注記

   パフォーマンス上の理由から、テンプレートは、エントリーキャッシュは使用しない **cn=config** 接尾辞ではなく、通常のバックエンドになければなりません。

   以下に例を示します。

   ```
   # ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

   dn: cn=anonymous_template,ou=people,dc=example,dc=com
   objectclass: nsContainer
   objectclass: top
   cn: anonymous_template
   nsSizeLimit: 250
   nsLookThroughLimit: 1000
   nsTimeLimit: 60
   ```

2. レプリケーショントポロジー内のすべてのサプライヤーで、テンプレートエントリーの DN を参照するサーバー設定に ***nsslapd-anonlimitsdn*** パラメーターを追加します。「コマンドラインを使用したユーザーおよびグローバルリソース制限の設定」に任意のリソース制限を設定できます。以下に例を示します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-
   anonlimitsdn="cn=anonymous_template,ou=people,dc=example,dc=com"
   ```

14.5.5. 範囲検索のパフォーマンス向上

範囲検索は演算子 (「検索フィルターでの演算子の使用」) を使用して括弧を設定して検索し、ディレクトリー内のエントリーのサブセット全体を返します。たとえば、これにより 1 月 1 日の午前 0 時以降に変更されたすべてのエントリーを検索します。

```
(modifyTimestamp>=20210101010101Z)
```

範囲検索の性質は、ディレクトリー内のすべてのエントリーを評価して、その範囲内にあるかどうかを確認する必要があることです。基本的に、範囲検索は常に ID 検索です。

ほとんどのユーザーの場合は、ルックスルーの制限が開始され、範囲の検索が全 ID 検索に変換するのを防ぎます。これにより、全体的なパフォーマンスが向上し、さまざまな検索結果を加速します。ただし、Directory Manager などの一部のクライアントまたは管理ユーザーには、ルックスルー制限が設定

されていない場合があります。この場合は、範囲検索が完了するまで数分かかるか、無限に続行することがあります。

範囲に対するルックスルー制限を個別に設定することも可能です。これにより、クライアントや管理者ユーザーは、パフォーマンスが低下する可能性のある範囲検索に合理的な制限を設けながらも、高いルックスルー制限を設定することができます。

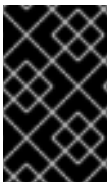
これは ***nsslapd-rangelookthroughlimit*** 属性で設定されます。デフォルト値は **5000** で、デフォルトの ***nsslapd-lookthroughlimit*** 属性値と同じです。

以下に例を示します。

```
# ldapmodify -a -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=config,cn=ldbm database,cn=plugins,cn=config
changetype: add
add: nsslapd-rangelookthroughlimit
nsslapd-rangelookthroughlimit: 7500
```

14.6. 永続検索の使用

永続的な検索は **ldapsearch** で、最初の検索結果が返されても開かれたままになります。



重要

Red Hat Enterprise Linux の OpenLDAP クライアントツールは、永続的な検索に対応していません。ただし、サーバー自体は機能します。その他の LDAP クライアントは、永続的な検索を実行するために使用する必要があります。

永続検索の目的は、ディレクトリーエントリーへの変更の継続的なリストと、ハイブリッド検索や changelog などの完全なエントリー自体を提供することです。そのため、検索コマンドでは、どのエントリーを返すか (検索パラメーター)、どのような変更によってエントリーが返されるか (エントリー変更パラメーター) を指定する必要があります。

永続的な検索は、Directory Server にアクセスするアプリケーションまたはクライアントで特に役立ち、2 つの重要な利点を提供します。

- 一貫性のあるローカルキャッシュと現在のローカルキャッシュを保持します。

  クライアントは、ディレクトリーに接続してクエリーを試行する前にローカルキャッシュをクエリーします。永続検索は、これらのクライアントのパフォーマンスを改善するのに必要なローカルキャッシュを提供します。

- ディレクトリーアクションを自動的に開始します。

  永続キャッシュはエントリーの変更時に自動的に更新され、永続検索結果ではエントリーに対して実行された変更の種類を表示できます。別のアプリケーションでは、その出力を利用してエントリーを自動的に更新することができます。たとえば、新しいユーザーのためにメールサーバーにメールアカウントを自動的に作成したり、固有のユーザー ID 番号を生成したりすることができます。

永続検索を実行する場合のパフォーマンスに関する考慮事項がいくつかあります。

- クライアントの接続解除時に **ldapsearch** は通知を送信せず、検索が切断されている間に変更についての通知が送信されません。これは、クライアントのキャッシュが切断されても更新さ

れないことを意味し、切断中に変更した新しいエントリー、変更されたエントリー、または削除されたエントリーでキャッシュを更新する適切な方法はありません。

- 攻撃者は、サービス拒否攻撃を開始するために多数の永続検索を開くことができます。
- 永続的な検索では、Directory Server とクライアント間で TCP 接続を開放する必要があります。これは、サーバーが多くのクライアント接続を許可し、アイドル状態の接続を閉じる方法を持つ場合にのみ行う必要があります。

アクセスログでは、永続検索はタグ **options=persistent** で識別されます。

```
[12/Jan/2009:12:51:54.899423510 -0500] conn=19636710736396323 op=0 SRCH
base="dc=example,dc=com" scope=2 filter="(objectClass=person)" attrs=ALL options=persistent
```

14.7. 指定したコントロールでの検索

Directory Server は、DSE の ***supportedControls*** 属性に制御を定義しています。これらの中には、レプリケーションのようなサーバーの操作を定義するものもあれば、Get Effective Rights や、クライアントが LDAP 操作でサーバーに渡すことができる制御の逆参照などの拡張操作を許可するものもあります。

これらの制御は、**-E** オプションを使用して指定できます。制御 OID、**ldapsearch** の重大度、およびその制御操作に必要な情報を指定します。

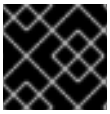
```
-E '[!]control_OID:control_information'
```

一部の制御 (サーバー側のソートやシンプルなページングされた結果など、) には、検索操作にコントロールを渡すために使用できるエイリアスがあります。制御エイリアスを使用すると、制御がクライアントによって認識されるため、結果がフォーマットされます。

14.7.1. 効果のあるユーザー権限の取得

有効な権利を取得する検索コントロールは、コントロール OID を使用して渡されます。以下に例を示します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b "dc=example,dc=com"
-s sub -x -E '!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=jsmith,ou=people,dc=example,dc=com' "
(objectclass=*)"
```



重要

コントロールの OID が渡されると、検索結果の形式化がされません。

Get Effective Rights の検索については、アクセス制御の章「エントリーのアクセス権利の確認 (Get Effective Rights)」で詳細に説明されています。

14.7.2. サーバー側のソートの使用

サーバー側のソートは、**-E** フラグと **sss** 制御エイリアスを使用して、他の制御操作として実行されます。操作の構造は、結果をソートし、任意でソート順序および順序ルールである属性を設定します。

```
-E sss=[-]attribute_name:[ordering_rule_OID]
```

ダッシュ (**-**) は、ソート順序を元に戻す任意のフラグで、降順の降順を逆に実行します。「一致するルールの使用」 のマッチングルールテーブルには、Directory Server でサポートされる順序ルールが含まれています。

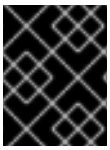
以下に例を示します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b "dc=example,dc=com"
-s sub -x -E sss=-uidNumber:2.5.13.15 "(objectclass=*)"
```

14.7.3. 逆参照検索の実行

逆参照 検索は、エントリーの相互参照を追跡し、参照されたエントリーに関する情報を返す簡単な方法です。たとえば、グループエントリーには、そのメンバーのユーザーエントリーへの参照が含まれます。通常検索は、最初にグループを検索し、そのメンバーをリスト表示し、各メンバーに個別の検索が必要になります。グループエントリーの逆参照検索は、メンバーに関する情報 (場所、メールアドレス、マネージャーなど) を 1 つの検索要求でグループの情報と **ともに** 返します。

逆参照は多くのクライアント操作を簡素化し、実行した検索操作の数を減らします。クロスリンクは、エントリー間の関係を表示します。一部の操作では、1 つのエントリーから複数のリンクのリストを取得し、後続の検索を実行してリスト上の各エントリーから情報を取得しないといけない場合があります。逆参照により、検索のシーケンスを 1 つの検索に統合することができます。



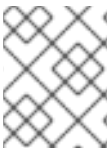
重要

逆参照操作は、OpenLDAP コマンドラインツールのバージョン 2.4.18 以降、または逆参照検索をサポートするその他のクライアントを使用して行う必要があります。

逆参照引数の形式は次のとおりです。

```
-E 'deref=deref_attribute:list_of_attributes'
```

deref_attribute は、参照が含まれる検索ターゲットの属性です。これには、**member** または **manager** などの値に DN を持つ属性を使用できます。

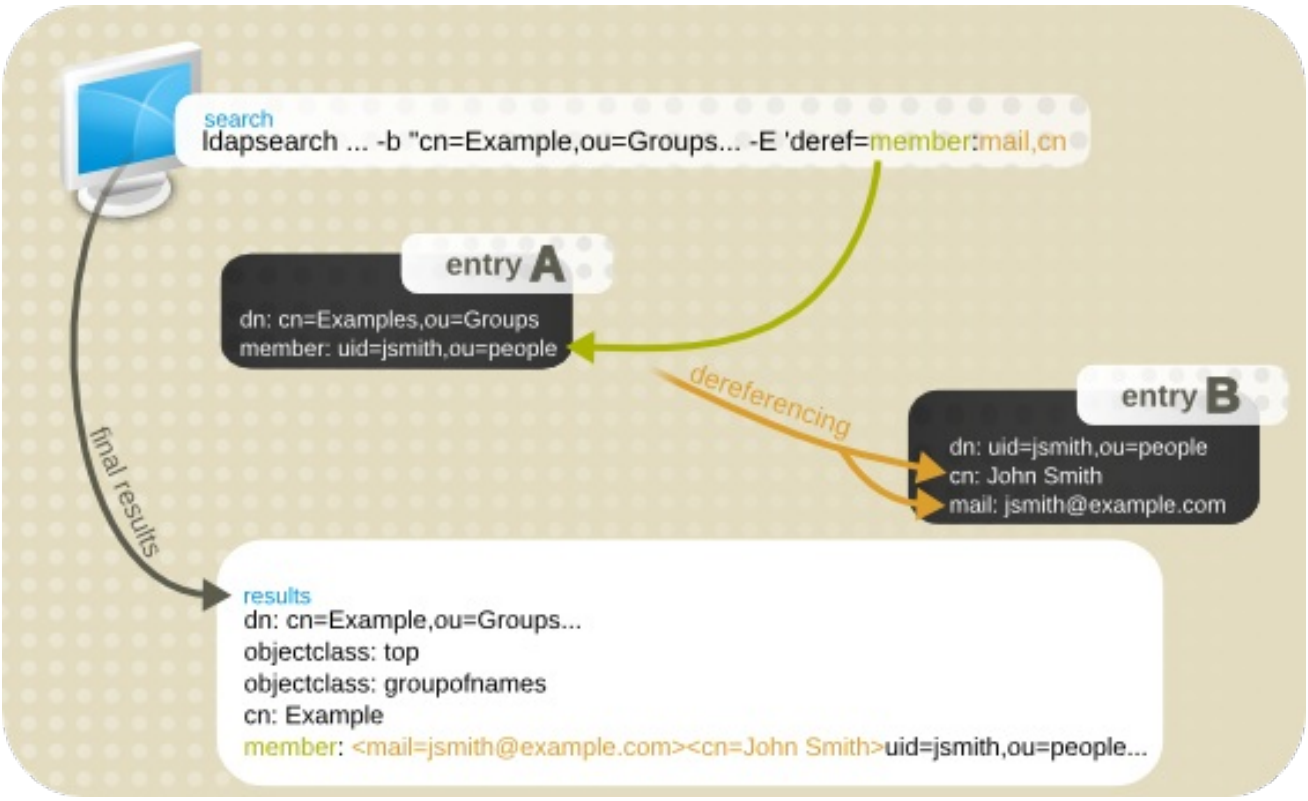


注記

deref_attribute の値は DN にする必要がありますが、この属性に対する実際の定義された構文は DN 構文 (**1.3.6.1.4.1.1466.115.121.1.12**) である必要があります。

list_of_attributes は、参照されたエントリーの 1 つ以上の属性で、プライマリーの検索結果とともに返されます。**l,mail,cn** のように、複数の属性をコンマで区切ることができます。

図14.1 簡単な参照検索コマンド



検索引数で要求された逆参照された情報は、残りの検索結果とともに返されます。たとえば、この逆参照検索は、検索ターゲットエントリー (エンジニアグループ) の ***member*** 属性を **deref_attribute** として使用するように指示します。その後、各メンバーの locality 属性を返します。

```
# ldapsearch -x -D "cn=Directory Manager" -W -b "cn=Example,ou=Groups,dc=example,dc=com" -E
'deref=member:mail,cn' "(objectclass=*)"

# Engineers, Groups, example.com
dn: cn=Engineers,ou=Groups,dc=example,dc=com
control: 1.3.6.1.4.1.4203.666.5.16 false MIQAAADNMIQAAAA1BAZtZW1iZXIEK2NuPURld

mVsb3BlcnMsIG91PUdyb3VwcywgZGM9ZXhhbXBsZSxkYz1jb20whAAAADIEBm1lbWJlcgQoY249VG

VzdGVycywgb3U9R3JvdXBzLCBkYz1leGFtcGxlLGRjPWNvbTCEAAAAVAQGbWVtYmVyBCp1aWQ9ZW5

nLCBvdT1lbmdpbmVlcmluZywgZGM9ZXhhbXBsZSxkYz1jb22ghAAAABowhAAAABQEAWwxhAAAAAs
E
 CUNhbWJyaWRnZQ==
# member: <mail=jsmith@example.com><cn=John
Smith>;uid=jsmith,ou=people,dc=example,dc=com
 objectClass: top
objectClass: inetuser
objectClass: groupofnames
cn: Engineers
member: uid=jsmith,ou=people,dc=example,dc=com
```

14.7.4. 単純なページ結果の使用

検索結果は非常に大きくなる可能性があり、結果処理の一部が結果を整理することです。その方法の一つとして、**ページングされた単純な結果** を使用することがあります。これは、結果を一定の長さのページに分割するコントロールです。

シンプルなページの結果で、一度に表示するエントリーの数を設定できます。結果は一度に 1 ページ経由でスクロールされるため、ダイジェストで結果が容易になります。制御の完全な動作は RFC 2696 で説明されています。

ページングされた単純な結果は、Directory Server の LDAP コントロール拡張機能として実装されます。OID は **1.2.840.113556.1.4.319** です。

簡単なページ結果の仕組み

ページ結果の簡単な検索を開始すると、以下のようになります。

1. クライアントは検索をサーバーに送り、ページ付けの結果が制御し、最初のページで返すレコードの数を制御します。

2. Directory Server がデータの返信を開始する前に、サーバーは合計で何件のレコードを返信できるかの見積もりを生成します。

   レコードの推定値は、正確な数ではありません。返されるレコードの合計数は推定値よりも小さくなります。このようなシナリオの理由は以下の通りです。

   - 検索フィルターで使用される属性はインデックスに存在しません。最適な結果を得るには、クエリーされた属性をすべてインデックス化する必要があります。

   - エントリーがクライアントに送信される前に、アクセス制御リスト (ACL) が検証されます。パーミッションが十分にないため、エントリーが返されなくなります。

   推定値を生成した後、サーバーは最初の結果セット、Cookie、および推定レコード数を送信します。

3. 返されたレコードがクライアントに表示されます。ユーザーは、次のリクエストで返されるレコードの数を入力できるようになりました。要求された番号は、Cookie と一緒にサーバーに送信されます。

4. サーバーは要求された数のレコードをデータベースから取得し、それらを Cookie と共にクライアントに送信します。

5. 前述の 2 つのステップは、すべてのレコードが送信されるか、検索がキャンセルされるまで繰り返されます。

シンプルなページ結および OpenLDAP ツール

ldapsearch の簡単なページの結果検索オプションの形式は以下のとおりです。

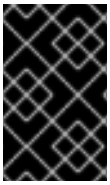
```
-E pg=size
```

size の値はページサイズまたはページごとに含むエントリー数です。以下に例を示します。

```
ldapsearch -x -D "cn=Directory Manager" -W -b "ou=Engineers,ou=People,dc=example,dc=com" -E
pg=3 "(objectclass=*)" cn

dn: uid=jsmith,ou=Engineers,ou=People,dc=example,dc=com
  cn: John Smith
```

```
dn: uid=bjensen,ou=Engineers,ou=People,dc=example,dc=com
  cn: Barbara Jensen

dn: uid=hmartin,ou=Engineers,ou=People,dc=example,dc=com
  cn: Henry Martin

Results are sorted.
next page size (3): 5
```

末尾のタグには、検索で設定されたページサイズ (カッコ内の数字) が表示されています。コロンの後に、次のページのページサイズが表示されるため、以下に示すように **5** と入力すると、次のページが 5 つのエントリーで開きます。



重要

簡素なページ結果操作は、OpenLDAP コマンドラインツールのバージョン 2.4.18 以降、または Perl Net::LDAP などの簡素なページング結果をサポートするその他のクライアントを使用して実行する必要があります。

ページングされた簡素な結果およびサーバー一致

ページングされた簡素な結果は、サーバー側のソートと併用できます。サーバー側のソートは、クライアントではなくサーバーでソートプロセスを実行する制御です。これは通常、特定のマッチングルールを使用する検索を行います。(この動作は RFC 2891 で定義されています。) OpenLDAP クライアントツールは、簡素なページングされた結果制御でサーバー側のソートをサポートしませんが、Perl Net::LDAP などのその他の LDAP ユーティリティーは両方をサポートします。

1 つの接続に複数の簡素なページングされた結果要求

一部のクライアントは Directory Server への接続を 1 つ開きますが、簡素なページングされた結果拡張を使用する複数の検索要求など、複数の操作要求を送信します。

Directory Server は、複数の簡素なページングされた検索を管理および解釈できます。各検索は、アレイ内のエントリーとして追加されます。ページングされた検索要求が最初に送信されると、Cookie が作成され、検索結果に関連付けられます。結果の各ページはその Cookie で返されます。Cookie は結果の次のページを要求するために使用されます。最後のページでは、Cookie が空になり、結果が終了したことを示します。これにより、各検索結果のセットが個別に保持されます。

1 つの接続に複数の簡素なページングされた結果がある場合、タイムアウト制限は引き続き監視されますが、**すべて** のオープン検索要求は、**いずれか** のページ検索が切断される前に、設定された時間制限に到達します。

VLV インデックスと対照的な、簡素なページングされた結果

VLV インデックスは簡素なページングされた結果に類似しているため、それらのインデックスは、結果の閲覧可能なリストも返すことになります。主な違いは、リストの生成方法です。簡素なページングされた結果は検索ごとに計算されますが、VLV インデックスは永続的なリストになります。全体的に、VLV インデックスは検索速度が速いですが、サーバー側の設定が必要で、サーバーが維持するためのオーバーヘッドがあります。

注記

シンプルなページの結果と VLV インデックスは、同じ検索では使用 **できません**。簡素なページングされた結果は、すでに参照しているインデックスである VLV インデックスの操作を試みます。VLV インデックスを使用した検索に制御が渡された場合、サーバーは **UNWILLING_TO_PERFORM** エラーを返します。

VLV インデックスの詳細は、「仮想リストビューコントロールを使用して、大規模な検索結果の連続したサブセットを要求する」を参照してください。

14.7.5. 読み取り前および後のエントリーレスポンス制御

Red Hat Directory Server は、RFC 4527 に準拠した、読み取り前および読み取り後のエントリー応答制御をサポートします。クライアントが応答制御を要求すると、LDAP 検索エントリーが返されます。これには、更新前および更新後に属性の値が含まれます。

事前読み取りの制御が使用されると、LDAP 検索クエリーには変更前に指定した属性の値が含まれます。読み取り後の制御が使用される場合、クエリーには変更後の属性の値が含まれます。両方の制御を同時に使用できます。たとえば、***description*** 属性を更新し、変更前および変更後に値を表示するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -x \
    -e \!preread=description -e \!postread=description
dn: uid=user,ou=People,dc=example,dc=com
changetype: modify
replace: description
description: new description

modifying entry "uid=user,ou=People,dc=example,dc=com"
control: 1.3.6.1.1.13.1 false ZCkEJXVpZD1qdXNlcixvdT1QZW9wbGUsZGM9ZXhhbXBsZSxk
 Yz1jb20wAA==
# ==> preread
dn: uid=user,ou=People,dc=example,dc=com
description: old description
# <== preread
control: 1.3.6.1.1.13.2 false ZEsEJXVpZD1qdXNlcixvdT1QZW9wbGUsZGM9ZXhhbXBsZSxk
Yz1jb20wIjAgBAtkZXNjcmlwdGlvbjERBA9uZXcgZGVzY3JpcHRpb24=
# ==> postread
dn: uid=user,ou=People,dc=example,dc=com
description: new description
# <== postread
```

第15章 レプリケーションの管理

レプリケーション は、ディレクトリーデータが 1 つの Red Hat Directory Server インスタンスから相互に自動的に同期されるメカニズムで、単一のサーバー設定以外にディレクトリーサービスを拡張する上で重要なメカニズムです。本章では、単一サプライヤーレプリケーション、マルチサプライヤーレプリケーション、およびカスケードレプリケーションを設定するサプライヤーサーバーおよびコンシューマーサーバーで実行するタスクについて説明します。

15.1. レプリケーションの概要

レプリケーションとは、Directory Server 間でディレクトリーデータを自動的に同期させる仕組みです。あらゆる種類 (エントリーの追加、変更、削除、または名前変更) の更新は、レプリケーションを使用して他の Directory Server に自動的にミラーリングされます。

- 「複製されるディレクトリーユニット」
- 「読み取り/書き込みレプリカおよび読み取り専用レプリカ」
- 「サプライヤーとコンシューマー」
- 「Changelog」
- 「レプリケーション ID」
- 「レプリカ合意」

15.1.1. 複製されるディレクトリーユニット

レプリケートできるディレクトリーの最小単位はデータベースです。つまり、データベース全体を複製できますが、データベース内のサブツリーは複製できません。したがって、ディレクトリーツリーを作成する際に、情報を配信する方法を決定する際に、レプリケーションプランを検討してください。

レプリケーションでは、1 つのデータベースが 1 つの接尾辞に対応する必要もあります。つまり、カスタム分散ロジックを使用して 2 つ以上のデータベースに分散される接尾辞 (または名前空間) は複製できません。このトピックの詳細については、「データベースの作成および維持」を参照してください。

15.1.2. 読み取り/書き込みレプリカおよび読み取り専用レプリカ

レプリケーションに参加するデータベースは **レプリカ** と呼ばれます。レプリカには、読み書き可能なものと、読み取り専用の 2 種類があります。**読み取り/書き込みレプリカ** には、ディレクトリー情報のサプライヤーコピーが含まれ、更新できます。**読み取り専用のレプリカ** サービスは読み取り、検索、および比較要求ですが、読み取り/書き込みレプリカに対する更新操作をすべて参照します。サーバーは、任意の数の読み取り専用または読み書きレプリカを保持できます。

15.1.3. サプライヤーとコンシューマー

別のサーバーのレプリカに送信されるレプリカを保持するサーバーは、そのレプリカの **サプライヤー** と呼ばれます。別のサーバーから受信したレプリカを保持するサーバーは、そのレプリカの **コンシューマー** と呼ばれます。通常、サプライヤーサーバーのレプリカは読み取り/書き込みレプリカで、コンシューマーサーバーのレプリカは 2 つの例外を持つ読み取り専用レプリカになります。

- レプリケーションをカスケードするとき、ハブサーバーは、コンシューマーに提供する読み取り専用レプリカを保持します。「カスケードレプリケーション」に詳細情報があります。

- マルチサプライヤーレプリケーションの場合、**サプライヤー** は同じ情報についてサプライヤーとコンシューマーの両方を設定します。詳細は、「マルチサプライヤーのレプリケーション」を参照してください。

レプリケーションは、常にサプライヤー・サーバーによって開始され、コンシューマーによって開始されることはありません (**サプライヤーが開始するレプリケーション**)。サプライヤーが開始するレプリケーションでは、サプライヤーサーバーを設定して、データを複数のコンシューマーサーバーに送信できます。

15.1.4. Changelog

すべてのサプライヤーサーバーは、**changelog** と、サプライヤーまたはハブがそのコンシューマーに送信する必要がある変更の記録です。changelog は、レプリカで発生した変更を保持する特別な種類のデータベースです。次に、サプライヤーサーバーは、マルチサプライヤーレプリケーションの場合には、コンシューマーサーバーに保存されているレプリカまたは他のサプライヤーにこの変更を再生します。

エントリーが変更されると、実行された LDAP 操作を記述する変更レコードが changelog に記録されます。

changelog は、メインデータベースと同じデータベース環境を使用します。メインのデータベースの一部として changelog を実装すると、データベースおよび changelog が常に同期され、必要なデータベースキャッシュサイズが減少し、バックアップと復元操作が簡素化されます。

重要

changelog は、サーバーがシャットダウンするときに changelog RUV エントリーをデータベースにのみ書き込み、それ以外は RUV がメモリー内で管理されます。サプライヤーのデータベースをバックアップする場合は、**dsctl db2bak** コマンドまたは Web コンソールを使用します。いずれの方法でも、バックアップを開始する前に RUV がデータベースに書き込まれます。

Directory Server では、changelog はサーバーによる内部使用のみを目的としています。

15.1.5. レプリケーション ID

2 つのサーバー間でレプリケーションが発生すると、レプリケーションプロセスは、**レプリケーションマネージャー** エントリーと呼ばれる特別なエントリーを使用して、レプリケーションプロトコルの交換を特定し、ディレクトリーデータへのアクセスを制御します。レプリケーションマネージャーエントリーまたはレプリケーション中に使用されるエントリーは、以下の基準を満たしている必要があります。

- これは、**cn=config** エントリーのコンシューマーサーバーに作成されます。
- 別のサーバーから更新を受け取る **すべて** のサーバー (つまり、すべてのハブまたは専用のコンシューマー) でこのエントリーを作成します。
- レプリカがコンシューマーまたはハブとして設定されている場合は、このエントリーを、レプリケーションの更新を実行する権限のあるものとして指定する必要があります。
- レプリカ合意はサプライヤーサーバーで作成され、このエントリーの DN をレプリカ合意に指定する必要があります。
- レプリケーションコンテキストでは、このエントリーは、その特別なユーザープロファイルを使用して、そのレプリケーションアグリーメントに含まれるデータベースのコンシューマー

サーバーで定義されたアクセス制御規則をバイパスします。レプリケーションコンテキストの外では、レプリケーションマネージャーは通常の操作を実行するときに ACI の影響を受けることに注意してください。

15.1.6. レプリカ合意

Directory Server はレプリカ合意を使用してレプリケーション設定を定義します。レプリカ合意は、**1つ**のサプライヤーと **1つ** のコンシューマーとの間のレプリケーションのみを説明します。この合意はサプライヤーサーバーに設定し、必要なレプリケーション情報をすべて指定する必要があります。

- 複製されるデータベース。
- データがプッシュされるコンシューマーサーバー
- レプリケーションが実行する曜日および時間帯
- サプライヤーサーバーがバインドに使用する必要のある DN および認証情報 (レプリケーションマネージャーエントリーまたはサプライヤーバインド DN)
- 接続をセキュアにする方法 (TLS、クライアント認証)。
- 複製されない属性 (一部レプリケーション)

15.1.7. 一部レプリケーションを使用した属性のサブセットの複製

一部レプリケーションは、サプライヤーからコンシューマー (または別のサプライヤー) に送信されない特定の属性のサブセットを設定します。したがって、管理者は、含まれるすべての情報を複製したり、全エントリーのすべての情報を複製せずに、データベースを複製できます。

一部レプリケーションは、エントリーごとではなく、レプリカ合意ごとに有効になり、設定されます。レプリケーションから属性を除外すると、レプリカ合意の範囲内のすべてのエントリーと同等に適用されます。

スキーマで任意 (**MAY** キーワード) として定義された属性については、増分更新と全体更新で複製する属性を別々に設定することができます。増分更新リスト (***nsDS5ReplicatedAttributeList***) は、一部レプリケーションを有効にするために常に設定される必要があります。唯一の属性が設定されている場合は、増分更新と合計更新の両方に適用されます。オプションの ***nsDS5ReplicatedAttributeListTotal*** 属性は、更新の合計に追加の一部レプリケーション一覧を設定します。これは、「合計更新および増分更新での異なる一部レプリケーション属性の設定」で説明されています。

注記

除外された属性への更新が依然として変更イベントをトリガーし、空のレプリケーション更新を生成します。***nsds5ReplicaStripAttrs*** 属性は、空のレプリケーションイベントでは送信できず、更新シーケンスから削除される属性の一覧を追加します。これには、***modifiersName*** のような運用上の利便性が含まれます。

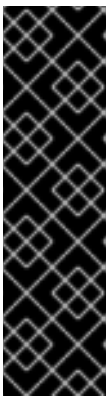
レプリケーションイベントが **空でない** 場合は、ストライピングされた属性 **が** 複製されます。これらの属性は、イベントが空である場合にのみ更新から削除されます。

15.1.8. TLS 上のレプリケーション

セキュリティー上の理由から、TLS 接続を介してデータのみを複製するように、レプリケーションに関連する Directory Server インスタンスを設定します。属性の暗号化を有効にすると、セキュアな接続がレプリケーションに常に必要になります。

前提条件
TLS でレプリケーションを設定する前に、以下の前提条件を満たす必要があります。

- サプライヤーサーバーとコンシューマーサーバーの両方が TLS を使用するように設定します。「Directory Server での TLS の有効化」を参照してください。
- コンシューマーサーバーが、サプライヤーサーバーの証明書を サプライヤー DN として認識するように設定します。これは、簡易認証ではなく TLS クライアント認証のみを使用します。「証明書ベースのクライアント認証の使用」を参照してください。



重要

TLS ハンドシェイク中に、サプライヤーの証明書がサーバー証明書としてのみ動作し、同時にクライアントとしては動作しない場合は、証明書ベースの認証を用いて TLS 上で設定されたレプリケーションが失敗するという問題がありました。証明書ベースの認証でのレプリケーションでは、リモートサーバーへの認証に Directory Server のサーバー証明書を使用します。

certutil を使用して証明書署名要求 (CSR) を生成する場合は、 ***--nsCertType=sslClient,sslServer*** オプションをコマンドに渡し、必要な証明書を設定します。

TLS でのレプリケーションの設定
レプリケーションの設定に関する詳細は、以下を参照してください。

- ユーザー名とパスワード認証を使用する場合:
  - 「単一サプライヤーレプリケーション」
  - 「マルチサプライヤーのレプリケーション」
  - 「カスケードレプリケーション」
- 証明書ベースの認証を使用する場合は、「証明書ベースの認証を使用するようにレプリケーションパートナーの設定」を参照してください。

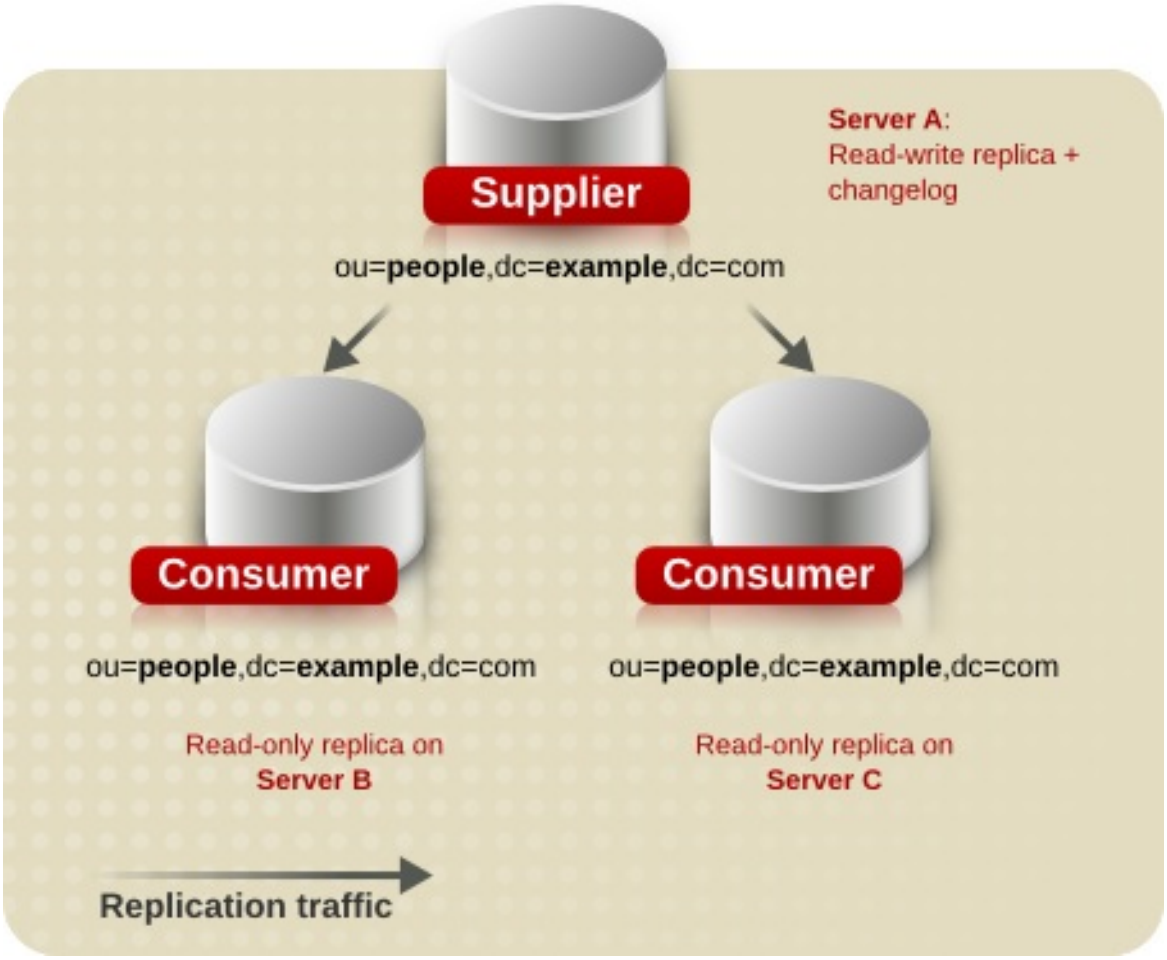
15.2. 単一サプライヤーレプリケーション

単一サプライヤーレプリケーションのシナリオでは、ディレクトリーデータのサプライヤーコピーが、**サプライヤーサーバー** と呼ばれる 1 台のサーバーの 1 つの読み取り/書き込みレプリカに保持されます。サプライヤーは、このレプリカの changelog も維持します。**コンシューマーサーバー** と呼ばれる別のサーバーでは、ディレクトリーの読み取り専用コピーが保存されます。単一サプライヤーレプリケーション環境では、複数のコンシューマーを実行できます。

接尾辞が多数の検索要求を受け取りますが、書き込み要求数が少ない場合などに、単一サプライヤーレプリケーショントポロジーを使用します。負荷を分散するために、クライアントはトポロジー内のすべてのサーバーで接尾辞を検索し、書き込み要求をサプライヤーに送信します。

以下の図は、2 つのコンシューマーを持つ単一サプライヤーレプリケーション環境を示しています。

図15.1 単一サプライヤーレプリケーション



コマンドラインまたは Web コンソールを使用して、単一サプライヤーレプリケーショントポロジーを設定します。参照:

- 「コマンドラインを使用した単一サプライヤーレプリケーションの設定」
- 「Web コンソールを使用した単一サプライヤーレプリケーションの設定」

15.2.1. コマンドラインを使用した単一サプライヤーレプリケーションの設定

以下の例では、既存の Directory Server インスタンスが **supplier.example.com** という名前のホストで実行されていることを前提としています。このホストは、レプリケーショントポロジーに設定されるサプライヤーとして機能します。以下の手順では、**consumer.example.com** という名前の読み取り専用コンシューマーをトポロジーに追加する方法と、**dc=example,dc=com** 接尾辞に単一サプライヤーレプリケーションを設定する方法を説明します。

コンシューマーで実行する手順
consumer.example.com ホスト:

1. Directory Server をインストールして、インスタンスを作成します。詳細は、『Red Hat Directory Server インストールガイド』 を参照してください。
2. データベースなしでインスタンスを作成した場合には、接尾辞のデータベースを作成します。たとえば、**dc=example,dc=com** 接尾辞に **userRoot** という名前のデータベースを作成するには、以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://consumer.example.com backend \
    create --suffix="dc=example,dc=com" --be-name="userRoot"
```

接尾辞のデータベース作成に関する詳細は、「接尾辞の作成」を参照してください。

3. 接尾辞のレプリケーションを有効にし、レプリケーションマネージャーアカウントを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://consumer.example.com replication \
    enable --suffix="dc=example,dc=com" --role="consumer" \
    --bind-dn="cn=replication manager,cn=config" --bind-passwd="password"
```

このコマンドは、**consumer.example.com** ホストを **dc=example,dc=com** 接尾辞のコンシューマーとして設定します。さらに、サーバーは指定のパスワードで **cn=replication manager,cn=config** ユーザーを作成し、このアカウントがこのホストに接尾辞の変更を複製することができます。

接尾辞に複数のコンシューマーを追加するには、各コンシューマーでこの手順を繰り返します。

サプライヤーで実施する手順

supplier.example.com ホスト:

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

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

このコマンドは、**supplier.example.com** ホストを **dc=example,dc=com** 接尾辞のサプライヤーとして設定し、このエントリーのレプリカ ID を **1** に設定します。

重要

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

2. レプリカ合意を追加し、コンシューマーを初期化します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt \
    create --suffix="dc=example,dc=com" --host="consumer.example.com" --port=636 \
    --conn-protocol=LDAPS --bind-dn="cn=replication manager,cn=config" \
    --bind-passwd="password" --bind-method=SIMPLE --init \
    example-agreement
```

このコマンドは、**example-agreement** という名前のレプリカ合意を作成します。レプリカ合意は、コンシューマーへのデータの接続や複製時にサプライヤーが使用するコンシューマーのホスト名、プロトコル、認証情報などの設定を定義します。

この合意の作成後、Directory Server はコンシューマーを初期化します。後ほどコンシューマーを初期化する場合は、**--init** オプションを省略します。コンシューマーを初期化する前にレプリケーションが起動しないことに注意してください。コンシューマーの初期化の詳細は、「コンシューマーの初期化」を参照してください。

コマンドで使用するオプションの詳細は、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt --help
```

3. 初期化が成功したかどうかを確認します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt \
    init-status --suffix="dc=example,dc=com" example-agreement
Agreement successfully initialized.
```

複製するデータ量によっては、初期化に時間がかかる場合があります。

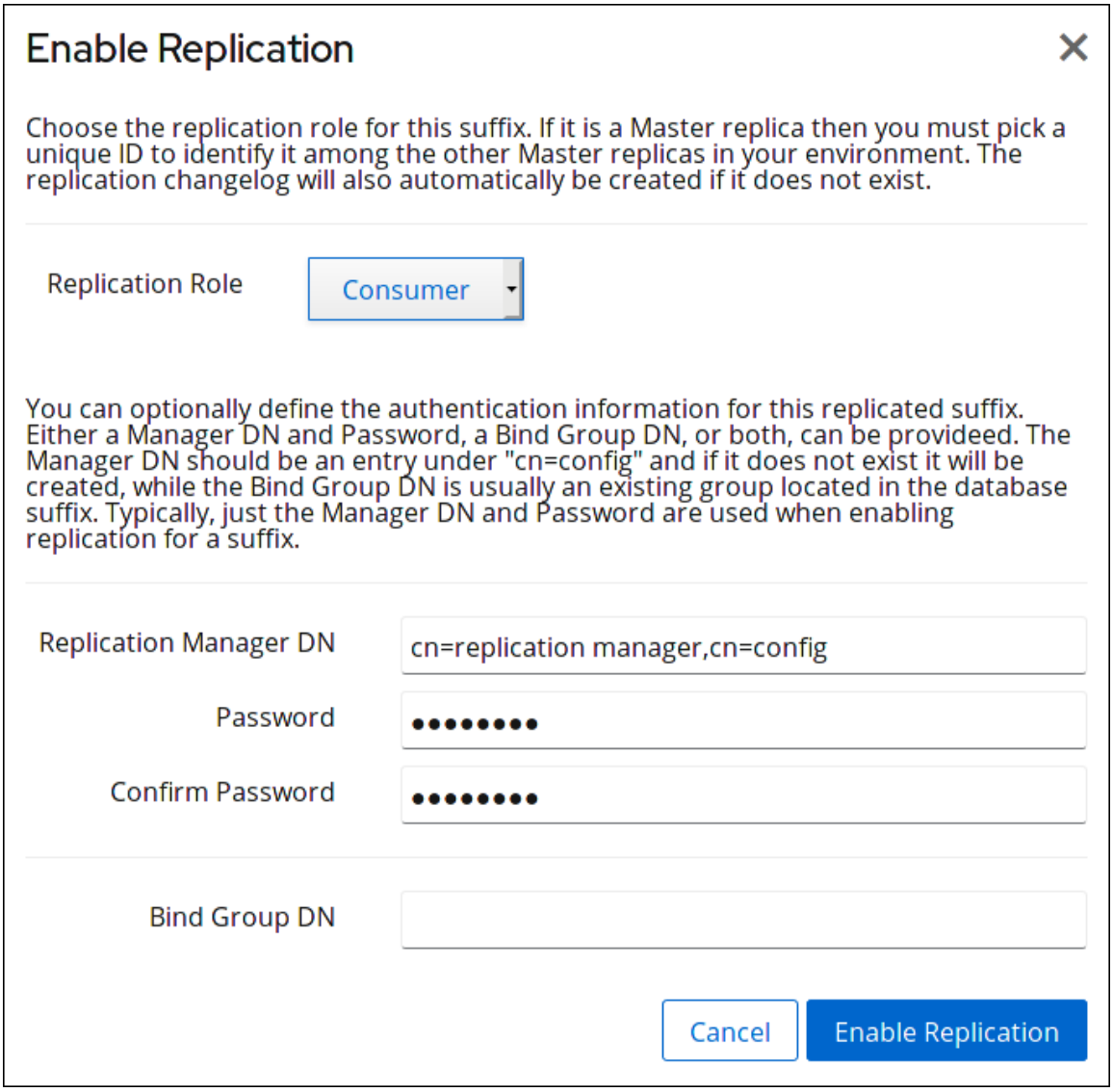
トポロジーに、接尾辞の複数のコンシューマーを追加する場合は、各コンシューマーのサプライヤーで手順を繰り返します。ただし、接尾辞のレプリケーションは、サプライヤーで 1 回のみ有効にする必要があります。

15.2.2. Web コンソールを使用した単一サプライヤーレプリケーションの設定

以下の例では、既存の Directory Server インスタンスが **supplier.example.com** という名前のホストで実行されていることを前提としています。このホストは、レプリケーショントポロジーに設定されるサプライヤーとして機能します。以下の手順では、**consumer.example.com** という名前の読み取り専用コンシューマーをトポロジーに追加する方法と、**dc=example,dc=com** 接尾辞に単一サプライヤーレプリケーションを設定する方法を説明します。

コンシューマーで実行する手順
consumer.example.com ホスト:

1. Directory Server をインストールして、インスタンスを作成します。詳細は、『Red Hat Directory Server インストールガイド』 を参照してください。
2. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
3. インスタンスを選択します。
4. データベースなしでインスタンスを作成した場合には、接尾辞のデータベースを作成します。接尾辞のデータベース作成に関する詳細は、「接尾辞の作成」を参照してください。
5. 接尾辞のレプリケーションを有効にします。
    a. **Replication** メニューを開きます。
    b. **dc=example,dc=com** 接尾辞を選択し、**Enable Replication** をクリックします。
    c. **Replication Role** フィールドで **Consumer** を選択し、作成するレプリケーションマネージャーアカウントの DN およびパスワードを入力します。以下に例を示します。



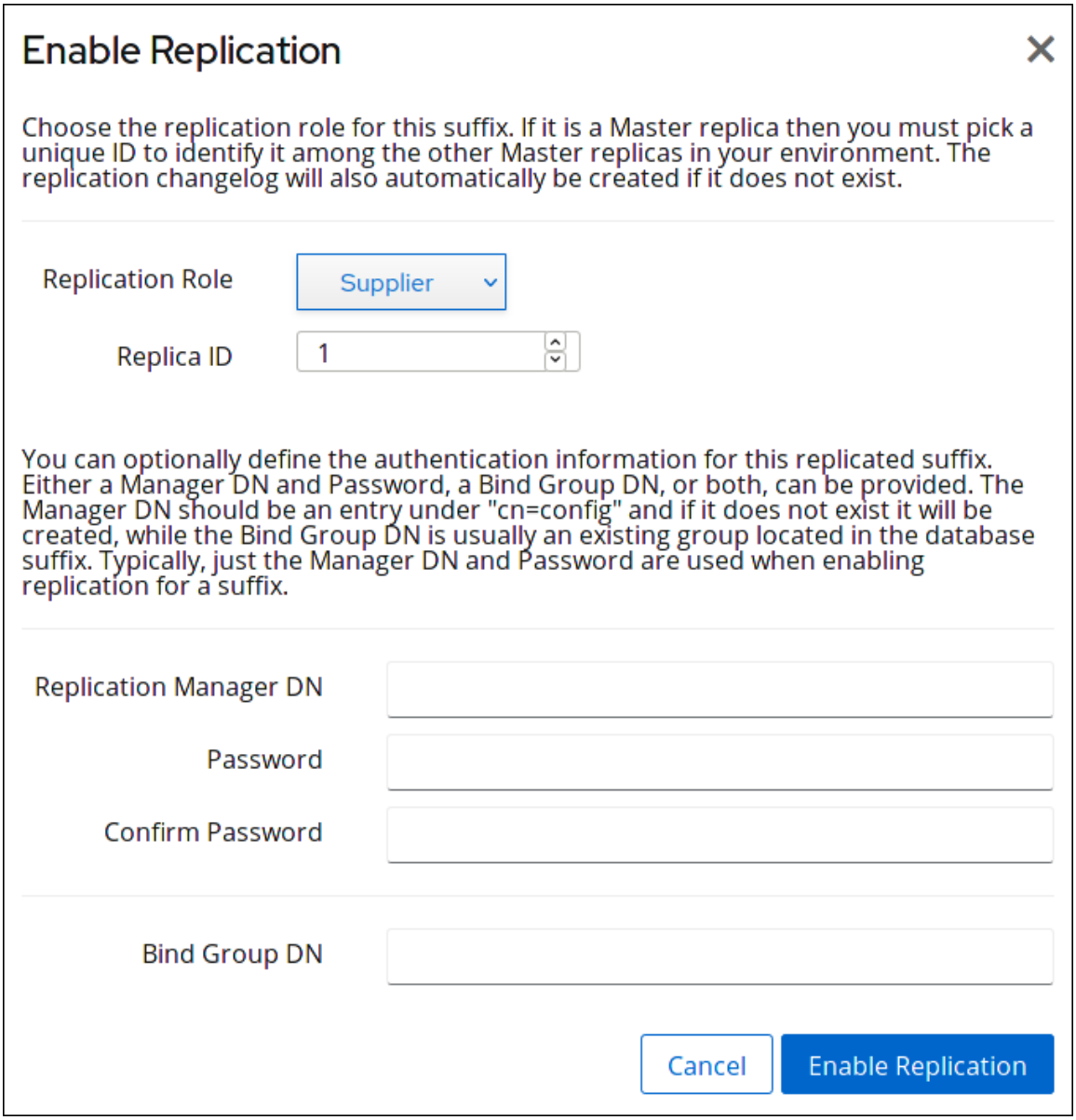
この設定により、ホストを **dc=example,dc=com** 接尾辞のコンシューマーとして設定します。さらに、サーバーは指定のパスワードで **cn=replication manager,cn=config** ユーザーを作成し、このアカウントがこのホストに接尾辞の変更を複製することができます。

d. **Enable Replication** をクリックします。

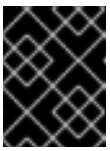
接尾辞に複数のコンシューマーを追加するには、各コンシューマーでこの手順を繰り返します。

サプライヤーで実施する手順
supplier.example.com ホスト:

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **dc=example,dc=com** 接尾辞のレプリケーションを有効にします。
   a. **Replication** メニューを開きます。
   b. **dc=example,dc=com** 接尾辞を選択し、**Enable Replication** をクリックします。
   c. **Replication Role** フィールドで **Supplier** を選択し、レプリカ ID を入力し、**Replication Authentication** エリアのフィールドを空のままにします。以下に例を示します。



これにより、ホストを **dc=example,dc=com** 接尾辞のサプライヤーとして設定し、このエントリーのレプリカ ID を **1** に設定します。



重要

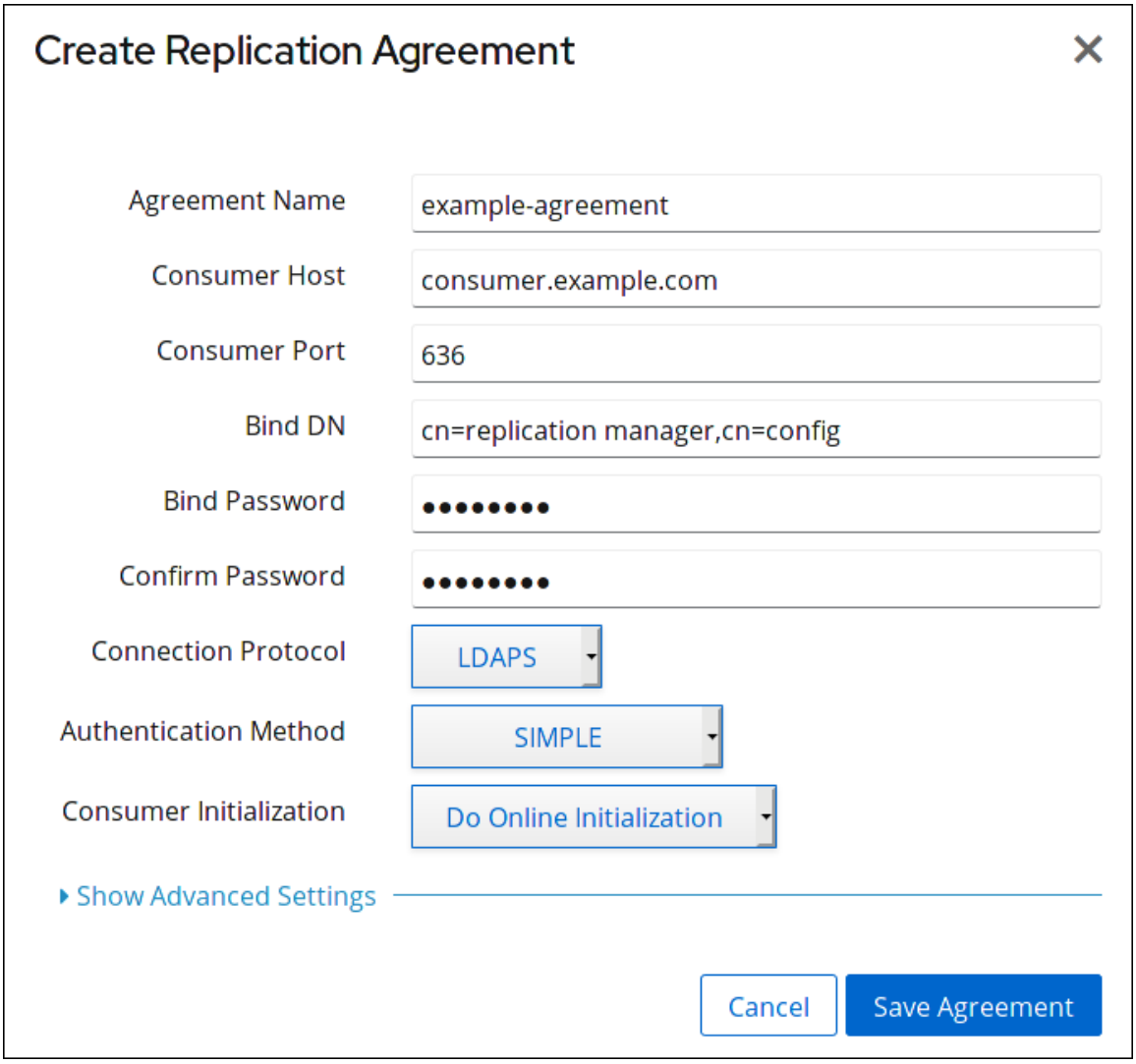
トポロジー内のすべてのサプライヤーの接尾辞については、レプリカ ID は **1** から **65534** の間の一意の整数である必要があります。

d. **Enable Replication** をクリックします。

4. レプリカ合意を追加し、コンシューマーを初期化します。

a. **Replication** メニューを開き、**dc=example,dc=com** 接尾辞を選択します。

b. **Replication Agreements** タブで **Create Agreement** をクリックし、フィールドに入力します。以下に例を示します。



この設定により、**example-agreement** という名前のレプリカ合意が作成されます。レプリカ合意は、コンシューマーへのデータの接続や複製時にサプライヤーが使用するコンシューマーのホスト名、プロトコル、認証情報などの設定を定義します。

c. **Consumer Initialization** フィールドで **Do Online Initialization** を選択し、合意の保存後にコンシューマーを自動的に初期化します。

後でコンシューマーを初期化する場合は、**Do Not Initialize** を選択します。コンシューマーを初期化する前にレプリケーションが起動しないことに注意してください。コンシューマーの初期化の詳細は、「コンシューマーの初期化」を参照してください。

d. **Save Agreement** をクリックします。

5. 初期化が成功したかどうかを確認します。

a. **Replication** メニューを開きます。

b. **Agreements** エントリーを選択します。

初期化が正常に完了したら、Web コンソールの **Last Update Status** 列に **Error (0) Replica acquired successfully: Incremental update succeeded** メッセージが表示されます。

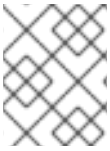
Last Update Status	Last Init Status
Error (0) Replica acquired successfully: Incremental update succeeded	*Initialized*

複製するデータ量によっては、初期化に時間がかかる場合があります。

トポロジーに、接尾辞の複数のコンシューマーを追加する場合は、各コンシューマーのサプライヤーで手順を繰り返します。ただし、接尾辞のレプリケーションは、サプライヤーで 1 回のみ有効にする必要があります。

15.3. マルチサプライヤーのレプリケーション

マルチサプライヤーレプリケーションシナリオでは、ディレクトリーデータのサプライヤーコピーが複数の読み取り/書き込みレプリカに保存されます。これらのサーバーはそれぞれ、読み取り/書き込みレプリカの changelog を維持します。Directory Server は、レプリケーショントポロジーでサプライヤーを最大 20 台サポートします。

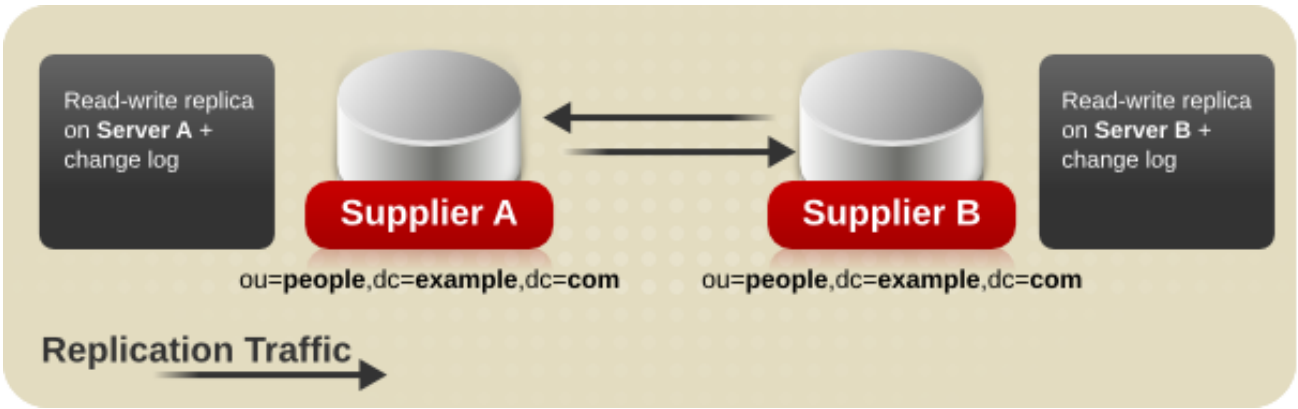


注記

マルチサプライヤーレプリケーション環境のそれぞれのサプライヤーは、自動的にコンシューマーです。

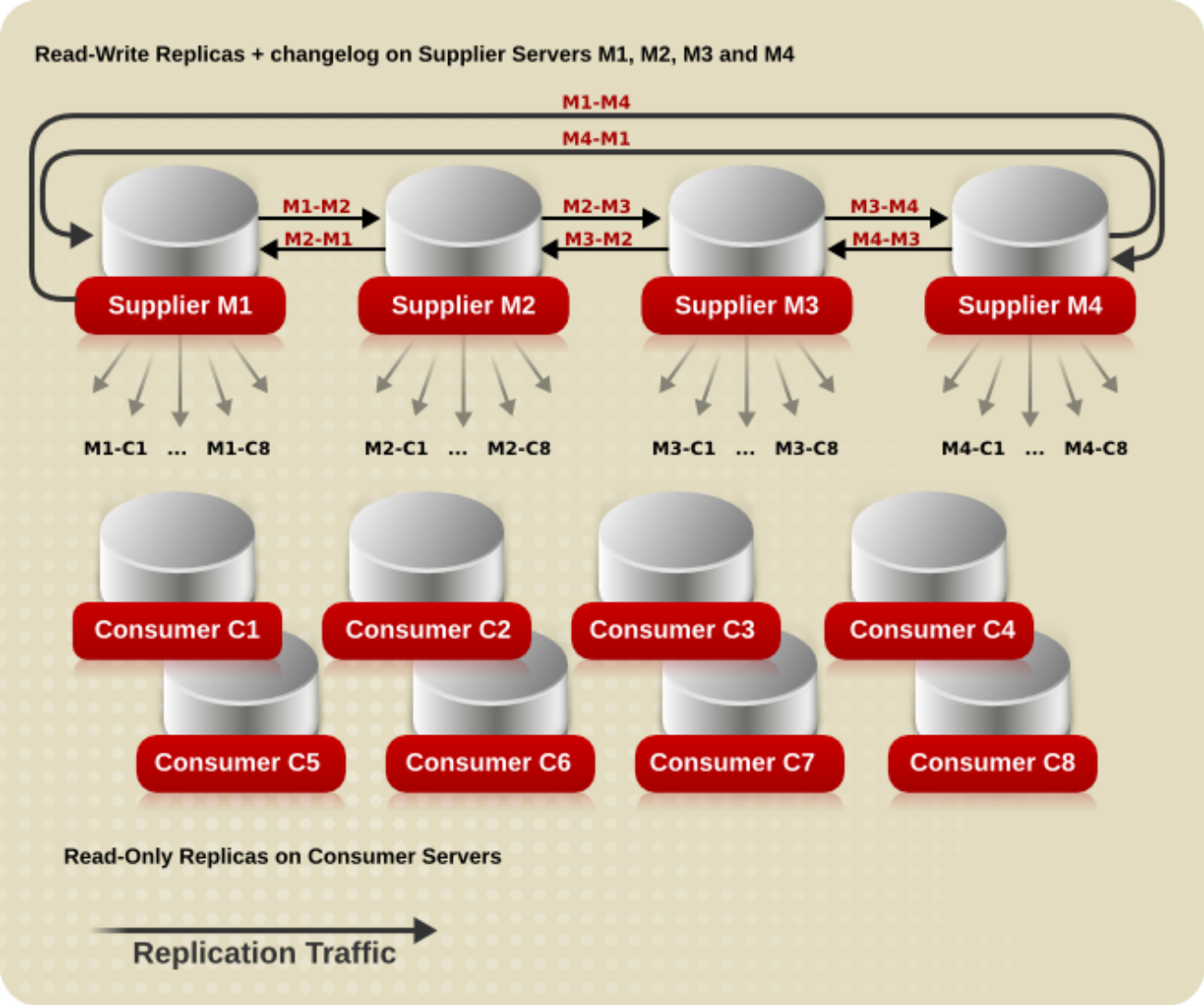
以下の図は、2 つのサプライヤーを持つマルチサプライヤーレプリケーション環境を示しています。

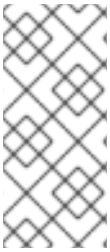
図15.2 2 つのサプライヤーを持つマルチサプライヤーレプリケーション



複雑な環境では、レプリケーショントポロジーには複数の読み取り/書き込みサプライヤーと読み取り専用コンシューマーが含まれることがよくあります。以下の図は、各サプライヤーが 2 つの他のサプライヤーと 8 つのコンシューマーにデータを複製するために 10 個のレプリカ合意で設定されたトポロジーを示しています。

図15.3 4 つのサプライヤーと 8 つのコンシューマーを持つ複雑なレプリケーションシナリオ





注記

レプリケーション速度は、以下に依存します。

- ネットワークの速度。
- 送信および受信のレプリカ合意の数。

コマンドラインまたは Web コンソールを使用して、マルチサプライヤーレプリケーショントポロジーを設定します。参照:

- 「コマンドラインを使用したマルチサプライヤーレプリケーションの設定」
- 「Web コンソールを使用したマルチサプライヤーレプリケーションの設定」

15.3.1. コマンドラインを使用したマルチサプライヤーレプリケーションの設定

以下の例では、既存の Directory Server インスタンスが **supplier1.example.com** という名前のホストで実行されていることを前提としています。以下の手順では、**supplier2.example.com** という名前の別の読み取り/書き込みレプリカをトポロジーに追加する方法と、**dc=example,dc=com** 接尾辞のマルチサプライヤーレプリケーションを設定する方法を説明します。

参加する新しいサーバーの準備

supplier2.example.com ホスト:

1. Directory Server をインストールして、インスタンスを作成します。詳細は、『Red Hat Directory Server インストールガイド』 を参照してください。

2. データベースなしでインスタンスを作成した場合には、接尾辞のデータベースを作成します。たとえば、**dc=example,dc=com** 接尾辞に **userRoot** という名前のデータベースを作成するには、以下のコマンドを実行します。

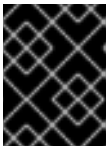
```
# dsconf -D "cn=Directory Manager" ldap://supplier2.example.com backend \
    create --suffix="dc=example,dc=com" --be-name="userRoot"
```

   接尾辞のデータベース作成に関する詳細は、「接尾辞の作成」を参照してください。

3. 接尾辞のレプリケーションを有効にし、レプリケーションマネージャーアカウントを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier2.example.com replication \
    enable --suffix="dc=example,dc=com" --role="supplier" --replica-id=1 \
    --bind-dn="cn=replication manager,cn=config" --bind-passwd="password"
```

   このコマンドは、**supplier2.example.com** ホストを **dc=example,dc=com** 接尾辞のサプライヤーとして設定し、このエントリーのレプリカ ID を **1** に設定します。さらに、サーバーは指定のパスワードで **cn=replication manager,cn=config** ユーザーを作成し、このアカウントがこのホストに接尾辞の変更を複製することができます。



重要

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

既存のサーバーをサプライヤーに設定

supplier1.example.com ホスト:

1. 参加する新しいサーバーで実行したコマンドと同様に、**dc=example,dc=com** 接尾辞のレプリケーションを有効にし、レプリケーションマネージャーアカウントを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier1.example.com replication \
    enable --suffix="dc=example,dc=com" --role="supplier" --replica-id=2 \
    --bind-dn="cn=replication manager,cn=config" --bind-passwd="password"
```

   レプリカ ID は、「参加する新しいサーバーの準備」で作成されるものとは異なるものでなければなりませんが、レプリケーションマネージャーアカウントは同じ DN を使用できます。

2. レプリカ合意を追加し、新しいサーバーを初期化します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier1.example.com repl-agmt \
    create --suffix="dc=example,dc=com" --host="supplier2.example.com" --port=636 \
    --conn-protocol=LDAPS --bind-dn="cn=replication manager,cn=config" \
    --bind-passwd="password" --bind-method=SIMPLE --init \
    example-agreement-supplier1-to-supplier2
```

このコマンドは、**example-agreement-supplier1-to-supplier2** という名前のレプリカ合意を作成します。レプリカ合意は、コンシューマーへのデータの接続や複製時にサプライヤーが使用するコンシューマーのホスト名、プロトコル、認証情報などの設定を定義します。

この合意の作成後、Directory Server はコンシューマーを初期化します。後ほどコンシューマーを初期化する場合は、**--init** オプションを省略します。コンシューマーを初期化する前にレプリケーションが起動しないことに注意してください。コンシューマーの初期化の詳細は、「コンシューマーの初期化」を参照してください。

コマンドで使用するオプションの詳細は、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier1.example.com repl-agmt --help
```

3. 初期化が成功したかどうかを確認します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier1.example.com repl-agmt \
    init-status --suffix="dc=example,dc=com" example-agreement-supplier1-to-supplier2
Agreement successfully initialized.
```

複製するデータ量によっては、初期化に時間がかかる場合があります。

新しいサーバーをサプライヤーに設定
supplier2.example.com ホスト:

警告

「既存のサーバーをサプライヤーに設定」で説明されているように、既存のサーバーで接尾辞 dc=example,dc=com を初期化していない場合は、続行しないでください。それ以外の場合は、新規サーバーの空のデータベースは、既存のサプライヤーのデータベースを上書きします。

- **supplier 2** から **supplier 1** に情報を複製する複製合意を追加します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier2.example.com repl-agmt \
    create --suffix="dc=example,dc=com" --host="supplier1.example.com" --port=636 \
    --conn-protocol=LDAPS --bind-dn="cn=replication manager,cn=config" \
    --bind-passwd="password" --bind-method=SIMPLE \
    example-agreement-supplier2-to-supplier1
```

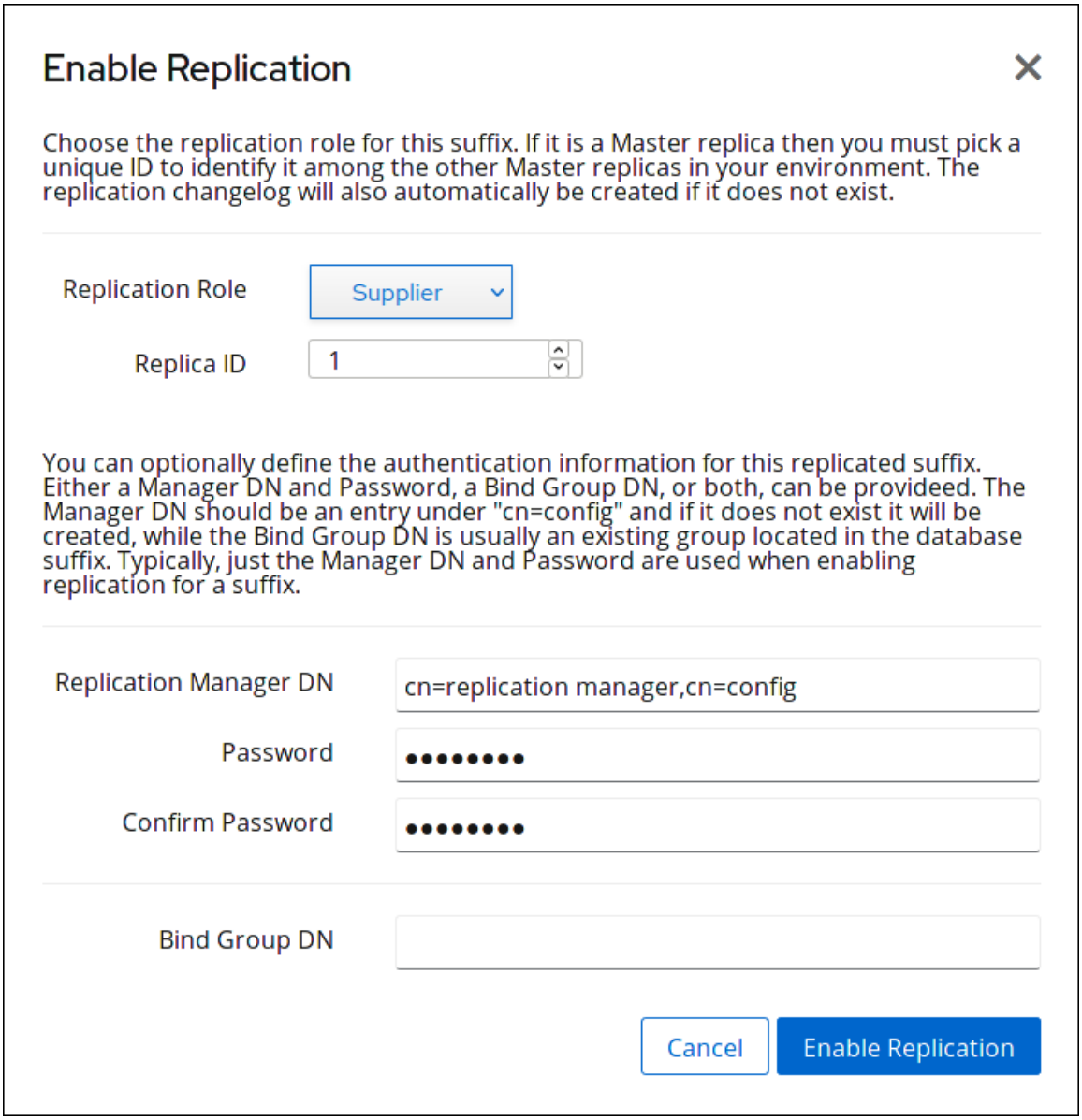
このコマンドは、**example-agreement-supplier2-to-supplier1** という名前のレプリカ合意を作成します。レプリカ合意は、コンシューマーへのデータの接続や複製時にサプライヤーが使用するコンシューマーのホスト名、プロトコル、認証情報などの設定を定義します。

15.3.2. Web コンソールを使用したマルチサプライヤーレプリケーションの設定

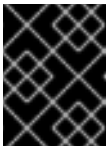
以下の例では、既存の Directory Server インスタンスが **supplier1.example.com** という名前のホストで実行されていることを前提としています。以下の手順では、**supplier2.example.com** という名前の別の読み取り/書き込みレプリカをトポロジーに追加する方法と、**dc=example,dc=com** 接尾辞のマルチサプライヤーレプリケーションを設定する方法を説明します。

参加する新しいサーバーの準備
supplier2.example.com ホスト:

1. Directory Server をインストールして、インスタンスを作成します。詳細は、『Red Hat Directory Server インストールガイド』 を参照してください。
2. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
3. インスタンスを選択します。
4. データベースなしでインスタンスを作成した場合は、接尾辞からデータベースを作成します。接尾辞のデータベース作成に関する詳細は、「接尾辞の作成」を参照してください。
5. 接尾辞のレプリケーションを有効にします。
    a. **Replication** メニューを開きます。
    b. **dc=example,dc=com** 接尾辞を選択し、**Enable Replication** をクリックします。
    c. **Replication Role** フィールドで **Supplier** を選択し、レプリカ ID を入力し、作成するレプリケーションマネージャーアカウントの DN およびパスワードを入力します。以下に例を示します。



この設定により、**supplier2.example.com** ホストを **dc=example,dc=com** 接尾辞のサプライヤーとして設定し、このエントリーのレプリカ ID を **1** に設定します。さらに、サーバーは指定のパスワードで **cn=replication manager,cn=config** ユーザーを作成し、このアカウントがこのホストに接尾辞の変更を複製することができます。



重要

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

d. **Enable Replication** をクリックします。

既存のサーバーをサプライヤーに設定
supplier1.example.com ホスト:

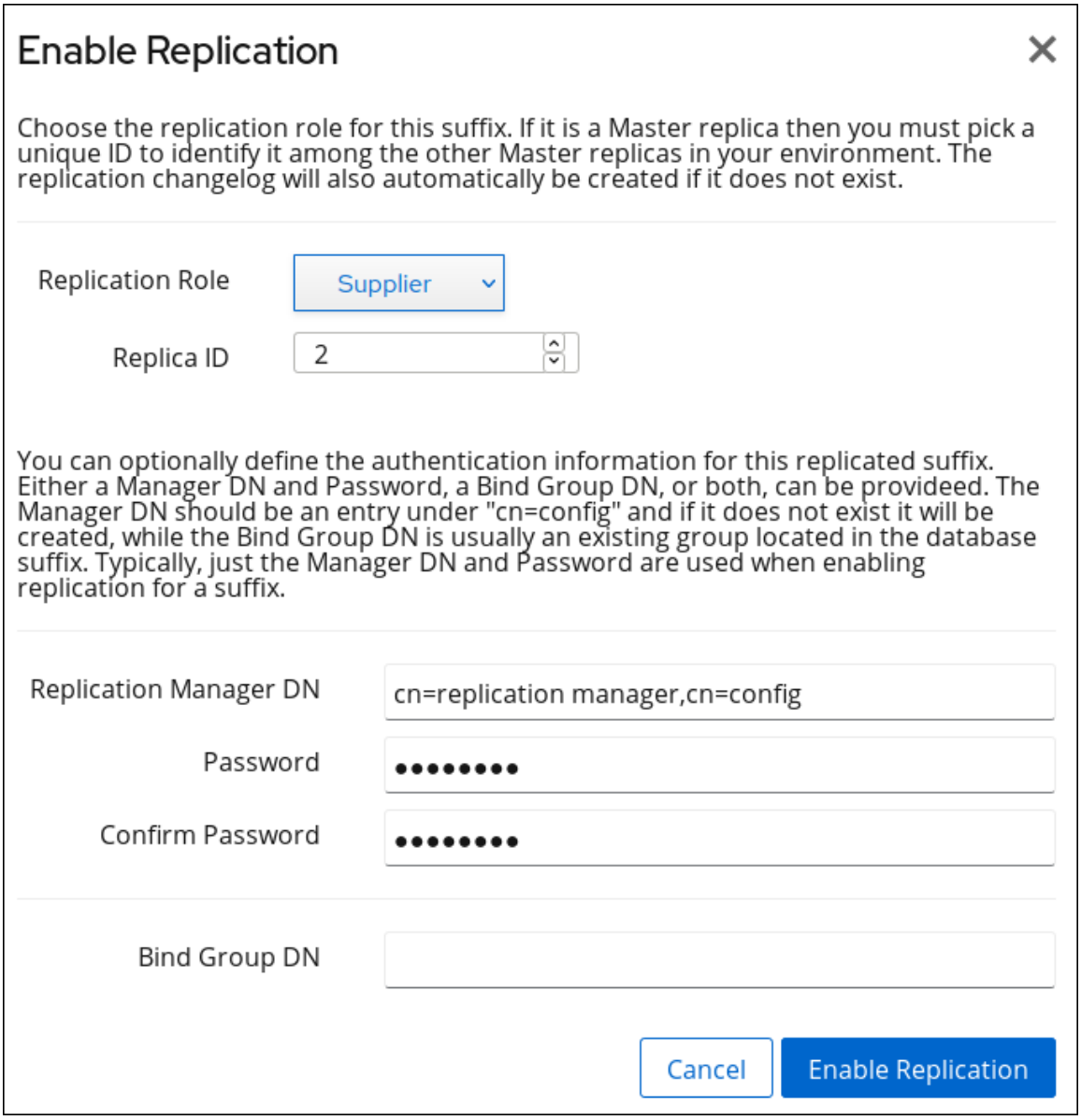
1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。

3. 参加する新規サーバーの設定と同様に、**dc=example,dc=com** 接尾辞のレプリケーションを有効にし、レプリケーションコントローラーアカウントを作成します。

   a. **Replication** メニューを開きます。

   b. **dc=example,dc=com** 接尾辞を選択し、**Enable Replication** をクリックします。

   c. **Replication Role** フィールドで **Supplier** を選択し、レプリカ ID を入力し、作成するレプリケーションマネージャーアカウントの DN およびパスワードを入力します。以下に例を示します。



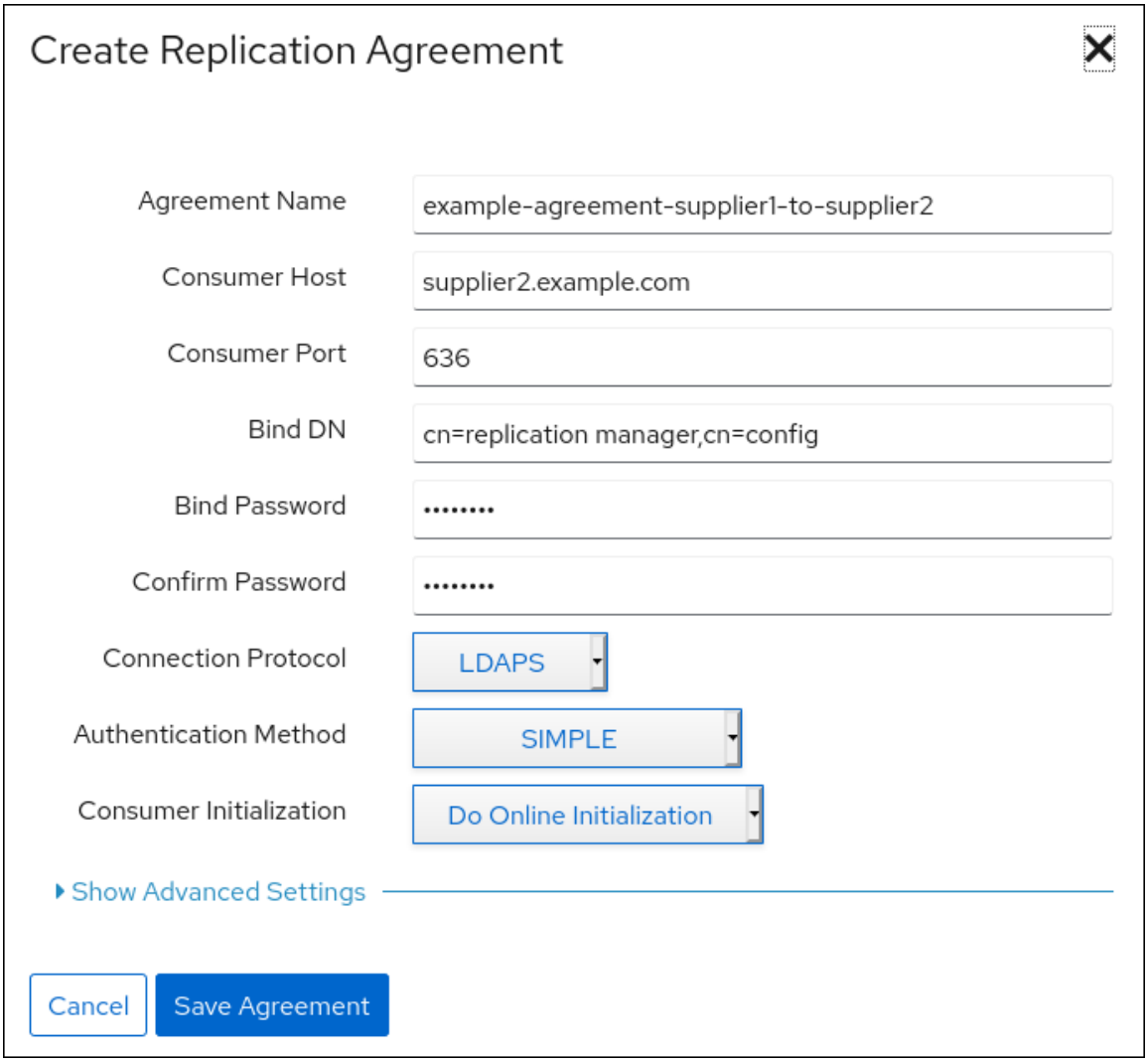
   レプリカ ID は、「参加する新しいサーバーの準備」で作成されるものとは異なるものでなければなりませんが、レプリケーションマネージャーアカウントは同じ DN を使用できます。

   d. **Enable Replication** をクリックします。

4. レプリカ合意を追加し、コンシューマーを初期化します。

   a. **Replication** メニューを開き、**Agreements** エントリーを選択します。

b. **Create Replication Agreement** をクリックし、フィールドに入力します。以下に例を示します。



この設定により、**example-agreement-supplier1-to-supplier2** という名前のレプリカ合意が作成されます。レプリカ合意は、コンシューマーへのデータの接続や複製時にサプライヤーが使用するコンシューマーのホスト名、プロトコル、認証情報などの設定を定義します。

c. **Consumer Initialization** フィールドで **Do Online Initialization** を選択し、合意の保存後にコンシューマーを自動的に初期化します。

後でコンシューマーを初期化する場合は、**Do Not Initialize** を選択します。コンシューマーを初期化する前にレプリケーションが起動しないことに注意してください。コンシューマーの初期化の詳細は、「コンシューマーの初期化」を参照してください。

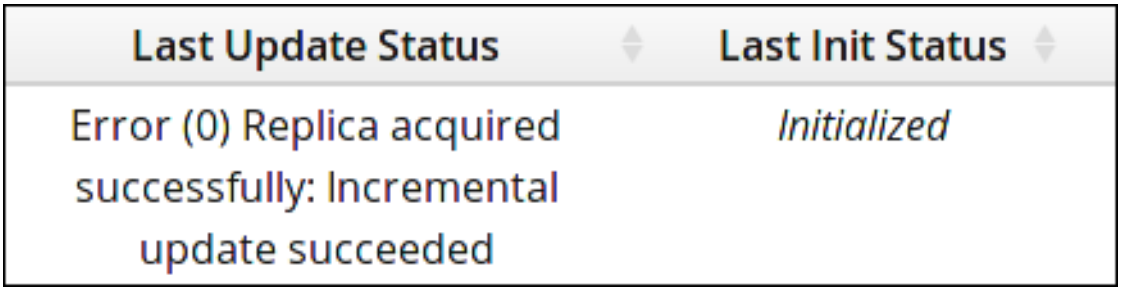
d. **Save Agreement** をクリックします。

5. 初期化が成功したかどうかを確認します。

a. **Replication** メニューを開きます。

b. **Agreements** エントリーを選択します。

初期化が正常に完了したら、Web コンソールの **Last Update Status** 列に **Error (0) Replica acquired successfully: Incremental update succeeded** メッセージが表示されます。



Last Update Status	Last Init Status
Error (0) Replica acquired successfully: Incremental update succeeded	*Initialized*

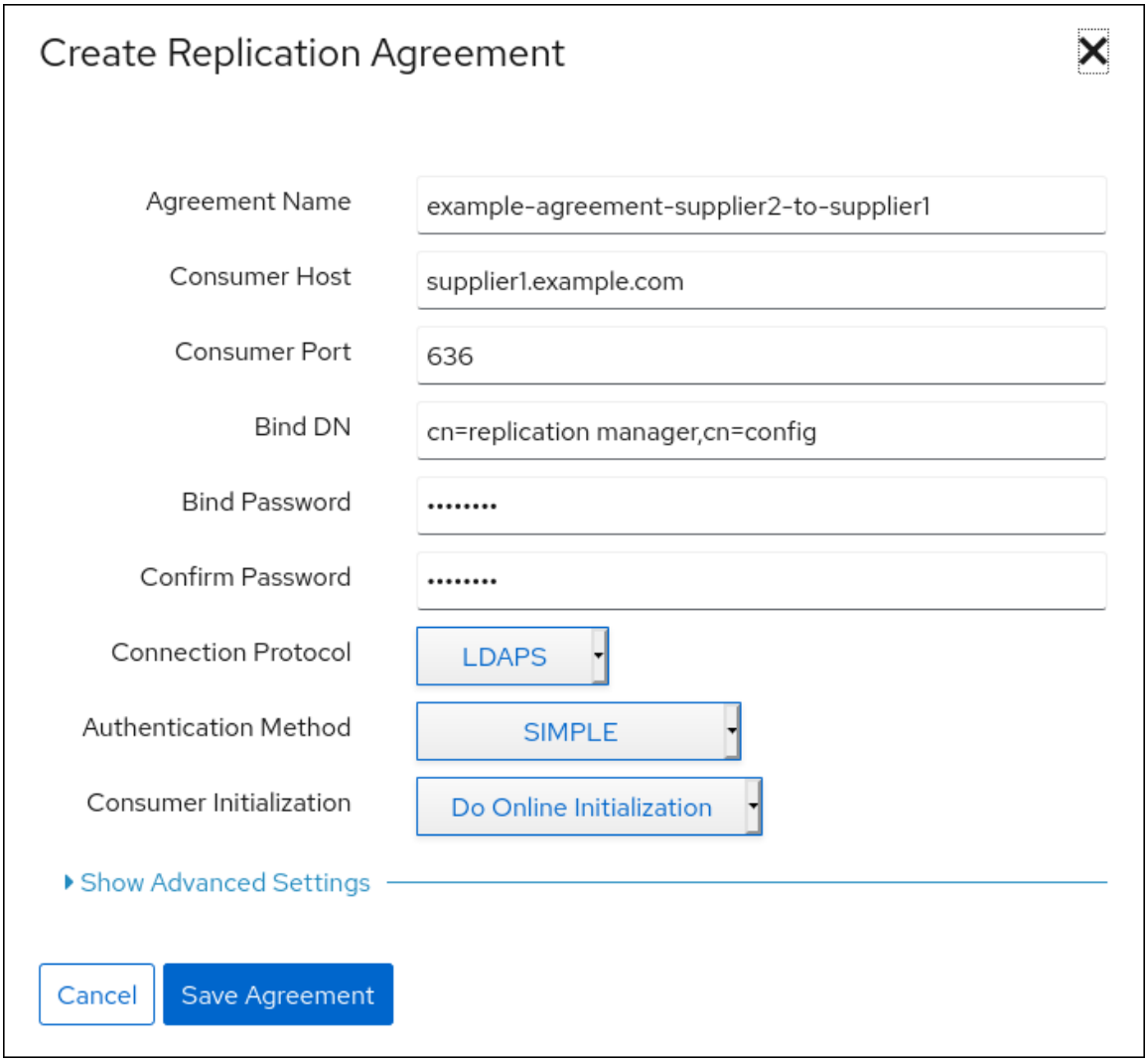
複製するデータ量によっては、初期化に時間がかかる場合があります。

新しいサーバーをサプライヤーに設定
supplier2.example.com ホスト:

警告

「既存のサーバーをサプライヤーに設定」 で説明されているように、既存のサーバーでレプリカ合意を初期化していない場合は、続行しないでください。それ以外の場合は、新規サーバーの空のデータベースは、既存のサプライヤーのデータベースを上書きします。

1. レプリカ合意を追加し、コンシューマーを初期化します。
   a. **Replication** メニューを開き、**Agreements** エントリーを選択します。
   b. **Create Replication Agreement** をクリックし、フィールドに入力します。以下に例を示します。



この設定により、**example-agreement-supplier2-to-supplier1** という名前のレプリカ合意が作成されます。

c. **Consumer Initialization** フィールドで **Do Online Initialization** を選択し、合意の保存後にコンシューマーを自動的に初期化します。

後でコンシューマーを初期化する場合は、**Do Not Initialize** を選択します。コンシューマーを初期化する前にレプリケーションが起動しないことに注意してください。コンシューマーの初期化の詳細は、「コンシューマーの初期化」を参照してください。

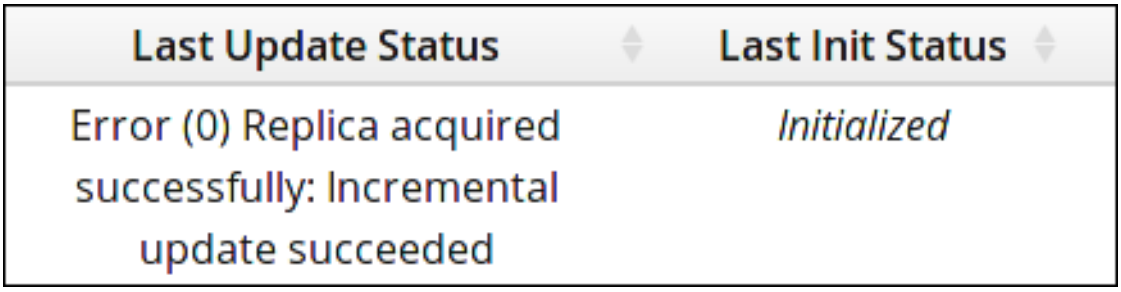
d. **Save Agreement** をクリックします。

2. 初期化が成功したかどうかを確認します。

a. **Replication** メニューを開きます。

b. **Agreements** エントリーを選択します。

初期化が正常に完了すると、Web コンソールは **Last Update Status** 列に **Error (0) Replica acquired successfully: Incremental update succeeded** メッセージを表示します。



複製するデータ量によっては、初期化に時間がかかる場合があります。

15.3.3. マルチサプライヤーレプリケーションにおけるコンシューマーの独占を防ぐ

マルチサプライヤーレプリケーションの機能の 1 つは、サプライヤーが複製されたエリアのコンシューマーへの排他的アクセスを取得することです。この間、他のサプライヤーは、コンシューマーによる直接通信がロックされます。ロックアウトされた状態でサプライヤーがアクセス権を取得しようとすると、コンシューマーはビジー応答を返し、サプライヤーは数秒間スリープしてから再度アクセスを試みます。更新負荷が低い間に、最初のコンシューマーがロックされたときにサプライヤーが別のコンシューマーに更新を送信し、最初のコンシューマーが再び解放されると更新を送信します。

ロックサプライヤーの更新負荷が高かったり、changelog に多くの保留中の更新があったりすると、問題が発生することがあります。ロックサプライヤーが更新の送信を終了し、送信する保留中の変更が複数ある場合は、直ちにコンシューマーの再取得を試みます。他のサプライヤーは通常スリープ状態であるため、ほとんどの場合、このような試みは成功します。これにより、単一のサプライヤーが数時間またはそれ以上にわたってコンシューマーを独占することになります。

以下の属性は、この問題に対応します。

nsds5ReplicaBusyWaitTime

別のアクセスの取得を試みる前に、コンシューマーがビジー応答を返した後、サプライヤーが待機する時間を秒単位で設定します。

たとえば、別の取得を試みる前に、サプライヤーが **5** 秒待機するように設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt \
    set --suffix="suffix" --busy-wait-time=5 agreement_name
```

nsds5ReplicaSessionPauseTime

2 つの更新セッションの間にサプライヤーが待機する時間を秒単位で設定します。***nsds5ReplicaBusyWaitTime*** で指定した値またはそれよりも小さい値を設定すると、Directory Server は、***nsds5ReplicaBusyWaitTime*** で設定した値よりも大きな値である ***nsds5ReplicaSessionPauseTime*** パラメーターの値を自動的に使用します。

たとえば、サプライヤーは 2 つの更新セッション間で **10** 秒待機するように設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt \
    set --suffix="suffix" --session-pause-time=10 agreement_name
```

nsds5ReplicaReleaseTimeout

更新の送信を終了したかどうか、サプライヤーがレプリカのリリース後のタイムアウトを設定します。これにより、単一サプライヤーがレプリカを独占しなくなります。

たとえば、サプライヤーが大きいレプリケーション環境で **90** 秒後にレプリカを解放するように設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com replication \
    set --suffix="suffix" --repl-release-timeout=90
```

詳細は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 のパラメーターの説明を参照してください。

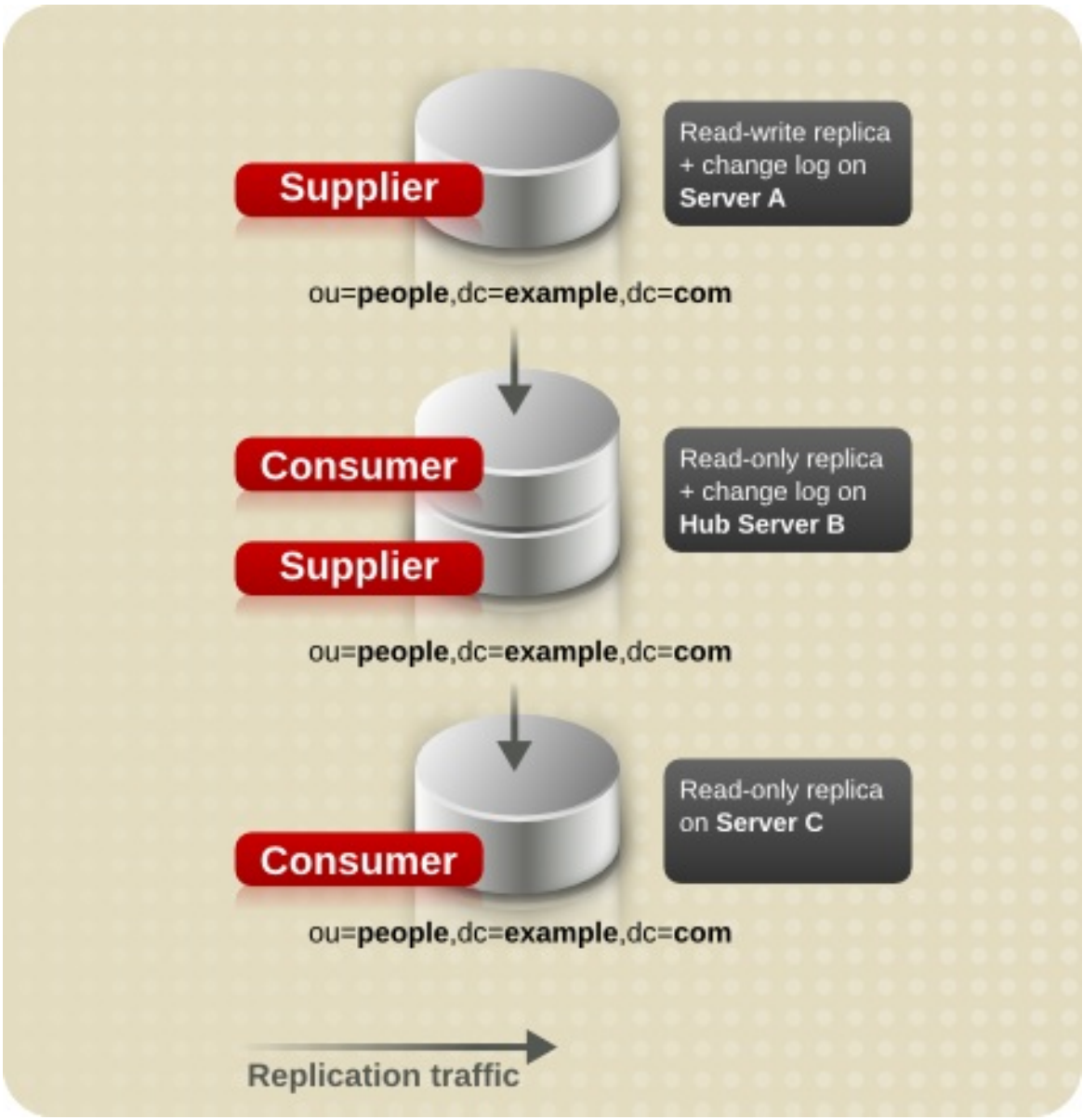
レプリカのビジーエラーをログに記録するには、**Replication** エラーログ (ログレベル **8192**) を有効にします。「ログレベルの設定」を参照してください。

15.4. カスケードレプリケーション

カスケードレプリケーションのシナリオでは、**ハブ** となる 1 台のサーバーがコンシューマーとサプライヤーの両方のロールを果たします。読み取り専用のレプリカを保持し、changelog を維持するため、データのサプライヤーコピーを保持するサプライヤーサーバーから更新を受け取って、その更新をコンシューマーに提供します。カスケードレプリケーションは、負荷の高いトラフィックのバランスをとる場合や、地理的に分散した環境でサプライヤーサーバーをローカルに配置する場合に使用します。

次の図は、シンプルなカスケードレプリケーションのシナリオを示しています。

図15.4 カスケードレプリケーション



注記

マルチサプライヤーとカスケードレプリケーションを組み合わせることができます。

コマンドラインまたは Web コンソールを使用して、カスケードレプリケーショントポロジーを設定します。参照:

- 「コマンドラインによるカスケードレプリケーションの設定」
- 「Web コンソールによるカスケードレプリケーションの設定」

15.4.1. コマンドラインによるカスケードレプリケーションの設定

以下の例では、既存の Directory Server インスタンスが **supplier.example.com** という名前のホストで実行されていることを前提としています。以下の手順では、**dc=example,dc=com** 接尾辞用にサプライヤーから更新を受信するトポロジーに **hub.example.com** という名前のハブを追加する方法を説明します。その後、接尾辞のハブサーバーから更新を受信する **consumer.example.com** という名前のコンシューマーを追加する方法を説明します。

参加する新しいハブサーバーの準備

hub.example.com ホスト:

1. Directory Server をインストールしてインスタンスを作成します。詳細は、『Red Hat Directory Server インストールガイド』 を参照してください。

2. データベースなしでインスタンスを作成した場合には、接尾辞のデータベースを作成します。たとえば、**dc=example,dc=com** 接尾辞に **userRoot** という名前のデータベースを作成するには、以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://hub.example.com backend \
    create --suffix="dc=example,dc=com" --be-name="userRoot"
```

   接尾辞のデータベース作成に関する詳細は、「接尾辞の作成」を参照してください。

3. 接尾辞のレプリケーションを有効にし、レプリケーションマネージャーアカウントを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://hub.example.com replication \
    enable --suffix="dc=example,dc=com" --role="hub" \
    --bind-dn="cn=replication manager,cn=config" --bind-passwd="password"
```

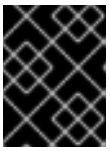
   このコマンドは、**dc=example,dc=com** 接尾辞のハブとして **hub.example.com** ホストを設定します。さらに、サーバーは指定のパスワードで **cn=replication manager,cn=config** ユーザーを作成し、このアカウントがこのホストに接尾辞の変更を複製することができます。

既存のサーバーをサプライヤーに設定

supplier.example.com ホスト:

1. 「参加する新しいハブサーバーの準備」に参加する新しいハブサーバーで実行したコマンドと同様に、**dc=example,dc=com** 接尾辞のレプリケーションを有効にして、レプリケーションマネージャーアカウントを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier1.example.com replication \
    enable --suffix="dc=example,dc=com" --role="supplier" --replica-id=1 \
    --bind-dn="cn=replication manager,cn=config" --bind-passwd="password"
```



重要

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

   レプリケーションマネージャーアカウントは、ハブで作成したものと同じ DN を使用できます。

2. レプリカ合意を追加し、ハブを初期化します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt \
    create --suffix="dc=example,dc=com" --host="hub.example.com" --port=636 \
    --conn-protocol=LDAPS --bind-dn="cn=replication manager,cn=config" \
    --bind-passwd="password" --bind-method=SIMPLE --init \
    example-agreement-supplier-to-hub
```

このコマンドは、**example-agreement-supplier-to-hub** という名前のレプリカ合意を作成します。レプリカ合意は、ハブへのデータの接続時や複製時にサプライヤーが使用するハブのホスト名、プロトコル、認証情報などの設定を定義します。

この合意の作成後、Directory Server はハブを初期化します。後でハブを初期化するには、**--init** オプションを省略します。コンシューマーを初期化する前にレプリケーションが起動しないことに注意してください。コンシューマーの初期化の詳細は、「コンシューマーの初期化」を参照してください。

コマンドで使用するオプションの詳細は、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt --help
```

3. 初期化が成功したかどうかを確認します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt \
    init-status --suffix="dc=example,dc=com" example-agreement-supplier-to-hub
Agreement successfully initialized.
```

複製するデータ量によっては、初期化に時間がかかる場合があります。

参加させる新規コンシューマーの準備

consumer.example.com ホスト:

1. Directory Server をインストールしてインスタンスを作成します。詳細は、『Red Hat Directory Server インストールガイド』 を参照してください。
2. データベースなしでインスタンスを作成した場合には、接尾辞のデータベースを作成します。たとえば、**dc=example,dc=com** 接尾辞に **userRoot** という名前のデータベースを作成するには、以下のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://hub.example.com backend \
    create --suffix="dc=example,dc=com" --be-name="userRoot"
```

接尾辞のデータベース作成に関する詳細は、「接尾辞の作成」を参照してください。

3. 接尾辞のレプリケーションを有効にし、レプリケーションマネージャーアカウントを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://consumer.example.com replication \
    enable --suffix="dc=example,dc=com" --role="consumer" \
    --bind-dn="cn=replication manager,cn=config" --bind-passwd="password"
```

このコマンドは、**consumer.example.com** ホストを **dc=example,dc=com** 接尾辞のコンシューマーとして設定します。さらに、サーバーは指定のパスワードで **cn=replication manager,cn=config** ユーザーを作成し、このアカウントがこのホストに接尾辞の変更を複製することができます。

ハブをコンシューマーのサプライヤーとして設定

hub.example.com ホスト:

1. レプリカ合意を追加し、サーバーを初期化します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://hub.example.com repl-agmt \
    create --suffix="dc=example,dc=com" --host="consumer.example.com" --port=636 \
```

```
    --conn-protocol=LDAPS --bind-dn="cn=replication manager,cn=config" \
    --bind-passwd="password" --bind-method=SIMPLE --init \
    example-agreement-hub-to-consumer
```

この合意の作成後、Directory Server はコンシューマーを初期化します。後ほどコンシューマーを初期化する場合は、**--init** オプションを省略します。

2. 初期化が成功したかどうかを確認します。

```
# dsconf -D "cn=Directory Manager" ldap://hub.example.com repl-agmt \
    init-status --suffix="dc=example,dc=com" example-agreement-hub-to-consumer
Agreement successfully initialized.
```

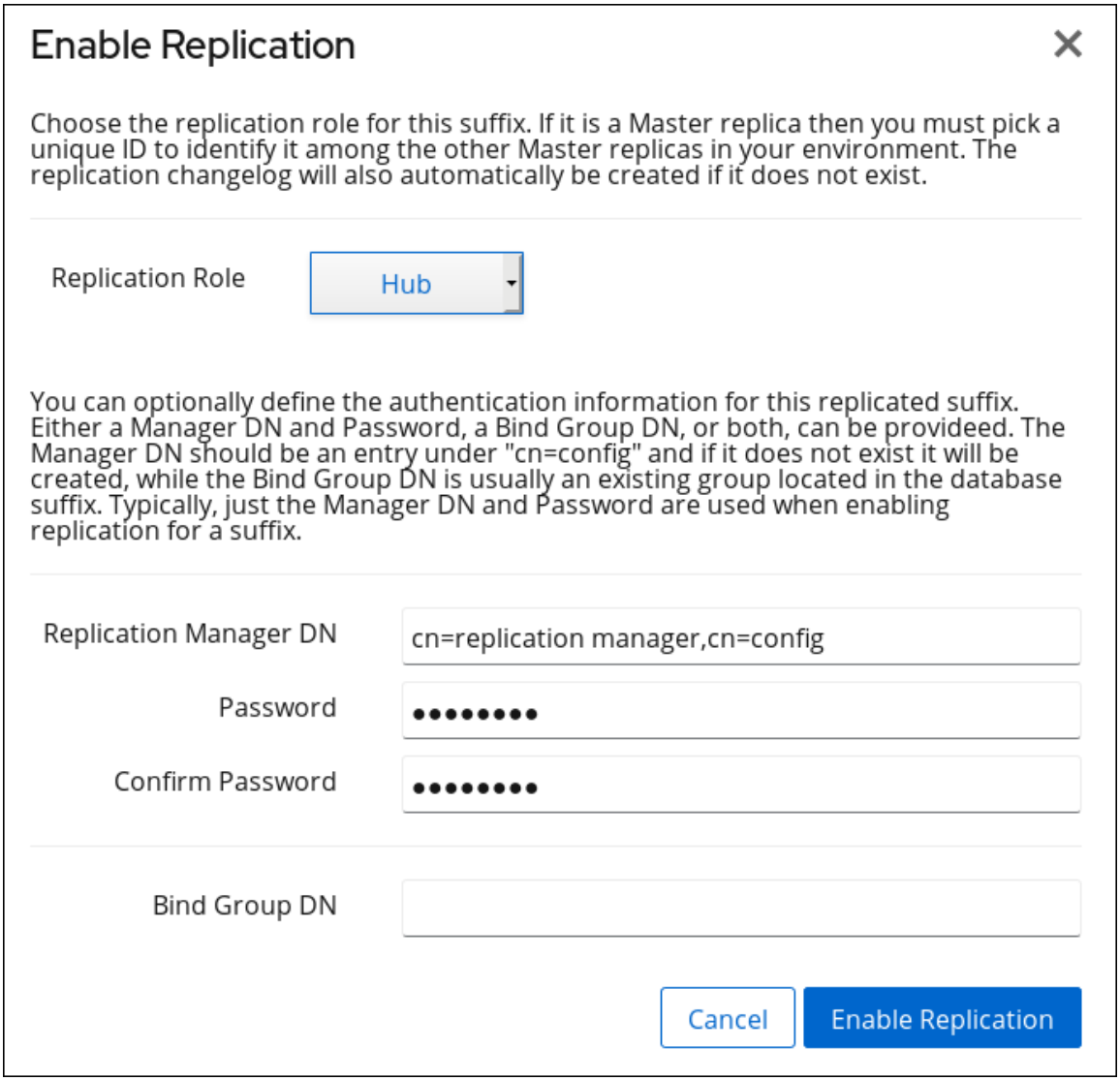
複製するデータ量によっては、初期化に時間がかかる場合があります。

15.4.2. Web コンソールによるカスケードレプリケーションの設定

以下の例では、既存の Directory Server インスタンスが **supplier.example.com** という名前のホストで実行されていることを前提としています。以下の手順では、**dc=example,dc=com** 接尾辞用にサプライヤーから更新を受信するトポロジーに **hub.example.com** という名前のハブを追加する方法を説明します。その後、接尾辞のハブサーバーから更新を受信する **consumer.example.com** という名前のコンシューマーを追加する方法を説明します。

参加する新しいハブサーバーの準備
hub.example.com ホスト:

1. Directory Server をインストールしてインスタンスを作成します。詳細は、『Red Hat Directory Server インストールガイド』 を参照してください。
2. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
3. インスタンスを選択します。
4. データベースなしでインスタンスを作成した場合には、接尾辞のデータベースを作成します。接尾辞のデータベース作成に関する詳細は、「接尾辞の作成」を参照してください。
5. 接尾辞のレプリケーションを有効にします。
    a. **Replication** メニューを開きます。
    b. **dc=example,dc=com** 接尾辞を選択し、**Enable Replication** をクリックします。
    c. **Replication Role** フィールドで **Hub** を選択し、作成するレプリケーションマネージャーアカウントの DN およびパスワードを入力します。以下に例を示します。



この設定により、**hub.example.com** ホストを **dc=example,dc=com** 接尾辞のハブとして設定します。さらに、サーバーは指定のパスワードで **cn=replication manager,cn=config** ユーザーを作成し、このアカウントがこのホストに接尾辞の変更を複製することができます。

d. **Enable Replication** をクリックします。

既存のサーバーをサプライヤーに設定
supplier.example.com ホスト:

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。

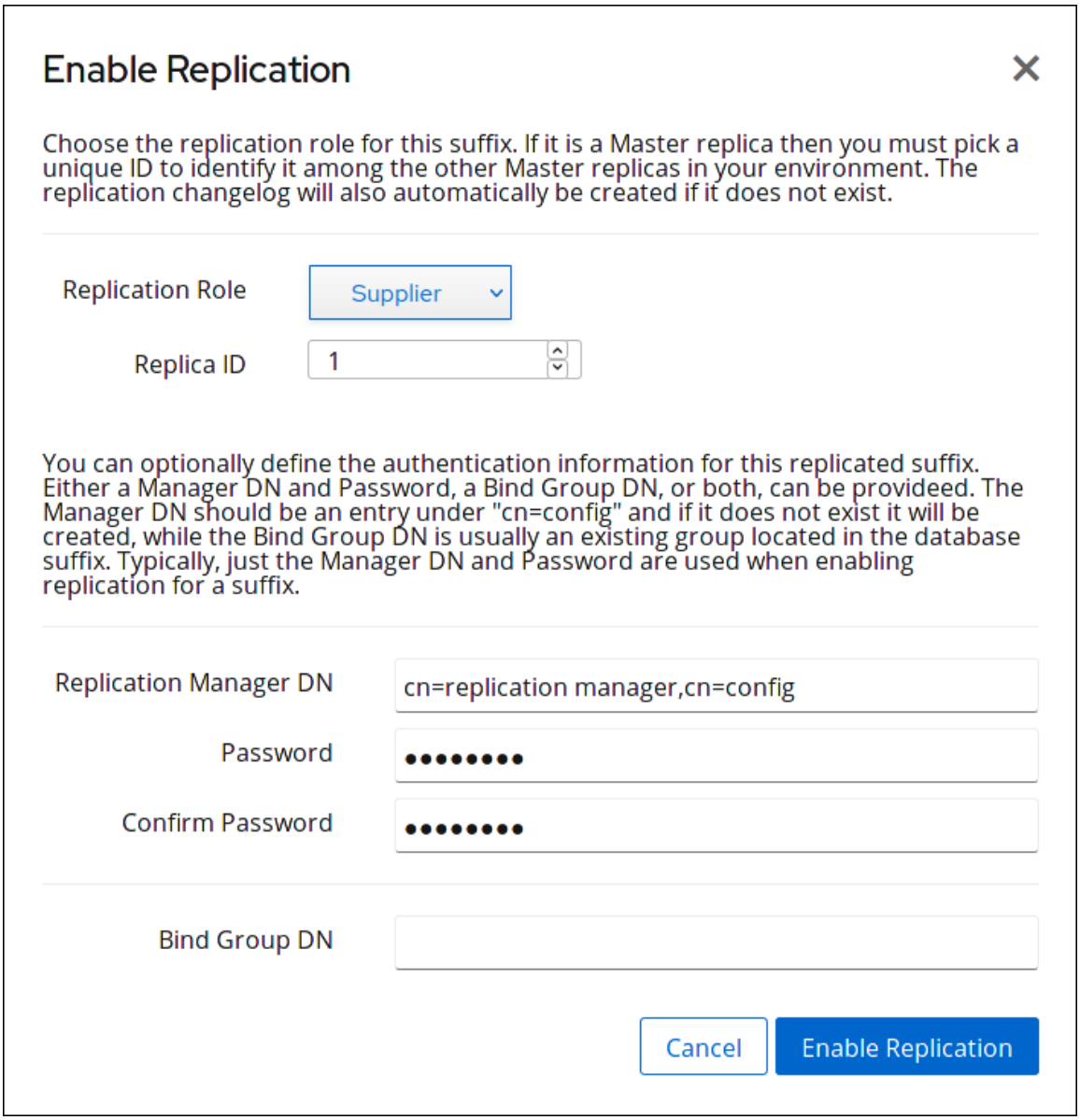
2. インスタンスを選択します。

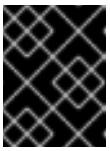
3. 「参加する新しいハブサーバーの準備」に参加する新しいハブサーバーの設定と同様、**dc=example,dc=com** 接尾辞のレプリケーションを有効にして、レプリケーションマネージャーアカウントを作成します。

   a. **Replication** メニューを開きます。

   b. **dc=example,dc=com** 接尾辞を選択し、**Enable Replication** をクリックします。

c. **Replication Role** フィールドで **Supplier** を選択し、レプリカ ID を入力し、作成するレプリケーションマネージャーアカウントの DN およびパスワードを入力します。以下に例を示します。





重要

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

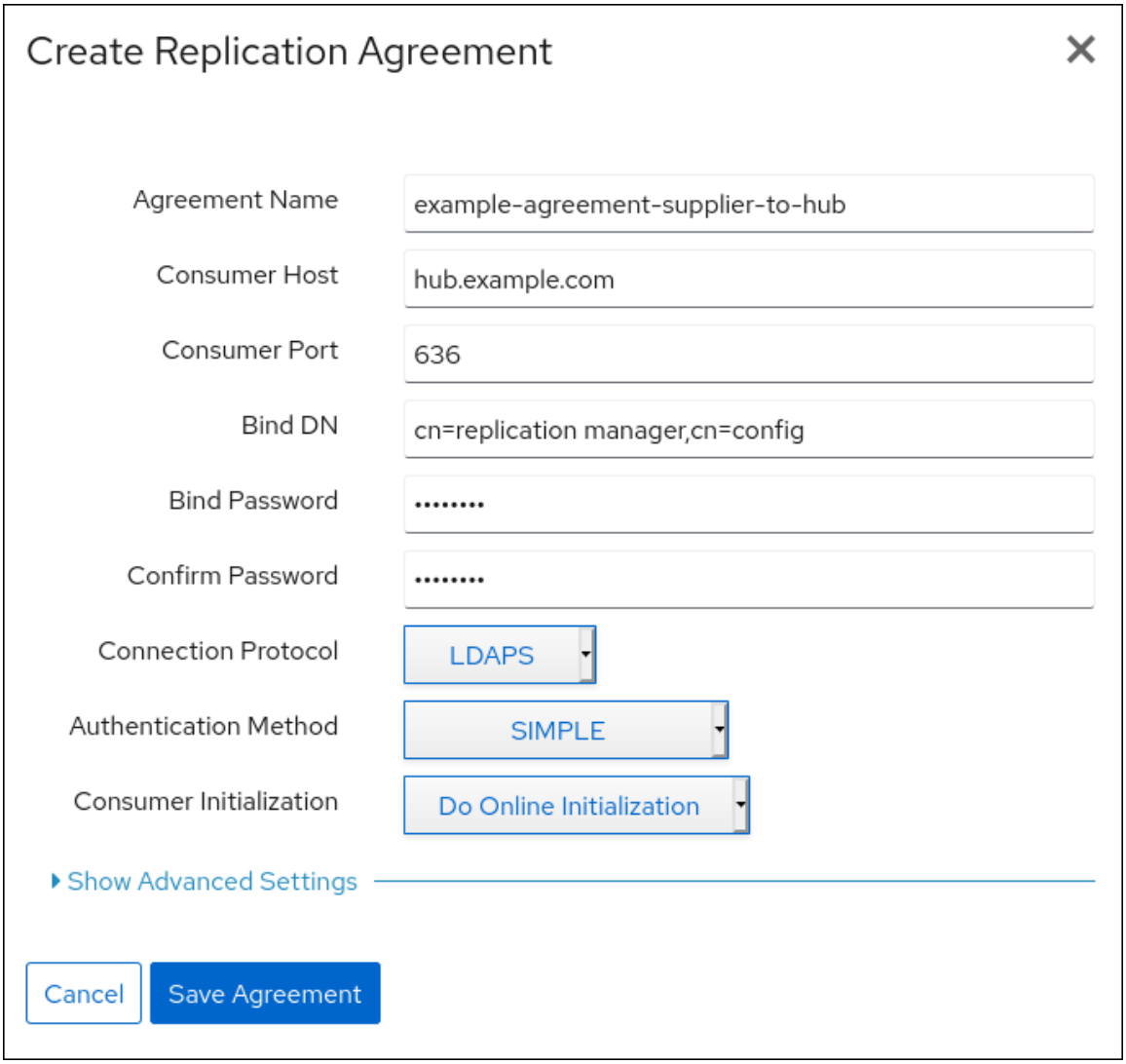
レプリケーションマネージャーアカウントは、ハブで作成したものと同じ DN を使用できます。

d. **Enable Replication** をクリックします。

4. レプリカ合意を追加し、ハブを初期化します。

a. **Replication** メニューを開き、接尾辞を選択します。

b. **Replication Agreements** タブで **Create Agreement** をクリックし、フィールドに入力します。以下に例を示します。



この設定により、**example-agreement-supplier-to-hub** という名前のレプリカ合意が作成されます。レプリカ合意は、ハブへのデータの接続時や複製時にサプライヤーが使用するハブのホスト名、プロトコル、認証情報などの設定を定義します。

c. **Consumer Initialization** フィールドで **Do Online Initialization** を選択し、合意の保存後にコンシューマーを自動的に初期化します。

後でハブを初期化するには、**Do Not Initialize** を選択します。コンシューマーを初期化する前にレプリケーションが起動しないことに注意してください。コンシューマーの初期化の詳細は、「コンシューマーの初期化」を参照してください。

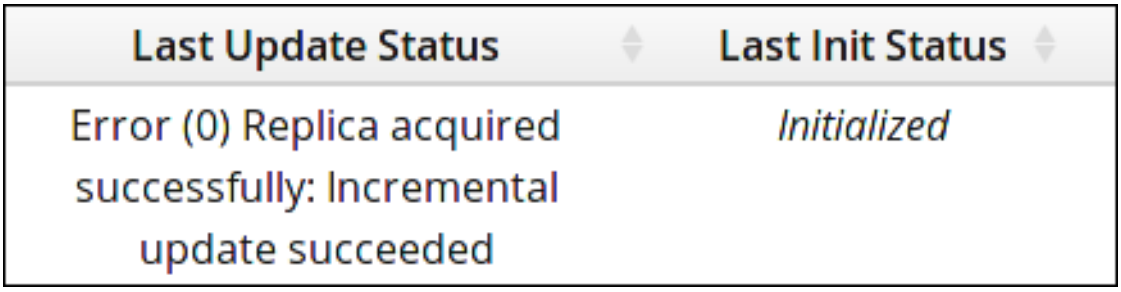
d. **Save Agreement** をクリックします。

5. 初期化が成功したかどうかを確認します。

a. **Replication** メニューを開きます。

b. **Agreements** エントリーを選択します。

初期化が正常に完了すると、Web コンソールは **Last Update Status** 列に **Error (0) Replica acquired successfully: Incremental update succeeded** メッセージを表示します。

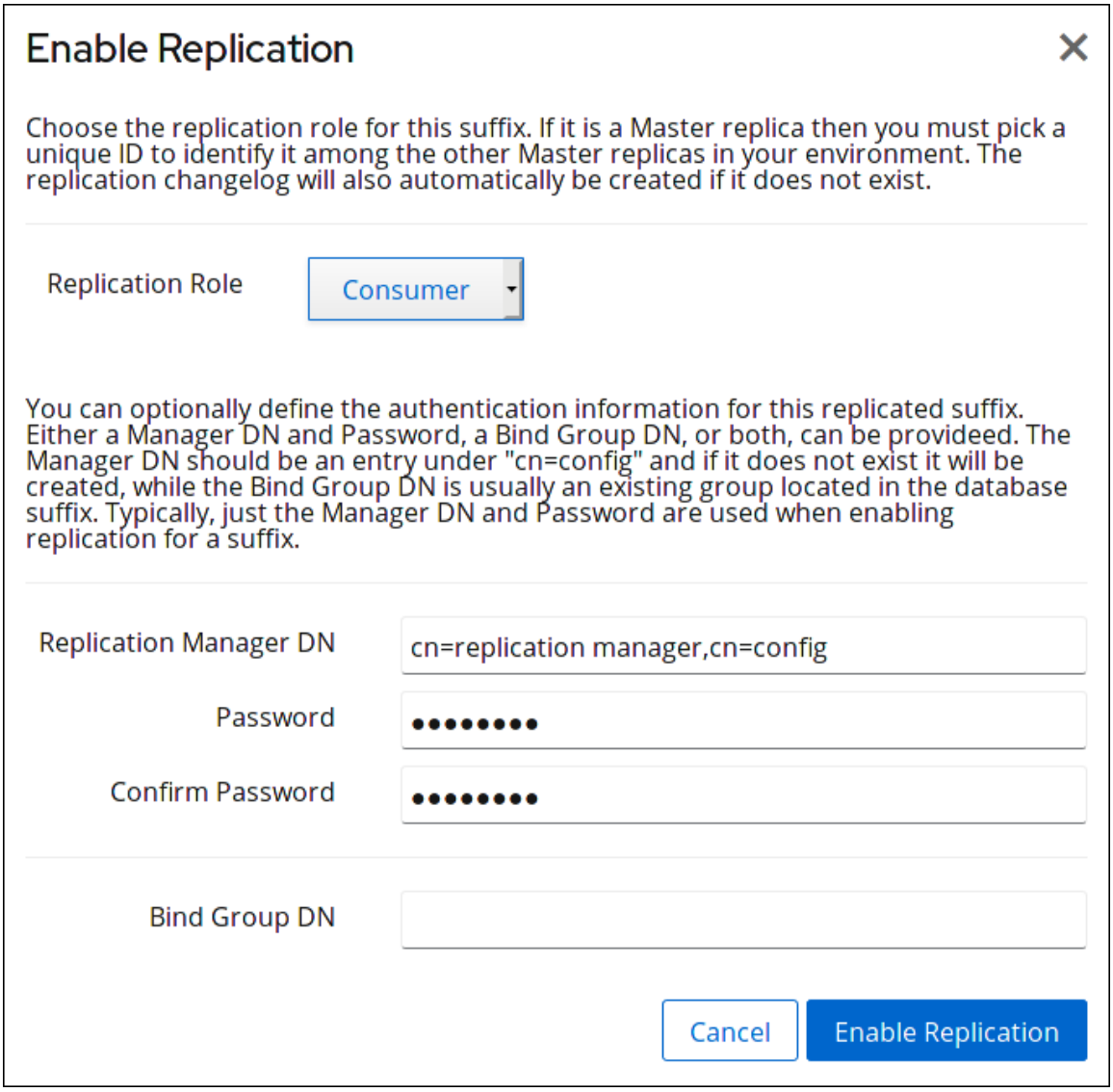


Last Update Status	Last Init Status
Error (0) Replica acquired successfully: Incremental update succeeded	*Initialized*

複製するデータ量によっては、初期化に時間がかかる場合があります。

参加する新規コンシューマーの設定
consumer.example.com ホスト:

1. Directory Server をインストールしてインスタンスを作成します。詳細は、『Red Hat Directory Server インストールガイド』 を参照してください。
2. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
3. インスタンスを選択します。
4. データベースなしでインスタンスを作成した場合には、接尾辞のデータベースを作成します。接尾辞のデータベース作成に関する詳細は、「接尾辞の作成」を参照してください。
5. 接尾辞のレプリケーションを有効にします。
    a. **Replication** メニューを開きます。
    b. **dc=example,dc=com** 接尾辞を選択し、**Enable Replication** をクリックします。
    c. **Replication Role** フィールドで **Consumer** を選択し、作成するレプリケーションマネージャーアカウントの DN およびパスワードを入力します。以下に例を示します。



この設定により、**consumer.example.com** ホストを **dc=example,dc=com** 接尾辞のコンシューマーとして設定します。さらに、サーバーは指定のパスワードで **cn=replication manager,cn=config** ユーザーを作成し、このアカウントがこのホストに接尾辞の変更を複製することができます。

d. **Enable Replication** をクリックします。

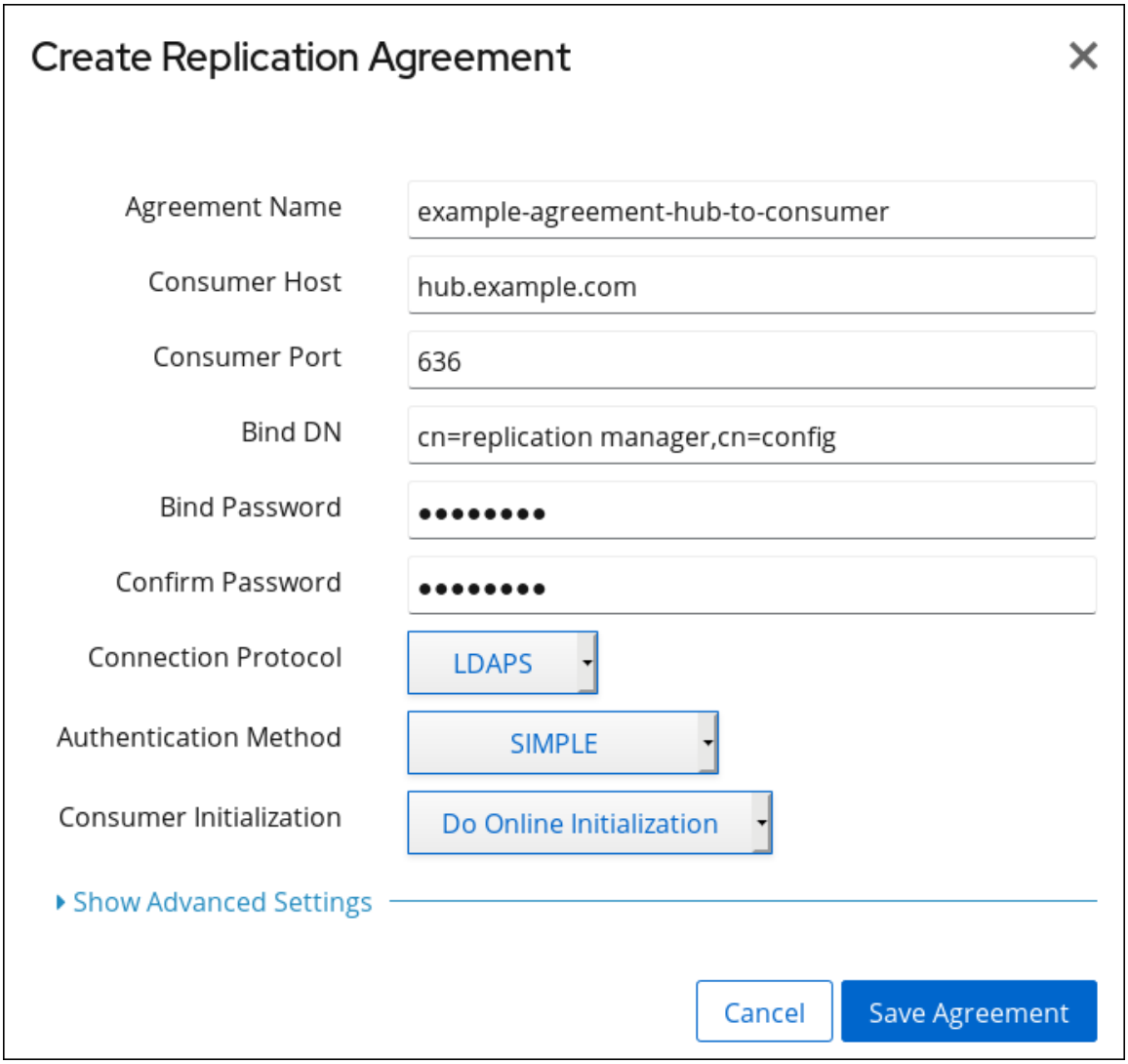
ハブをコンシューマーのサプライヤーとして設定

consumer.example.com ホスト:

1. レプリカ合意を追加し、コンシューマーを初期化します。

   a. **Replication** メニューを開き、接尾辞を選択します。

   b. **Replication Agreements** タブで **Create Agreement** をクリックし、フィールドに入力します。以下に例を示します。



この設定により、**example-agreement-hub-to-consumer** という名前のレプリカ合意が作成されます。

c. **Consumer Initialization** フィールドで **Do Online Initialization** を選択し、合意の保存後にコンシューマーを自動的に初期化します。

後でコンシューマーを初期化する場合は、**Do Not Initialize** を選択します。コンシューマーを初期化する前にレプリケーションが起動しないことに注意してください。コンシューマーの初期化の詳細は、「コンシューマーの初期化」を参照してください。

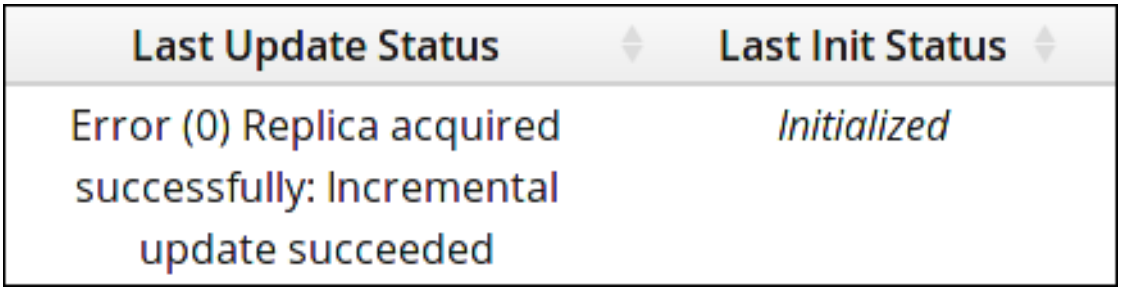
d. **Save Agreement** をクリックします。

2. 初期化が成功したかどうかを確認します。

a. **Replication** メニューを開きます。

b. **Agreements** エントリーを選択します。

初期化が正常に完了すると、Web コンソールは **Last Update Status** 列に **Error (0) Replica acquired successfully: Incremental update succeeded** メッセージを表示します。



複製するデータ量によっては、初期化に時間がかかる場合があります。

15.5. ブートストラップ認証情報の設定

レプリカ合意でバインド識別名 (DN) グループを使用する場合は、グループが存在しないか、古い状態になる可能性があります。

- データベースの初期化前にレプリカに対して認証する必要があるオンライン初期化
- GSSAPI を認証方法として使用する場合、Kerberos 認証情報が変更される

ブートストラップのクレデンシャルをレプリカ合意に設定した場合、Directory Server は、以下のエラーの 1 つにより接続が失敗した場合にこれらのクレデンシャルを使用します。

- **LDAP_INVALID_CREDENTIALS (err=49)**
- **LDAP_INAPPROPRIATE_AUTH (err=48)**
- **LDAP_NO_SUCH_OBJECT (err=32)**

ブートストラップ認証情報でバインドに成功すると、サーバーはレプリケーション接続を確立し、新しいレプリケーションセッションが開始されます。これにより、バインド DN グループメンバーを更新できるようになります。デフォルトでは、次のレプリケーションセッションでは、Directory Server は合意で今回成功したデフォルトの認証情報を使用します。

ブートストラップの認証情報も失敗すると、Directory Server は接続の試行を停止します。

手順
レプリカ合意の作成時にブートストラップ認証情報を設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt create ... --bootstrap-bind-dn "bind_DN" --bootstrap-bind-passwd "password" --bootstrap-bind-method bind_method --bootstrap-conn-protocol connection protocol ...
```

既存のレプリカ合意でブートストラップ認証情報を設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt set --suffix="suffix" --bootstrap-bind-dn "bind_DN" --bootstrap-bind-passwd "password" --bootstrap-bind-method bind_method --bootstrap-conn-protocol connection protocol agreement_name
```

15.6. 証明書ベースの認証を使用するようにレプリケーションパートナーの設定

レプリケーションパートナーに対するバインド DN およびパスワードを使用する代わりに、証明書ベースの認証を使用できます。

以下の手順では、レプリケーショントポロジーに **server2.example.com** という名前の新規サーバーを追加する方法を説明します。また、証明書ベースの認証を使用して、新規ホストと既存の **server1.example.com** との間でレプリカ合意を設定する方法を説明します。

1. 両方のホストで、証明書ベースの認証を設定します。詳細については、「証明書ベースの認証の設定」 を参照してください。

2. **server1.example.com** ホスト:

   a. 両方のサーバー (**cn=server1,example,dc=com** や **cn=server2,dc=example,dc=com** など) にアカウントを作成し、クライアント証明書を対応するアカウントに追加します。詳細については、以下を参照してください。

      - 「**ldapadd** を使用したエントリーの追加」
      - 「ユーザーへの証明書の追加」

      両方のサーバーは、後でこれらのアカウントと証明書を使用して、相互にレプリケーション接続を確立するときに認証を行います。

   b. **cn=repl_server,ou=Groups,dc=example,dc=com** などのグループを作成し、両方のサーバーアカウントを追加します。「グループの使用」を参照してください。

   c. レプリカエントリーを作成し、***nsds5ReplicaBindDNGroup*** 属性を直前の手順で作成されたグループの DN に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server1.example.com replication \
    enable --suffix="dc=example,dc=com" --role="supplier" --replica-id="7" \
    --bind-group-dn="cn=repl_server,ou=Groups,dc=example,dc=com"
```

   d. グループが変更したかどうかを Directory Server が確認するレプリカエントリーの間隔を、**0** に設定します。

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

3. 新しいサーバーを初期化します。

   a. **server2.example.com** で、**cn=Replication Manager,cn=config** などの一時的なレプリケーションマネージャーアカウントを作成します。

   b. **server1.example.com** で、認証用に直前の手順でアカウントを使用する一時的なレプリカ合意を作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server2.example.com repl-agmt \
    create --suffix="dc=example,dc=com" --host="server1.example.com" --port=636 \
    --conn-protocol=LDAPS --bind-dn="cn=Replication Manager,cn=config" \
    --bind-passwd="password" --bind-method=SIMPLE --init \
    temporary_agreement
```

   この合意は、以前に作成したレプリケーションマネージャーアカウントを使用してデータベースを初期化します。この初期化前に、**server2.example.com** のデータベースが空で、関連する証明書を持つアカウントは存在しません。したがって、データベースの初期化前に、証明書を使用してレプリケーションすることはできません。

4. 新しいサーバーが初期化された後

   a. **server1.example.com** から一時的なレプリカ合意を削除します。

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

   b. **server2.example.com** から一時的なレプリケーションマネージャーアカウントを削除します。

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

5. 証明書ベースの認証を使用する両サーバーでレプリカ合意を作成します。

   a. **server1.example.com**:

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

   b. **server2.example.com**:

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

6. レプリケーションが正しく機能していることを確認するには、レプリカ合意に ***nsds5replicaLastUpdateStatus*** 属性を表示します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server1.example.com repl-agmt status --
   suffix="dc=example,dc=com" example_agreement
   ```

   使用できるステータスの詳細は『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の付録 『レプリカ合意の状況』 を参照してください。

15.7. コンシューマーまたはハブを 1 つのサプライヤーにプロモート

ハードウェアの停止によりレプリケーショントポロジー内のサプライヤーが利用できない状況において、管理者は読み取り専用コンシューマーまたはハブを書き込み可能なサプライヤーにプロモートするなどの状況です。

15.7.1. コマンドラインを使用したコンシューマーまたはハブのサプライヤーへのプロモート

たとえば、**server.example.com** ホストを **dc=example,dc=com** 接尾辞のサプライヤーにプロモートするには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication \
    promote --suffix="dc=example,dc=com" --newrole="supplier" --replica-id=2
```

重要

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

必要に応じて、新しいサプライヤーを設定して、接尾辞の変更をトポロジー内の他のサーバーに複製できるようになりました。レプリケーションの設定に関する詳細は、以下を参照してください。

- 「コマンドラインを使用した単一サプライヤーレプリケーションの設定」
- 「コマンドラインを使用したマルチサプライヤーレプリケーションの設定」
- 「コマンドラインによるカスケードレプリケーションの設定」

15.7.2. Web コンソールを使用したコンシューマーまたはハブのサプライヤーへのプロモート

たとえば、コンシューマーまたはハブを、**dc=example,dc=com** 接尾辞のサプライヤーにプロモートするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Replication** メニューを開き、**Configuration** エントリーを選択します。
4. **dc=example,dc=com** 接尾辞を選択します。
5. **Promote** をクリックします。
6. **Replication Role** フィールドで **Supplier** を選択し、レプリカ ID を入力します。

重要

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

7. **Yes, I am sure** を選択します。
8. **Change Role** をクリックして新規ロールを確認します。

必要に応じて、新しいサプライヤーを設定して、接尾辞の変更をトポロジー内の他のサーバーに複製できるようになりました。レプリケーションの設定に関する詳細は、以下を参照してください。

- 「Web コンソールを使用した単一サプライヤーレプリケーションの設定」
- 「Web コンソールを使用したマルチサプライヤーレプリケーションの設定」
- 「Web コンソールによるカスケードレプリケーションの設定」

15.8. コンシューマーの初期化の概要

レプリカ合意を作成したら、コンシューマーを初期化します。この操作中、サプライヤーは既存のデータをコンシューマーにコピーします。

重要

レプリケーションは、コンシューマーを初期化するまで開始しません。

15.8.1. コンシューマーの初期化のタイミング

コンシューマーの初期化には、サプライヤーサーバーからコンシューマーサーバーにデータをコピーする必要があります。サブツリーがコンシューマーに物理的に配置されたら、サプライヤーサーバーはコンシューマーサーバーへの更新操作のリプレイを開始できます。

通常の操作では、コンシューマーは再初期化する必要はありません。ただし、サプライヤーのデータとコンシューマーのデータの間に大きな不一致がある可能性があり、コンシューマーを再初期化します。たとえば、サプライヤーサーバーのデータがバックアップから復元されると、そのサーバーによって提供されたすべてのコンシューマーが再初期化される必要があります。別の例として、サプライヤーがコンシューマーと長期間 (例: 1 週間) にわたって通信できない場合、サプライヤーはコンシューマーが更新できないほど古くなっている可能性があると判断し、再初期化する必要があります。

コンシューマーは、Web コンソールを使用してオンラインで初期化することも、コマンドラインを使用して手動で初期化できます。Web コンソールを使用したオンラインコンシューマーの初期化は、少数のコンシューマーを初期化する効果的な方法です。ただし、各レプリカは順番に初期化されるため、この方法は大量のレプリカを初期化するのに適していません。オンラインコンシューマーの初期化は、コンシューマーがサプライヤーサーバーでのレプリカ合意の設定の一部として初期化される時に使用する方法です。

コマンドラインを使用した手動コンシューマーの初期化は、1 つの LDIF ファイルから多数のコンシューマーを初期化するより効果的な方法です。

15.8.2. 初期タイムアウトの設定

大規模なデータベースの初期化がタイムアウトにより失敗する場合は、以下のいずれかを十分な時間または時間無制限に設定して、操作がタイムアウトする前に Directory Server がデータベース全体を初期化できるようにしてください。

- **cn=config** エントリーの ***nsslapd-idletimeout*** 設定パラメーターは、サーバー上のすべてのレプリカ合意のタイムアウトを設定します。たとえば、グローバルにタイムアウトを無効にするには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-idletimeout=0
```

- レプリケーションマネージャーの DN の ***nsIdleTimeout*** パラメーターは、このレプリケーションマネージャーエントリーを使用するすべての合意のタイムアウトを設定します。たとえば、**cn=replication manager,cn=config** エントリーのタイムアウトを無効にするには、以下のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -w -h server.example.com -p 389 -x
dn: cn=replication manager,cn=config
changetype: modify
add: nsIdleTimeout
nsIdleTimeout: 0
```

15.8.3. コンシューマーの初期化

本セクションでは、コマンドラインと Web コンソールを使用してコンシューマーを初期化する方法を説明します。

15.8.3.1. コマンドラインを使用したコンシューマーの初期化

コンシューマーがオンラインでもオフラインでも、コマンドラインを使用して初期化できます。本セクションでは、両方の手順を説明します。

15.8.3.1.1. コンシューマーオンラインの初期化

レプリカ合意を作成したら、**dsconf repl-agmt init** コマンドを使用してオンラインでコンシューマーを初期化します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt \
    init --suffix="suffix" agreement_name
```

15.8.3.1.2. コンシューマーオフラインの初期化

コンシューマーをオフラインで初期化するには、以下を行います。

1. サプライヤーで以下を行います。
   a. サプライヤーでインスタンスをシャットダウンします。

      ```
      # dsctl instance_name stop
      ```

   b. 複製する接尾辞が含まれる **userRoot** データベースをエクスポートし、レプリケーションの情報を有する **/tmp/example.ldif** ファイルにエクスポートします。

      ```
      # dsctl instance_name db2ldif --replication userRoot /tmp/example.ldif
      ```

   c. サプライヤーでインスタンスを起動します。

      ```
      # dsctl instance_name start
      ```

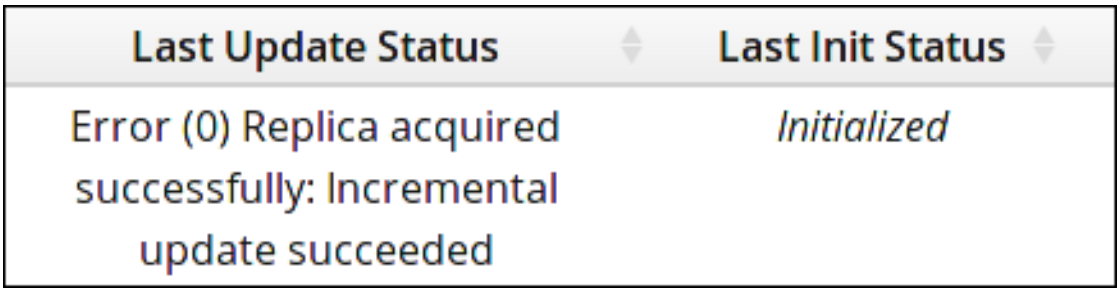
2. エクスポートしたファイルをコンシューマーにコピーします。
3. コンシューマーにデータをインポートします。詳細については、「サーバーがオフライン時のデータのインポート」 を参照してください。

15.8.4. Web コンソールを使用したコンシューマーの初期化

Web コンソールを使用してコンシューマーをオンラインに初期化するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Replication** メニューを開き、接尾辞を選択します。
4. **Replication Agreements** タブで、接尾辞のレプリカ合意の横にある **Choose Action** メニューを開き、**Initialize Agreement** を選択します。

初期化が正常に完了すると、Web コンソールは **Last Update Status** 列に **Error (0) Replica acquired successfully: Incremental update succeeded** メッセージを表示します。



複製するデータ量によっては、初期化に時間がかかる場合があります。

15.9. レプリケーションの無効化および再有効化

デフォルトでは、レプリカ合意の作成時にレプリケーションが有効になります。メンテナンスのためにサーバーを停止するときなど、管理者が一時的にレプリケーションを無効にしたい場合があります。

重要

dsconf replication disable コマンドを使用してレプリケーションエントリーを無効にすると、Directory Server によりレプリカ合意も自動的に削除されます。レプリケーションを再度有効にするには、レプリカ合意を再作成する必要があります。

レプリカ合意を一時的に無効にするには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt disable --suffix="dc=example,dc=com" agreement_name
```

レプリカ合意を再度有効にするには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt enable --suffix="dc=example,dc=com" agreement_name
```

15.10. レプリケーショントポロジーからの DIRECTORY SERVER インスタンスの削除

ハードウェアの停止や構造の変更など、特定の状況では、管理者はレプリケーショントポロジーから Directory Server インスタンスを削除します。本セクションでは、インスタンスの削除に関する詳細を説明します。

15.10.1. Replication Topology からのコンシューマーまたはハブの削除

レプリケーショントポロジーからコンシューマーまたはハブを削除するには、以下を実行します。

1. 削除するホストがハブであり、トポロジー内の他のサーバーのサプライヤーでもある場合は、他のサプライヤーまたはハブを設定して、これらのサーバーにデータを複製します。これらのサーバーに他のサプライヤーが設定されておらず、ハブを削除すると、これらのサーバーはレプリケーショントポロジーから分離されます。レプリケーションの設定に関する詳細は、以下を参照してください。

   - 「単一サプライヤーレプリケーション」

- 「マルチサプライヤーのレプリケーション」
- 「カスケードレプリケーション」

2. 削除するホストで、更新を防ぐためにデータベースを読み取り専用モードに設定します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h host-to-remove.example.com -x

dn: cn=userRoot,cn=ldbm database,cn=plugins,cn=config
changetype: modify
replace: nsslapd-readonly
nsslapd-readonly: on
```

3. 削除するホストとのレプリカ合意があるすべてのサプライヤーで、レプリカ合意を削除します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt \
    delete --suffix="dc=example,dc=com" agreement_name
```

4. コンシューマーまたはハブで、すべての接尾辞のレプリケーションを無効にします。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://host-to-remove.example.com replication \
    disable --suffix="dc=example,dc=com"
```

レプリケーションを無効にすると、このサーバーのこの接尾辞のレプリカ合意がすべて自動的に削除されます。

15.10.2. レプリケーショントポロジーからのサプライヤーの削除

レプリケーショントポロジーからサプライヤーをきれいに削除することは、シューマーまたはハブを削除するよりも複雑です。これは、トポロジー内のすべてのサプライヤーが他のサプライヤーに関する情報を保存し、サプライヤーが利用できない状態になった場合でも、その情報を保持するためです。

Directory Server は、レプリカ更新ベクトル (RUV) と呼ばれるメタデータセットに、レプリケーショントポロジーに関する情報を維持します。RUV には、ID と URL 等のサプライヤーに関する情報、ローカルサーバー上の最新の変更状態番号 (CSN)、および最初の変更の CSN などが含まれています。サプライヤーとコンシューマーはいずれも RUV 情報を保存し、これを使用してレプリケーションの更新を制御します。

サプライヤーを完全に削除するには、設定エントリーと共にそのメタデータを削除する必要があります。

1. 削除するレプリカがトポロジー内の他のサーバーのサプライヤーでもある場合は、他のサプライヤーまたはハブを設定して、そのサーバーにデータを複製します。これらのサーバーに他のサプライヤーが設定されておらず、サプライヤーを削除すると、これらのサーバーはレプリケーショントポロジーから分離されます。レプリケーションの設定に関する詳細は、以下を参照してください。

- 「単一サプライヤーレプリケーション」
- 「マルチサプライヤーのレプリケーション」
- 「カスケードレプリケーション」

2. 削除するサプライヤーで、以下を行います。

   a. データベースを読み取り専用モードにして、更新ができないようにします。詳細については、「読み取り専用モードでのデータベースの設定」 を参照してください。

   b. トポロジー内の他のすべてのサーバーが、このサプライヤーからすべてのデータを受け取るまで待ちます。確認のため、他のサーバーの CSN が、削除するサプライヤーの CSN と同等以上であることを確認します。以下に例を示します。

```
# ds-replcheck online -D "cn=Directory Manager" -w password -m ldap://replica-to-
remove.example.com:389 -r ldap://server.example.com:389 -b dc=example,dc=com
================================================================================
========
        Replication Synchronization Report  (Tue Mar  5 09:46:20 2019)
================================================================================
========


Database RUV's
======================================================

Supplier RUV:
  {replica 1 ldap://replica-to-remove.example.com:389} 5c7e8927000100010000
5c7e89a0000100010000
  {replicageneration} 5c7e8927000000010000

Replica RUV:
  {replica 1 ldap://replica-to-remove.example.com:389} 5c7e8927000100010000
5c7e8927000400010000
  {replica 2 ldap://server.example.com:389}
  {replicageneration} 5c7e8927000000010000
```

   c. レプリカ ID を表示します。

```
# dsconf -D "cn=Directory Manager" ldap://replica-to-remove.example.com replication get
--suffix="dc=example,dc=com" | grep -i "nsds5replicaid"
nsDS5ReplicaId: 1
```

   この例では、レプリカ ID は **1** です。この手順の最後のステップのレプリカ ID を書き留めます。

3. 削除するレプリカとのレプリカ合意があるすべてのサプライヤーで、レプリカ合意を削除します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt \
    delete --suffix="dc=example,dc=com" agreement_name
```

4. 削除するレプリカで、すべての接尾辞のレプリケーションを無効にします。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://replica-to-remove.example.com replication \
    disable --suffix="dc=example,dc=com"
```

レプリケーションを無効にすると、このサーバーのこの接尾辞のレプリカ合意がすべて自動的に削除されます。

5. トポロジー内の残りのサプライヤーのいずれかで、レプリカ ID の RUV をクリーンアップします。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-tasks \
    cleanallruv --suffix="dc=example,dc=com" --replica-id=1
```

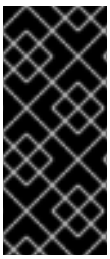
このコマンドは、この手順の直前の手順で表示されるレプリカ ID を指定する必要があります。

15.11. 一部レプリケーションによる属性の管理

「一部レプリケーションを使用した属性のサブセットの複製」で説明するように、一部レプリケーションでは、管理者がレプリケーションの更新から **除外** する属性を設定することができます。管理者は、さまざまなパフォーマンスの理由により、ネットワーク上で送信される大きな属性の数を制限したり、修正タスク (***memberOf*** 計算など) が実行される回数を減らすために実行できます。

レプリケーションから除外する属性のリストは、***nsDS5ReplicatedAttributeList*** 属性で定義されます。この属性はレプリカ合意の一部で、Web コンソールのレプリカ合意ウィザードで設定するか、レプリカ合意の作成時にコマンドラインから設定できます。

```
nsDS5ReplicatedAttributeList: (objectclass=*) $ EXCLUDE memberof authorityRevocationList
accountUnlockTime
```

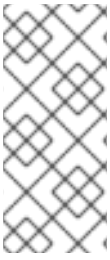


重要

Directory Server には、***nsDS5ReplicatedAttributeList*** 属性の値に **(objectclass=*) $ EXCLUDE** の部分が必要です。**ldapmodify** ユーティリティーなどを使用して属性を直接編集する場合は、上記の例で示されている属性の一覧とともにこの部分を指定する必要があります。ただし、**dsconf** および Web コンソールはどちらも **(objectclass=*) $ EXCLUDE** の部分を自動的に追加し、指定する必要があるのは属性のみです。

15.11.1. 合計更新および増分更新での異なる一部レプリケーション属性の設定

一部レプリケーションが最初に設定されている場合、除外された属性のリストは更新操作ごとに適用されます。つまり、属性のリストは、完全更新と通常の増分更新に対して除外されます。しかし、パフォーマンスを向上させるために増分更新から属性を除外しても、ディレクトリーデータセットを完全にするために全体更新に含めるべき場合があります。この場合は、全体更新 (***nsDS5ReplicatedAttributeListTotal***) から除外する属性の別のリストを定義する 2 番目の属性を追加できます。



注記

nsDS5ReplicatedAttributeList プライマリーの一部レプリケーション属性です。***nsDS5ReplicatedAttributeList*** のみが設定されている場合、増分更新と合計更新の両方に適用されます。***nsDS5ReplicatedAttributeList*** と ***nsDS5ReplicatedAttributeListTotal*** の両方が設定されている場合、***nsDS5ReplicatedAttributeList*** は増分更新にのみ適用されます。

たとえば、***memberOf*** 属性がエントリーに追加されるたびに、memberOf 修正タスクが実行してグループメンバーシップを解決します。これにより、レプリケーションが発生するたびにそのタスクが実行する場合に、サーバーでオーバーヘッドが発生する可能性があります。合計の更新は、レプリケー

ションに新たに追加されたり、長期間オフラインになったデータベースでのみ実行されるため、合計更新後の memberOf 修正タスクを実行すると、合計値が許容オプションになります。この場合、***nsDS5ReplicatedAttributeList*** 属性には ***memberOf*** というリストが記載されるため、増分更新から除外されるようにしますが、***nsDS5ReplicatedAttributeListTotal*** は ***memberOf*** を一覧表示しないため、すべての更新に含まれるようにします。

増分更新の除外リストは、レプリカ合意の ***nsDS5ReplicatedAttributeList*** 属性に設定されます。以下に例を示します。

```
nsds5replicatedattributelist: (objectclass=*) $ EXCLUDE authorityRevocationList accountUnlockTime memberof
```

nsDS5ReplicatedAttributeList 属性を設定するには、**dsconf repl-agmt set** コマンドを使用します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt set \
    --suffix="suffix" --frac-list="authorityRevocationList accountUnlockTime memberof" \
    agreement_name
```

nsDS5ReplicatedAttributeList が唯一の属性セットである場合、そのリストは増分更新と合計更新の両方に適用されます。更新の合計に別のリストを設定するには、レプリカ合意に ***nsDS5ReplicatedAttributeListTotal*** 属性を追加します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt set \
    --suffix="suffix" --frac-list-total="accountUnlockTime" \
    agreement_name
```

注記

nsDS5ReplicatedAttributeList 属性は、すべての更新に対して ***nsDS5ReplicatedAttributeListTotal*** を設定する前に増分更新のために設定される必要があります。

15.11.2. レプリケーションのキープアライブエントリー

サプライヤーの属性を更新すると、サプライヤーに対する changelog 変更シーケンス番号 (CSN) が増加されます。レプリケーショントポロジーでは、このサーバーは最初のコンシューマーに接続し、ローカル CSN をコンシューマーの CSN と比較できるようになりました。ローカル CSN の方が低い場合は、ローカルの changelog から更新内容を取得し、コンシューマーに複製します。一部レプリケーションを有効にしたレプリケーショントポロジーでは、これが問題になることがあります。たとえば、レプリケーションから除外された属性のみがサプライヤー上で更新された場合、複製するための更新が見つからないため、コンシューマー上で CSN が更新されません。特定のシナリオでは、レプリケーションから除外されたサプライヤーで属性のみが更新されると、サプライヤーの更新に不要な検索を行うと、他のサーバーが、必要に応じて後にデータを受け取る可能性があります。この問題を回避するために、Directory Server ではキープアライブエントリーを使用します。

サプライヤー上の更新された属性がすべてレプリケーションから除外され、スキップされた更新の数が 100 を超えると、サプライヤーで ***keepalivetimestamp*** 属性が更新され、コンシューマーに複製されます。***keepalivetimestamp*** 属性はレプリケーションから除外されないため、キープアライブエントリーの更新は複製され、コンシューマー上の CSN が更新され、サプライヤー上のものと等しくなります。次回サプライヤーをコンシューマーに接続する際には、コンシューマーの CSN より新しい更新のみが検索されます。これにより、サプライヤーが送信する新規更新の検索に費やされた時間が短縮されます。

Directory Server は、サプライヤー上でオンデマンドで、レプリケーションのキープアライブエントリーを自動的に作成します。識別名 (DN) にサプライヤーのレプリカ ID が含まれます。キープアライブエントリーはそれぞれ特定のサプライヤーに固有のものです。たとえば、非表示のキープアライブエントリーを表示するには、次のコマンドを実行します。

```
# ldapsearch -D "cn=Directory Manager" -b "dc=example,dc=com" -W -p 389 -h server.example.com
-x 'objectClass=ldapsubentry'

dn: cn=repl keep alive 1,dc=example,dc=com
objectclass: top
objectclass: ldapsubentry
objectclass: extensibleObject
cn: repl keep alive 1
keepalivetimestamp: 20181112150654Z
```

キープアライブエントリーは、以下の状況で更新されます (更新前に存在しない場合には最初に作成されます)。

- 一部レプリカ合意が 100 を超える更新を省略し、レプリケーションセッションの終了前に更新を送信しません。
- サプライヤーがコンシューマーを初期化すると、最初に独自のキープアライブエントリーを作成します。サプライヤーでもあるコンシューマーは、別のコンシューマーも初期化しない限り、独自のキープアライブエントリーを作成しません。

15.11.3. 一部レプリケーションによる空の更新の防止

一部レプリケーションでは、レプリケーション更新 (***nsDS5ReplicatedAttributeList***) から削除される属性の一覧が許可されます。しかし、除外された属性への変更があっても、修正イベントが発生し、空のレプリケーション更新が生成されます。

nsds5ReplicaStripAttrs 属性は、空のレプリケーションイベントでは送信できず、更新シーケンスから削除される属性の一覧を追加します。これには、***modifiersName*** のような運用上の利便性が含まれます。

たとえば、***accountUnlockTime*** 属性が除外されたとします。John Smith のユーザーアカウントがロックされ、期間が期限切れになり、自動的にロック解除されます。***accountUnlockTime*** 属性のみが変更し、その属性はレプリケーションから除外されます。ただし、操作属性 ***internalmodifytimestamp*** も変更されています。John Smith のユーザーアカウントが変更しているため、レプリケーションイベントがトリガーされます。ただし、送信する唯一のデータは新しい変更タイムスタンプであり、更新は空になります。(たとえば) ログイン時間やパスワードの有効期限などに関連する多くの属性がある場合は、空のレプリケーション更新が作成され、サーバーのパフォーマンスに悪影響を与えるか、関連するアプリケーションを妨げる可能性があります。

これを防ぐには、一部レプリケーションの動作を調整するのに役立つように、***nsds5ReplicaStripAttrs*** 属性をレプリカ合意に追加します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com repl-agmt set \
    --suffix="suffix" \
    --strip-list="modifiersname modifytimestamp internalmodifiersname" \
    agreement_name
```

レプリケーションイベントが **空でない** 場合は、削除済みの属性 **は** 他の変更で複製されます。これらの属性は、イベントが空である場合にのみ更新から削除されます。

15.12. レプリケーションを使用した削除されたエントリーの管理

エントリーが削除されると、すぐにデータベースから削除されません。代わりに、**tombstone** エントリーに変換されます。これは、レプリケーションのサーバーで使用されるバックアップエントリーを使用して、特定の競合 (孤立したエントリー) を解決します。tombstone エントリーは、変更した DN (追加された **nsTombstone** オブジェクトクラス) を持つ元のエントリーですが、属性はインデックスから削除されます。

廃棄は無限に保持されません。パージジョブは、指定した間隔で定期的に実行します (***nsDS5ReplicaTombstonePurgeInterval*** 属性の設定)。パージは古い tombstone エントリーを削除します。tombstone エントリーは、指定の期間 (***nsDS5ReplicaPurgeDelay*** 属性で設定) に保存されます。tombstone エントリーが遅延期間よりも古い場合が、次のパージジョブで復元します。

パージ遅延とパージの間隔は、**cn=replica,cn=**replicated suffix**,cn=mapping tree,cn=config** 設定エントリーでレプリカエントリーに設定されます。レプリケーションのパージ設定を定義する場合、2 つの考慮事項があります。

- パージ操作は、特にサーバーが多くの削除操作を処理する場合に時間がかかりません。パージの間隔が低すぎるか、サーバーのリソースを過剰に消費してパフォーマンスに影響を及ぼす可能性があります。

- サプライヤーは、tombstone エントリーなどの変更情報を使用して、初期化後に Prime レプリケーションを実行します。コンシューマーを効果的に再初期化し、レプリケーションの競合を解決するには、変更のバックログが十分にある必要があります。パージ遅延 (tombstone エントリーの経過時間) を低く設定しすぎないでください。または、レプリケーションの競合の解決に必要な情報が失われる可能性があります。

  レプリケーショントポロジーの長いレプリケーションスケジュールよりもわずかに長くパージ遅延を設定します。たとえば、最長のレプリケーションの間隔が 24 時間の場合は、tombstone エントリーを 25 時間保持します。これにより、コンシューマーを初期化するのに十分な変更履歴が確保され、異なるサプライヤーに保存されているデータが乖離するのを防ぐことができます。

dsconf replication set コマンドを使用する場合、**--repl-tombstone-purge-interval=*seconds*** オプションは ***nsDS5ReplicaTombstonePurgeInterval*** 属性を設定し、**--repl-purge-delay=*seconds*** オプションは ***nsDS5ReplicaPurgeDelay*** 属性を設定します。

たとえば、tombstone パージ間隔を **43200** (12 時間) に設定し、レプリカパージの遅延を **90000** (25 時間) に設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com replication set \
    --repl-tombstone-purge-interval=43200 --repl-purge-delay=90000
```

注記

tombstone エントリーおよび状態情報をすぐにクリーンアップするには、非常に小さい値を ***nsDS5ReplicaTombstonePurgeInterval*** 属性および ***nsDS5ReplicaPurgeDelay*** 属性に設定します。どちらの属性も値が秒単位で設定されているため、パージ操作をほぼ即座に開始することができます。

警告

常にパージ期間を使用して、データベースから tombstone エントリーを消去します。**tombstone エントリーを手動で削除しないでください。**

15.13. CHANGELOG 暗号化の設定

セキュリティーを強化するために、Directory Server は changelog の暗号化をサポートします。本セクションでは、この機能を有効にする方法を説明します。

前提条件

サーバーに、ネットワークセキュリティーサービス (NSS) データベースに証明書およびキーを保存する必要があります。したがって、「Directory Server での TLS の有効化」 で説明されているように、サーバーで TLS 暗号化を有効にします。

手順

changelog 暗号化を有効にするには、以下を実行します。

1. changelog 暗号化を有効にするサーバーを除き、以下のコマンドを入力してレプリケーショントポロジー内のすべてのインスタンスを停止します。

```
# dsctl instance_name stop
```

2. changelog 暗号化を有効にするサーバーで、以下を実行します。

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

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication dump-changelog -o /tmp/changelog.ldif
```

   b. インスタンスを停止します。

```
# dsctl instance_name stop
```

   c. **/etc/dirsrv/slapd-*instance_name*/dse.ldif** ファイルの **dn: cn=changelog5,cn=config** エントリーに、以下の設定を追加します。

```
nsslapd-encryptionalgorithm: AES
```

   d. インスタンスを起動します。

```
# dsctl instance_name start
```

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

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

3. 以下のコマンドを実行して、レプリケーショントポロジー内の他のサーバー上のインスタンスをすべて起動します。

```
# dsctl instance_name start
```

検証

changelo が暗号化されていることを確認するには、暗号化された changelo を使用して、サーバー上で以下の手順を実行します。

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

```
# dsctl stop instance_name
```

3. 以下のコマンドを実行して、changelog の一部を表示します。

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

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

4. インスタンスを起動します。

```
# dsctl start instance_name
```

関連情報

- 「レプリケーション変更ログのエクスポート」
- 「レプリケーション changelog の LDIF 形式の changelog ダンプからのインポート」

15.14. CHANGELOG の削除

changelog は、サプライヤーがコンシューマーサーバー (マルチサプライヤーレプリケーションの場合はサプライヤー) のレプリカにこれらの変更を再生するために使用する特定のレプリカに対するすべての変更の記録です。

サプライヤーサーバーがオフラインになると、すべての変更の true レコードを保持しなくなったため、変更ログを削除することが重要です。そのため、レプリケーションのベースとして使用することはできません。ログファイルを削除して、changelog を効果的に削除できます。

15.14.1. コマンドラインを使用した Changelog の削除

サプライヤーサーバーから変更ログを削除するには、次のコマンドを実行します。

1. レプリケーションがすべて接尾辞で無効かどうかを確認します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com replication list
There are no replicated suffixes
```

2. changelog を削除します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com replication delete-changelog
```

15.14.2. Web コンソールを使用した changelog の削除

サプライヤーサーバーから変更ログを削除するには、次のコマンドを実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Replication** メニューを開き、**Replication Changelog** エントリーを選択します。
4. **Delete Changelog** をクリックします。

15.15. レプリケーション変更ログのエクスポート

レプリケーション changelog を暗号化する場合など、特定の状況では、プロセスの一部として changelog をエクスポートする必要があります。changelog をエクスポートするには、以下の手順を実行します。

前提条件

- レプリケーションは Directory Server インスタンスで有効になります。

手順

変更ログを **/tmp/changelog.ldif** ファイルにエクスポートするには、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication dump-changelog -o /tmp/changelog.ldif
```

dirsrv ユーザーには、指定したファイルを作成するには、適切なファイルシステムの権限が必要になることに注意してください。

15.16. レプリケーション CHANGELOG の LDIF 形式の CHANGELOG ダンプからのインポート

この手順では、LDIF 形式のレプリケーション changelog ダンプを Directory Server にインポートします。

前提条件

- レプリケーションは Directory Server インスタンスで有効になります。
- 「レプリケーション変更ログのエクスポート」で説明されているように、changelog ダンプが作成されています。

手順

/tmp/changelog.ldif ファイルから changelog ダンプをインポートするには、以下を入力します。

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

dirsrv ユーザーには、指定したファイルを読み取るパーミッションが必要です。

15.17. レプリケーション CHANGELOG ディレクトリーの移動

特定の状況では、Directory Server レプリケーション changelog ディレクトリーを変更したい場合があります。たとえば、ディレクトリーを **/var/lib/dirsrv/slapd-*instance_name*/new_changelogdb/** に変更するには、以下のコマンドを実行します。

1. 変更ログへの現在のパスを表示し、新しいパスを設定します。
   - コマンドラインの使用するには、以下を行います。
     1. 現在のディレクトリーを表示します。

        ```
        # ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -x \
            -b "cn=changelog5,cn=config" nsslapd-changelogdir
        ...
        nsslapd-changelogdir: /var/lib/dirsrv/slapd-instance_name/changelogdb/
        ```

        ディレクトリーを移動するには、後のステップで表示されたパスが必要です。

     2. 新しいパスを設定します。

        ```
        # ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

        dn: cn=changelog5,cn=config
        changetype: modify
        replace: nsslapd-changelogdir
        nsslapd-changelogdir: /var/lib/dirsrv/slapd-instance_name/new_changelogdb/
        ```

   - Web コンソールの使用
     1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
     2. インスタンスを選択します。
     3. **Replication** メニューを開き、**Replication Changelog** エントリーを選択します。
     4. **Show Advanced Settings** をクリックします。
     5. **Changelog Location** フィールドで現在のパスを特定します。ディレクトリーを移動するには、後のステップで表示されたパスが必要です。
     6. **Changelog Location** フィールドに新しいパスを設定します。
     7. **Save** をクリックします。
2. Directory Server インスタンスを停止します。

   ```
   # dsctl instance_name stop
   ```

3. 前のディレクトリーのコンテンツを **/var/lib/dirsrv/slapd-*instance_name*/new_changelogdb/** に移動します。

```
# mv /var/lib/dirsrv/slapd-instance_name/changelogdb/ \
    /var/lib/dirsrv/slapd-instance_name/new_changelogdb/
```

4. 以前のディレクトリーを削除します。

```
# rm /var/lib/dirsrv/slapd-instance_name/changelogdb/
```

5. Directory Server インスタンスを起動します。

```
# dsctl instance_name start
```

15.18. レプリケーション CHANGELOG のトリム

ディレクトリーサーバーの changelog は、受け取ったおよび処理された変更の一覧を維持します。これには、クライアントの変更やレプリケーションパートナーから受け取った変更が含まれます。

デフォルトでは、ディレクトリーサーバーは 7 日間経過した変更ログエントリーを削除します。ただし、これを変更して以下を設定できます。

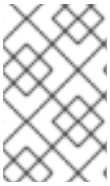
- 変更ログにおけるエントリーの最大経過時間 (***nsslapd-changelogmaxage*** パラメーター)。
- 変更ログ内のレコード総数 (***nsslapd-changelogmaxentries*** パラメーター)。

これらの設定のうち少なくとも 1 つが有効にされると、ディレクトリーサーバーはデフォルトで変更ログを 5 分ごとにトリミングします (***nsslapd-changelogtrim-interval***)。

すべてのレコードとその後に作成されたレコードは、トポロジー内のすべてのサーバーで正常にレプリケートされるまで、変更ログに残ります。「レプリケーショントポロジーからのサプライヤーの削除」で説明されているように、トポロジーからサプライヤーを削除する必要がある場合、Directory Server は、このサプライヤーのすべての更新を他のサーバーの変更ログから削除します。

15.18.1. レプリケーション変更ログのトリミング設定

デフォルトでは、ディレクトリーサーバーは 7 日間経過した変更ログエントリーを削除します。ただし、ディレクトリーサーバーがエントリーを削除できるようになるまでの時間を設定できます。エントリー数が設定値を超えた場合にエントリーを自動的に削除するようにディレクトリーサーバーを設定することもできます。



注記

Red Hat は、エントリーの最大数ではなく、最大経過時間を設定することを推奨します。最大経過時間は、***nsDS5ReplicaPurgeDelay*** parameter in the **cn=replica,cn=suffixDN,cn=mapping tree,cn=config** エントリーのパラメーター。

サプライヤーで次の手順を実行します。

1. 変更ログのトリミングを設定します。
   - 変更ログエントリーの最長期間を設定するには、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication set-changelog
--max-age "4w"
```

このコマンドは、最大経過時間を 4 週間に設定します。パラメーターは、以下の単位をサポートします。

- **s (S)** (秒)
- **m (M)** (分)
- **h (H)** (時間)
- **d (D)** (日)
- **w (W)** (週間)

- エントリーの最大数を設定するには、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication set-changelog
--max-entries "100000"
```

このコマンドは、変更ログのエントリーの最大数を 100,000 に設定します。

2. デフォルトでは、Directory Server は変更ログを 5 分 (300 秒) ごとにトリミングします。別の間隔を設定するには、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication set-changelog --
trim-interval 600
```

このコマンドは、間隔を 10 分 (**600** 秒) に設定します。

15.18.2. 大きな changelog のサイズを手動で縮小

レプリケーション変更ログのトリミングが有効になっていない場合など、特定の状況では、変更ログが過度に大きなサイズに増大する可能性があります。これを修正するには、変更ログのサイズを手動で減らすことができます。

サプライヤーでこの手順を実行します。

前提条件
レプリケーションを有効にしている。

手順

1. (オプション) 変更ログのサイズを表示します。

```
# ls -lh /var/lib/dirsrv/slapd-instance_name/changelogdb/
total 159M
rw------. 1 dirsrv dirsrv 159M  Nov 21 04:01 a1cf5703-697a11ed-896ed7a0-
04f329b5_637b3daf000000010000.db
rw------. 1 dirsrv dirsrv 30 Nov 21 03:58 DBVERSION
```

この例は */var/lib/dirsrv/slapd-instance_name/changelogdb/* ディレクトリーに、サイズが 159M の changelog ファイルが 1 つだけ含まれていることを示しています。

2. 変更ログサイズを縮小した後にパラメーターをリセットできるようにするには、対応するパラメーターの現在の値を表示して書き留めます。

```
# dsconf instance_name replication get-changelog
dn: cn=changelog5,cn=config
cn: changelog5
nsslapd-changelogdir: /var/lib/dirsrv/slapd-instance_name/changelogdb/
nsslapd-changelogmaxage: 7d
nsslapd-changelogtrim-interval: 300
objectClass: top
objectClass: nsChangelogConfig
```

出力に特定の属性が表示されない場合、Directory Server はデフォルト値を使用します。

3. 一時的に、トリミングに関連するパラメーターを減らします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication set-changelog --max-age "300s" --max-entries 500 --trim-interval 60
```

重要

パフォーマンス上の理由から、短い間隔設定を永続的に使用しないでください。

4. ***--trim-interval*** パラメーターに設定した時間が経過するのを待ちます。

5. 変更ログを圧縮して、ディスク領域を再度確保します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com backend compact-db --only-changelog
```

6. 変更ログパラメーターを、一時的に減らす前の値にリセットします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication set-changelog --max-age "7d" --max-entries 0 --trim-interval 300
```

検証

- 変更ログのサイズを表示します。

```
# ls -lh /var/lib/dirsrv/slapd-instance_name/changelogdb/

total 14M
rw------. 1 dirsrv dirsrv 14M Nov 21 05:08 a1cf5703-697a11ed-896ed7a0-04f329b5_637b3daf000000010000.db
rw------. 1 dirsrv dirsrv 30 Nov 21 05:01 DBVERSION
```

15.19. レプリケーション更新の強制

通常のメンテナンスでレプリケーションに関連する Directory Server インスタンスを停止する場合は、オンラインに戻ったら直ちにレプリケーションを更新する必要があります。マルチサプライヤー環境でのサプライヤーでは、セットアップ内の他のサプライヤーがディレクトリー情報を更新する必要があります。それ以外の場合は (たとえば、メンテナンスのためにハブや専用コンシューマーを停止し、後でオンラインに復帰する)、サプライヤーサーバーはステータスを更新する必要があります。

前提条件

- レプリケーションが設定されている。
- コンシューマーが初期化されている。

手順

1. レプリカ合意に更新スケジュールが設定されているかどうかを確認します。

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

コマンドの出力に **nsDS5ReplicaUpdateSchedule:** * が含まれている、または ***nsDS5ReplicaUpdateSchedule*** パラメーターが存在しない場合、更新スケジュールは設定されていません。

nsDS5ReplicaUpdateSchedule に、以下のようなスケジュールが含まれる場合には、値を書き留めておきます。

```
nsDS5ReplicaUpdateSchedule: 0800-2200 0246
```

2. 更新スケジュールが設定されている場合は、以下のコマンドを実行して一時的に無効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt set --schedule \* --
suffix="dc=example,dc=com" agreement_name
```

3. レプリカ合意を一時的に無効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt disable --
suffix="dc=example,dc=com" agreement_name
```

4. レプリカ合意を再度有効にして、レプリケーションの更新を強制的に実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt enable --
suffix="dc=example,dc=com" agreement_name
```

5. この手順の最初にレプリケーションスケジュールが設定されていた場合、スケジュールを以前の値に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt set --schedule
"0800-2200 0246" --suffix="dc=example,dc=com" agreement_name
```

15.20. レプリケーションのタイムアウト期間の設定

ディレクトリーに更新を送信するには、サプライヤーには、コンシューマーへの排他的な接続が必要です。「マルチサプライヤーレプリケーションにおけるコンシューマーの独占を防ぐ」で説明したように、コンシューマーへの接続を試みるサプライヤーに待機時間を設定し、コンシューマーが別のサプライヤーと関連付けられている間にサプライヤーがハングしないようにすることができます。

また、サプライヤーにタイムアウト期間を設定し、低速な接続や破損した接続で更新の送信をやり直してもコンシューマーに接続し続けないようにすることもできます。

タイムアウト期間を設定する属性は 2 つあります。

- ***nsDS5ReplicaTimeout*** は、レプリケーション操作がコンシューマーからの応答を待ってから、タイムアウトして失敗するまでの秒数を設定します。最適な数値を設定するには、アクセスログでレプリケーション処理にかかる平均時間を確認し、それに合わせてタイムアウト期間を設定します。

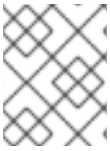
- ***nsDS5DebugReplicaTimeout*** は、デバッグロギングが有効な場合にレプリケーション操作のタイムアウト期間を設定します。デバッグロギングではディレクトリー操作が遅くなる可能性があるため、この設定は ***nsDS5ReplicaTimeout*** 設定よりも大幅に高くなる可能性があります。この属性は任意で、このパラメーターが適用されるエラーログレベルを設定できます。デフォルトはレプリケーションのデバッグ (8192) です。

これらの属性はどちらも、レプリケートされた接尾辞のレプリケーションアグリーメントで設定されます。

```
# ldapmodify -D "cn=Directory Manager" -W -x

dn: cn=example-agreement,cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
changetype: modify
add: nsDS5ReplicaTimeout
nsDS5ReplicaTimeout: 600
-
add: nsDS5DebugReplicaTimeout
nsDS5DebugReplicaTimeout: 6000
```

15.21. RETRO CHANGELOG プラグインの使用

Retro Changelog プラグインは、Directory Server 4.x に実装された changelog との互換性を維持するように Directory Server を設定します。



注記

Directory Server 4.x 形式の changelog に依存するディレクトリークライアントの changelog を維持する必要がある場合は、Retrolog プラグインのみを有効にします。

Retro Changelog プラグインを使用するには、ディレクトリーサーバーインスタンスをシングルサプライヤーレプリカとして設定する必要があります。

Directory Server が Retro Changelog を維持するように設定されていると、この changelog は特別な接尾辞 **cn=changelog** の下に別のデータベースに保存されます。

Retro Changelog は単一レベルのエントリーで設定されます。changelog の各エントリーには、オブジェクトクラス ***changeLogEntry*** があります。changelog エントリーで可能な属性のリストは、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の 『changelog 属性』 セクションを参照してください。

15.21.1. Retro Changelog プラグインの有効化

本セクションでは、**Retro Changelog** プラグインを有効にする方法を説明します。

警告

retro 変更ログバックエンドでレプリケーションを有効にしないでください。retro 変更ログでレプリケーションを有効にすると、次のような結果になる場合があります。

- 過剰な量のレプリケーショントラフィックが生成され、その半分は重複した更新です。
- retro 変更ログのトリミングに関連する削除操作でエラーが発生します。
- レプリケーションのパフォーマンスが非常に低く、サプライヤーでの更新が収束しません。

15.21.1.1. コマンドラインを使用した Retro Changelog プラグインの有効化

コマンドラインを使用して **Retro Changelog** プラグインを有効にするには、以下を実行します。

1. **dsconf** ユーティリティーを使用してプラグインを有効にします。

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

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

15.21.1.2. Web コンソールを使用した Retro Changelog プラグインの有効化

Web コンソールを使用して **Retro Changelog** プラグインを有効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Plugins** メニューを選択します。
4. 左側の一覧で **Retro Changelog** プラグインを選択します。
5. ステータスを **On** に変更します。
6. **Save Config** をクリックします。
7. インスタンスを再起動します。「Web コンソールを使用した Directory Server インスタンスの起動および停止」を参照してください。

15.21.2. Retro Changelog のトリム

nsslapd-changelogmaxage パラメーターで設定したレコードの最大年齢を下げ、***nsslapd-changelog-trim-interval*** で設定した次のトリミング間隔を実行すると、Retro Changelog のサイズが自動的に縮小されます。

たとえば、Retro Changelog のレコードの最大期間を 2 日に設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin retro-changelog set --max-age="2d"
```

15.21.3. Retro Changelog の検索および変更

Changelog は検索操作に対応し、**(&(changeNumber>=X)(changeNumber<=Y))** 形式のフィルターが含まれる検索に対して最適化されます。

一般的なルールとして、changelog のエントリーの追加操作や変更操作は実行しないでください。ただし、エントリーを削除して changelog のサイズをトリミングすることができます。デフォルトのアクセス制御ポリシーを変更するには、Retro Changelog エントリーのみを変更します。

15.21.4. Retro Changelog およびアクセス制御ポリシーの見直し

Directory Server が Retro Changelog を作成するとき、アクセス制御規則 (ACI) は作成されず、Directory Manager のみにアクセス制御規則 (読み取り、検索、比較、書き込み、削除) が適用されます。

Retro Changelog に適用されるデフォルトのアクセス制御ポリシーを変更するには、***cn=changelog*** エントリーの ***aci*** 属性を変更します。たとえば、すべての許可されたユーザーに **読み取り**、**検索**、および **比較** 権限を付与する場合は、次の ACI を ***cn=changelog*** に追加します。

```
dn: cn=changelog
aci: (targetattr="changeNumber || objectClass")(targetfilter="(objectClass=changelogentry)")
 (version 3.0; acl "Enable authenticated users to read the retro changelog"; allow (read, search, compare)
 (userdn="ldap:///all");)
```

警告

Changelog エントリーにはパスワードなどの機密情報が含まれている可能性があるため、***aci*** 属性を修正する際は、匿名ユーザー (***userdn=anyone***) に読み取り権限を付与しないでください。認証されたアプリケーションとユーザー () に対してのみ、この情報へのアクセスを許可してください。

15.22. 特定のレプリカ合意の状態の表示

コマンドラインと Web コンソールで、特定のレプリカ合意のステータスを表示できます。

15.22.1. コマンドラインを使用した特定のレプリカ合意の状態の表示

コマンドラインを使用してレプリカ合意の状態を表示するには、以下を実行します。

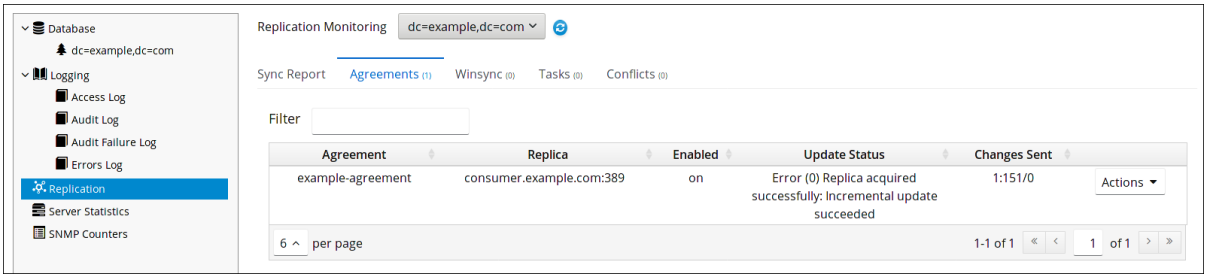
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-agmt status --
suffix="dc=example,dc=com" agreement_name
agreement_name Status for agreement_name agmt consumer.example.com:636
Replica Enabled: on
Update In Progress: FALSE
Last Update Start: 19700101000000Z
Last Update End: 19700101000000Z
Number Of Changes Sent: 0
Number Of Changes Skipped: None
Last Update Status: Error (-1) Problem connecting to replica - LDAP error: Can't contact LDAP server
(connection error)
Init In Progress:
Last Init Start: 19700101000000Z
Last Init End: 19700101000000Z
Last Init Status: unavailable
Reap Active: 0
Replication Status: Not in Synchronization: supplier (5bed8467000100010000) consumer
(Unavailable)  Reason(Unknown)
Replication Lag Time: Unavailable
```

この例では、**consumer.example.com** ホストが利用できないため、現在レプリケーションが失敗することを示しています。

15.22.2. Web コンソールを使用した特定のレプリカ合意のステータスの表示

Web コンソールでレプリカ合意のステータスを表示するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Monitoring** メニューを開き、**Replication**を選択します。
4. 左側のペインの一覧から **Replication** エントリーを選択します。
5. Directory Server インスタンスまたは Winsync Agreement 間のレプリカ合意の状態を表示するかどうかに応じて、該当するタブを選択します。



Update Status 列には、レプリカ合意のステータスが表示されます。

15.23. レプリケーショントポロジーの監視

dsconf replication monitor コマンドを使用して、レプリケーションステータスと、サプライヤー、コンシューマー、およびハブのレプリカ ID、Change State Numbers (CSN) などの追加情報を表示します。

```
# dsconf -D "cn=Directory Manager" ldap://supplier.example.com replication monitor
Enter password for cn=Directory Manager on ldap://supplier.example.com: password

Enter a bind DN for consumer.example.com:389: cn=Directory Manager
Enter a password for cn=Directory Manager on consumer.example.com:389: password
Supplier: server.example.com:389
--------------------------------
Replica Root: dc=example,dc=com
Replica ID: 1
Replica Status: Available
Max CSN: 5e3acb77001d00010000

Status For Agreement: "example-agreement" (consumer.example.com:389)
Replica Enabled: on
Update In Progress: FALSE
Last Update Start: 20200205140439Z
Last Update End: 20200205140440Z
Number Of Changes Sent: 1:166/0
Number Of Changes Skipped: None
Last Update Status: Error (0) Replica acquired successfully: Incremental update succeeded
Last Init Start: 20200205133709Z
Last Init End: 20200205133711Z
Last Init Status: Error (0) Total update succeeded
Reap Active: 0
Replication Status: In Synchronization
Replication Lag Time: 00:00:00

Supplier: consumer.example.com:389
-----------------------------------
Replica Root: dc=example,dc=com
Replica ID: 65535
Replica Status: Available
Max CSN: 00000000000000000000
```

15.23.1. .dsrc ファイルでのレプリケーション監視の認証情報の設定

デフォルトでは、**dsconf replication monitor** コマンドは、リモートインスタンスに対して認証する際に、バインド DN およびパスワードを要求します。または、ユーザーの ~/**.dsrc** ファイルのトポロジーで、バインド DN (および必要に応じて) パスワードを設定できます。

例15.1 .dsrc ファイルの例と各フィールドの説明

~/**.dsrc** ファイルの例を以下に示します。

```
[repl-monitor-connections]
connection1 = server1.example.com:389:cn=Directory Manager:*
connection2 = server2.example.com:389:cn=Directory Manager:[~/pwd.txt]
connection3 = hub1.example.com:389:cn=Directory Manager:S3cret
```

この例では、各エントリーのキーとして ***connection1*** から ***connection3*** を使用しています。ただし、キーが一意であれば、任意のキーを使用できます。

dsconf replication monitor コマンドを実行すると、**dsconf** ユーティリティーはインスタンスのレプリカ合意で設定されたすべてのサーバーに接続します。このユーティリティーが ~/**.dsrc** でホスト名を見つけると、定義された認証情報を使用してリモートサーバーに対して認証されます。上記の

例では、**dsconf** はサーバーへの接続時に以下の認証情報を使用します。

ホスト名	バインド DN	Password
server1.example.com	**cn=Directory Manager**	パスワードの入力を要求します。
server2.example.com	**cn=Directory Manager**	**~/pwd.txt** からパスワードを読み取る
hub1.example.com	**cn=Directory Manager**	**S3cret**

15.23.2. レプリケーショントポロジーモニタリング出力でのエイリアスの使用

デフォルトでは、**dsconf replication monitor** コマンドは監視レポートにサーバーのホスト名を表示します。または、以下のいずれかの方法でエイリアスを表示できます。

- **~/.dsrc** ファイルでエイリアスを定義します。

```
[repl-monitor-aliases]
M1 = server1.example.com:389
M2 = server2.example.com:389
```

- ***-a alias=host_name:port*** パラメーターを **dsconf replication monitor** コマンドに渡してエイリアスを定義します。

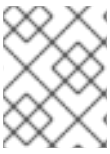
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com replication monitor -a
M1=server1.example.com:389 M2=server2.example.com:389
```

いずれの場合も、コマンドはコマンド出力にエイリアスを表示します。

```
...
Supplier: M1 (server1.example.com:389)
...
Supplier: M2 (server2.example.com:389)
...
```

15.24. 2 つの DIRECTORY SERVER インスタンスの比較

特定の状況では、管理者が 2 つの Directory Server を同期すれば比較する必要があります。**ds-replcheck** ユーティリティーを使用すると、2 つのオンラインサーバーを比較できます。また、**ds-replcheck** では、オフラインモード 2 つの LDIF 形式のファイルを比較し、オンラインモードで 2 つのサーバーを比較することができます。



注記

2 つのデータベースをオフラインで比較するには、**db2ldif -r** コマンドを使用して複製の状態情報を追加します。

2 つのオンラインサーバーを比較すると、負荷が大きい場合、そのデータベースの内容が異なります。この問題を回避するには、***-l time_in_seconds*** パラメーターを **ds-replcheck** に渡すことで、スクリプ

トでラグ時間値を使用します。デフォルトでは、この値は **300** 秒 (5 分) に設定されます。ユーティリティーがラグ時間内にある不整合を検出すると、報告されません。これにより、誤検出が軽減されます。

デフォルトでは、レプリカ合意の特定の属性を複製から除外した場合、**ds-replcheck** はこれらの属性を異なると報告します。これらの属性を無視するには、***-i attribute_list*** パラメーターをユーティリティーに渡します。

たとえば、2 つの Directory Server の ***dc=example,dc=com*** 接尾辞を比較するには、以下のコマンドを実行します。

```
# ds-replcheck -D "cn=Directory Manager" -W \
    -m ldap://server1.example.com:389 \
    -r ldap://server2.example.com:389 \
    -b "dc=example,dc=com"
```

このユーティリティーの出力には、以下のセクションが含まれます。

データベース RUV

データベースの Replication Update Vectors (RUV) と、最小および最大の Change Sequence Numbers (CSN) をリスト表示します。以下に例を示します。

```
Supplier RUV:
 {replica 1 ldap://server1.example.com:389} 58e53b92000200010000 58e6ab46000000010000
 {replica 2 ldap://server2.example.com:389} 58e53baa000000020000 58e69d7e000000020000
 {replicageneration} 58e53b7a000000010000

Replica RUV:
 {replica 1 ldap://server1.example.com:389} 58e53ba1000000010000 58e6ab46000000010000
 {replica 2 ldap://server2.example.com:389} 58e53baa000000020000 58e7e8a3000000020000
 {replicageneration} 58e53b7a000000010000
```

エントリー数

tombstone エントリーを含む、両方のサーバー上のエントリーの合計数を表示します。以下に例を示します。

```
Supplier:  12
Replica: 10
```

Tombstones

各レプリカの tombstone エントリーの数を表示します。これらのエントリーは、合計エントリー数に追加されます。以下に例を示します。

```
Supplier:  4
Replica: 2
```

競合エントリー

各競合エントリーの識別名 (DN)、競合タイプ、および作成された日付をリスト表示します。以下に例を示します。

```
Supplier Conflict Entries: 1
```

```
 - nsuniqueid=48177227-2ab611e7-afcb801a-ecef6d49+uid=user1,dc=example,dc=com
   - Conflict:   namingConflict (add) uid=user1,dc=example,dc=com
   - Glue entry: no
   - Created:    Wed Apr 26 20:27:40 2017


Replica Conflict Entries: 1

 - nsuniqueid=48177227-2ab611e7-afcb801a-ecef6d49+uid=user1,dc=example,dc=com
   - Conflict:   namingConflict (add) uid=user1,dc=example,dc=com
   - Glue entry: no
   - Created:    Wed Apr 26 20:27:40 2017
```

エントリーがありません

足りない各エントリーの DN と、エントリーが存在する他のサーバーからの作成日をリスト表示します。以下に例を示します。

```
Entries missing on Supplier:
 - uid=user2,dc=example,dc=com (Created on Replica at: Wed Apr 12 14:43:24 2017)
 - uid=user3,dc=example,dc=com (Created on Replica at: Wed Apr 12 14:43:24 2017)

Entries missing on Replica:
 - uid=user4,dc=example,dc=com (Created on Supplier at: Wed Apr 12 14:43:24 2017)
```

エントリーの不整合

相手のサーバーとは異なる属性を持つエントリーの DN をリスト表示します。状態情報が利用可能な場合は、これも表示されます。属性の状態情報が利用できない場合には、元の値としてリスト表示されます。レプリケーションが初めて初期化されてから、値が更新されていないことを意味します。以下に例を示します。

```
cn=group1,dc=example,dc=com
----------------------------
Replica missing attribute "objectclass":

 - Supplier's State Info: objectClass;vucsn-58e53baa000000020000: top
 - Date: Wed Apr  5 14:47:06 2017

 - Supplier's State Info: objectClass;vucsn-58e53baa000000020000: groupofuniquenames
 - Date: Wed Apr  5 14:47:06 2017
```

15.25. 一般的なレプリケーションの競合の解決

マルチサプライヤーレプリケーションは、最終的に調整されたレプリケーションモデルを使用します。つまり、同じエントリーを別のサーバーで変更できることを意味します。この 2 つのサーバー間でレプリケーションが発生した場合は、競合する変更を解決する必要があります。多くの場合は、各サーバーでの変更に関連するタイムスタンプに基づいて解決が自動的に行われます。最新の変更が優先されます。

ただし、解像度に到達するには、競合を手動で介入する必要があるケースがあります。変更の競合のあるエントリーは、レプリケーションプロセスで自動的に解決できません。

競合エントリーをリスト表示するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-conflict list dc=example,dc=com
```

15.25.1. ネーミングの競合の解決

異なるサーバーの同じ DN で 2 つのエントリーを作成すると、レプリケーション中に自動的に競合解決が行われ、DN のエントリーの一意識別子を含む、最後に作成されたエントリーの名前が変更します。すべてのディレクトリーエントリーには、操作属性 ***nsuniqueid*** に保存されている一意の識別子が含まれます。命名の競合が発生すると、この一意の ID が一意でない DN に追加されます。

たとえば、2 つの異なるサーバーに **uid=*user_name*,ou=People,dc=example,dc=com** エントリーが作成された場合、レプリケーションは一意の ID を、作成した最後のエントリーの DN に追加します。つまり、以下のエントリーが存在します。

- **uid=*user_name*,ou=People,dc=example,dc=com**
- **nsuniqueid=*66446001-1dd211b2*+uid=*user_name*,ou=People,dc=example,dc=com**

レプリケーションの競合を解決するには、以下の手順を手動で決定する必要があります。

- 有効なエントリー (**uid=*user_name*,ou=People,dc=example,dc=com**) を維持し、競合エントリーを削除するには、次のコマンドを実行します。

  ```
  # dsconf -D "cn=Directory Manager" ldap://server.example.com repl-conflict delete
  nsuniqueid=66446001-1dd211b2+uid=user_name,ou=People,dc=example,dc=com
  ```

- 競合エントリー (**nsuniqueid=*66446001-1dd211b2*+uid=*user_name*,ou=People,dc=example,dc=com**) のみを維持するには、次のコマンドを実行します。

  ```
  # dsconf -D "cn=Directory Manager" ldap://server.example.com repl-conflict swap
  nsuniqueid=66446001-1dd211b2+uid=user_name,ou=People,dc=example,dc=com
  ```

- 両方のエントリーを保持するには、次のコマンドを実行します。

  ```
  # dsconf -D "cn=Directory Manager" ldap://server.example.com repl-conflict convert --new-
  rdn=uid=user_name_NEW nsuniqueid=66446001-
  1dd211b2+uid=user_name,ou=People,dc=example,dc=com
  ```

  競合エントリーを保持するには、**--new-rdn** オプションで新しい Relative Distinguished Name (RDN) を指定して競合エントリーの名前を変更する必要があります。上記のコマンドは、競合エントリーの名前を **uid=user_name_NEW,ou=People,dc=example,dc=com** に変更します。

15.25.2. 孤立エントリーの競合の解決

削除操作が複製され、コンシューマーサーバーが、削除されるエントリーに子エントリーがあることを検出すると、競合解決の手順により、ディレクトリーに孤立したエントリーが存在しないように、**glue** エントリーが作成されます。

同様に、追加操作が複製され、コンシューマーサーバーが親エントリーを検出できない場合は、競合解決の手順により、新しいエントリーが孤立エントリーではないように、親を表す glue エントリーが作成されます。

glue エントリー は、オブジェクトクラス **glue** および **extensibleObject** を含む一時エントリーです。チャンネルエントリーは、複数の方法で作成できます。

- 競合解決手続で、一致する一意の識別子を持つ削除されたエントリーが見つかった場合、glue エントリーは、そのエントリーの再生であり、**glue** のオブジェクトクラスと ***nsds5ReplConflict*** の属性が追加されます。

  このような場合は、glue エントリーを変更して **glue** オブジェクトクラスと ***nsds5ReplConflict*** 属性を削除して、エントリーを通常のエントリーとして維持するか、その子エントリーを削除します。

- サーバーは **glue** および **extensibleObject** オブジェクトクラスを使用して最小のエントリーを作成します。

このような場合は、エントリーを変更して意味のあるエントリーに変換するか、そのすべての子エントリーを削除します。

すべての glue エントリーをリスト表示するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-conflict list-glue suffix
```

glue エントリーおよびその子エントリーを削除するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-conflict delete-glue
DN_of_glue_entry
```

glue エントリーを通常のエントリーに変換するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-conflict convert-glue
DN_of_glue_entry
```

15.25.3. 廃止または不明なエラーの解決

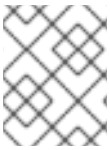
レプリケーショントポロジーに関する情報、つまり、相互に、および同じレプリケーショングループ内の他のレプリカに更新を提供するすべてのサプライヤーは、**レプリカ更新ベクトル (RUV)** と呼ばれるメタデータのセットに含まれています。RUV には、ID と URL、ローカルサーバー上の最新の変更状態番号 (CSN)、最初の変更の CSN などのサプライヤーに関する情報が含まれています。サプライヤーとコンシューマーはいずれも RUV 情報を保存し、これを使用してレプリケーションの更新を制御します。

あるサプライヤーがレプリケーショントポロジーから削除されると、別のレプリカの RUV に残っている場合があります。他のレプリカが再起動すると、ログにエラーを記録し、レプリケーションプラグインが削除されたサプライヤーを認識しないことを警告します。エラーは以下の例のようになります。

```
[22/Jan/2021:17:16:01 -0500] NSMMReplicationPlugin - ruv_compare_ruv: RUV [changelog max
RUV] does not contain element [{replica 8 ldap://m2.example.com:389} 4aac3e59000000080000
4c6f2a02000000080000] which is present in RUV [database RUV]

<...several more samples...>

[22/Jan/2021:17:16:01 -0500] NSMMReplicationPlugin - replica_check_for_data_reload: Warning: for
replica dc=example,dc=com there were some differences between the changelog max RUV and the
database RUV. If there are obsolete elements in the database RUV, you should remove them using
the CLEANALLRUV task. If they are not obsolete, you should check their status to see why there are
no changes from those servers in the changelog.
```

レプリカとその ID (この場合はレプリカ **8** を書き留めます。

サプライヤーがトポロジーから永久に削除されると、そのサプライヤーに関する残存するメタデータは、他のすべてのサプライヤーの RUV エントリーから消去されるはずです。**cleanallruv** ディレクトリータスクを使用して、トポロジー内のすべてのサプライヤーから RUV エントリーを削除します。



注記

cleanallruv タスクが複製されます。そのため、1 つのサプライヤーでのみ実行する必要があります。

手順15.1 cleanallruv タスク操作を使用した、廃止または欠落したサプライヤーの削除

1. 削除されたサプライヤーが他のサプライヤーにメタデータを残す可能性があるため、有効無効を問わず、すべての RUV レコードとレプリカ ID をリスト表示します。

```
# ldapsearch -o ldif-wrap=no -xLLL -H m1.example.com -D "cn=Directory Manager" -W -b
dc=example,dc=com '(&(nsuniqueid=ffffffff-ffffffff-ffffffff-ffffffff)(objectclass=nstombstone))'
nsDS5ReplicaId nsDS5ReplicaType nsds50ruv
dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
nsDS5ReplicaId: 1
nsDS5ReplicaType: 3
nsds50ruv: {replicageneration} 55d5093a000000010000
nsds50ruv: {replica 1 ldap://m1.example.com:389} 55d57026000000010000
55d57275000000010000
nsds50ruv: {replica 20 ldap://m2.example.com:389} 55e74b8c000000140000
55e74bf7000000140000
nsds50ruv: {replica 9 ldap://m2.example.com:389}
nsds50ruv: {replica 8 ldap://m2.example.com:389} 506f921f000000080000
50774211000500080000
```

返されたレプリカ ID **1**、**20**、**9**、および **8** を書き留めます。

2. **cn=config** 接尾辞のレプリカ設定エントリー DN **cn=replica** を検索して、データベースを複製するすべてのサプライヤーに現在定義され、有効なレプリカ ID を一覧表示します。

注記

コンシューマーおよび読み取り専用ノードには、レプリカ ID が **65535** に常に設定され、**nsDS5ReplicaType: 3** はサプライヤーを署名します。

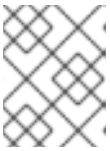
```
# ldapsearch -o ldif-wrap=no -xLLL -H m1.example.com m2.example.com -D "cn=Directory
Manager" -W -b cn=config cn=replica nsDS5ReplicaId nsDS5ReplicaType
dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
nsDS5ReplicaId: 1
nsDS5ReplicaType: 3

dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
nsDS5ReplicaId: 20
nsDS5ReplicaType: 3
```

最初のステップで返されるすべての URI を検索すると (この手順では **m1.example.com** および **m2.example.com**)、返されたサプライヤーのリスト (**nsDS5ReplicaType: 3** があるもの) を直前の手順の RUV の一覧と比較します。上記の例では、この検索で ID **1** と **20** のみが返されますが、以前の検索では URI **m2.example.com** で **9** および **8** も返されます。これは、後者の 2 つが削除済みで、その RUV を消去する必要があることを意味します。

3. クリーニングが必要な RUV を判別した後に、新しい **cn=cleanallruv,cn=tasks,cn=config** エントリーを作成し、レプリケーション設定に関する以下の情報を提供します。

   - 複製されたデータベースのベース DN (**replica-base-dn**)
   - レプリカ ID (**replica-id**)
   - 欠落しているサプライヤーからの最大変更状態番号 (CSN) に追いつくか、あるいはすべての RUV エントリーを削除して更新を見逃すか (**replica-force-cleaning**)。この属性を **no** に設定すると、タスクは設定されているすべてのレプリカが、まず削除されたレプリカからのすべての変更に追いつくのを待ってから、RUV を削除することになります。

```
# dsconf -D "cn=Directory Manager" ldap://m2.example.com repl-tasks \
    cleanallruv --suffix="dc=example,dc=com" --replica-id=8
```



注記

cleanallruv タスクが複製されます。そのため、1 つのサプライヤーでのみ実行する必要があります。

   整理したい各 RUV に同じことを繰り返す (この手順では ID **9**)。

4. 先に確認したすべてのレプリカの RUV をクリーンアップした後、最初の手順からの検索を再度使用して、追加の RUV がすべて削除されていることを確認します。

```
# ldapsearch -o ldif-wrap=no -xLLL -H m1.example.com -D "cn=Directory Manager" -W -b
dc=example,dc=com '(&(nsuniqueid=ffffffff-ffffffff-ffffffff-ffffffff)(objectclass=nstombstone))'
nsDS5ReplicaId nsDS5ReplicaType nsds50ruv
dn: cn=replica,cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
nsDS5ReplicaId: 1
nsDS5ReplicaType: 3
nsds50ruv: {replicageneration} 55d5093a000000010000
nsds50ruv: {replica 1 ldap://m1.example.com:389} 55d57026000000010000
55d57275000000010000
nsds50ruv: {replica 20 ldap://m2.example.com:389} 55e74b8c000000140000
55e74bf7000000140000
```

   上記の出力でわかるように、レプリカ ID **8** および **9** は存在しなくなり、RUV が正常に消去されていることを示しています。

15.26. レプリケーション関連の問題のトラブルシューティング

ここでは、いくつかのエラーメッセージを挙げ、その原因と対処法を説明します。

エラーログレベルを **8192** (レプリケーションのデバッグ) に設定すると、レプリケーションに関する詳細なデバッグ情報を取得できます。「ログレベルの設定」を参照してください。

エラーログレベルを **8192** に変更するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-errorlog-
level=8192
```

ログレベルは加算されるため、上記のコマンドを実行すると、エラーログに過剰なメッセージが表示されます。だから、判断して使用してください。

15.26.1. 考えられるレプリケーション関連のエラーメッセージ

以下のセクションでは、数多くの一般的なレプリケーションの問題を説明します。

agmt=%s (%s:%d) Replica has a different generation ID than the local data

- 理由: このメッセージの最初に指定されたコンシューマーがまだ (正常に) 初期化されていないか、異なるルートサプライヤーから初期化されました。
- 影響: ローカルのサプライヤーは、コンシューマーにデータを複製することはありません。
- 対策: コンシューマーを初期化する前に発生する場合は、このメッセージを無視してください。または、メッセージが永続的であれば、コンシューマーを再初期化します。マルチサプライヤー環境では、すべてのサーバーは、ルートサプライヤーから直接または間接的に 1 回のみ初期化する必要があります。たとえば、M1 は M2 および M4 を初期化し、M2 は M3 の初期化を行います。重要なことは、M2 は、M2 自身の初期化が完了するまで M3 の初期化を開始しないでください (M1 の Web コンソール、もしくは M1 または M2 のエラーログから合計更新ステータスを確認してください)。また、M2 は M1 を再び初期化しないでください。

Warning: data for replica's was reloaded, and it no longer matches the data in the changelog.Recreating the changelog file.This could affect replication with replica's consumers, in which case the consumers should be reinitialized.

- 理由: このメッセージは、サプライヤーが再開した場合にのみ表示されます。これは、サプライヤーが changelog を書き込みできないか、最後のシャットダウン時に RUV を消去できなかったことを示しています。前者はディスク領域の問題、後者はサーバーのクラッシュや不適切なシャットダウンなどが原因です。
- 影響: コンシューマーの ***maxcsn*** がサーバーの changelog に存在しない場合、サーバーはコンシューマーに変更を送信できません。
- 対策: ディスク領域と考えられるコアファイル (サーバーの logs ディレクトリー配下) を確認します。これが単一サプライヤーレプリケーションの場合は、コンシューマーを再初期化します。そうでなければ、後にサーバーがコンシューマーの CSN を見つけられないと訴えた場合に、コンシューマーが他のサプライヤーから CSN を入手できるかどうかを確認します。そうでない場合には、コンシューマーを再初期化します。

agmt=%s(%s:%d): Can't locate CSN %s in the changelog (DB rc=%d).The consumer may need to be reinitialized.

- 理由: ディスクが満杯になったり、またはサーバーが適切にシャットダウンしたりしたたあります。あります。
- 影響: ローカルサーバーは、コンシューマーが再初期化されたり、他のサプライヤーから CSN を取得するまで、そのコンシューマーへの追加の変更を送信できません。
- 対策: これが単一サプライヤーレプリケーションの場合は、コンシューマーを再初期化します。そうでない場合は、コンシューマーが他のサプライヤーから CSN を取得できるかどうかを確認します。そうでない場合には、コンシューマーを再初期化します。

Too much time skew

- 理由: ホストマシンのシステムクロックは同期が非常に低下します。
- 影響: システムクロックは、CSN の一部を生成するのに使用されます。複数のサプライヤー間で変更シーケンスを反映させるために、サプライヤーは、他のサプライヤーのリモートクロックに基づいて、ローカルクロックを転送します。調整は一定量に制限されているため、許可さ

れる制限を超過すると、レプリケーションセッションが中止します。

- 対策: Directory Server ホストマシンでシステムクロックを同期します。該当する場合は、それらのホストでネットワークタイムプロトコル (**ntp**) デーモンを実行します。

agmt=%s(%s:%d): Warning: Unable to send endReplication extended operation (%s)

- 理由: コンシューマーが応答しない。
- 影響: コンシューマーが再起動せずに回復すると、サプライヤーからリリースロックメッセージを受け取らないと、コンシューマーのレプリカが永久にロックされる可能性が高くなります。
- 対策: コンシューマーがあらゆるサプライヤーから新しい変更を受け取ったり、レプリケーションモニターを開始できる場合に、このコンシューマーの全サプライヤーが、レプリカがビジーであることを警告しているかどうかを確認します。レプリカが永久にロックされ、サプライヤーを取得できない場合は、コンシューマーを再起動します。

Changelog is getting too big.

- 理由: changelog のパージがオフになっていて、それがデフォルトの設定になっているか、changelog のパージがオンになっていても一部のコンシューマーがサプライヤーよりもずっと遅れているかのどちらかです。
- 対策: デフォルトでは、changelog のパージはオフになっています。コマンドラインからこれを有効にするには、以下のように **ldapmodify** を実行します。

```
ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=changelog5,cn=config
changetype: modify
add: nsslapd-changelogmaxage
nsslapd-changelogmaxage: 1d
```

1d は 1 日を意味します。その他の有効な時間単位は、秒 (s)、分 (m)、時 (h)、および週 (w) になります。0 の場合は、パージをオフにします。

changelog パージがオンになっている場合、5 分ごとに起動するパージスレッドは、そのエイジが nsslapd-changelogmaxage の値よりも大きく、そのサプライヤー (サプライヤーまたはハブ) のすべての直接コンシューマーに再生されている場合にその変更を削除します。

パージのしきい値に達したときに changelog がパージされていないと思われる場合は、すべてのコンシューマー間でレプリケーションモニターからの最大遅延時間を確認します。パージのしきい値に関わらず、すべてのコンシューマーによって再生される前に変更がパージされることはありません。

The Replication Monitor is not responding.

- 理由: LDAPS ポートは一部のレプリカ合意で指定されますが、証明書データベースは Replication Monitor によって指定されず、アクセスできません。LDAPS ポートに問題がある場合は、レプリケーショントポロジー内のいずれかのサーバーがハングする可能性があります。
- 対策: Replication Monitor の設定ファイルで TLS ポートを TLS 以外のポートにマッピングします。たとえば、636 が TLS ポートであり、389 が TLS 以外のポートである場合は、**[connection]** セクションに以下の行を追加します。

```
*:636=389:*:password
```

In the Replication Monitor, some consumers show just the header of the table.

- 理由: 対応するサプライヤーから作成された変更はありません。この場合、ヘッダー部分の ***MaxCSN***: は **"None"** である必要があります。
- 対策: サプライヤーからの変更がない場合は、何も間違ってありません。

第16章 RED HAT DIRECTORY SERVER と MICROSOFT ACTIVE DIRECTORY の同期

Windows 同期は、ディレクトリーの変更 (Red Hat Directory Server と Microsoft Active Directory の間のグループ、ユーザー、およびパスワードの追加、削除、変更) を行います。これにより、ディレクトリー全体で一貫した情報を維持する方がはるかに効率的で効果的になります。

16.1. WINDOWS 同期の概要

同期により、Active Directory のユーザーエントリーおよびグループエントリーが Red Hat Directory Server のエントリーと一致するようになります。エントリーは作成、変更、または削除されるため、対応する変更が同期ピアサーバーに加えられ、ユーザー、パスワード、およびグループの双方向の同期が可能になります。

同期プロセスはレプリケーションプロセスに似ています。同期はプラグインによって有効にされ、同期合意を介して開始して、ディレクトリーの変更の記録はその changelog に従って送信されます。これにより、Directory Server と Windows サーバーの間のユーザーおよびグループが同期されます。

Windows Synchronization には、ユーザーエントリーおよびグループエントリー用の部分が 2 つあり、もう 1 つはパスワード用です。

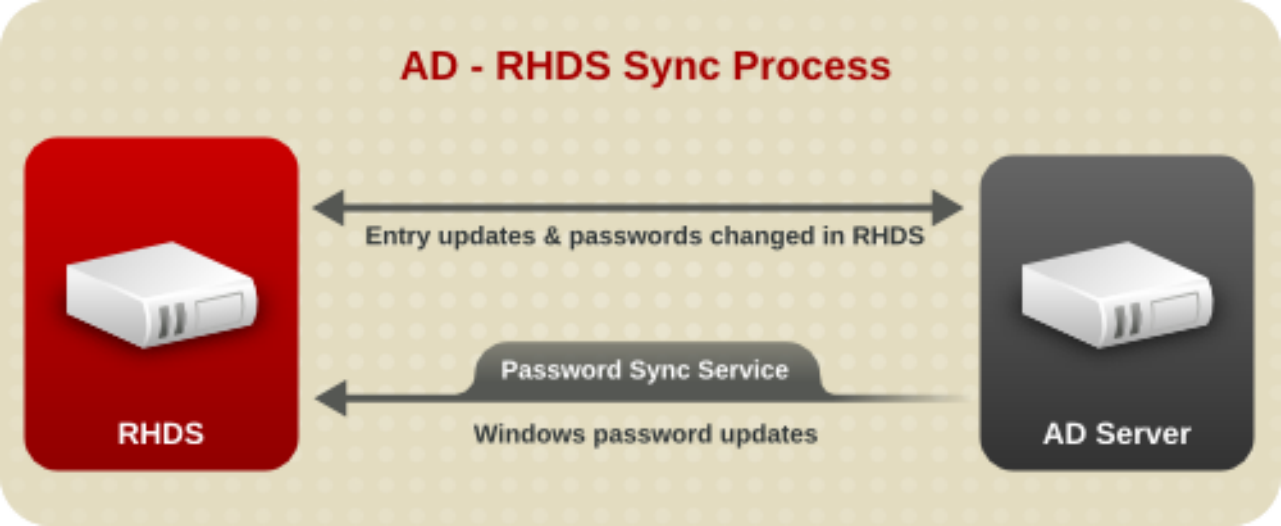
- **Directory Server Windows 同期**ユーザーエントリーおよびグループエントリーの同期は同期合意で設定されます。同様に、レプリケーションがレプリカ合意で設定されます。同期合意は、同期するエントリーの種類 (ユーザー、グループ、またはその両方) と、同期する方向の変更 (Directory Server から Active Directory へ、Active Directory から Directory Server へ、またはその両方) を定義します。

  Directory Server は、マルチサプライヤーレプリケーションプラグインを使用して、ユーザーエントリーおよびグループエントリーを同期します。マルチサプライヤーレプリケーションに使用されるものと同じ changelog は、LDAP 操作として Directory Server から Active Directory に更新を送信するために使用されます。サーバーは、Windows サーバーに対して LDAP 検索操作を実行し、Windows エントリーに加えた変更を対応する Directory Server エントリーと同期します。

- **パスワード同期サービス。cn=config** エントリーの ***nsslapd-unhashed-pw-switch*** パラメーターを **on** に設定すると、Directory Server で行ったパスワードの変更は Active Directory に自動的に同期されます。しかし、Active Directory 上のパスワード変更を認識して Directory Server に伝えるためには、特別なフックが必要です。これは、パスワード同期サービスにより実行されます。このアプリケーションは、Active Directory ドメインコントローラーのパスワード変更をキャプチャーして、LDAPS 経由で Directory Server に送信します。

  パスワード同期サービスは、すべての Active Directory ドメインコントローラーにインストールする必要があります。

図16.1 Active Directory - Directory Server の同期プロセス



同期は、1つ以上の **同期合意** により設定され、制御され、**同期ピア** と同期されるディレクトリーサーバー間の同期を確立します。これらはレプリカ合意と似ており、ホスト名 (IPv4 または IPv6 アドレス) や Active Directory のポート番号などに同様の情報が含まれています。Directory Server は、LDAP/LDAPS を使用してピア Windows サーバーに接続して、更新の送受信を行います。

LDAP (標準接続) は、ユーザーエントリーおよびグループエントリーのみの同期に使用することができますが、パスワードを同期するためには、セキュアな接続の一部が必要になります。セキュアな接続を使用しない場合、Windows ドメインは Directory Server からのパスワードの変更を受け入れず、Password Synchronization サービスは Active Directory ドメインから Directory Server にパスワードを送信しません。Windows Synchronization では、TLS と STARTTLS を使用して LDAPS の両方を許可します。

複数のサブツリーのペアを設定して、相互に同期することができます。**データベース** に接続するレプリケーションとは異なり、同期はディレクトリーツリー構造の **接尾辞** 間で行われます。同期された Active Directory と Directory Server の接尾辞はいずれも、同期合意で指定します。各サブツリー内のすべてのエントリーは、指定の接尾辞 DN の子ではないエントリーを含む、同期用の候補となります。

注記

管理者によって Active Directory とは別に子コンテナーエントリーを作成する必要があります。Windows Synchronization はコンテナーエントリーを作成しません。

Directory Server は、発生した変更を記録するデータベースである **changelog** を維持します。changelog は、Windows Synchronization により Active Directory ピアに追加された変更を調整および送信するために使用されます。Active Directory のエントリーへの変更は、Active Directory の Dirsync 検索機能を使用して確認できます。Directory Server は、デフォルトで 5 分ごとに定期的に Dirsync 検索を実行し、Active Directory サーバーの変更を確認します。**cn=*syncAgreement_Name*,cn=WindowsReplica,cn=*suffix_Name*,cn=mapping tree,cn=config** エントリーの ***winSyncInterval*** パラメーターを設定して、このデフォルトを変更することができます。Dirsync を使用すると、以前の検索以降に変更されたエントリーのみが取得されます。

同期が設定されている場合や、ディレクトリーデータに大きな変更があった場合など、状況によっては全体の更新 (**再同期**) を実行することができます。これにより、同期ピアのすべてのエントリーを調べ、変更または不足しているエントリーを送信します。更新全体が実行するたびに、完全な Dirsync 検索が開始します。詳細は、「同期更新の送信」を参照してください。

Windows 同期機能では、同期するエントリーを管理者が細かく制御できるように、また、さまざまな導入シナリオをサポートするための十分な柔軟性を持たせるために、同期するエントリーをある程度制御

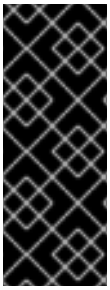
することができます。このコントロールは、Directory Server に設定された異なる設定属性を使用して設定されます。

- 同期合意の作成時に、作成時に新しい Windows エントリー (***nsDS7NewWinUserSyncEnabled*** および ***nsDS7NewWinGroupSyncEnabled***) を同期するオプションがあります。これらの属性が **on** に設定されている場合は、既存の Windows ユーザー/グループが Directory Server に同期され、作成されるユーザー/グループが Directory Server と同期されます。

  Windows サブツリー内では、ユーザーまたはグループのオブジェクトクラスを持つエントリーのみを Directory Server に同期できます。

- Directory Server で、**ntUser** または **ntGroup** オブジェクトクラス、および属性を持つエントリーのみを同期できます。

同期合意の配置は、同期される接尾辞によって異なります。単一の接尾辞の場合、同期合意はその接尾辞に対してのみ行われます。複数の接尾辞の場合、同期合意はディレクトリーツリーの上位ブランチで行われます。Directory Server のデプロイメント全体で Windows エントリーおよび更新を伝播するには、図16.2「マルチサプライヤーディレクトリーサーバー - Windows ドメインの同期」 に示すように、マルチサプライヤーレプリケーション環境でサプライヤー間で合意を作成し、そのサプライヤーを使用して変更を Directory Server デプロイメント全体に複製します。



重要

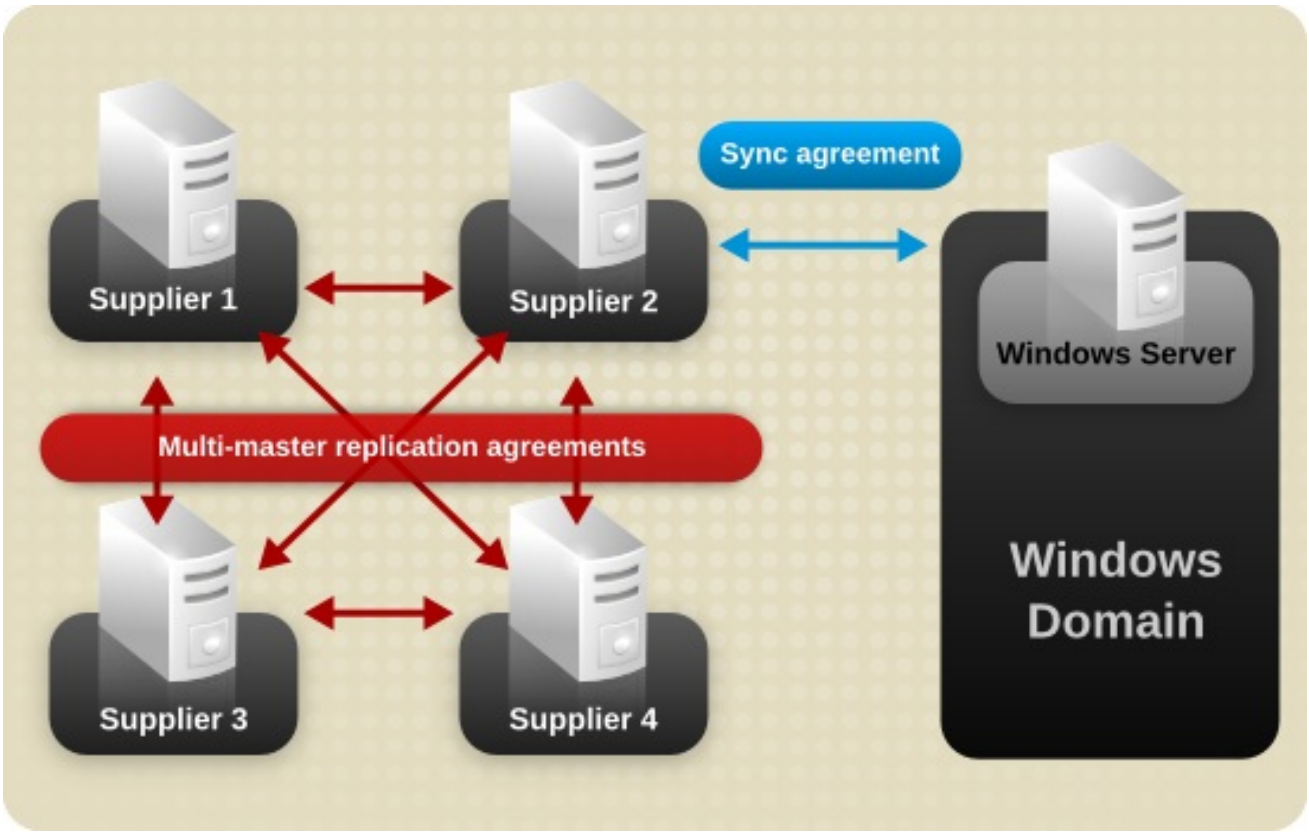
ハブサーバーで同期合意を設定することは可能ですが、Red Hat Directory Server から Active Directory への一方向の同期のみが許可されます。Active Directory サーバーは、変更内容をハブに同期することができません。

同期合意を設定するために、マルチサプライヤーレプリケーションのサプライヤーのみを使用することが強く推奨されます。

警告

Directory Server 環境と Active Directory 環境との間の同期合意は 1 つのみです。同じ Active Directory ドメインに同期合意が複数存在すると、エントリーの競合を作成できます。

図16.2 マルチサプライヤーディレクトリーサーバー - Windows ドメインの同期



Directory Server の changelog で平文のパスワードが保持されるため、Directory Server のパスワードは他のエントリー属性と同期されます。Active Directory で行われたパスワード変更を取得するには、パスワード同期サービスが必要です。パスワード同期サービスがないと、パスワードが Active Directory でハッシュ化され、Windows ハッシュ機能が Directory Server が使用するものと同じであるため、Windows パスワードが同期できません。

16.2. サポート対象の ACTIVE DIRECTORY のバージョン

『Red Hat Directory Server リリースノート』 で、最新の Directory Server バージョンの該当する項を参照してください。

16.3. パスワードの同期

Directory Server エントリーのパスワード変更は、**Password Sync** ユーティリティーを使用して Active Directory エントリーのパスワード属性に同期できます。

パスワードの同期時に、パスワードポリシーは各同期ピアに対してローカルに強制されます。Directory Server でパスワードが変更すると、Directory Server の構文または最小長の要件が適用されます。変更したパスワードが Windows サーバーと同期すると、Windows パスワードポリシーが適用されます。

重要

パスワードポリシー自体は同期されません。

パスワード変更履歴やアカウントロックアウトカウンターなどの設定情報はローカルに保持され、同期できません。

同期用のパスワードポリシーを設定する場合は、以下の点を考慮してください。

- **Password Sync** ユーティリティーは、Directory Server と同期する Windows マシンにローカルにインストールする必要があります。
- **Password Sync** は、Windows マシンを 1 つの Directory Server にのみリンクできます。複数の Directory Server インスタンスと変更を同期するには、マルチサプライヤーレプリケーション用に Directory Server を設定します。
- パスワードの有効期限の警告および時間、バインド試行の失敗、その他のパスワード関連の情報はサーバーごとにローカルで適用され、同期ピアサーバー間で同期されません。
- Windows サーバーが設定された Directory Server インスタンスで、**cn=config** エントリーの ***nsslapd-unhashed-pw-switch*** パラメーターを **on** に設定します。
- バインド動作は、すべてのサーバーで発生します。Directory Server サーバーおよび Active Directory サーバーの両方で、同じパスワードポリシーまたは同様のパスワードポリシーを作成してください。
- 同期用に作成されるエントリー (例: サーバーアイデンティティー) には有効期限のないパスワードが必要です。これらの特別なユーザーが期限切れにならないパスワードを持っていることを確認するために、Directory Server のエントリーに ***passwordExpirationTime*** 属性を追加し、それに **20380119031407Z** の値 (有効範囲の一番上) を指定します。

16.4. ACTIVE DIRECTORY と DIRECTORY SERVER の同期の設定

同期の設定は、レプリケーションの設定と非常に似ています。このデータベースを changelog を使用してサプライヤーとして設定し、同期を定義する合意を作成する必要があります。同期ユーザーである一般的なユーザーアイデンティティーは、Active Directory (AD) ドメインコントローラー (DC) に接続して、Directory Server から AD に更新を適用し、更新を Directory Server と同期するために AD をチェックします。

注記

ユーザーが Directory Server および AD のアカウントを使用できるようにするには、パスワードを同期します。パスワードの同期には、暗号化された接続を使用する必要があります。

ユーザーとグループのエントリーの同期は、AD 側からのパッシブなものです。Directory Server は AD に更新し、AD ドメインの更新をポーリングします。パスワードには、AD サーバーには別のパスワードサービスが必要です。このサービスは、AD ドメインから Directory Server にパスワードの変更をアクティブに送信します。

16.4.1. ステップ 1: Directory Server ホストで TLS の有効化

Password Sync サービスは、暗号化された接続でパスワードを同期する必要があります。Directory Server インスタンスで TLS が有効になっていない場合は、有効にします。詳細については、「Directory Server での TLS の有効化」 を参照してください。

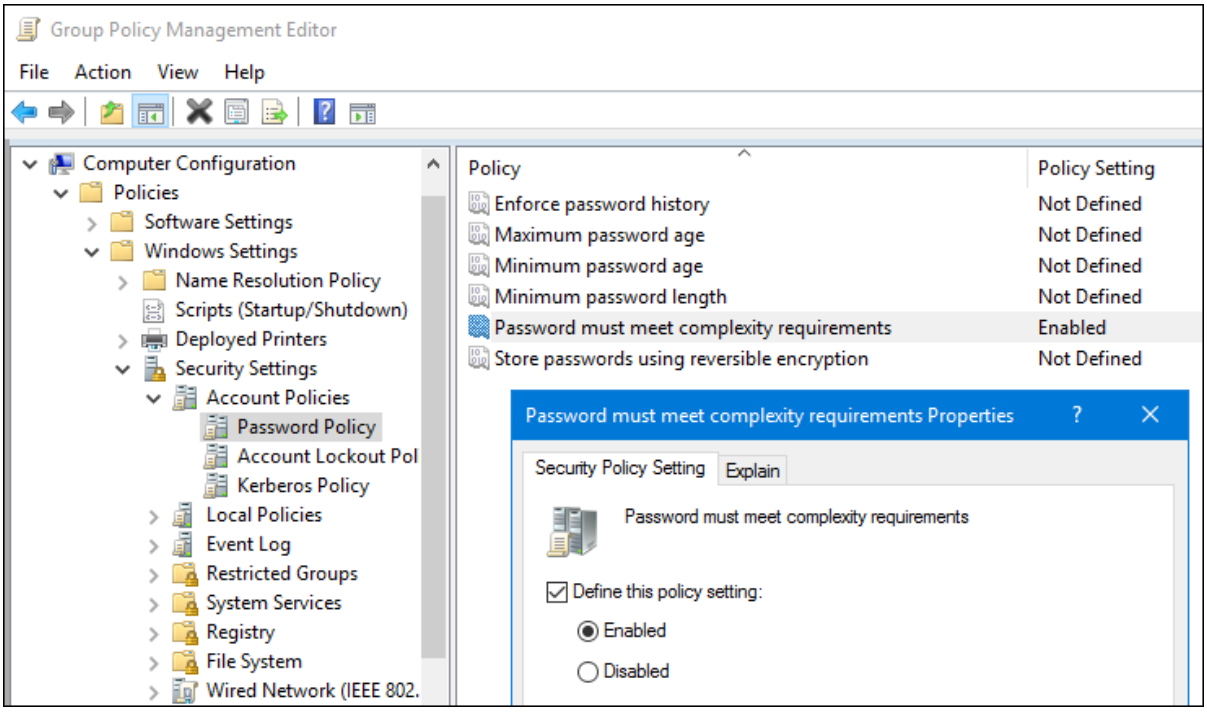
16.4.2. ステップ 2: AD ドメインでのパスワード複雑性の有効化

グループポリシーを使用して、AD ドメインでパスワードの複雑さを有効にします。以下に例を示します。

1. **Group Policy Management** コンソールを開き、ドメインに新しい Group Policy Object (GPO) を作成します。

Group Policy Management コンソールの使用に関する詳細は、Windows ドキュメントを参照してください。

2. GPO を右クリックし、**Edit** を選択して **Group Policy Management Editor** を開きます。
3. **Computer Configuration** → **Windows Settings** → **Security Settings** → **Account Policies** → **Password Policy** に移動し、**Password must meet complexity requirements** という名前のポリシーをダブルクリックします。
4. ポリシーを有効にし、**OK** をクリックします。

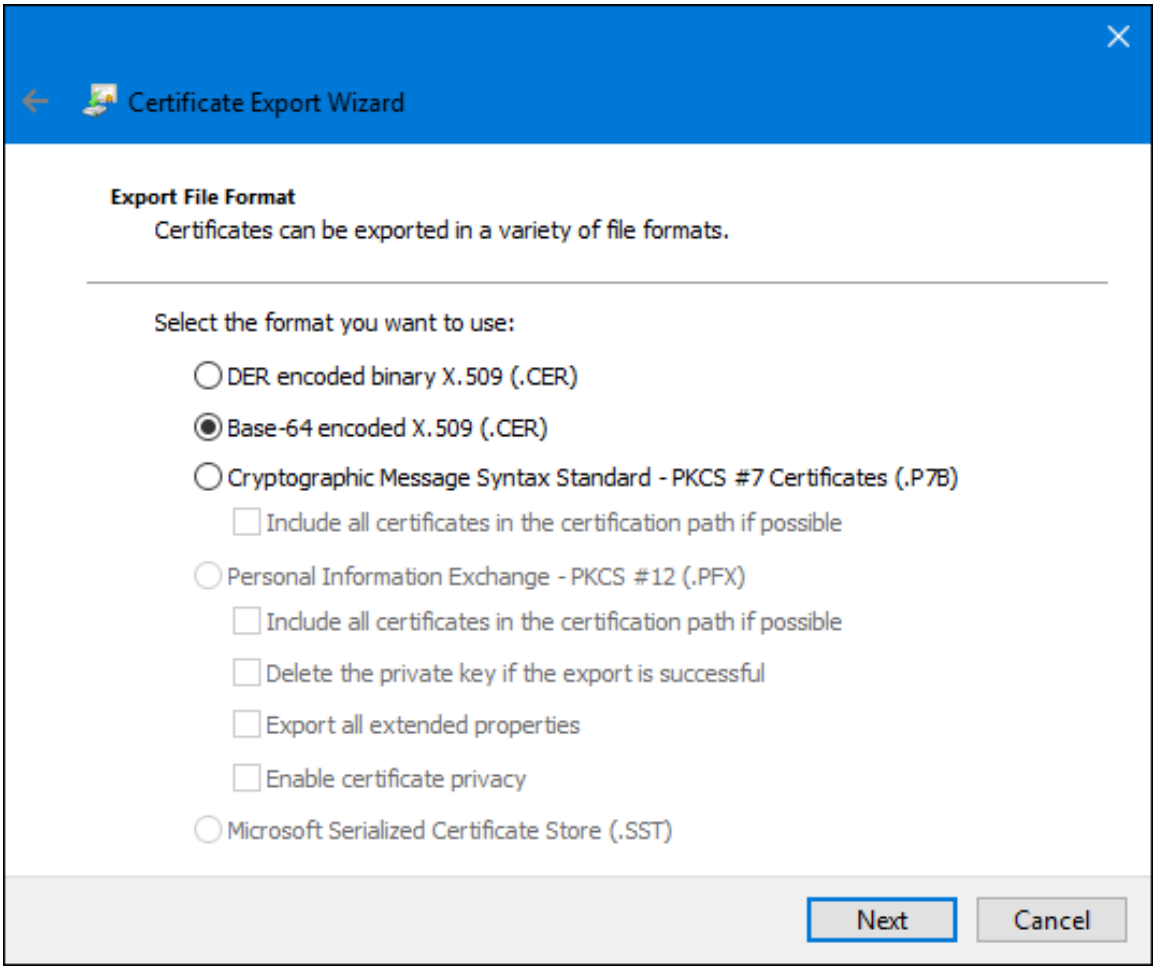


5. **Group Policy Management Editor** および **Group Policy Management** コンソールを閉じます。

16.4.3. ステップ 3: AD からの CA 証明書の抽出

ルート認証局 (CA) 証明書を抽出し、Directory Server ホストにコピーします。

- AD CA 証明書が自己署名の場合は、以下を実行します。
  1. **Certification Authority** アプリケーションがインストールされている AD DC で **Super key+R** の組み合わせを押して **Run** ダイアログを開きます。
  2. **certsrv.msc** コマンドを入力して **OK** をクリックすると、**Certification Authority** アプリケーションが開きます。
  3. ローカルの認証局の名前を右クリックし、**Properties** を選択します。
  4. **全般** タブで、**CA certificates** フィールドでエクスポートする証明書を選択し、**View Certificate** をクリックします。
  5. **詳細** タブで、 **ファイルにコピー** をクリックして **証明書のエクスポートウィザード** を起動します。
  6. **次へ** をクリックしてから、**base-64 でエンコードされた X.509 (.CER)** を選択します。



7. エクスポートされたファイルに適切なディレクトリーおよびファイル名を指定します。**次へ** をクリックして証明書をエクスポートしてから、**完了**をクリックします。

8. ルート CA 証明書を Directory Server ホストにコピーします。

- AD CA 証明書が外部 CA により署名されている場合は、以下を行います。

1. ルート CA を指定します。以下に例を示します。

```
# openssl s_client -connect adserver.example.com:636
CONNECTED(00000003)
depth=1 C = US, O = Demo Company, OU = IT, CN = Demo CA-28
verify error:num=20:unable to get local issuer certificate
verify return:0
---
Certificate chain
 0 s:/C=US/O=Demo Company/OU=IT/CN=adserver.example.com
   i:/C=US/O=Demo Company/OU=IT/CN=Demo CA-1
 1 s:/C=US/O=Demo Company/OU=IT/CN=Demo CA-1
   i:/C=US/O=Demo Company/OU=IT/CN=Demo Root CA 2
```

上記の例では、AD サーバーの CA 証明書は、**CN=Demo Root CA 2** で署名された **CN=Demo CA-1** で署名されています。つまり、**CN=Demo Root CA 2** が root CA であることがわかります。

2. CA 証明書の取得方法についてのルート CA の Operator にお問い合わせください。

3. ルート CA 証明書を Directory Server ホストにコピーします。

16.4.4. ステップ 4: Directory Server の NSS データベースから CA 証明書の拡張

Directory Server の NSS データベースから CA 証明書を抽出するには、以下を実行します。

1. データベースの証明書をリスト表示します。

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

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

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

2. データベースから CA 証明書を抽出します。たとえば、**Example CA** ニックネームで CA 証明書を抽出し、**/root/ds-ca.crt** ファイルに保存します。

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

3. CA 証明書を AD DC にコピーします。

16.4.5. ステップ 5: 同期アカウントの作成

AD と Directory Server の同期には、AD にアカウントが 1 つ、Directory Server にアカウントが 1 つ必要です。本セクションでは、これらのアカウントの作成に関する詳細を説明します。

Directory Server でのアカウントの作成

AD DC は、**Password Sync** サービスの Directory Server アカウントを使用して、パスワードを Directory Server と同期します。たとえば、Directory Server で **cn=pw_sync_user,dc=config** ユーザーを作成するには、次のコマンドを実行します。

1. ユーザーアカウントを作成します。

```
# ldapadd -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=pw_sync_user,cn=config
objectClass: inetorgperson
objectClass: person
objectClass: top
cn: pw_sync_user
sn: pw_sync_user
userPassword: password
passwordExpirationTime: 20380101000000Z
```

これにより、**cn=pw_sync_user,dc=config** アカウントが作成され、その有効期限が 2038 年 1 月 1 日に設定されます。

重要

セキュリティー上の理由から、同期されたサブツリーにアカウントを作成しないでください。

2. 同期されるサブツリーの上部に ACI を設定し、**write** および **compare** パーミッションを **cn=pw_sync_user,dc=config** ユーザーに付与します。たとえば、このような ACI を **ou=People,dc=example,dc=com** エントリーに追加するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="userPassword")(version 3.0;acl "Password synchronization";
 allow (write,compare) userdn="ldap:///cn=pw_sync_user,dc=config";)
```

3. Directory Server が、パスワードをクリアテキストで保存できるように設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-unhashed-pw-switch=on
```

Directory Server は Active Directory 以外のパスワード暗号化を使用するため、Directory Server は、パスワードをクリアテキストで Windows サーバーに送信する必要があります。ただし、クリアテキストパスワードは、パスワードの同期に必要な TLS 暗号化接続で送信されるため、ネットワークに公開されません。

AD でのアカウントの作成

Directory Server は、更新の送受信に、AD への接続時に AD アカウントを使用します。このアカウントは **Domain Admins** グループのメンバーであるか、AD に同等のパーミッションがある必要があります。AD アカウントの作成方法は、AD ドキュメントを参照してください。

16.4.6. ステップ 6: Password Sync サービスのインストール

AD のすべての書き込み可能な DC に **Password Sync** をインストールします。**Password Sync** サービスのインストールの詳細は、Red Hat Directory Server Installation Guide の 『Installing the password synchronization service』 セクションを参照してください。

Red Hat がサポートする **Password Sync** サービスを実行しているオペレーティングシステムの一覧は、『Red Hat Directory Server リリースノート』 を参照してください。

16.4.7. ステップ 7: Password Sync サービスの証明書データベースを使用するように CA Certificate Directory Server を追加

Password Sync サービスがインストールされているすべての DC で、Directory Server が使用する CA 証明書を **Password Sync** サービスの証明書データベースに追加します。

1. **C:\Program Files\Red Hat Directory Password Synchronization** ディレクトリーに移動します。

```
> cd "C:\Program Files\Red Hat Directory Password Synchronization\"
```

2. 現在のディレクトリーに証明書データベースを作成します。

```
> certutil.exe -d . -N
```

certutil.exe ユーティリティーは、作成する新規データベースにパスワードを設定するようにプロンプトを表示します。

3. Directory Server インスタンスが使用する CA 証明書をインポートします。「ステップ 4: Directory Server の NSS データベースから CA 証明書の拡張」にあるこの証明書を Windows DC にコピーしました。たとえば、**C:\ds-ca.crt** ファイルから証明書をインポートし、**Example CA** ニックネームを使用してこれをデータベースに保存するには、以下を実行します。

```
> certutil.exe -d . -A -n "Example CA" -t CT,, -a -i "C:\ds-ca.crt"
```

4. 必要に応じて、CA 証明書がデータベースに正しく保存されていることを確認します。

```
> certutil.exe -d . -L

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

Example CA                              CT,,
```

5. Windows DC を再起動します。システムを再起動するまで、**Password Sync** サービスは利用できません。

注記

Password Sync のインストール時に AD ユーザーアカウントが存在する場合は、パスワードが変更するまで、サービスはこれらのアカウントのパスワードを同期できません。これは、**Password Sync** が Active Directory に格納されたらパスワードを復号できないために発生します。AD ユーザーのパスワードリセットを有効にする方法は、Active Directory のドキュメントを参照してください。

16.4.8. ステップ 8: AD が使用する CA 証明書を Directory Server の証明書データベースに追加

Directory Server ホストで、AD が使用する CA 証明書を証明書データベースに追加します。

1. AD が使用する CA 証明書をインポートします。「ステップ 3: AD からの CA 証明書の抽出」にある証明書を Directory Server ホストにコピーしている。たとえば、**/root/ad-ca.crt** ファイルから証明書をインポートし、**Example CA** ニックネームを使用してこれをデータベースに保存するには、以下を実行します。

```
> certutil -d /etc/dirsrv/slapd-instance_name/ -A -n "Example CA" -t CT,, -a -i /root/ad-ca.crt
```

2. 必要に応じて、CA 証明書がデータベースに正しく保存されていることを確認します。

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

Certificate Nickname                    Trust Attributes
                                        SSL,S/MIME,JAR/XPI
...
Example CA                              CT,,
```

16.4.9. ステップ 9: 同期用のデータベースの設定および同期合意の作成

本セクションでは、同期用にデータベースを設定し、同期合意を作成する方法を説明します。

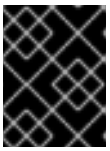
16.4.9.1. コマンドラインを使用した同期用のデータベースの設定および同期合意の作成

以下の例では、Directory Server が **ds.example.com** という名前のホストで実行され、AD DC が **win-server.ad.example.com** という名前のホストで実行していることを前提としています。以下の手順では、これらのホスト間で同期を設定する方法を説明します。

1. 接尾辞のレプリケーションを有効にします。

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

このコマンドは、**ds.example.com** ホストを **dc=example,dc=com** 接尾辞のサプライヤーとして設定し、このエントリーのレプリカ ID を **1** に設定します。



重要

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

2. 同期合意を追加し、合意を初期化します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://ds.example.com repl-winsync-agmt \
    create --suffix="dc=example,dc=com" --host="win-server.ad.example.com" --port=636 \
    --conn-protocol="LDAPS" --bind-
dn="cn=user_name,cn=Users,dc=ad,dc=example,dc=com" \
    --bind-passwd="password" --win-subtree="cn=Users,dc=example,dc=com" \
    --ds-subtree="ou=People,dc=example,dc=com" --win-domain="AD" \
    --init example-agreement
```

このコマンドは、*example-agreement* という名前のレプリカ合意を作成します。レプリカ合意は、AD DC のホスト名、プロトコル、認証情報などの設定を定義します。Directory Server は、データを DC に接続して同期するときに使用します。

この合意の作成後、Directory Server は合意を初期化します。後で合意を初期化するには、**--init** オプションを省略します。合意を初期化する前に同期は開始されないことに注意してください。同期合意の初期化に関する詳細は、「コマンドラインを使用した完全同期の実行」を参照してください。

必要に応じて、**--sync-users="on"** および **--sync-groups="on"** オプションをコマンドに渡して、新しい Windows ユーザーおよびグループを Directory Server に自動的に同期します。

コマンドで使用するオプションの詳細は、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://ds.example.com repl-agmt --help
```

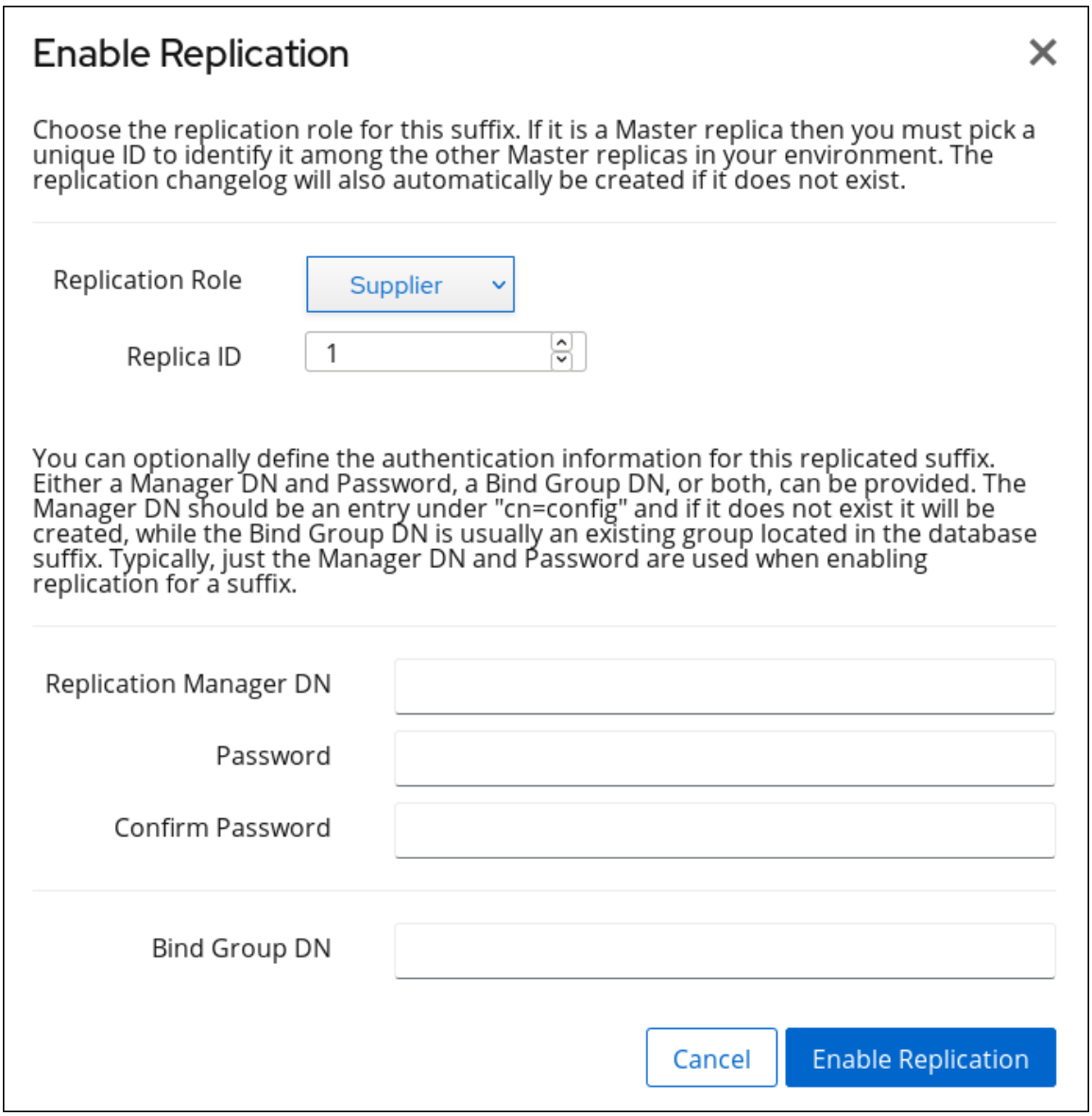
3. 初期化が成功したことを確認します。

```
# dsconf -D "cn=Directory Manager" ldap://ds.example.com repl-winsync-agmt \
    init-status --suffix="dc=example,dc=com" example-agreement
Agreement successfully initialized.
```

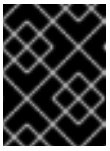
16.4.9.2. Web コンソールを使用した同期用のデータベースの設定および同期合意の作成

以下の例では、Directory Server が **ds.example.com** という名前のホストで実行され、AD DC が **win-server.ad.example.com** という名前のホストで実行していることを前提としています。以下の手順では、これらのホスト間で同期を設定する方法を説明します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. 接尾辞のレプリケーションを有効にします。
   a. **Replication** メニューを開きます。
   b. **dc=example,dc=com** 接尾辞を選択し、**Enable Replication** をクリックします。
   c. **Replication Role** フィールドで **Supplier** を選択し、レプリカ ID を入力します。以下に例を示します。



これらの設定は、**ds.example.com** ホストを **dc=example,dc=com** 接尾辞のサプライヤーとして設定し、このエントリーのレプリカ ID を **1** に設定します。



重要

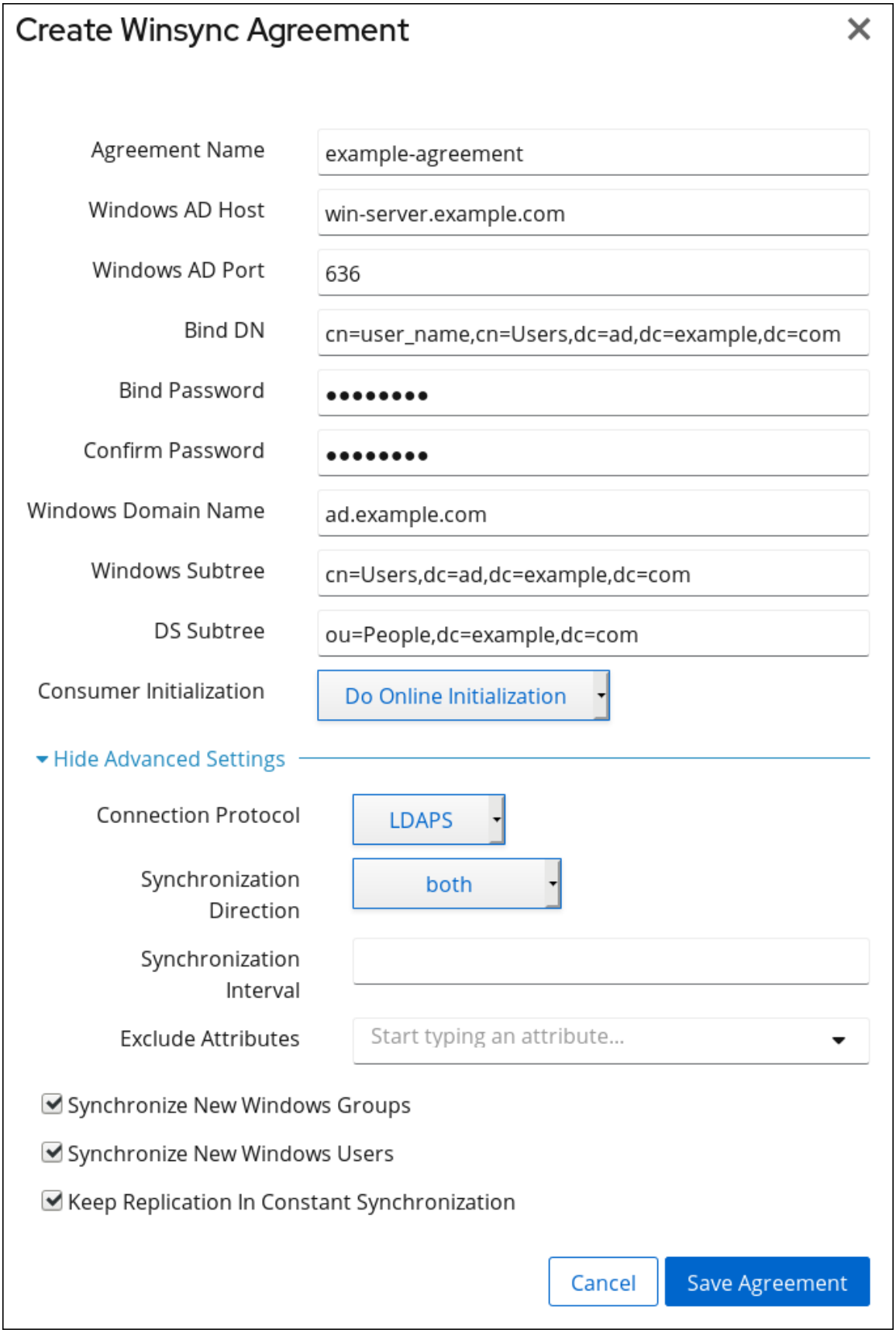
トポロジー内のすべてのサプライヤーの接尾辞については、レプリカ ID は **1** から **65534** の間の一意の整数である必要があります。

d. **Enable Replication** をクリックします。

4. 同期合意を追加し、合意を初期化します。

a. Replication メニューを開き、Winsync Agreements エントリーを選択します。

b. **Create Agreement** をクリックし、フィールドに入力します。以下に例を示します。



この設定により、**example-agreement** という名前の同期合意が作成されます。同期契約では、DC のホスト名、プロトコル、認証情報など、Directory Server が接続してデータを同期する際に使用する設定を定義します。

必要に応じて、**Sync New Windows Users** および **Sync New Windows Groups** を選択すると、新しい Windows ユーザーおよびグループが Directory Server に自動的に同期されます。

この合意の作成後、Directory Server は合意を初期化します。後で合意を初期化するには、**Do Online Initialization** を選択しないでください。合意を初期化する前に同期は開始されないことに注意してください。同期合意の初期化に関する詳細は、「Web コンソールを使用した完全同期の実行」を参照してください。

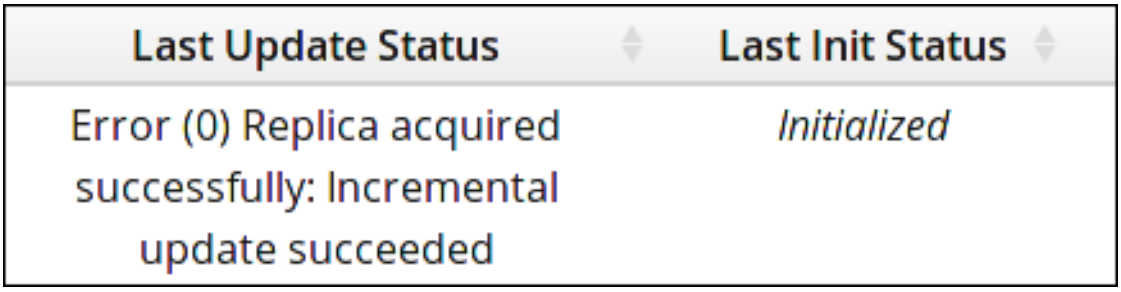
c. **Save Agreement** をクリックします。

5. 初期化が成功したことを確認します。

a. **Replication** メニューを開きます。

b. **Agreements** エントリーを選択します。

初期化が正常に完了すると、Web コンソールは **Last Update Status** 列に **Error (0) Replica acquired successfully: Incremental update succeeded** メッセージを表示します。



同期するデータの量によっては、初期化に最大数時間かかる場合があります。

16.5. ユーザーの同期

ユーザーは、Directory Server と Active Directory 間で自動的に同期されません。両方向の同期を設定する必要があります。

- Active Directory ドメインのユーザーは、**Sync New Windows Users** オプションを選択し、同期合意に設定すると同期されます。同期が開始すると、すべての Windows ユーザーが Directory Server にコピーされ、その後、新しいユーザーが作成されると、そのユーザーが同期されます。

- Directory Server のユーザーアカウントは、Directory Server エントリーにある特定の属性を使用して Active Directory と同期されます。Directory Server エントリーには、**ntUser** オブジェクトクラスと ***ntUserCreateNewAccount*** 属性が必要です。***ntUserCreateNewAccount*** 属性 (既存のエントリーでも) は、Active Directory サーバーにエントリーを書き込むように Directory Server Windows Synchronization プラグインに通知します。

  ntUser オブジェクトクラスが追加された新規または変更されたユーザーエントリーが作成され、エントリーの標準的なポーリングである次の定期更新時に Windows マシンに同期されます。

注記

Active Directory ドメインでは、パスワードが使用されるまでユーザーがアクティブではありません。既存のユーザーが必要な Windows 属性を持つように変更されると、そのユーザーエントリーは Active Directory ドメインに同期されますが、Directory Server 側でパスワードが変更されるか、管理者が Active Directory にパスワードを設定するまでログインできません。これは、Directory Server に保存されているパスワードが暗号化され、**Password Sync** はすでに暗号化されたパスワードを同期できないためです。

Active Directory ドメインでユーザーを有効にするには、ユーザーのパスワードをリセットします。

Directory Server で同期されたエントリーは、それが Directory Server で発生したものであるか、Active Directory で発生したものであるかに関わらず、すべて特別な同期属性を持っています。

- **ntUserDomainId.** これは、Active Directory エントリーの ***sAMAccountName*** 属性に対応します。
- **ntUniqueId.** これには、対応する Windows エントリーの ***objectGUID*** 属性の値が含まれます。この属性は同期プロセスで設定され、手動で設定または変更しないでください。
- **ntUserDeleteAccount.** この属性は、Windows エントリーが同期され、Directory Server エントリーに対して手動で設定する必要がある場合に自動的に設定されます。***ntUserDeleteAccount*** の値が **true** であれば、Directory Server エントリーが削除された場合に対応する Windows エントリーが削除されます。それ以外の場合、エントリーは Active Directory のままになりますが、Directory Server で削除されている場合は Directory Server データベースから削除されます。

Directory Server エントリーで ***ntUserCreateNewAccount*** および ***ntUserDeleteAccount*** を設定すると、Directory Manager では、同期されたサブツリー内のどのユーザーが Active Directory で同期されるかを正確に制御できます。

16.5.1. Directory Server と Active Directory との間で同期されるユーザー属性

Directory Server 属性および Active Directory 属性のサブセットのみが同期されます。これらの属性はハードコーディングされ、エントリーの同期方法に関わらず定義されます。Directory Server または Active Directory のいずれかにあるエントリーにあるその他の属性は、同期の影響を受けないままになります。

Directory Server および Active Directory で使用される属性の一部は同一です。これは通常、すべての LDAP サービスに共通する LDAP 標準で定義された属性です。これらの属性は、相互に正確に同期されます。表16.2「Directory Server および Windows サーバーで同一のユーザースキーマ」 は、Directory Server と Windows サーバーとの間で同じ属性を示しています。

同じ情報を定義する属性もありますが、属性やスキーマ定義の名前が異なります。これらの属性は Active Directory と Directory Server の間でマッピングされるため、1 つのサーバーの属性 A がもう 1 つのサーバーの属性 B として扱われます。同期の場合、これらの属性の多くは Windows 固有の情報に関連します。表16.1「Directory Server と Active Directory との間でマッピングされるユーザースキーマ」 は、Directory Server と Windows サーバーとの間で同じ属性を示しています。

Directory Server および Active Directory が一部のスキーマ要素を処理する方法の違いについての詳細は、 「Red Hat Directory Server と Active Directory との間のユーザースキーマの相違点」 を参照してください。

表16.1 Directory Server と Active Directory との間でマッピングされるユーザースキーマ

Directory Server	Active Directory
cn[a]	name
ntUserDomainId	sAMAccountName
ntUserHomeDir	homeDirectory
ntUserScriptPath	scriptPath
ntUserLastLogon	lastLogon
ntUserLastLogoff	lastLogoff
ntUserAcctExpires	accountExpires
ntUserCodePage	codePage
ntUserLogonHours	logonHours
ntUserMaxStorage	maxStorage
ntUserProfile	profilePath
ntUserParms	userParameters
ntUserWorkstations	userWorkstations

[a] ***cn*** は、他の同期属性とは異なる方法で処理されます。Directory Server から Active Directory に同期する際に、直接 (***cn*** から ***cn*** に) マッピングされます。ただし、Active Directory から Directory Server に同期する場合、***cn*** は Windows の ***name*** 属性から Directory Server の ***cn*** 属性にマッピングされます。

表16.2 Directory Server および Windows サーバーで同一のユーザースキーマ

cn[a]	physicalDeliveryOfficeName
description	postOfficeBox
destinationIndicator	postalAddress
facsimileTelephoneNumber	postalCode
givenname	registeredAddress
homePhone	sn

homePostalAddress	st
initials	street
l	telephoneNumber
mail	teletexTerminalIdentifier
mobile	telexNumber
o	title
ou	usercertificate
pager	x121Address

[a] ***cn*** は、他の同期属性とは異なる方法で処理されます。Directory Server から Active Directory に同期する際に、直接 (***cn*** から ***cn*** に) マッピングされます。ただし、Active Directory から Directory Server に同期する場合、***cn*** は Windows の ***name*** 属性から Directory Server の ***cn*** 属性にマッピングされます。

16.5.2. Red Hat Directory Server と Active Directory との間のユーザースキーマの相違点

Active Directory は Directory Server と同じ基本的な X.500 オブジェクトクラスをサポートしますが、管理者が認識すべき非互換性がいくつかあります。

16.5.2.1. cn 属性の値

Directory Server では、***cn*** 属性に複数の値を指定できますが、Active Directory ではこの属性に単一の値しか持たせません。Directory Server の ***cn*** 属性が同期されると、単一の値のみが Active Directory ピアに送信されます。

これは、同期の意味としては、***cn*** の値が Active Directory エントリーに追加され、その値が Directory Server の ***cn*** の値のいずれでもない場合、Directory Server の ***cn*** 値がすべて単一の Active Directory 値で上書きされます。

もう 1 つの重要な相違点として、Active Directory は ***cn*** 属性を命名属性として使用するのに対し、Directory Server では ***uid*** を使用する点があります。つまり、***cn*** 属性が Directory Server で編集されると、エントリーの名前が完全に (誤って) 変更される可能性があります。この ***cn*** の変更が Active Directory エントリーに書き込まれると、エントリーの名前が変更になり、新しい名前付きエントリーが Directory Server に書き込まれます。

16.5.2.2. パスワードポリシー

Active Directory と Directory Server の両方は、パスワードの最小長や最大期間などのパスワードポリシーを強制できます。Windows 同期を使用すると、ポリシーの一貫性、強制、同期がなくなります。Directory Server と Active Directory の両方においてパスワードポリシーの一貫性がないため、他のシステムと同期すると、システムに加えられたパスワードの変更が失敗する可能性があります。Directory Server におけるデフォルトのパスワード構文設定は、Active Directory が実施するデフォルトのパスワードの複雑さルールに準拠します。

16.5.2.3. street および streetAddress の値

Active Directory は、ユーザーまたはグループの住所に ***streetAddress*** 属性を使用します。これは、Directory Server が ***street*** 属性を使用する方法です。Active Directory および Directory Server が ***streetAddress*** 属性および ***street*** 属性を使用する方法には 2 つの重要な相違点があります。

- Directory Server では、***streetAddress*** は ***street*** のエイリアスです。Active Directory にも ***street*** 属性がありますが、***streetAddress*** のエイリアスではなく、別の属性で個別の値を保持することができます。
- Active Directory は ***streetAddress*** と ***street*** を単一値の属性として定義しますが、Directory Server は RFC 4519 で指定されるように ***street*** を多値属性として定義します。

Directory Server および Active Directory が ***streetAddress*** および ***street*** 属性を処理する方法が異なるため、Active Directory および Directory Server で address 属性を設定する際に従う 2 つのルールがあります。

- Windows 同期は、Windows エントリーの ***streetAddress*** を Directory Server の ***street*** にマッピングします。競合を避けるために、***street*** 属性は Active Directory では使用しないようにしてください。
- 1 つの Directory Server ***street*** 属性値のみが Active Directory に同期されます。 ***streetAddress*** 属性が Active Directory で変更され、新しい値が Directory Server に存在しない場合は、Directory Server のすべての ***street*** 属性値が新しい Active Directory 値に置き換えられます。

16.5.2.4. initials 属性の制約

initials 属性では、Active Directory は最大長 6 文字の制限を課しますが、Directory Server には長さ制限がありません。6 文字を超える ***initials*** 属性が Directory Server に追加されると、その値は Active Directory エントリーと同期したときにトリミングされます。

16.5.3. Directory Server ユーザーのユーザー同期の設定

Directory Server ユーザーが Active Directory に同期するには、ユーザーエントリーに適切な同期属性を設定する必要があります。

コマンドラインで同期を有効にするには、必要な同期属性をエントリーに追加するか、これらの属性でエントリーを作成します。

同期には、以下の 3 つのスキーマ要素が必要です。

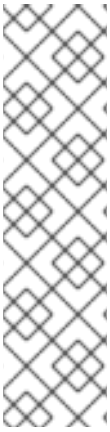
- **ntUser** オブジェクトクラス。
- Windows ID を指定する ***ntUserDomainId*** 属性
- ***ntUserCreateNewAccount*** 属性は、同期プラグインに Active Directory 経由で Directory Server エントリーを同期するように通知します。

たとえば、以下のように **ldapmodify** ユーティリティーを使用します。

```
dn: uid=scarter,ou=People,dc=example,dc=com
changetype: modify
add: objectClass
objectClass:ntUser
-
add: ntUserDomainId
```

```
ntUserDomainId: Sam Carter
-
add: ntUserCreateNewAccount
ntUserCreateNewAccount: true
-
add: ntUserDeleteAccount
ntUserDeleteAccount: true
```

エントリーに、さらに多くの Windows およびユーザー属性を追加できます。同期されたスキーマは、「Directory Server と Active Directory との間で同期されるユーザー属性」 にすべてリストされます。**ntUser** オブジェクトクラスに属する Windows 固有の属性については、『Red Hat Directory Server 11 Configuration, Command, and File Reference』 で詳しく説明されています。



注記

ユーザーのパスワードをリセットします。

Active Directory ドメインでは、パスワードが使用されるまでユーザーがアクティブではありません。既存のユーザーが必要な Windows 属性を持つように変更されると、そのユーザーエントリーは Active Directory ドメインに同期されますが、Directory Server 側でパスワードが変更されるか、管理者が Active Directory にパスワードを設定するまでログインできません。**Password Sync** は、暗号化したパスワードを同期できません。

したがって、Active Directory ドメインでユーザーをアクティブにするには、ユーザーのパスワードをリセットします。

16.5.4. Active Directory ユーザーのユーザー同期の設定

Windows ユーザー (Active Directory ドメインにあるユーザー) の同期は、同期合意で設定されます。

ユーザー同期を有効にするには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-winsync-agmt set --sync-users="on" --suffix="dc=example,dc=com" example-agreement
```

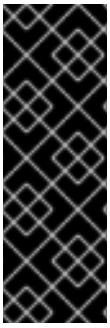
ユーザーの同期を無効にするには、**--sync-users** オプションを **off** に設定します。

16.6. グループの同期

ユーザーエントリーと同様に、グループは Directory Server と Active Directory の間で自動的に同期されません。両方向の同期を設定する必要があります。

- 同期合意で設定されている場合、Active Directory ドメインのグループは、**新規 Windows グループの同期** オプションを選択すると同期されます。同期を開始すると、すべての Windows グループが Directory Server にコピーされ、新規グループは作成時に同期されます。
- Directory Server のグループアカウントは、Directory Server エントリーにある特定の属性を使用して Active Directory と同期します。Directory Server エントリーには、**ntGroup** オブジェクトクラスと ***ntGroupCreateNewGroup*** 属性が必要です。***ntGroupCreateNewGroup*** 属性 (既存のエントリーでも) は、Active Directory サーバーにエントリーを書き込むように Directory Server Windows Synchronization に通知します。

  ntGroup オブジェクトクラスを持つ新規または変更されたグループが作成され、次の通常の更新時に Windows マシンと同期されます。



重要

グループを同期すると、そのメンバーのリストも同期されます。ただし、ユーザー同期が有効で、これらのエントリーに適用する限り、メンバーエントリー自体は同期されません。

これにより、アプリケーションやサービスが Active Directory サーバー上のグループのすべてのメンバーに対して修正操作を行おうとしたときに、それらのユーザーの一部が存在しない場合に問題が発生する可能性があります。

また、グループには、その他の一般的な属性がいくつかあります。

- Active Directory では、Directory Server グループが作成/削除されるかどうかを制御する 2 つの属性 (***ntGroupCreateNewGroup*** および ***ntGroupDeleteGroup***) を制御します。

  ntGroupCreateNewGroup は、Active Directory に Directory Server グループを同期するために必要です。

- **ntUserDomainId** には、Active Directory ドメインのエントリーの一意の ID が含まれます。これは、**ntGroup** オブジェクトクラスの唯一の必須属性です。

- **ntGroupType** は Windows グループのタイプです。Windows のグループタイプには、global/security、domain local/security、builtin、universal/security、global/distribution、domain local/distribution、universal/distribution があります。この属性は、同期をとる Windows グループには自動的に設定されますが、Directory Server エントリーには、同期をとる前にこの属性を手動で設定する必要があります。

16.6.1. Windows グループタイプの概要

Active Directory には、セキュリティーとディストリビューションの 2 つの主要なグループタイプがあります。セキュリティーグループは、アクセス制御、リソースの制限、およびその他のパーミッションに対してポリシーを設定することができるため、Directory Server のグループには最も似ています。配信グループは、メール配信のためのグループです。これはさらに、グローバルグループおよびローカルグループに分けられます。Directory Server **ntGroupType** は、以下の 4 つのグループタイプをすべてサポートします。

- グローバル/セキュリティーの場合 (デフォルト) は **-2147483646**
- ドメインローカル/セキュリティーの場合は **-2147483644**
- 組み込みの場合は **-2147483643**
- 汎用/セキュリティーの場合は **-2147483640**
- グローバル/ディストリビューションの場合は **2**
- ドメインローカル/ディストリビューションの場合は **4**
- ユニバーサル/ディストリビューションの場合は **8**

16.6.2. Directory Server と Active Directory との間で同期されるグループ属性

Directory Server 属性および Active Directory 属性のサブセットのみが同期されます。これらの属性はハードコーディングされ、エントリーの同期方法に関わらず定義されます。Directory Server または Active Directory のいずれかにあるエントリーにあるその他の属性は、同期の影響を受けないままになります。

Directory Server エントリーおよび Active Directory グループエントリーで使用される属性の一部は同一です。これは通常、すべての LDAP サービスに共通する LDAP 標準で定義された属性です。これらの属性は、相互に正確に同期されます。表16.4「Directory Server と Active Directory との間のグループエントリー属性」は、Directory Server と Windows サーバーとの間で同じ属性を示しています。

同じ情報を定義する属性もありますが、属性やスキーマ定義の名前が異なります。これらの属性は Active Directory と Directory Server の間でマッピングされるため、1 つのサーバーの属性 A がもう 1 つのサーバーの属性 B として扱われます。同期の場合、これらの属性の多くは Windows 固有の情報に関連します。表16.3「Directory Server と Active Directory との間のグループエントリー属性のマッピング」は、Directory Server と Windows サーバーとの間で同じ属性を示しています。

Directory Server および Active Directory が一部のスキーマ要素を処理する方法の違いについての詳細は、「Red Hat Directory Server と Active Directory のグループスキーマの相違点」 を参照してください。

表16.3 Directory Server と Active Directory との間のグループエントリー属性のマッピング

Directory Server	Active Directory
cn	name
ntUserDomainID	name
ntGroupType	groupType
uniqueMember member	メンバー[a]

[a] Active Directory の ***Member*** 属性は、Directory Server の ***uniqueMember*** 属性に同期されます。

表16.4 Directory Server と Active Directory との間のグループエントリー属性

cn	o
description	ou
l	seeAlso
mail	

16.6.3. Red Hat Directory Server と Active Directory のグループスキーマの相違点

Active Directory は Directory Server と同じ基本的な X.500 オブジェクトクラスをサポートしますが、管理者が認識すべき非互換性がいくつかあります。

ネスト化されたグループ (グループに別のグループをメンバーとして追加) がサポートされ、Windows Synchronization では同期します。ただし、Active Directory では、ネストされたグループの設定として特定の制約が適用されます。たとえば、グローバルグループには、ドメインローカルグループをメン

バーとして追加することはできません。Directory Server にはローカルグループとグローバルグループの概念がないため、同期時に Active Directory の制約に違反する Directory Server 側でエントリーを作成できます。

16.6.4. Directory Server グループのグループ同期の設定

Directory Server グループが Active Directory に同期するには、グループエントリーに適切な同期属性を設定する必要があります。

コマンドラインで同期を有効にするには、必要な同期属性をエントリーに追加するか、これらの属性でエントリーを作成します。

同期には、以下の 3 つのスキーマ要素が必要です。

- **ntGroup** オブジェクトクラス。
- エントリーの Windows ID を与える ***ntUserDomainId*** 属性。
- ***ntGroupCreateNewGroup*** 属性は、同期プラグインに Active Directory 経由で Directory Server エントリーを同期するように通知します。

  ntGroupDeleteGroup 属性は任意ですが、Directory Server で削除される場合に、自動的に Active Directory ドメインからエントリーを削除するかどうかを設定します。

また、***ntGroupType*** 属性を追加することも推奨されます。この属性が指定されていない場合、グループはグローバルセキュリティーグループ (**ntGroupType:-2147483646**) として自動的に追加されます。

たとえば、**ldapmodify** を使用するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=Example Group,ou=Groups,dc=example,dc=com
changetype: modify
add: objectClass
objectClass:ntGroup
-
add: ntUserDomainId
ntUserDomainId: example-group
-
add: ntGroupCreateNewGroup
ntGroupCreateNewGroup: true
-
add: ntGroupDeleteGroup
ntGroupDeleteGroup: true
-
add: ntGroupType
ntGroupType: 2
```

エントリーには、多くの Windows やグループの属性を追加することができます。同期されたスキーマは、「Directory Server と Active Directory との間で同期されるグループ属性」 にすべてリストされます。**ntGroup** オブジェクトクラスに属する Windows 固有の属性については、『Red Hat Directory Server 11 Configuration, Command, and File Reference』 で詳しく説明されています。

16.6.5. Active Directory グループのグループ同期の設定

Windows ユーザー (Active Directory ドメインにあるユーザー) の同期は、同期合意で設定されます。

グループの同期を有効にするには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-winsync-agmt set --sync-groups="on" --suffix="dc=example,dc=com" example-agreement
```

グループの同期を無効にするには、**--sync-groups** オプションを **off** に設定します。

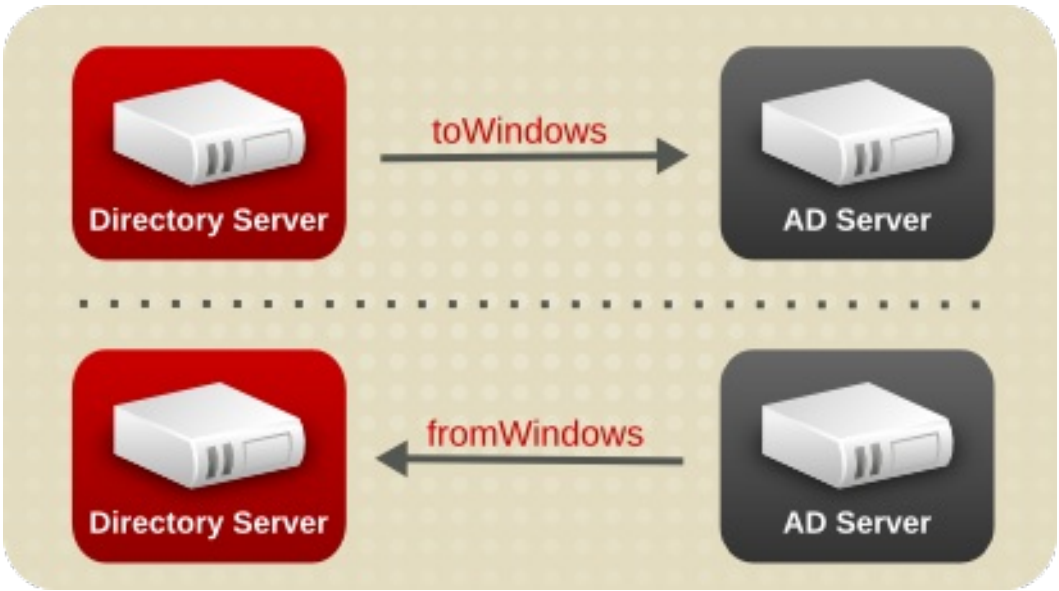
16.7. 一方向の同期の設定

図16.1「Active Directory - Directory Server の同期プロセス」 に示すように、同期はデフォルトで **双方向** となります。つまり、Active Directory の変更が Directory Server に送信され、Directory Server の変更が Active Directory に送信されることを意味します。

変更が一方向のみに送信される場合は、**一方向同期** を作成できます。これはサプライヤーとコンシューマーの関係と似ています。[1] マルチサプライヤーとは対照的です。

同期合意の追加属性 ***oneWaySync*** は、一方向の同期を有効にし、変更を送信する方向を指定します。使用できる値は ***fromWindows*** (Active Directory から Directory Server への同期の場合) および ***toWindows*** (Directory Server から Active Directory の同期の場合) です。この属性がない場合、同期は双方向になります。

図16.3 一方向の同期



同期プロセス自体は、双方向と一方向の同期にほぼ同じです。これは、同じ同期間隔と設定を使用します。唯一の違いは、同期情報の要求方法にあります。

Windows Active Directory から Directory Server への同期では、定期的な同期の更新間隔で、Directory Server が Active Directory Server に接続し、DirSync コントロールを送信して更新を要求します。ただし、Directory Server は、その側から変更やエントリーは送信されません。つまり、同期更新は、Active Directory の変更内容が Directory Server のエントリーに送信され、更新されることで設定されています。

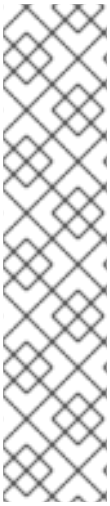
Directory Server から Active Directory 同期では、Directory Server は通常の更新で Active Directory サーバーにエントリー変更を送信しますが、Active Directory 側から更新を要求しないように DirSync 制御は含まれません。

--one-way-sync="*direction*" オプションを使用して、以下のいずれかの状況で一方向同期を有効にします。

1. 「ステップ 9: 同期用のデータベースの設定および同期合意の作成」 で新しい同期合意を作成する場合は、オプションを **dsconf repl-winsync-agmt create** コマンドに渡します。

2. 同期合意がすでに存在する場合は、合意を更新します。たとえば、AD から Directory Server への同期を設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-winsync-agmt set --one-way-sync="fromWindows" --suffix="dc=example,dc=com" example-agreement
```



注記

一方向同期を有効にすると、同期されていないサーバーで自動的に変更ができなくなる **わけではないため**、同期更新間の同期ピア間で不整合が生じる可能性があります。たとえば、一方向性同期は、Active Directory から Directory Server に行くように設定されているため、Active Directory が (実質的に) データサプライヤーとなります。Directory Server でエントリーを変更または削除すると、Directory Server 情報はその情報とは異なるため、これらの変更は Active Directory に引き継がれません。次の同期更新時に、編集内容は Directory Server で上書きされ、削除済みのエントリーが再追加されます。

データの不整合が発生するのを防ぐには、アクセス制御ルールを使用して、同期 **されていない** サーバーの同期サブツリー内のエントリーを編集または削除しないようにします。Directory Server のアクセス制御については、18章 *アクセス制御の管理*で説明しています。Active Directory の場合は、適切な Windows ドキュメントを参照してください。

一方向の同期はパスワードの同期には影響しません。同期方向が ***toWindows*** に設定されている場合でも、Active Directory サーバー上のパスワードを更新した後に、パスワードは Directory Server に送信されます。

16.8. WINDOWS 同期での複数のサブツリーおよびフィルターの設定

Windows Synchronization は、Directory Server (DS) と Active Directory (AD) のサブツリーの複数のペアとの間で同期するように作られています。フィルターを使用すると、サブツリーの配下にあるエントリーのみが同期されます。

Windows 同期における複数のサブツリー

複数のサブツリーのペア間で同期するには、Windows 同期合意の **winSyncSubtreePair** パラメーターに Directory Server サブツリーと Active Directory サブツリーを設定します。たとえば、複数の **ou=OU1,dc=DSexample,dc=com** および **ou=OU1,DC=ADexample,DC=com** サブツリーを設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-winsync-agmt set --subtree-pair="ou=OU1,dc=DSexample,dc=com:ou=OU1,DC=ADexample,DC=com" --suffix="dc=example,dc=com" example-agreement
```

winSyncSubtreePair が設定されていない場合は、代わりに **nsds7WindowsReplicaSubtree** AD サブツリーパラメーターと **nsds7DirectoryReplicaSubtree** DS サブツリーパラメーターが同期ターゲットチェックに使用されます。それ以外の場合は、この 2 つのパラメーターは無視されます。

Windows 同期のフィルター

以下のパラメーターで同期されるデータを選択するフィルターを設定できます。

- ***--win-filter*** は、Active Directory サーバーに追加のフィルターを設定します。
- ***--ds-filter*** パラメーターは、Directory Server に追加のフィルターを設定します。

以下の例では、***user*** 属性および ***group*** 属性が含まれるエントリーを **example_agreement** が同期するように設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-winsync-agmt \
    set --win-filter="(|(cn=*user*)(cn=*group*))" --ds-filter="(|(uid=*user*)(cn=*group*))" \
    example_agreement
```

16.9. ユーザーとグループの POSIX 属性の同期

すべての可能なユーザーと属性のサブセットが、Active Directory と Red Hat Directory Server の間で同期されます。一部の属性はマッピングされ、Active Directory と Directory Server スキーマには違いがあり、一部の属性が直接照合されます。同期される属性 (一致およびマップされた)「Directory Server と Active Directory との間で同期されるユーザー属性」および「Directory Server と Active Directory との間で同期されるグループ属性」にリスト表示されます。

デフォルトでは、これらの属性のみが同期されます。

その同期リストにない属性のタイプの 1 つは、POSIX 関連の属性です。Linux システムでは、システムユーザーおよびグループは POSIX エントリーとして識別され、LDAP POSIX 属性に必要な情報が含まれています。しかし、Windows ユーザーが同期すると、Windows アカウントであることを示す ***ntUser*** 属性および ***ntGroup*** 属性が自動的に追加されますが、POSIX 属性は同期されず (Active Directory エントリーに存在していても)、Directory Server 側でも POSIX 属性は追加されません。

POSIX Winsync API プラグインは、Active Directory エントリーと Directory Server エントリーとの間で POSIX 属性を同期します。

注記

すべての POSIX 属性 (***uidNumber***、***gidNumber***、および ***homeDirectory***) は、Active Directory エントリーと Directory Server エントリー間で同期されます。ただし、新しい POSIX エントリーまたは POSIX 属性が Directory Server の既存のエントリーに追加されると、**POSIX 属性のみが Active Directory に対応するエントリーと同期します**。POSIX オブジェクトクラス (ユーザーの場合は **posixAccount**、グループの場合は **posixGroup**) は Active Directory エントリーに追加されません。

16.9.1. POSIX 属性同期の有効化

Posix Winsync API プラグインはデフォルトで無効になっており、Active Directory ユーザーおよびグループのエントリーから対応する Directory Server エントリーに同期するように POSIX 属性に対して有効にする必要があります。

Posix Winsync API プラグインを有効にするには、以下を実行します。

1. プラグインを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin "cn=Posix Winsync
API,cn=plugins,cn=config" enable
```

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

16.9.2. Posix グループ属性の同期設定の変更

POSIX グループの属性とグループメンバーを Active Directory エントリーから対応する Directory Server のグループエントリーおよびユーザーエントリーに同期する方法を制御するために、複数のプラグイン属性を設定することができます。詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の該当するセクションを参照してください。

デフォルト設定はほとんどのデプロイメントに使用できますが、Active Directory 環境に応じて設定を変更できます。たとえば、ネスト化されたグループマッピングを有効にするには、次のコマンドを実行します。

1. 以下のコマンドを使用して、ネストされたグループマッピングを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin posix-winsync set --map-nested-grouping="true"
```

2. Directory Server を再起動して、新しい設定を読み込みます。

```
# dsctl instance_name restart
```

16.9.3. posixGroup エントリーの member 属性と uniqueMember 属性の値の不一致の修正

Directory Server および Active Directory (AD) の ***posixGroup*** エントリーで ***member*** 属性値および ***uniqueMember*** 属性値が一致しない場合は、**dsconf plugin posix-winsync fixup** コマンドを使用して問題を修正します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin posix-winsync fixup DN
```

このコマンドは、Directory Server で ***memberUid*** の値を再作成し、AD で定義された値に一致するように ***member*** 属性値および ***uniqueMember*** 属性値を自動的に変更します。

必要に応じて、***-f filter*** パラメーターをコマンドに渡して、コマンドが ***memberUid*** 属性を修正するエントリーを指定します。フィルターなしでは、コマンドは **inetuser**、**inetadmin**、および **nsmemberof** オブジェクトクラスが含まれるすべてのエントリーで動作します。

16.10. エントリーの削除および復元

ここでは、同期を有効にすることで、同期先の削除されたエントリーにどのような影響があるか、また復活したエントリーがどのように処理されるかについて説明します。

16.10.1. エントリーの削除

Active Directory ピアのすべての変更は、常に Directory Server と同期されます。つまり、Active Directory ドメイン上で Active Directory グループやユーザーアカウントが削除されると、その削除内容が Directory Server の同期ピアサーバーに自動的に同期して戻ってくるということです。

一方、Directory Server では、Directory Server アカウントが削除されると、Directory Server エントリーに ***ntUserDeleteAccount*** 属性または ***ntGroupDeleteGroup*** 属性が **true** に設定されている場合のみ、Active Directory の対応するエントリーが削除されます。

注記

Directory Server エントリーが Active Directory に初めて同期すると、Active Directory は自動的に一意の ID を割り当てます。次の同期間隔で、一意の ID が Directory Server エントリーに同期され、***ntUniqueId*** 属性として保存されます。一意の ID が Directory Server に同期する **前** に Active Directory で Directory Server エントリーを削除すると、このエントリーは Directory Server で削除 **されません**。Directory Server は ***ntUniqueId*** 属性を使用して、Active Directory に追加された変更を対応する Directory Server エントリーに識別し、同期します。その属性がないと、Directory Server は削除を認識しません。

Active Directory のエントリーを削除し、Directory Server で削除を同期するには、***ntUniqueId*** 属性が削除される前に、エントリーの作成後に ***winSyncInterval*** の長さ (デフォルトでは 5 分) 待ちます。

16.10.2. エントリーのレスキュー

削除済みのエントリーを Directory Server に戻すことができます。削除されたエントリーは **tombstone** エントリーと呼ばれます。Directory Server と Active Directory の間で同期されていた削除エントリーが Directory Server に再び追加されると、再開する Directory Server エントリーには元の属性と値がすべて含まれます。これは **tombstone reanimation** と呼ばれます。再取得されたエントリーには、エントリーの同期に使用された元の ***ntUniqueId*** 属性が含まれます。これは、この新規エントリーが tombstone エントリーである Active Directory サーバーに通知されます。

Active Directory は古いエントリーを再取得し、エントリーの元の一意の ID を保持します。

Active Directory エントリーの場合、tombstone エントリーが Directory Server 上で復活すると、元の Directory Server の属性がすべて保持され、復活した Active Directory エントリーにも含まれます。

16.11. 同期更新の送信

同期は、***winSyncInterval*** (Active Directory ドメインから変更内容を取得する場合) または***nsds5replicaupdateschedule*** 設定 (Directory Server から変更内容をプッシュする場合) で設定された頻度で行われます。デフォルトでは、変更は 5 分ごとに Active Directory から取得され、Directory Server からの変更がすぐに送信されます。

同期の更新は手動でトリガーできます。また、完全な再同期を行うことも可能で、Directory Server と Active Directory のすべてのエントリーを、あたかも新しいもののように送信したり、引き出したりすることができます。完全な再同期には、以前同期されていない既存の Directory Server エントリーが含まれます。

16.11.1. 手動増分同期の実行

通常の運用では、Directory Server のエントリーに行われた更新のうち、Active Directory に送信する必要のあるものはすべて changelog に集められ、増分更新の際に再生されます。

変更を手動で同期するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-winsync-agmt poke --
suffix="dc=example,dc=com" example-agreement
```

16.11.2. 完全同期の実行

データに大きな変更があった場合や、既存の Directory Server エントリーに同期属性を追加する場合

は、**再同期** を開始する必要があります。再同期は合計更新であり、同期されたサブツリーのコンテンツ全体が検証され、必要に応じて更新されます。再同期は changelog を使用せずに行われます。これは、レプリケーションのコンシューマーの初期化または再初期化に似ています。

16.11.2.1. コマンドラインを使用した完全同期の実行

コマンドラインで完全同期を開始するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-winsync-agmt init --suffix="suffix"
agreement_name
```

同期の状態を表示するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-winsync-agmt init-status --
suffix="suffix" agreement_name
```

16.11.2.2. Web コンソールを使用した完全同期の実行

完全同期を開始するには、以下を実行します。

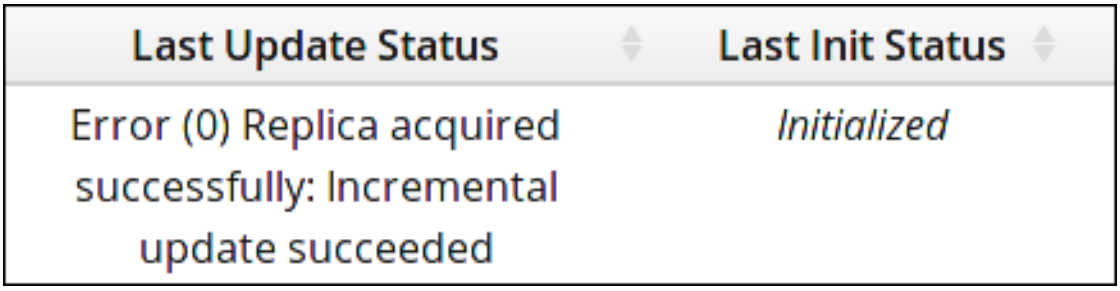
1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Replication** メニューを開き、**Winsync Agreements** エントリーを選択します。
4. 同期合意の横にある **Choose Action** メニューを開き、**Full Re-Synchronization** を選択します。

   再同期は、同期ピアのデータを削除しません。プロセスは、すべての更新を送受信して、新規または変更した Directory Server エントリーを追加します。たとえば、このプロセスは、**ntUser** オブジェクトクラスが追加される既存の Directory Server ユーザーを追加します。

Web コンソールで同期ステータスを表示するには、以下を実行します。

1. **Replication** メニューを開きます。
2. **Winsync Agreements** エントリーを選択します。

   同期が正常に完了すると、Web コンソールは **Last Update Status** 列に **Error (0) Replica acquired successfully: Incremental update succeeded** メッセージを表示します。



16.11.3. 同期スケジュールの設定

同期は 2 つの方法で機能します。Directory Server は、***nsds5replicaupdateschedule*** 属性を使用し

て、レプリケーションと同様に設定可能なスケジュールの Active Directory に更新を適用します。Directory Server は Active Directory をポーリングして変更の有無 (Active Directory サーバーが ***winSyncInterval*** 属性に設定されている頻度) を確認します。

デフォルトでは、Directory Server の更新スケジュールは常に同期されます。Active Directory の間隔は、5 分ごとに Active Directory をポーリングする間隔です。

Directory Server が更新を Active Directory に送信するために使用するスケジュールを変更するには、***nsds5replicaupdateschedule*** 属性を編集します。スケジュールは、24 時間クロックを使用して **HHMM** 形式の開始 (**SSSS**) および終了 (**EEEE**) 時間で設定されます。同期の更新スケジュールを設定する日は、**0** (日曜日) から **6** (土曜日) までです。

```
nsds5replicaupdateschedule: SSSS EEEE DDDDDDD
```

たとえば、日曜日、火曜日、木曜日、土曜日の正午から午後 2 時まで同期を実行するようにスケジュールを設定します。

```
nsds5replicaupdateschedule: 1200 1400 0246
```

注記

同期時間は真夜中を含むことはできないため、設定 **2300 0100** は有効ではありません。

Directory Server が Active Directory エントリーへの変更をチェックする頻度を変更するには、***winSyncInterval*** 属性をリセットします。この属性は秒単位で設定されるため、デフォルトの **300** は、Directory Server が Active Directory サーバーを 300 秒または 5 分間隔でポーリングすることを意味します。これを高い値に設定すると、ディレクトリーの検索に時間がかかり、パフォーマンスに影響する場合に便利です。

```
winSyncInterval: 1000
```

16.11.4. 同期接続の変更

同期合意について接続の 2 つの側面を変更することができます。

- バインドユーザー名およびパスワード (***nsDS5ReplicaBindDN*** および ***nsDS5ReplicaCredentials***)
- 接続方法 (***nsDS5ReplicaTransportInfo***)。

  nsDS5ReplicaTransportInfo は **LDAP** から **StartTLS** に (その逆も) しか変更できません。LDAPS への変更、または **LDAPS** からの変更はポート番号を変更することができないため、LDAP と LDAPS の切り替えにはポート番号の変更が必要となります。

以下に例を示します。

```
nsDS5ReplicaBindDN: cn=sync user,cn=Users,dc=ad1
nsDS5ReplicaCredentials: {DES}ffGad646dT0nnsT8nJOaMA==
nsDS5ReplicaTransportInfo: StartTLS
```

警告

Active Directory 同期ピアのポート番号を変更することはできません。そのため、標準ポートとセキュアでないポートの間で変更する必要があるため、標準/STARTTLS 接続と TLS 接続を切り替えることはできません。

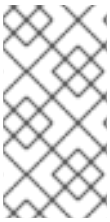
TLS に変更するには、同期合意を削除し、更新されたポート番号と新しいトランスポート情報を再度追加します。

16.11.5. 同期しているサブツリーから移動するエントリーの処理

同期合意は、Active Directory と Directory Server の両方において、またその 2 つの間で同期するサブツリーを定義します。スコープ **内**のエントリー (サブツリー) が同期され、他のエントリーは無視されます。

ただし、同期プロセスは実際にルート DN で開始し、同期のエントリーの評価を開始します。エントリーは、Active Directory の ***samAccount*** と Directory Server の ***uid*** 属性に基づいて相関します。同期プラグインは、(***samAccount/uid*** 関係に基づいて) エントリーが削除または移動されたために、同期されたサブツリーから削除された場合、その旨を通知します。これは、同期プラグインに対して、そのエントリーがもう同期されないことを示す信号です。

この問題は、同期プロセスで、移動したエントリーの処理方法を決定するための設定が必要となることです。対応するエントリーの削除、エントリー (デフォルト) の無視、またはエントリーの同期解除を 3 つのオプションがあります。



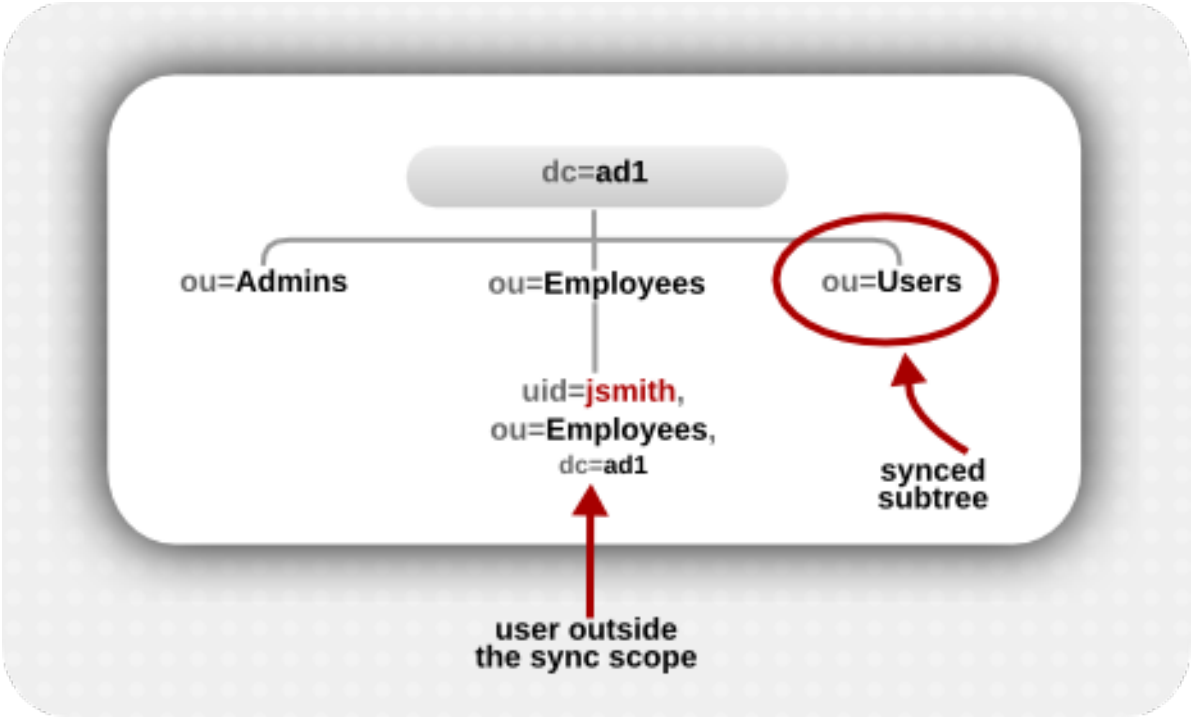
注記

これらの同期アクションは、Active Directory 側でエントリーが範囲外に移動した場合の **Directory Server 側での** 処理方法にのみ関連しています。エントリーが Directory Server 側で同期されたサブツリーからエントリーを移動しても、Active Directory エントリーには影響はありません。

Directory Server 9.0 のデフォルトの動作では、対応する Directory Server エントリーが削除されるようになりました。これは、Active Directory 側のエントリーが Directory Server 側に同期されていなかった場合でも該当します。Directory Server 9.1 以降、デフォルトの動作ではエントリーを無視して、何も実行しません。

たとえば、***samAccount*** ID が **jsmith** のユーザーは、Active Directory の **ou=Employees** サブツリーに作成されています。同期されたサブツリーは **ou=Users** であるため、**jsmith** ユーザーは Directory Server に同期されませんでした。

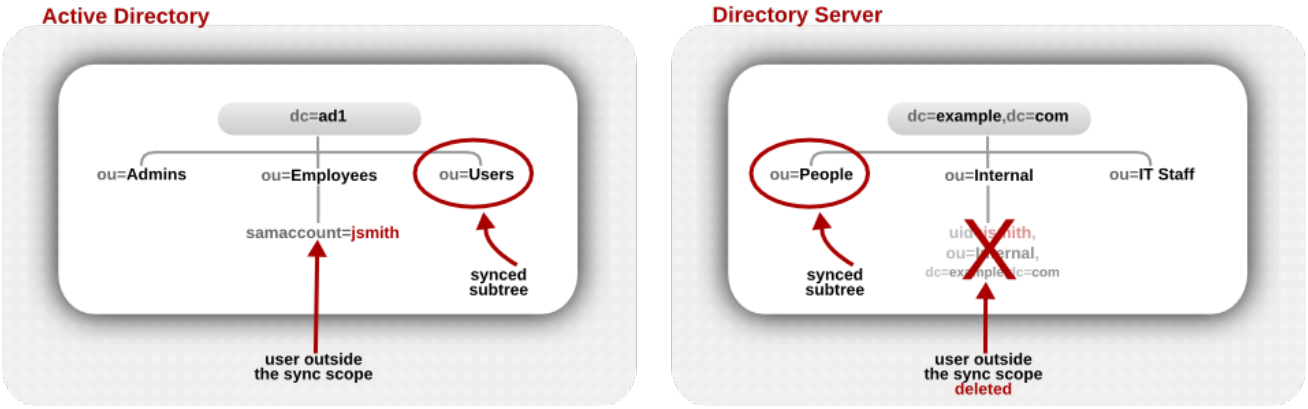
図16.4 Active Directory ツリー



バージョン 7.x および 8.x の Directory Server では、そのユーザーは同期されたサブツリーの外にあったため、同期は単にそのユーザーを無視していました。

Directory Server 9.0 以降、Directory Server はサブツリーの名前変更をサポートします。つまり、既存のエントリーはディレクトリーツリーのブランチ間で移動できるようになりました。同期プラグインは、Active Directory ツリーの中で、Directory Server のユーザー (***samAccount/uid*** 関係) に対応し、同期サブツリーの外にあるエントリーを、**意図的に** 同期サブツリーの外に移動させることを前提としています (基本的には名前の変更操作)。対応する Directory Server エントリーを削除する必要があることを前提とします。

図16.5 Active Directory と Directory Server ツリーの比較



この仮定は必ずしも正確なものではなく、特に同期サブツリーの外側に常に存在するユーザーエントリーの場合は注意が必要です。

同期合意の ***winSyncMoveAction*** 属性は、これらの移動したエントリーの処理方法を設定します。

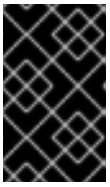
- **none** は何もしないため、同期した Directory Server エントリーが存在する場合は、同期するか、スコープ **内** に Active Directory エントリーを作成したりできます。同期された Directory Server エントリーが存在しない場合は、何も起こりません (Directory Server バージョン 9.1 以降では、これがデフォルトの動作です)。

- **unsync** は、Directory Server エントリーから同期関連の属性 (***ntUser*** または ***ntGroup***) を削除しますが、Directory Server エントリーはそのまま残されます。

重要

エントリーの同期を解除すると、Active Directory のエントリーが後から削除され、Directory Server のエントリーがそのまま残ってしまう危険性があります。これにより、特に Active Directory 側でエントリーを再作成するのに Directory Server エントリーを使用する場合などに、データが不整合になる可能性があります。

- **delete** は、Active Directory と同期していたかどうかに関わらず、Directory Server で該当するエントリーを削除します (これは 9.0 のデフォルト動作です)。



重要

対応する Active Directory エントリーを削除せずに Directory Server エントリーを削除することはありません。このオプションは、Directory Server 9.0 システムとの互換性でのみ利用できます。

デフォルトを変更する必要がある場合:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com repl-winsync-agmt --move-
action="action" --suffix="suffix" agreement_name
```

16.12. トラブルシューティング

同期が正常に行われない場合は、Windows のイベントログや Directory Server のエラーログで、問題がないか確認してください。

レプリケーションロギングを有効にして同期エラーを記録する
レプリケーションロギングを有効にすると、同期に関するより詳細な情報がエラーログに記録されます。レプリケーションログレベルは、同期コードからより詳細なログを生成します。同期トラフィックに関連するメッセージ (レプリケーショントラフィックと同じ) は、問題を診断するのに役立ちます。

ログレベルの設定に関する詳細は、「ログレベルの設定」を参照してください。

エラー #1: 同期後、ステータスは error 81 を返します。
同期ピアサーバーの 1 つは TLS 通信に対して適切に設定されていません。Directory Server のアクセスログファイルを調べ、Directory Server が接続試行を受け取っているかどうかを確認します。Directory Server のエラーログファイルには有用なメッセージがあります。

設定ミスの原因を突き止めるために、Directory Server への LDAPS 接続を試みます。この接続に失敗した場合は、すべての値 (ポート番号、ホスト名、IPv4/IPv6 アドレス、検索ベース、およびユーザー認証情報など) を確認して、それらのいずれかが問題かどうかを確認します。すべてが失敗した場合は、新しい証明書で Directory Server を再設定します。

Directory Server への LDAPS 接続に成功すると、設定が間違っていることが Active Directory にある可能性があります。Windows イベントログファイルでエラーメッセージを確認します。

注記

典型的な問題は、Windows 同期サービス証明書データベースが設定されたときに認証局が信頼できるものとして設定されていないことです。

エラー #2: エントリーは Active Directory のサブツリーから別のサブツリーに移動していますが、ユーザーは Directory Server の対応するサブツリーに移動していません。
これは、Active Directory の modrdn 操作と Directory Server のエントリーとの同期に関する既知の問題です。これを回避するには、Active Directory のエントリーを削除して、新しいサブツリーに追加します。削除と追加は、Directory Server ピアに対して適切に同期されます。

---

[1] コンシューマーとは異なり、同期されていないサーバーで変更は引き続き可能です。ACL を使用して、同期されていないサーバーでエントリーを編集または削除し、データの整合性を維持します。

第17章 SYNCREPL プロトコルを使用したコンテンツ同期の設定

Content Synchronization プラグインを使用すると、Directory Server は RFC 4533 に従って **SyncRepl** プロトコルをサポートします。このプロトコルにより、LDAP サーバーとクライアントは Red Hat Directory Server をソースとして使用し、ローカルデータベースを Directory Server の変更するコンテンツと同期させることができます。

SyncRepl プロトコルを使用するには、以下を実行します。

- Directory Server で **Content Synchronization** プラグインを有効にし、必要に応じてクライアントが Directory Server にバインドするために使用する新規ユーザーを作成します。アカウントには、ディレクトリー内のコンテンツを読み取るパーミッションが必要です。
- クライアントを設定します。たとえば、同期するサブツリーの検索ベースを設定します。詳細は、クライアントのドキュメントを参照してください。

17.1. コマンドラインを使用した CONTENT SYNCHRONIZATION プラグインの設定

コマンドラインを使用して **Content Synchronization** プラグインを設定するには、以下を実行します。

1. **Content Synchronization** プラグインでは、***nsuniqueid*** 属性をログに記録するのに **Retro Changelog** プラグインが必要です。

   a. Retro Changelog が有効になっているかどうかを確認するには、次のコマンドを実行します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin retro-changelog show
   ...
   nsslapd-pluginEnabled: off
   ```

   nsslapd-pluginEnabled パラメーターが **off** に設定されている場合、Retro Changelog は無効になります。有効にする場合は、「Retro Changelog プラグインの有効化」 を参照してください。

   b. ***nsuniqueid*** 属性を、Retro Changelog プラグインの設定に追加します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin retro-changelog set --attribute nsuniqueid:targetUniqueId
   ```

   c. 必要に応じて、パフォーマンスを向上させるために、以下の推奨事項を適用します。

      i. Retro Changelog のエントリーの最大有効期間を設定します。たとえば、2 日 (**2d**) を設定するには、以下のコマンドを実行します。

      ```
      # ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

      dn: cn=changelog5,cn=config
      changetype: modify
      replace: nsslapd-changelogmaxage
      nsslapd-changelogmaxage: 2d
      ```

ii. データを同期するバックエンドまたはサブツリーのクライアントアクセスを把握している場合は、**Retro Changelog** プラグインのスコープを制限します。たとえば、***cn=demo,dc=example,dc=com*** サブツリーを除外するには、以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin retro-
changelog set --exclude-suffix "cn=demo,dc=example,dc=com"
```

2. **Content Synchronization** プラグインを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin set --enabled on
"Content Synchronization"
```

3. デフォルトの Directory Server は、***oid=1.3.6.1.4.1.4203.1.9.1.1,cn=features,cn=config*** エントリーにアクセス制御命令 (ACI) を作成し、すべてのユーザーが **SyncRepl** プロトコルを使用できるようにします。

```
aci: (targetattr != "aci")(version 3.0; acl "Sync Request Control";
  allow( read, search ) userdn = "ldap:///all";)
```

必要に応じて、**SyncRepl** コントロールを使用して ACI を制限するように更新します。ACI の詳細は、「バインドルールの定義」を参照してください。

4. Directory Server を再起動します。

```
# dsctl instance_name restart
```

クライアントは、**SyncRepl** プロトコルを使用して、Directory Server とデータを同期できるようになりました。

第18章 アクセス制御の管理

本章では、Red Hat Directory Server の Access Control Instructions (ACI) を使用してエントリーへのアクセスを管理する方法を説明します。

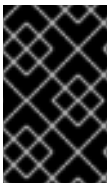
18.1. アクセス制御要件

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

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

- ディレクトリー全体、サブツリー、または特定のエントリーの場合
- 特定のユーザー、特定のユーザーまたはグループに属するすべてのユーザー、またはディレクトリー内のすべてのユーザーの場合
- IP アドレス、IP 範囲、または DNS 名などの特定の場所。

  ロードバランサーは場所固有のルールに影響を及ぼす可能性があることに注意してください。



重要

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

18.2. ACI 配置

Directory Server は、ディレクトリーエントリーの複数値の ***aci*** 操作属性に ACI を保存します。ACI を設定するには、***aci*** 属性を対応するディレクトリーエントリーに追加します。Directory Server は ACI を適用します。

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

  注記

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

- ACI を含むエントリーと、(子エントリーがある場合は) その下のすべてのエントリーへ。これにより、サーバーが指定のエントリーに対するアクセスパーミッションを評価すると、リクエストされたディレクトリー接尾辞と、エントリー自体の ACI との間のすべてのエントリーについて ACI を検証します。

  たとえば、ACI は **dc=example,dc=com** と **ou=People,dc=example,dc=com** のエントリーに設定されています。クライアントが ACI セットのない **uid=*user_name*,ou=People,dc=example,dc=com** オブジェクトにアクセスする場合、ディレクトリーサーバーは、まず **dc=example,dc=com** と **ou=People,dc=example,dc=com** からの

ACI でセットを作成します。Directory Server は、適用可能な ACI のリストを、ターゲットエントリーから一番上の接尾辞までボトムアップで作成します。ただし、このリストはセットとして考えてください。クライアントアプリケーションは、ACI 評価の順序を予測するべきではありません。

サーバーは、この初期セットから適用可能な ACI の最終セットを作成するリソースエントリーに一致する ACI を選択します。次に、最初に権限を拒否する ACI を評価します。**DENY** ACI が正常に評価された場合、操作は失敗します。**DENY** ACI が見つからない場合、Directory Server は **ALLOW** パーミッションを付与する ACI が存在するかどうかを確認します。ACI の少なくとも 1 つがアクセスを許可する場合、Directory Server はアクセスを許可します。**ALLOW** パーミッションを付与する ACI がない場合、Directory Server はアクセスを拒否し、操作は失敗します。

注記

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

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

注記

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

18.3. ACI 構造

aci 属性は以下の構文を使用します。

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

- ***target_rule*** は、アクセスを制御するためのエントリー、属性、またはエントリーと属性のセットを指定します。詳細については、「ターゲットの定義」 を参照してください。
- ***version 3.0*** は、ACI バージョンを識別する必須の文字列です。
- ***aci "ACL_name"*** は、ACI を説明する名前または文字列を設定します。
- ***permission_rule*** は、**read** または **write** など、どの権限が許可または拒否されるかを設定します。詳細については、「パーミッションの定義」 を参照してください。
- ***bind_rules*** は、アクセスを許可または拒否するために、バインド時に一致するルールを指定します。詳細については、「バインドルールの定義」 を参照してください。

注記

パーミッションとバインドルールのペアはアクセス制御ルールと呼ばれます。

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

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

18.4. ACI 評価

特定のエントリーへのアクセス権を評価するために、サーバーは、エントリー自体と、Directory Server に格納されている最上位のエントリーまでの親エントリーに存在する ACI のリストを作成します。ACI は、特定のインスタンス用のデータベース全体で評価されますが、異なるインスタンスもすべて評価されます。

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

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

注記

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

18.5. ACI の制限

ACI を設定すると、以下の制限が適用されます。

- ディレクトリーデータベースが複数のサーバーに分散されている場合は、ACI で使用できるキーワードに以下の制限が適用されます。
  - **groupdn** キーワードを使用したグループエントリーに依存する ACI は、グループエントリーと同じサーバーに置く必要があります。

    グループが動的の場合、グループのすべてのメンバーに、サーバーのエントリーが必要です。静的グループのメンバーエントリーは、リモートサーバーに配置できます。
  - **roledn** キーワードを使用したロール定義に依存する ACI は、ロール定義エントリーと同じサーバーにある必要があります。ロールを持つすべてのエントリーは、同じサーバーに配置する必要があります。

  ただし、ターゲットエントリーに保存されている値を、たとえば **userattr** キーワードを使用して、bind ユーザーのエントリーに保存されている値と一致させることができます。この場合、通常、バインドユーザーに ACI を格納するサーバーにエントリーがない場合でも、アクセスが評価されます。

  詳細は、「データベースリンクおよびアクセス制御評価」を参照してください。
- 以下の ACI キーワードでは、Class of Service (CoS) 属性などの仮想属性を使用することはできません。
  - **targetfilter**
  - **targattrfilters**

- **userattr**

詳細については、8章 エントリーの編成とグループ化 を参照してください。

- アクセス制御ルールは、ローカルサーバーでのみ評価されます。たとえば、ACI キーワードの LDAP URL にサーバーのホスト名を指定すると、URL は無視されます。

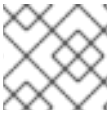
18.6. DIRECTORY SERVER がレプリケーショントポロジーで ACI を処理する方法

ACI はエントリーの ***aci*** 属性に保存します。したがって、ACI を含むエントリーが複製されたデータベースの一部である場合、ACI は複製されます。

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

18.7. コマンドラインを使用した ACI 管理

このセクションでは、コマンドラインを使用して ACI を管理する方法について説明します。



注記

Web コンソールで Directory Server ACI の管理はサポートされません。

18.7.1. ACI の表示

ldapsearch ユーティリティーを使用して、コマンドラインを使用して ACI を表示します。たとえば、**dc=example,dc=com** およびサブエントリーに設定された ACI を表示するには、以下のコマンドを実行します。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -x \
    -b "dc=example,dc=com" -s sub '(aci=*)' aci
```

18.7.2. ACI の追加

ldapmodify ユーティリティーを使用して ACI を追加します。以下に例を示します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="userPassword") (version 3.0; acl "Allow users updating their password";
 allow (write) userdn= "ldap:///self";)
```

18.7.3. ACI の削除

コマンドラインで ACI を削除するには、次を実行します。

1. エントリーに設定された ACI を表示します。「ACI の表示」を参照してください。
2. ACI を削除します。

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: delete
delete: aci
```

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
delete: aci
aci: (targetattr="userPassword") (version 3.0; acl "Allow users
 updating their password"; allow (write) userdn= "ldap:///self";)
```

属性の削除に関する詳細は 「エントリーからの属性の削除」を参照してください。

18.7.4. ACI の更新

コマンドラインを使用して ACI を更新するには、以下を実行します。

1. 既存の ACI を削除します。「ACI の削除」を参照してください。
2. 更新された設定で新しい ACI を追加します。「ACI の追加」を参照してください。

18.8. WEB コンソールを使用した ACI 管理

ここでは、Web コンソールで LDAP ブラウザーウィザードを使用してアクセス制御命令 (ACI) を管理するための基本について説明します。

18.8.1. LDAP ブラウザーでのアクセス制御命令の作成

Web コンソールで LDAP ブラウザーを使用して、Red Hat Directory Server (RHDS) エントリーのアクセス制御命令 (ACI) を作成および追加できます。

前提条件

- Web コンソールへのアクセス。
- Red Hat Directory Server に存在する親エントリー。

手順

1. Web コンソールにログインし、**Red Hat Directory Server** をクリックします。
2. Web コンソールが **Red Hat Directory Server** インターフェイスをロードしたら、**LDAP browser** をクリックします。
3. LDAP エントリーを選択し、Options メニューをクリックします。
4. ドロップダウンメニューから **ACIs** を選択します。

5. LDAP ブラウザーウィザードを使用して ACI を作成する場合、次の 2 つのオプションがあります。

    a. **Add ACI Wizard** をクリックして、ウィザードを使用して ACI を作成します。次の手順に進みます。

    b. **Add ACI Manually** をクリックし、テキストフィールドに指示を指定して **Save ACI** をクリックします。

6. ウィザードの手順に従い、各手順を完了したら **Next** ボタンをクリックします。

7. ACI を作成するには、ウィザードが生成したデータを確認し、**Add ACI** をクリックします。

8. ウィザードウィンドウを閉じるには、**Finish** ボタンをクリックします。

検証

- **Manage ACIs** ウィンドウに新しい ACI が表示されることを確認します。

18.8.2. LDAP ブラウザーでのアクセス制御命令の編集

Web コンソールで **LDAP Browser** の **Manage ACIs** ウィンドウを使用して、Red Hat Directory Server エントリーのアクセス制御命令 (ACI) を編集できます。

前提条件

- Web コンソールへのアクセス。
- Red Hat Directory Server に存在する親エントリー。

手順

1. Web コンソールにログインし、**Red Hat Directory Server** をクリックします。
2. Web コンソールが **Red Hat Directory Server** インターフェイスをロードしたら、**LDAP browser** をクリックします。
3. LDAP エントリーを選択し、Options メニューをクリックします。
4. ドロップダウンメニューから **ACIs** を選択します。
5. Options メニューをクリックし、**Edit ACI** を選択します。
6. テキストフィールドの命令を変更し、**Save ACI** をクリックします。

検証

- **Manage ACIs** ウィンドウで、変更した ACI を展開し、変更を確認します。

18.8.3. LDAP ブラウザーでのアクセス制御命令の削除

Web コンソールで **LDAP Browser** を使用して、Red Hat Directory Server エントリーのアクセス制御命令 (ACI) を削除できます。

前提条件

- Web コンソールへのアクセス。

- Red Hat Directory Server に存在する親エントリー。

手順

1. Web コンソールにログインし、**Red Hat Directory Server** をクリックします。
2. Web コンソールが **Red Hat Directory Server** インターフェイスをロードしたら、**LDAP browser** をクリックします。
3. LDAP エントリーを選択し、Options メニューをクリックします。
4. ドロップダウンメニューから **ACIs** を選択し、**Manage ACIs** ウィンドウを開きます。
5. 削除する ACI のノードオプションアイコンをクリックし、**Remove ACI** を選択します。
6. **Yes, I'm sure** のチェックボックスを選択し、**Delete ACI** ボタンをクリックします。

検証

- **Manage ACIs** ウィンドウで、削除した ACI が ACI リストに表示されないことを確認します。

18.9. ターゲットの定義

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

ACI では、以下の強調表示された部分がターゲットルールになります。

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

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

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

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

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

構文
ターゲットルールの一般的な構文は、以下のとおりです。

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

- ***keyword***: ターゲットの種類を設定します。「よく使用されるターゲットキーワード」を参照してください。
- ***comparison_operator***: 有効な値は **=** および **!=** で、ターゲットが式で指定されたオブジェクトであるかを示します。

警告

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

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

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

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

18.9.1. よく使用されるターゲットキーワード

管理者は、以下のターゲットキーワードを頻繁に使用します。

- **target**: 「ディレクトリーエントリーのターゲット」 を参照してください。
- **targetattr**: 「ターゲット属性」 を参照してください。
- **targetfilter**: 「LDAP フィルターを使用したエントリーと属性の対象」 を参照してください。
- **targattrfilters**: 「LDAP フィルターを使用した属性値のターゲット」 を参照してください。

18.9.1.1. ディレクトリーエントリーのターゲット

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

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

注記

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

例18.1 target キーワードの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (target = "ldap:///ou=People,dc=example,dc=com") (version 3.0;
 acl "Allow users to read and search attributes of own entry"; allow (search, read)
 (userdn = "ldap:///self");)
```

target キーワードでのワイルドカードの使用
* ワイルドカード文字ターゲットに複数のエントリーを使用できます。

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

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

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

例18.2 ワイルドカードを使用したディレクトリーエントリーのターゲット

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

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

以前のターゲットルールは、以下のような複数のエントリーと一致します。

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

重要

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

18.9.1.2. ターゲット属性

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

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

注記

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

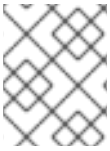
重要

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

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

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

式に設定された属性はスキーマに定義する必要があります。



注記

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

例18.3 targetattr キーワードの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "userPassword") (version 3.0;
 acl "Allow users updating own userPassword";
 allow (write) (userdn = "ldap:///self");)
```

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

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

警告

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

18.9.1.3. LDAP フィルターを使用したエントリーと属性の対象

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

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

フィルター式は、14章*ディレクトリーエントリーの検索*で説明されているように、標準の LDAP 検索フィルターです。

例18.4 targetfilter キーワードの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetfilter = "(|(department=Engineering)(department=Sales)")
 (version 3.0; acl "Allow HR updating engineering and sales entries";
 allow (write) (groupdn = "ldap:///cn=Human Resources,dc=example,dc.com");)
```

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

注記

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

targetfilter キーワードでのワイルドカードの使用

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

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

18.9.1.4. LDAP フィルターを使用した属性値のターゲット

アクセス制御を使用すると、属性の特定値を対象にできます。つまり、ある属性の値が ACI で定義されている基準を満たしていれば、その属性に対してパーミッションを付与したり、拒否したりすることができるのです。属性の値に基づいてアクセスを許可または拒否する ACI は、値ベースの ACI と呼ばれます。

注記

これは、**ADD** および **DEL** 操作にのみ適用されます。検索権限を持つユーザーは、特定の値で制限できません。

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

- 1 つの属性とフィルターの組み合わせが含まれる操作の場合:

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

- 複数の属性とフィルターの組み合わせのある操作の場合:

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

- 複数の属性とフィルターを組み合わせた 2 つの操作の場合。

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

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

以下では、フィルターを一致させる方法を説明します。

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

例18.5 targattrfilters キーワードの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targattrfilters="add=nsroledn:(!(nsroledn=cn=Admin)) &&
 telephoneNumber:(telephoneNumber=123*)") (version 3.0;
 acl "Allow adding roles and telephone";
 allow (add) (userdn = "ldap:///self");)
```

18.9.2. 詳細なターゲットキーキーワード

このセクションでは、頻繁に使用されないターゲットキーワードを説明します。

18.9.2.1. ソースおよび宛先 DN のターゲット

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

- ACI に設定される別のソースからエントリーを移動します。
- エントリーを ACI のセットとして別の宛先に移動するには、以下のコマンドを実行します。
- ソース DN から既存のエントリーを削除するには、以下を実行します。
- 宛先 DN に新規エントリーを追加するには、以下を行います。

例18.6 target_from および target_to キーワードの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (target_from="ldap:///uid=*,cn=staging,dc=example,dc=com")
 (target_to="ldap:///cn=People,dc=example,dc=com")
 (version 3.0; acl "MODDN from"; allow (moddn))
 userdn="ldap:///uid=user,dc=example,dc=com";)
```

注記

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

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

18.9.3. ターゲットルールの高度な使用方法

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

18.9.3.1. グループの作成およびメンテナンスへのパーミッションの委譲

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

例18.7 グループの作成およびメンテナンスへのパーミッションの委譲

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (target = "ldap:///cn=*,ou=Groups,dc=example,dc=com")
 (targattrfilters="add=objectclass:(|(objectclas=top)(objectclass=groupOfUniqueNames)))
 (targetattr="cn || uniqueMember || objectClass")
 (version 3.0; acl "example"; allow (read, search, write, add)
 (userdn = "ldap:///uid=test,ou=People,dc=example,dc=com");)
```

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

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

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

18.9.3.2. エントリーと属性の両方をターゲットに設定

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

例18.8 エントリーと属性の両方をターゲットに設定

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
```

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

18.9.3.3. フィルターに一致するエントリーの個別属性のターゲット設定

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

例18.9 フィルターに一致するエントリーの個別属性のターゲット設定

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "jpegPhoto || manager")
 (targetfilter = "(department=Engineering)") (version 3.0;
 acl "Allow engineering admins updating jpegPhoto and manager of department members";
 allow (write) (groupdn = "ldap:///cn=Engineering Admins,dc=example,dc=com");)
```

18.9.3.4. 単一ディレクトリーエントリーのターゲット設定

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

例18.10 単一ディレクトリーエントリーのターゲット設定

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=Engineering,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "ou || cn")
 (targetfilter = "(ou=Engineering)") (version 3.0;
 acl "Allow uid=user to search and read engineering attributes";
 allow (read, search) (userdn = "ldap:///uid=user,ou=People,dc=example,dc=com");)
```

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

重要

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

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

18.10. パーミッションの定義

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

ACI では、以下の強調表示された部分はパーミッションルールになります。

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

構文
パーミッションルールの一般的な構文は、以下のとおりです。

> *permission* (*rights*)

- ***permission***: ACI がパーミッションを許可するか、拒否するかを設定します。
- ***rights***: ACI が許可または拒否する権限を設定します。「ユーザーの権利」を参照してください。

> **例18.11 パーミッションの定義**
>
> **ou=People,dc=example,dc=com** エントリーに保存されているユーザーが、独自のエントリー内の全属性を検索し、表示するには、以下を実行します。
>
> ```
> # ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
> dn: ou=People,dc=example,dc=com
> changetype: modify
> add: aci
> aci: (target = "ldap:///ou=People,dc=example,dc=com") (version 3.0;
>  acl "Allow users to read and search attributes of own entry"; allow (search, read)
>  (userdn = "ldap:///self");)
> ```

18.10.1. ユーザーの権利

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

表18.1 ユーザーの権利

権利	説明
read	ユーザーがディレクトリーデータを読み込めるかどうかを設定します。このパーミッションは、LDAP の検索操作にのみ適用されます。

権利	説明
write	属性を追加、変更、または削除してユーザーがエントリーを変更できるかどうかを設定します。このパーミッションは、LDAP の **modify** および **modrdn** 操作に適用されます。
add	ユーザーがエントリーを作成できるかどうかを設定します。このパーミッションは、LDAP の **add** 操作にのみ適用されます。
削除	ユーザーがエントリーを削除できるかどうかを設定します。このパーミッションは、LDAP の **delete** 操作にのみ適用されます。
search	ユーザーがディレクトリーデータを検索できるかどうかを設定します。検索結果の一部として返されたデータを表示するには、**search** および **read** 権限を付与します。このパーミッションは、LDAP の検索操作にのみ適用されます。
compare	ユーザーが提供したデータとディレクトリーに保存されているデータを比較できるかどうかを設定します。**compare** 権限では、ディレクトリーは問い合わせに対して成功または失敗のメッセージを返しますが、ユーザーはエントリーや属性の値を見ることはできません。このパーミッションは、LDAP の比較操作にのみ適用されます。
selfwrite	ユーザーがグループから独自の DN を追加または削除できるかどうかを設定します。この権限は、グループ管理にのみ使用されます。
proxy	指定した DN が他のエントリーの権限でターゲットにアクセスできるかどうかを設定します。**proxy** 権限は ACL の範囲内で付与され、その権限が付与されたユーザーやグループは、Directory Server のユーザーとしてコマンドを実行することができます。プロキシー権限を特定のユーザーに制限することはできません。 セキュリティー上の理由から、**proxy** 権限を使用する ACI は、ディレクトリーの最も対象となるレベルに設定してください。
all	**proxy** 以外のすべての権限を設定します。

18.10.2. LDAP 操作に必要な権限

このセクションでは、実行を承認する LDAP 操作のタイプに応じて、ユーザーに付与する必要のある権限を説明します。

- エントリーの追加:
  - 追加するエントリーの **add** パーミッションを付与します。
  - エントリーの各属性の値に **write** パーミッションを付与します。この権限はデフォルトで付与されますが、**targattrfilters** キーワードを使用して制限できます。
- エントリーの削除:
  - 削除するエントリーの **delete** パーミッションを付与します。

    - エントリーの各属性の値に **write** パーミッションを付与します。この権限はデフォルトで付与されますが、**targattrfilters** キーワードを使用して制限できます。

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

18.10.3. アクセス制御と modrdn 操作

ACI を使用して **modrdn** 操作を明示的に拒否するには、関連するエントリーをターゲットにしますが、**targetattr** キーワードは省略します。たとえば、**cn=example,ou=Groups,dc=example,dc=com** グループを定義する ACI を追加するには、***cn*** 属性が含まれる **ou=people,dc=example,dc=com** のエントリーの名前を変更できません。

```
ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///cn=*,ou=people,dc=example,dc=com")
 (version 3.0; acl "Deny modrdn rights to the example group";
 deny(write) groupdn="ldap:///cn=example,ou=groups,dc=example,dc=com";)
```

18.11. バインドルールの定義

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

- DNS
- グループメンバーシップまたは割り当てられたロール

- エントリーがバインドする場所
- バインド時に使用する必要のある認証の種類
- バインドが実行される回数または日数

ACI では、以下の強調表示された部分はバインドルールになります。

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

構文
バインドルールの一般的な構文は以下のとおりです。

> *keyword comparison_operator* "*expression*"

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

18.11.1. 頻繁に使用されるバインドルール

管理者は、以下のバインドキーワードを使用します。

- **userDN**: 「ユーザーベースのアクセスの定義」 を参照してください。
- **groupdn**: 「グループベースのアクセスの定義」 を参照してください。

さらに、バインドルールはブール値演算子を使用して頻繁に組み合わせられます。詳細については、「ブール演算子を使用したバインドルールの組み合わせ」 を参照してください。

18.11.1.1. ユーザーベースのアクセスの定義

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

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

式の DN を以下のように設定します。

- DN: 「**userdn** キーワードでの DN の使用」 を参照してください。
- LDAP フィルター: 「LDAP フィルターで **userdn** キーワードの使用」を参照してください。
- **anyone** エイリアス: 「匿名アクセスの付与」 を参照してください。
- **all** エイリアス: 「認証済みユーザーへのアクセスの付与」 を参照してください。
- **self** エイリアス: 「ユーザーが空のエントリーにアクセスできるようにする」 を参照してください。

- **parent** エイリアス: 「ユーザーの子エントリーへのアクセス設定」 を参照してください。

注記

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

18.11.1.1.1. userdn キーワードでの DN の使用

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

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

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

例18.12 userdn キーワードでの DN の使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="manager") (version 3.0; acl "Allow uid=admin reading manager attribute";
 allow (search, read) userdn = "ldap:///uid=admin,ou=People,dc=example,dc=com";)
```

18.11.1.1.2. LDAP フィルターで userdn キーワードの使用

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

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

注記

LDAP フィルターは * ワイルドカードをサポートします。

例18.13 LDAP フィルターで userdn キーワードの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
```

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

18.11.1.1.3. 匿名アクセスの付与

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

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

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

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

例18.14 匿名アクセスの付与

認証のないすべてのユーザーが、**ou=People,dc=example,dc=com** エントリーで ***sn*** 属性、***givenName*** 属性、および ***telephoneNumber*** 属性を読み取りおよび検索できるようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="sn" || targetattr="givenName" || targetattr = "telephoneNumber")
 (version 3.0; acl "Anonymous read, search for names and phone numbers";
 allow (read, search) userdn = "ldap:///anyone";)
```

18.11.1.1.4. 認証済みユーザーへのアクセスの付与

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

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

例18.15 認証済みユーザーへのアクセスの付与

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=example,ou=Groups,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="member") (version 3.0;
 acl "Allow users to add/remove themselves from example group";
 allow (selfwrite) userdn = "ldap:///all";)
```

18.11.1.1.5. ユーザーが空のエントリーにアクセスできるようにする

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

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

例18.16 ユーザーが空のエントリーにアクセスできるようにする

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="userPassword") (version 3.0;
 acl "Allow users updating their password";
 allow (write) userdn = "ldap:///self";)
```

18.11.1.1.6. ユーザーの子エントリーへのアクセス設定

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

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

例18.17 ユーザーの子エントリーへのアクセス設定

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=user,ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="manager") (version 3.0;
 acl "Allow cn=user to update manager attributes";
 allow (write) userdn = "ldap:///parent";)
```

18.11.1.2. グループベースのアクセスの定義

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

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

- ***member***
- ***uniqueMember***
- ***memberURL***
- ***memberCertificateDescription***

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

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

式の DN を以下のように設定します。

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

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

注記

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

18.11.1.2.1. groupdn キーワードでの DN の使用

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

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

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

例18.18 groupdn キーワードでの DN の使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="manager") (version 3.0;
 acl "Allow example group to read manager attribute";
 allow (search, read) groupdn = "ldap:///cn=example,ou=Groups,dc=example,dc=com";)
```

18.11.1.2.2. LDAP フィルターで groupdn キーワードの使用

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

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

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

注記

LDAP フィルターは * ワイルドカードをサポートします。

例18.19 LDAP フィルターで groupdn キーワードの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="homePostalAddress") (version 3.0;
 acl "Allow manager=example setting homePostalAddress"; allow (write)
 userdn = "ldap:///dc=example,dc=com??sub?(manager=example)";)
```

18.11.2. さらなるバインドルール

このセクションでは、頻繁に使用されないバインドルールを説明します。

18.11.2.1. 値の一致に基づくアクセスの定義

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

userattr キーワードは、以下の構文を使用します。

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

詳細は、以下を参照してください。

- 「**USERDN** バインドタイプの使用」
- 「**GROUPDN** バインドタイプの使用」
- 「**ROLEDN** バインドタイプの使用」
- 「**SELFDN** バインドタイプの使用」
- 「**LDAPURL** バインドタイプの使用」

重要

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

継承の詳細は、「継承による **userattr** キーワードの使用」を参照してください。

18.11.2.1.1. USERDN バインドタイプの使用

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

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

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

例18.20 USERDN バインドタイプの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "telephoneNumber")
 (version 3.0; acl "Manager: telephoneNumber";
 allow (all) userattr = "manager#USERDN";)
```

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

18.11.2.1.2. GROUPDN バインドタイプの使用

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

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

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

例18.21 GROUPDN バインドタイプの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=Social Committee,ou=Groups,dc=example,dc=com
changetype: modify
add: aci
```

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

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

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

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

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

18.11.2.1.3. ROLEDN バインドタイプの使用

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

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

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

例18.22 ROLEDN バインドタイプの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (version 3.0; acl "Allow example role owners to read manager attribute";
 allow (search, read) userattr = manager#ROLEDN;)
```

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

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

18.11.2.1.4. SELFDN バインドタイプの使用

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

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

userattr *comparison_operator* "*attribute_name*#SELFDN"

例18.23 SELFDN バインドタイプの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=otp,dc=example,dc=com
changetype: modify
add: aci
aci: (target = "ldap:///ipatokenuniqueid=*,cn=otp,dc=example,dc=com")
 (targetfilter = "(objectClass=ipaToken)")(version 3.0;
 acl "token-add-delete"; allow (add) userattr = "ipatokenOwner#SELFDN";)
```

18.11.2.1.5. LDAPURL バインドタイプの使用

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

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

userattr *comparison_operator* "*attribute_name*#LDAPURL"

例18.24 LDAPURL バインドタイプの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*")
 (version 3.0; acl "Allow read,search "; allow (read,search)
 (userattr = "aciurl#LDAPURL);)
```

18.11.2.1.6. 継承による userattr キーワードの使用

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

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

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

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

以下に例を示します。

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

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

例18.25 継承による userattr キーワードの使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=Profiles,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="*") (version 3.0; acl "Profile access",
 allow (read,search) userattr="parent[0,1].owner#USERDN" ;)
```

18.11.2.2. 特定の IP アドレスまたは範囲からのアクセスの定義

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

ip キーワードでルールをバインドするには、以下の構文を使用します。

```
ip comparison_operator "IP_address_or_range"
```

例18.26 バインドルールでの IPv4 アドレス範囲の使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*") (version 3.0;acl "Deny 192.0.2.0/24"; deny (all)
 (userdn = "ldap:///anyone") and (ip != "192.0.2.");)
```

例18.27 バインドルールでの IPv6 アドレス範囲の使用

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*") (version 3.0;acl "Deny 2001:db8::/64"; deny (all)
 (userdn = "ldap:///anyone") and (ip != "2001:db8::");)
```

18.11.2.3. 特定のホストまたはドメインからアクセスの定義

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

警告

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

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

dns キーワードでルールをバインドするには、以下の構文を使用します。

```
dns comparison_operator "host_name_or_domain_name"
```

例18.28 特定のホストからのアクセスの定義

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*") (version 3.0;acl "Deny client.example.com"; deny (all)
 (userdn = "ldap:///anyone") and (dns != "client.example.com");)
```

例18.29 特定のドメインからアクセスの定義

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "*") (version 3.0;acl "Deny example.com"; deny (all)
 (userdn = "ldap:///anyone") and (dns != "*.example.com");)
```

18.11.2.4. 接続に一定レベルのセキュリティーの要求

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

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

ssf キーワードでルールをバインドするには、以下の構文を使用します。

```
ssf comparison_operator key_strength
```

以下の比較演算子を使用できます。

- **=** (等しい)
- **!** (等しくない)
- **<** (未満)
- **>** (超過)
- **<=** (以下)
- **>=** (以上)

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

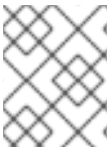
例18.30 接続に一定レベルのセキュリティーの要求

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr = "userPassword") (version 3.0;
 acl "Allow users updating own userPassword";
 allow (write) (userdn = "ldap:///self") and (ssf >= "128");)
```

18.11.2.5. 曜日の特定の日におけるアクセスの定義

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



注記

Directory Server はサーバー上で時間を使用して ACI を評価しますが、クライアントの時間ではありません。

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

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

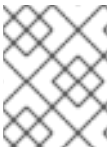
例18.31 特定の曜日にアクセスの付与

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (version 3.0; acl "Deny access on Saturdays and Sundays";
 deny (all)
 (userdn = "ldap:///uid=user,ou=People,dc=example,dc=com") and
 (dayofweek = "Sun,Sat");)
```

18.11.2.6. 特定の時刻におけるアクセスの定義

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



注記

Directory Server はサーバー上で時間を使用して ACI を評価しますが、クライアントの時間ではありません。

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

```
timeofday comparison_operator "time"
```

以下の比較演算子を使用できます。

- **=** (等しい)
- **!** (等しくない)
- **<** (未満)

- **>** (超過)
- **<=** (以下)
- **>=** (以上)

重要

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

例18.32 特定の時刻におけるアクセスの定義

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
 aci: (version 3.0; acl "Deny access between 6pm and 0am";
 deny (all)
 (userdn = "ldap:///uid=user,ou=People,dc=example,dc=com") and
 (timeofday >= "1800" and timeofday < "2400");)
```

18.11.2.7. 認証方法に基づいたアクセスの定義

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

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

```
authmethod comparison_operator "authentication_method"
```

以下の認証方法を設定できます。

- **none**: 認証は不要で、匿名のアクセスを表します。これはデフォルトになります。
- **simple**: クライアントは、ディレクトリーにバインドするユーザー名とパスワードを提供する必要があります。
- **SSL**: クライアントは、データベース、スマートカード、または他のデバイスのいずれかで TLS 証明書を使用してディレクトリーにバインドする必要があります。証明書ベースの認証の詳細は、「証明書ベースのクライアント認証の使用」を参照してください。
- **SASL**: クライアントは、Simple Authentication and Security Layer (SASL) 接続を介してディレクトリーにバインドする必要があります。bind ルールでこの認証方法を使用する場合は、**EXTERNAL** などの SASL メカニズムも指定します。

例18.33 EXTERNAL SASL 認証方法を使用した接続でのみアクセスのみの有効化

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (version 3.0; acl "Deny all access without certificate"; deny (all)
 (authmethod = "none" or authmethod = "simple");)
```

18.11.2.8. ロールに基づくアクセスの定義

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

注記

Red Hat は、ロールの代わりにグループを使用することを推奨します。ロールおよび制限の詳細は、「ロールの概要」を参照してください。

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

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

注記

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

例18.34 ロールに基づくアクセスの定義

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

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (targetattr="manager") (version 3.0;
 acl "Allow manager role to update manager attribute";
 allow (search, read) roledn = "ldap:///cn=Human Resources,ou=People,dc=example,dc=com";)
```

18.11.3. ブール演算子を使用したバインドルールの組み合わせ

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

バインドルールとブール演算子を組み合わせた構文は以下の通りです。

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

例18.35 ブール演算子を使用したバインドルールの組み合わせ

cn=Administrators,ou=Groups,dc=example,com および **cn=Operators,ou=Groups,dc=example,com** の両方のグループのメンバーであるユーザーが、**ou=People,dc=example,dc=com** のエントリーを読み取り、検索、追加、更新、および削除できるように設定するには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: ou=People,dc=example,dc=com
changetype: modify
add: aci
aci: (target="ldap:///ou=People,dc=example,dc=com") (version 3.0;
 acl "Allow members of administrators and operators group to manage users";
 allow (read, search, add, write, delete)
 groupdn = "ldap:///cn=Administrators,ou=Groups,dc=example,com" AND
 groupdn = "ldap:///cn=Operators,ou=Groups,dc=example,com";)
```

Directory Server によるブール値演算子の評価方法

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

- 左から右へのすべての式。

  以下の例では、***bind_rule_1*** が最初に評価されます。

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

- 一番内側から外側に向かって、親表現が優先されます。

  以下の例では、***bind_rule_2*** を最初に評価し、次に ***bind_rule_3*** を評価します。

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

- **AND** または **OR** 演算子の前に **NOT**。

  以下の例では、***bind_rule_2*** が最初に評価されます。

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

  AND および **OR** 演算子には優先順位がありません。

18.12. エントリーのアクセス権利の確認 (GET EFFECTIVE RIGHTS)

ユーザーが特定のエントリー内の属性に対して持っているアクセス権を見つけることは、管理者がアクセス権限を見つけて制御するのための便利な方法を提供します。

Get effective rights は、ディレクトリー検索を拡張して、ユーザーが特定のエントリーに対してどのようなアクセス権 (読み取り、検索、書き込みと自己書き込み、追加、削除など) を持っているかを表示する方法です。

Directory Server では、通常のユーザーは、表示できるエントリーに対する権限を確認して、他の人による個人エントリーへのアクセスを確認することができます。Directory Manager は、あるユーザーが別のユーザーに属する権限を確認できます。

エントリーの実効権限を確認することが便利な状況は 2 つあります。

- 管理者は、ディレクトリーに対するアクセス制御手順をより適切に整理するために、get effective rights コマンドを使用できます。あるグループのユーザーが閲覧または編集できる内容を、別のグループと比較して制限する必要があることがよくあります。たとえば、**QA Managers** グループのメンバーには、***manager*** や ***salary*** などの属性を検索および読み取る権利がありますが、**HR Group** メンバーのみが変更または削除する権限を持ちます。ユーザーまたはグループの実効権限を確認する方法は、適切なアクセス制御が有効であることを確認する方法です。

- ユーザーは、get effective rights コマンドを実行して、個人エントリーで表示または変更できる属性を確認することができます。たとえば、ユーザーは ***homePostalAddress*** や ***cn*** などの属性にアクセスできますが、***manager*** 属性および ***salary*** 属性への読み取りアクセスしかできません。

getEffectiveRights 検索には、エンティティーが 3 つあります。1 つ目は、**getEffectiveRights** 検索操作の実行時に認証済みエントリーである **requester** です。2 つ目は、権限の評価に使用される **サブジェクト** で、GER 制御の **承認 DN** として定義されます。3 つ目は、要求の検索ベース、検索フィルター、属性リストで定義される **ターゲット** です。

18.12.1. Get Effective Rights 検索表示される権限

コマンドラインでの get effective rights の検索時に、要求側がターゲットエントリーに対して必要とする権限が表示されます。

任意のエントリーに指定できるアクセス権には、2 種類があります。1 つ目は上位の権利で、**エントリー自体に対する権利** です。つまり、ユーザー A がユーザー B のエントリー全体に対して実行できる操作の種類を意味します。第 2 レベルのアクセス権はより詳細で、ユーザー A が **ある属性に対してどのような権利** を有しているかを示しています。この場合、ユーザー A は同じエントリーで異なる属性のアクセスパーミッションがある可能性があります。ユーザーに許可されるアクセス制御は、そのエントリーに対する **有効な** 権限です。

以下に例を示します。

```
entryLevelRights: vadn
attributeLevelRights: givenName:rscWO, sn:rscW, objectClass:rsc, uid:rsc, cn:rscW
```

表18.2「エントリーの権限」および 表18.3「属性権」は、エントリーおよび属性へのアクセス権限をそれぞれ表示し、get effective rights 検索で返されます。

表18.2 エントリーの権限

パーミッション	説明
a	エントリーを追加します。
d	このエントリーを削除します。
n	DN の名前を変更します。
v	エントリーを表示します。

表18.3 属性権

パーミッション	説明
r	読み取り。
s	検索。
w	書き込み (**mod-add**)。
o	抹消 (**mod-del**)。削除に類似しています。
c	比較。
W	自己書き込み。
O	自己削除。

18.12.2. Get Effective Rights 検索の形式

Get effective rights (GER と呼ばれることもあります) は、拡張ディレクトリー検索です。GER パラメーターは、**-E** オプションを定義して、**ldapsearch** コマンドで LDAP コントロールを渡します。(**-E** オプションなしで **ldapsearch** を実行すると、get effective rights 情報なしに、エントリーが通常通りに返されます。)

```
# ldapsearch -x -D bind_dn -W -p server_port -h server_hostname -E
[!]1.3.6.1.4.1.42.2.27.9.5.2=:GER_subject (searchFilter) attributeList
```

- **-b** は、GER サブジェクトの検索に使用されるサブツリーまたはエントリーのベース DN です。

  検索ベースが特定のエントリー DN である場合や、1 つのエントリーのみが返される場合、結果には要求元がその特定のエントリーに対して所有している権利が表示されます。検索ベースの下にある複数のエントリーがフィルターと一致する場合、検索は一致するすべてのエントリーを返し、各エントリーに対する要求側の権限で返します。

- **1.3.6.1.4.1.42.2.27.9.5.2** は、get effective rights control の OID です。

- 感嘆符 (**!**) は、サーバーがこの制御 (**!**) をサポートしていない場合に、検索操作でエラーを返すか、無視して通常通りの検索を行うか (nothing) を指定します。

- **GER_subject** は、権限を確認するユーザーです。**GER_subject** を空白 (**dn:**) のままにすると、匿名ユーザーの権限が返されます。

- 任意の **attributeList** は、Get Effective Rights 結果を指定された属性またはオブジェクトクラスに制限します。通常の **ldapsearch** の場合のように、***mail*** などの特定の属性を指定できます。属性が表示されない場合は、エントリーの present 属性がすべて返されます。アスタリスク (*) を使用すると、エントリーに対して可能なすべての属性の権限が返されます (既存の属性と存在しない属性の両方)。プラス記号 (**+**) を使用すると、エントリーの操作属性が返されます。特定の属性の権限を確認するための例は、「Non-Existent 属性の Get Effective Rights 検索の例」 および 「特定の属性またはオブジェクトクラスの Get Effective Rights 検索の例」 に記載されています。

get effective rights 検索の要点は、GER 対象者 (**-E**) が検索対象者 (**-b**) に対してどのような権利を持っているかを確認できることです。get effective rights 検索は通常の **ldapsearch** で、検索パラメーター

に一致するエントリーを検索し、その情報を返します。get effective rights オプションは、これらの検索結果に追加の情報を加え、特定のユーザーがそれらの検索結果に対してどのような権利を持っているかを示します。その GER サブジェクトユーザーは、リクエスター (**-D** が **-E** と同じ) でも、別のユーザーでも構いません。

要求元が (Directory Manager ではなく) 一般ユーザーである場合、要求元は GER サブジェクトが要求元独自のエントリーに存在することのみを確認できます。つまり、John Smith が Babs Jensen の持つ有効な権利を確認するために要求を実行した場合、John Smith は Babs Jensen が自分のエントリーで持つ有効な権利しか得ることができません。他のエントリーはすべて、有効権限が不十分なアクセスエラーを返します。

get effective rights 検索を実行する際の通常ユーザーには、一般的な 3 つのシナリオがあります。

- ユーザー A は、他のディレクトリーエントリーに対する権利を確認します。
- ユーザー A は、自身のエントリーに必要な権限をチェックします。
- ユーザー A は、ユーザー B がユーザー A のエントリーに対して持っている権利をチェックします。

get effective rights 検索には、属性の権利を確認するための柔軟な方法がいくつかあります。

18.12.3. GER 検索の例

GER 検索を実行する方法は複数あります。返される情報のタイプや、検索されるエントリーおよび属性のタイプによって異なります。

18.12.3.1. アクセス権限の確認に関する一般的な例

効果的な権利検索のための一般的なシナリオとして、一般ユーザーが自身の個人的なエントリーにどのような変更を加えることができるかを判断することがあります。

たとえば、Ted Morris は、エントリーに対する権限を確認します。**-D** と **-E** オプションの両方により、そのエントリーは要求元として付与されます。**-b** オプションには、個人エントリーを確認するため、その DN も含まれます。

例18.36 個人の権利の確認 (ユーザー A からユーザー A)

```
# ldapsearch -x -p 389 -h server.example.com -D "uid=tmorris,ou=people,dc=example,dc=com" -
W -b "uid=tmorris,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=tmorris,ou=people,dc=example,dc=com' "(objectClass=*)"

 dn: uid=tmorris,ou=People,dc=example,dc=com
 givenName: Ted
 sn: Morris
 ou: IT
 ou: People
 l: Santa Clara
 manager: uid=jsmith,ou=People,dc=example,dc=com
 roomNumber: 4117
 mail: tmorris@example.com
 facsimileTelephoneNumber: +1 408 555 5409
 objectClass: top
 objectClass: person
 objectClass: organizationalPerson
 objectClass: inetOrgPerson
```

```
uid: tmorris
cn: Ted Morris
userPassword: {SSHA}bz0uCmHZM5b357zwrCUCJs1IOHtMD6yqPyhxBA==
entryLevelRights: v
attributeLevelRights: givenName:rsc, sn:rsc, ou:rsc, l:rsc, manager:rsc, roomNumber:rscwo,
mail:rscwo, facsimileTelephoneNumber:rscwo, objectClass:rsc, uid:rsc, cn:rsc, userPassword:wo
```

Ted Morris は、たとえば、管理職であったり、IT や人事など、他のユーザーのエントリーを編集しなければならない部署で働いていたりします。この場合、例18.37「別のユーザーの権限を個人的に確認 (ユーザー A からユーザー B へ)」のように、Ted (**-D**) が Dave Miller のエントリー (**-b**) に対する自分の権利 (**-E**) を確認しているように、他のユーザーのエントリーに対して自分がどのような権利を持っているかを確認したい場合があります。

例18.37 別のユーザーの権限を個人的に確認 (ユーザー A からユーザー B へ)

```
# ldapsearch -p 389 -h server.example.com -D "uid=tmorris,ou=people,dc=example,dc=com" -W -
b "uid=dmiller,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=tmorris,ou=people,dc=example,dc=com' "(objectClass=*)"

dn: uid=dmiller,ou=People,dc=example,dc=com
...
entryLevelRights: vad
attributeLevelRights: givenName:rscwo, sn:rscwo, ou:rscwo, l:rscwo, manager:rsc,
roomNumber:rscwo, mail:rscwo, facsimileTelephoneNumber:rscwo, objectClass:rscwo, uid:rscwo,
cn:rscwo, userPassword:rswo
```

すべての属性について、Ted Morris は、Dave Miller のエントリーへのパーミッションを読み取り、検索、比較、修正、および削除するパーミッションがあります。これらの結果は、Ted Morris が自分のエントリーへのアクセスをチェックしたときに返されたものとは異なります。Ted Morris は個人的に、これらの属性のほとんどに対して読み取り、検索、比較の権利しか持っていなかったからです。

Directory Manager には、あるユーザーが別のユーザーのエントリーに対する権限をチェックする機能があります。例18.38「Directory Manager での別のユーザーに対する (User A からユーザー B に対する) 権利の確認」では、Directory Manager が、マネージャーである Jane Smith (**-E**) が部下である Ted Morris (**-b**) に対して持っている権限をチェックしています。

例18.38 Directory Manager での別のユーザーに対する (User A からユーザー B に対する) 権利の確認

```
# ldapsearch -p 389 -h server.example.com -D "cn=Directory Manager" -W -b
"uid=tmorris,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=jsmith,ou=people,dc=example,dc=com' "(objectClass=*)"

dn: uid=tmorris,ou=People,dc=example,dc=com
...
entryLevelRights: vadn
attributeLevelRights: givenName:rscwo, sn:rscwo, ou:rscwo, l:rscwo, manager:rscwo,
roomNumber:rscwo, mail:rscwo, facsimileTelephoneNumber:rscwo, objectClass:rscwo, uid:rscwo,
cn:rscwo, userPassword:rscwo
```

管理者のみが、異なるユーザーがエントリーにある実効権限を取得することができます。Ted Morris が Dave Miller のエントリーを判定しようとすると、アクセスエラーが不十分になります。

```
# ldapsearch -p 389 -h server.example.com -D "uid=dmiller,ou=people,dc=example,dc=com" -W -b
"uid=tmorris,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=tmorris,ou=people,dc=example,dc=com' "(objectClass=*)"

ldap_search: Insufficient access
ldap_search: additional info: get-effective-rights: requester has no g permission on the entry
```

ただし、一般ユーザーは get effective rights 検索を実行して、別のユーザーが個人エントリーに対して持っている権利を確認することができます。例18.39「個人のエントリーに対する他ユーザーの権利の確認」では、Ted Morris は、Dave Miller が Ted Morris のエントリーに対してどのような権利を持っているかを確認します。

例18.39 個人のエントリーに対する他ユーザーの権利の確認

```
# ldapsearch -p 389 -h server.example.com -D "uid=tmorris,ou=people,dc=example,dc=com" -W -
b "uid=tmorris,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=dmiller,ou=people,dc=example,dc=com' "(objectClass=*)"

dn: uid=tmorris,ou=people,dc=example,dc=com
...
entryLevelRights: v
attributeLevelRights: givenName:rsc, sn:rsc, ou:rsc, l:rsc,manager:rsc, roomNumber:rsc, mail:rsc,
facsimileTelephoneNumber:rsc, objectClass:rsc, uid:rsc, cn:rsc, userPassword:none
```

この場合、Dave Miller は、エントリーの DN を閲覧する権利と、***ou***、***givenName***、***l***、およびその他の属性を読み取り、検索し、比較する権利を有しており、***userPassword*** 属性については権利を有していません。

18.12.3.2. Non-Existent 属性の Get Effective Rights 検索の例

デフォルトでは、値を持たないエントリーの属性には情報は提供されません。たとえば、***userPassword*** の値が削除された場合、上記のエントリーで将来的に有効な権利を検索しても、自己書き込みおよび自己削除の権利が許可されていても、***userPassword*** に対する有効な権利は返されません。

get effective rights 検索でアスタリスク (*) を使用すると、エントリーに設定されていない属性も含めて、そのエントリーで利用可能なすべての属性が返されます。

例18.40 Non-Existent 属性の有効な権限を返す

```
# ldapsearch -D "cn=Directory Manager" -W -b "uid=scarter,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com' "(objectclass=*)" "*"

dn: uid=scarter,ou=People,dc=example,dc=com
givenName: Sam
telephoneNumber: +1 408 555 4798
sn: Carter
ou: Accounting
ou: People
l: Sunnyvale
```

```
manager: uid=dmiller,ou=People,dc=example,dc=com
roomNumber: 4612
mail: scarter@example.com
facsimileTelephoneNumber: +1 408 555 9700
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: scarter
cn: Sam Carter
userPassword: {SSHA}Xd9Jt8g1UsHC8enNDrEmxj3iJPKQLItlDYdD9A==
entryLevelRights: vadn
attributeLevelRights: objectClass:rscwo, aci:rscwo, sn:rscwo, cn:rscwo, description:rscwo,
seeAlso:rscwo, telephoneNumber:rscwo, userPassword:rscwo, destinationIndicator:rscwo,
facsimileTelephoneNumber:rscwo, internationaliSDNNumber:rscwo, l:rscwo, ou:rscwo,
physicalDeliveryOfficeName:rscwo, postOfficeBox:rscwo, postalAddress:rscwo,
postalCode:rscwo, preferredDeliveryMethod:rscwo, registeredAddress:rscwo, st:rscwo,
street:rscwo, teletexTerminalIdentifier:rscwo, telexNumber:rscwo, title:rscwo, x121Address:rscwo,
audio:rscwo, businessCategory:rscwo, carLicense:rscwo, departmentNumber:rscwo,
displayName:rscwo, employeeType:rscwo, employeeNumber:rscwo, givenName:rscwo,
homePhone:rscwo, homePostalAddress:rscwo, initials:rscwo, jpegPhoto:rscwo, labeledUri:rscwo,
manager:rscwo, mobile:rscwo, pager:rscwo, photo:rscwo, preferredLanguage:rscwo, mail:rscwo,
o:rscwo, roomNumber:rscwo, secretary:rscwo, uid:rscwo,x500UniqueIdentifier:rscwo,
userCertificate:rscwo, userSMIMECertificate:rscwo, userPKCS12:rscwo
```

secretary など、エントリーに使用できる属性はすべて、その属性が存在しない場合でも表示されます。

18.12.3.3. 特定の属性またはオブジェクトクラスの Get Effective Rights 検索の例

属性関連の GER 検索を追加で取り込むと、特定の属性への権限を検索し、属性セットを検索し、エントリーに設定したオブジェクトクラスのいずれかで使用できる属性をすべて表示することができます。

「Get Effective Rights 検索の形式」 のフォーマット例に記載されているオプションの 1 つは **attributeList** です。特定属性のみの実効権限を返すには、search コマンドの最後に、属性をスペースで区切って指定します。

例18.41 特定の属性に対する get effective rights の結果

```
# ldapsearch -D "cn=Directory Manager" -W -b "uid=scarter,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com' "(objectclass=*)" cn
mail initials

dn: uid=scarter,ou=People,dc=example,dc=com
cn: Sam Carter
mail: scarter@example.com
entryLevelRights: vadn
attributeLevelRights: cn:rscwo, mail:rscwo, initials:rscwo
```

例18.41「特定の属性に対する get effective rights の結果」 の ***initials*** 属性のように、**attributeList** に存在しない属性を指定することで、アスタリスクを使用してすべての属性をリストアップするのと同様に、利用可能な権利を確認することができます。

Directory Manager は、特定のオブジェクトクラスで利用可能なすべての属性の権限をリスト表示することもできます。このオプションの形式は **attribute@objectClass** です。これにより、2 つのエントリーが返されます。1 つ目は指定された GER サブジェクトのエントリー、2 つ目はオブジェクトクラスのテンプレートエントリーです。

例18.42 オブジェクトクラス内の属性に対する get effective rights 結果

```
# ldapsearch -D "cn=Directory Manager" -W -b "uid=scarter,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com' "(objectclass=*)"
uidNumber@posixAccount
...
dn: cn=template_posixaccount_objectclass,uid=scarter,ou=people,dc=example,dc=com
uidnumber: (template_attribute)
entryLevelRights: v
attributeLevelRights: uidNumber:rsc
```

注記

検索形式 **attribute@objectClass** は、要求元 (**-D**) が Directory Manager の場合のみ利用できます。

特定の属性の代わりにアスタリスク (*) を使用すると、指定された GER サブジェクトのすべての属性 (present と non-existent) と、オブジェクトクラステンプレートの属性の全リストが返されます。

例18.43 オブジェクトクラスのすべての属性に対する get effective rights 結果

```
# ldapsearch -D "cn=Directory Manager" -W -b "uid=scarter,ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com' "(objectclass=*)"
*@posixaccount
...
dn: cn=template_posixaccount_objectclass,uid=scarter,ou=people,dc=example,dc=com
objectClass: posixaccount
objectClass: top
homeDirectory: (template_attribute)
gidNumber: (template_attribute)
uidNumber: (template_attribute)
uid: (template_attribute)
cn: (template_attribute)
entryLevelRights: v
attributeLevelRights: cn:rsc, uid:rsc, uidNumber:rsc, gidNumber:rsc, homeDirectory:rsc,
objectClass:rsc, userPassword:none, loginShell:rsc, gecos:rsc, description:rsc, aci:rsc
```

18.12.3.4. 存在しないエントリーの get effective rights 検索の例

管理者は、既存のアクセス制御ルールに基づいて、特定のユーザー (**jsmith**) が存在しないユーザーにどのような権限を確認したい場合があります。存在しないエントリーをチェックする場合、サーバーはそのサブツリー内にテンプレートエントリーを生成します。たとえば、テンプレートエントリー **cn=joe new user,cn=accounts,ou=people,dc=example,dc=com** を確認するには、サーバーは **cn=template,cn=accounts,ou=people,dc=example,dc=com** を作成します。

存在しないエントリーをチェックする場合、get effective rights 検索は、指定したオブジェクトクラス

を使用して、(存在しない) エントリーのすべての属性を持つテンプレートエントリーを生成することができます。**person** のオブジェクトクラス (**@person**) を持つ **cn=joe new user,cn=accounts,ou=people,dc=example,dc=com** の場合、サーバーは **cn=template_person_objectclass,cn=accounts,ou=people,dc=example,dc=com** を生成します。

サーバーがテンプレートエントリーを作成すると、オブジェクトクラス定義の最初の MUST 属性を使用して RDN 属性を作成します (または、MUST 属性がない場合は MAY を使用します)。しかし、これは誤った RDN 値になる可能性があり、その結果、与えられたサブツリーに対して確立された ACI に違反または回避することになります。この場合は、使用する RDN 値をオブジェクトクラスを渡すことで指定できます。これには **@objectclass:rdn_attribute** の形式があります。

たとえば、RDN として、***uidNumber*** の存在しない Posix エントリーについて **scarter** の権限を確認するには、次のコマンドを実行します。

```
# ldapsearch -D "cn=Directory Manager" -W -b "ou=people,dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com' "(objectclass=*)"
@posixaccount:uidnumber

dn: uidNumber=template_posixaccount_objectclass,ou=people,dc=example,dc=com
entryLevelRights: v
attributeLevelRights: description:rsc, gecos:rsc, loginShell:rsc, userPassword
 :rsc, objectClass:rsc, homeDirectory:rsc, gidNumber:rsc, uidNumber:rsc, uid:
 rsc, cn:rsc
```

18.12.3.5. 操作属性の get effective rights 検索の例

操作属性は、get effective rights 検索など、通常の **ldapsearch** で返されません。操作属性の情報を返すには、プラス記号 (**+**) を使用します。これにより、エントリーで使用できる操作属性のみが返されます。

例18.44 操作属性に対する get effective rights の結果

```
# ldapsearch -D "cn=Directory Manager" -W -x -b "uid=scarter,ou=people,dc=example,dc=com" -
E '!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com' "(objectclass=*)" "+"


dn: uid=scarter,ou=People,dc=example,dc=com
entryLevelRights: vadn
attributeLevelRights: nsICQStatusText:rscwo, passwordGraceUserTime:rscwo,
pwdGraceUserTime:rscwo, nsYIMStatusText:rscwo, modifyTimestamp:rscwo,
passwordExpWarned:rscwo, pwdExpirationWarned:rscwo, entrydn:rscwo, aci:rscwo,
nsSizeLimit:rscwo, nsAccountLock:rscwo, passwordExpirationTime:rscwo, entryid:rscwo,
nsSchemaCSN:rscwo, nsRole:rscwo, retryCountResetTime:rscwo, ldapSchemas:rscwo,
nsAIMStatusText:rscwo, copiedFrom:rscwo, nsICQStatusGraphic:rscwo, nsUniqueId:rscwo,
creatorsName:rscwo, passwordRetryCount:rscwo, dncomp:rscwo, nsTimeLimit:rscwo,
passwordHistory:rscwo, pwdHistory:rscwo, nscpEntryDN:rscwo, subschemaSubentry:rscwo,
nsYIMStatusGraphic:rscwo, hasSubordinates:rscwo, pwdpolicysubentry:rscwo,
nsAIMStatusGraphic:rscwo, nsRoleDN:rscwo, createTimestamp:rscwo,
accountUnlockTime:rscwo, copyingFrom:rscwo, nsLookThroughLimit:rscwo,
nsds5ReplConflict:rscwo, modifiersName:rscwo, parentid:rscwo,
passwordAllowChangeTime:rscwo, nsBackendSuffix:rscwo, nsIdleTimeout:rscwo,
ldapSyntaxes:rscwo, numSubordinates:rscwo
```

18.12.3.6. get effective rights 結果とアクセスコントロールルールの例

get effective 権限は、get effective rights サブジェクトエントリーに対して有効な ACL に応じて返されます。

たとえば、この ACL が設定され、この例の目的でのみ ACL が設定されます。

```
dn: dc=example,dc=com
objectClass: top
objectClass: domain
dc: example
aci: (target=ldap:///ou=Accounting,dc=example,dc=com)(targetattr="*")(version
 3.0; acl "test acl"; allow (read,search,compare) (userdn = "ldap:///anyone") ;)

dn: ou=Accounting,dc=example,dc=com
objectClass: top
objectClass: organizationalUnit
ou: Accounting
```

ACL には **dc=example,dc=com** サブツリーが含まれていないため、get effective rights 検索では、ユーザーに **dc=example,dc=com** エントリーに対する権限がないことが表示されます。

例18.45 ACL を設定しない (Directory Manager) Get Effective Rights の結果

```
# ldapsearch -D "cn=Directory Manager" -W -b "dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com' "(objectclass=*)"
"*@person"

dn: cn=template_person_objectclass,uid=scarter,ou=people,dc=example,dc=com
objectClass: person
objectClass: top
cn: (template_attribute)
sn: (template_attribute)
description: (template_attribute)
seeAlso: (template_attribute)
telephoneNumber: (template_attribute)
userPassword: (template_attribute)
entryLevelRights: none
attributeLevelRights: sn:none, cn:none, objectClass:none, description:none, seeAlso:none,
telephoneNumber:none, userPassword:none, aci:none
```

Directory Manager ではなく通常のユーザーが同じコマンドを実行しようとすると、結果は単に空白になります。

例18.46 ACL を設定しない (通常ユーザー) Get Effective Rights の結果

```
# ldapsearch -D "uid=scarter,ou=people,dc=example,dc=com" -W -b "dc=example,dc=com" -E
'!1.3.6.1.4.1.42.2.27.9.5.2=:dn:uid=scarter,ou=people,dc=example,dc=com' "(objectclass=*)"
"*@person"
```

18.12.4. Get Effective Rights 戻りコード

Get Effective Rights 検索とエラーに重大度が設定されていない場合は、通常のエントリー情報が返されますが、**entryLevelRights** および **attributeLevelRights** の権限の代わりに、エラーコードが返されます。このコードにより、クエリーされたエントリーの設定に関する情報を取得できます。表18.4「返された結果コード」 は、エラーコードと、リレーが可能な潜在的な設定情報を要約します。

表18.4 返された結果コード

コード	説明
0	正常に完了しました。
1	操作エラー。
12	重要な拡張機能は利用できません。重大度式が **true** に設定され、クエリー対象のエントリーに有効な権限がない場合は、このエラーが返されます。
16	そのような属性はありません。アクセス権のために特定の属性をクエリーしましたが、その属性がスキーマに存在しない場合は、このエラーが返されます。
17	未定義の属性タイプ。
21	無効な属性構文。
50	権限が不十分。
52	利用できません。
53	不本意なパフォーマンス。
80	その他。

18.13. アクセス制御情報のロギング

アクセス制御情報をログに記録するには、***nsslapd-errorlog-level*** パラメーターを **128** (アクセス制御リストの処理) が含まれる値に設定します。エラーログレベルの設定に関する詳細は「ログレベルの設定」を参照してください。

18.14. 高度なアクセス制御: マクロ ACI の使用

マクロ ACI により柔軟性が向上します。たとえば、サブツリーを追加できます。また、ACI を追加しなくても、他のサブツリーと同じ調整されたアクセス制御を自動的に取得できます。副作用として、ACI の数は小さくなりますが、Macro ACI 処理の数は通常の ACI よりも高くなります。

マクロは、ACI の DN または DN の一部を表すために使用されるプレースホルダーです。マクロを使って、ACI のターゲット部分、バインドルール部分、またはその両方で DN を表現することができます。実際には、Directory Server が受信 LDAP 操作を取得すると、ACI マクロは LDAP 操作によってターゲットとなっているリソースと照合されます。一致する場合、マクロはターゲットリソースの DN の値に置き換えられます。次に、Directory Server は ACI を正常に評価します。

18.14.1. マクロ ACI の例

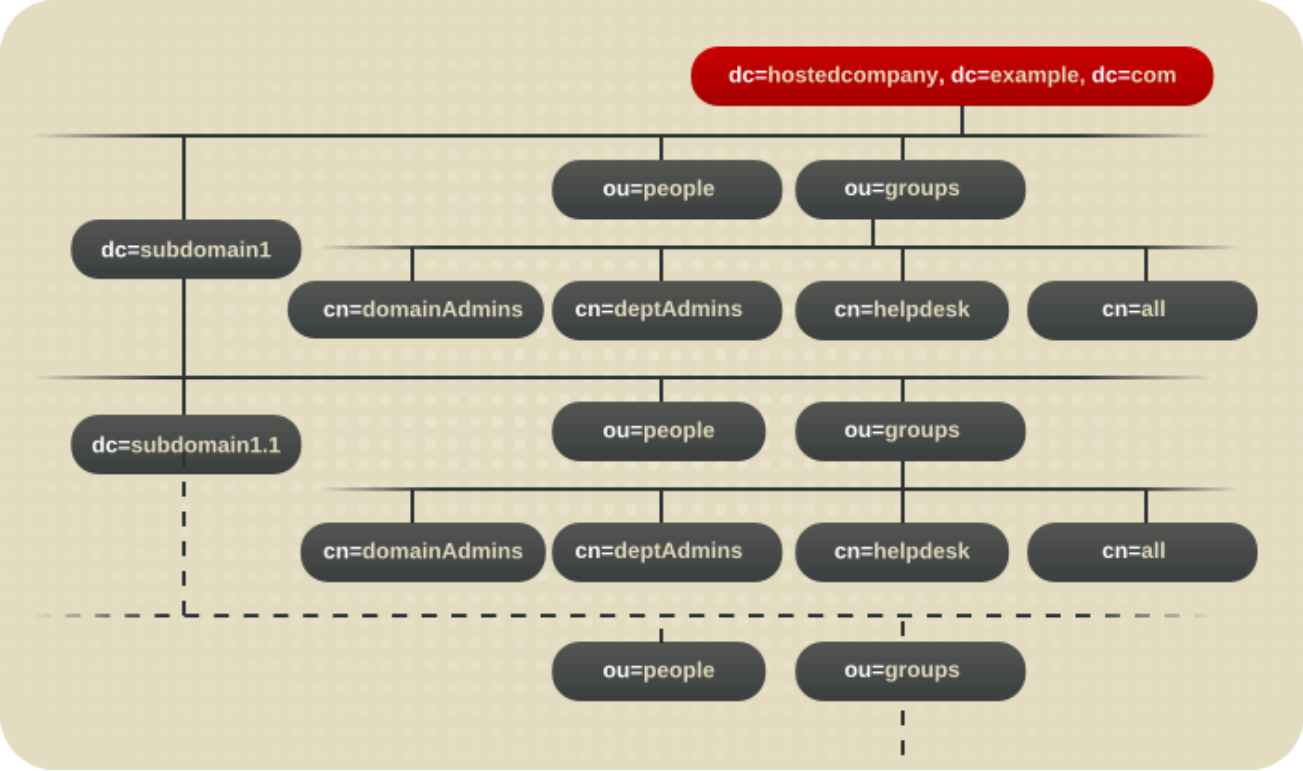
図18.1「マクロ ACI のディレクトリーツリーの例」は、マクロ ACI を使用して ACI の全体的な数を効果的に減らすディレクトリーツリーを表示します。この図は、同じツリー構造 (**ou=groups**、**ou=people**) を持つサブドメインの繰り返しパターンを使用します。Example Corp. ディレクトリーツリーが接尾辞 **dc=hostedCompany2,dc=example,dc=com** および **dc=hostedCompany3,dc=example,dc=com** を格納するため、このパターンはツリー全体で繰り返されます。

ディレクトリーツリーに適用される ACI には、繰り返しパターンがあります。たとえば、以下の ACI は **dc=hostedCompany1,dc=example,dc=com** ノードにあります。

```
aci: (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
    (version 3.0; acl "Domain access"; allow (read,search)
    groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com";)
```

この ACI は、**dc=hostedCompany1,dc=example,dc=com** ツリー内の任意のエントリーに **DomainAdmins** グループに対する読み取り権限および検索権限を付与します。

図18.1 マクロ ACI のディレクトリーツリーの例



以下の ACI は **dc=hostedCompany1,dc=example,dc=com** ノードにあります。

```
aci: (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
    (version 3.0; acl "Domain access"; allow (read,search)
    groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com";)
```

以下の ACI は **dc=subdomain1,dc=hostedCompany1,dc=example,dc=com** ノードにあります。

```
aci: (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
    (version 3.0; acl "Domain access"; allow (read,search)

groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany1,dc=example,dc=
com";)
```

以下の ACI は **dc=hostedCompany2,dc=example,dc=com** ノードにあります。

```
aci: (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
    (version 3.0; acl "Domain access"; allow (read,search)
    groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany2,dc=example,dc=com";)
```

以下の ACI は **dc=subdomain1,dc=hostedCompany2,dc=example,dc=com** ノードにあります。

```
aci: (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
    (version 3.0; acl "Domain access"; allow (read,search)

groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany2,dc=example,dc=
com";)
```

上記の 4 つの ACI での唯一の違いは **groupdn** キーワードで指定される DN です。DN にマクロを使用して、これらの ACI を **dc=example,dc=com** ノード上のツリーのルートにある単一の ACI に置き換えることができます。この ACI は以下のように読み取ります。

```
aci: (target="ldap:///ou=Groups,($dn),dc=example,dc=com")
    (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
    (version 3.0; acl "Domain access"; allow (read,search)
    groupdn="ldap:///cn=DomainAdmins,ou=Groups,[$dn],dc=example,dc=com";)
```

以前使用されていなかった **target** キーワードが新しい ACI で使用されます。

この例では、ACI の数が 4 から 1 に減ります。実際の効果は、ディレクトリーツリーにどれだけ多くの繰り返しパターンがあるかにかかっています。

18.14.2. マクロ ACI 構文

マクロ ACI には、DN、または DN の一部を置き換える以下のタイプの式が含まれます。

- ($dn)
- [$dn]
- ($attr.**attrName**) ここで、**attrName** はターゲットエントリーに含まれる属性を表します。

このセクションでは、**userdn**、**roledn**、**groupdn**、および **userattr** などのバインド認証情報を提供するために使用される ACI キーワードは、ACI の **ターゲット** とは異なり、**サブジェクト** をまとめて呼びます。マクロ ACI は、ターゲット部分または ACI のサブジェクト部分で使用できます。

表18.5「ACI キーワードのマクロ」 は、DN マクロを使用できる ACI の概要を示します。

表18.5 ACI キーワードのマクロ

マクロ	ACI キーキーワード
($dn)	target, targetfilter, userdn, roledn, groupdn, userattr
[$dn]	targetfilter, userdn, roledn, groupdn, userattr
($attr.**attrName**)	userdn, roledn, groupdn, userattr

以下の制限が適用されます。

- **targetfilter**、**userdn**、**roledn**、**groupdn**、**userattr** で **($dn)** を使用する場合は、**($dn)** を含むターゲットを定義する **必要があります**。
- **targetfilter**、**userdn**、**roledn**、**groupdn**、**userattr** で **[$dn]** を使用する場合は、**($dn)** を含むターゲットを定義する **必要があります**。

注記

マクロを使用する場合は、**($dn)** マクロが含まれるターゲット定義を **常に** 必要とします。

($dn) マクロと **($attr.attrName)** マクロを組み合わせることができます。

18.14.2.1. ($dn) のマクロ一致

($dn) マクロは、LDAP 要求でターゲットとするリソースの一致する部分に置き換えられます。たとえば、**cn=all,ou=groups,dc=subdomain1,dc=hostedCompany1,dc=example,dc=com** エントリーで LDAP 要求と、以下のようにターゲットを定義する ACI があります。

```
(target="ldap:///ou=Groups,($dn),dc=example,dc=com")
```

($dn) マクロは **dc=subdomain1,dc=hostedCompany1** と一致します。

ACI のサブジェクトも **($dn)** を使用する場合は、ターゲットと一致する部分文字列を使用してサブジェクトを展開します。以下に例を示します。

```
aci: (target="ldap:///ou=*,($dn),dc=example,dc=com")
    (targetattr = "*") (version 3.0; acl "Domain access"; allow (read,search)
    groupdn="ldap:///cn=DomainAdmins,ou=Groups,($dn),dc=example,dc=com";)
```

この場合、ターゲットの **($dn)** に一致する文字列が **dc=subdomain1,dc=hostedCompany1** の場合、サブジェクトで同じ文字列が使用されます。その後、ACI は以下のように拡張されます。

```
aci: (target="ldap:///ou=Groups,dc=subdomain1,dc=hostedCompany1,
    dc=example,dc=com") (targetattr = "*") (version 3.0; acl "Domain
    access"; allow (read,search) groupdn="ldap:///cn=DomainAdmins,ou=Groups,
    dc=subdomain1,dc=hostedCompany1,dc=example,dc=com";)
```

マクロの拡張後、Directory Server は通常のプロセスに従って ACI を評価し、アクセスが付与されているかどうかを確認します。

18.14.2.2. [$dn] のマクロ一致

[$dn] と一致するメカニズムは、**($dn)** とは若干異なります。対象となるリソースの DN は、一致するものが見つかるまで、左端の RDN コンポーネントを削除するたびに複数回調べられます。

たとえば、**cn=all,ou=groups,dc=subdomain1,dc=hostedCompany1,dc=example,dc=com** サブツリーをターゲットとする LDAP 要求と、以下の ACI があります。

```
aci: (target="ldap:///ou=Groups,($dn),dc=example,dc=com")
    (targetattr = "*") (version 3.0; acl "Domain access"; allow (read,search)
    groupdn="ldap:///cn=DomainAdmins,ou=Groups,[$dn],dc=example,dc=com";)
```

この ACI を拡張する手順は以下のとおりです。

1. ターゲットの **($dn)** は **dc=subdomain1,dc=hostedCompany1** と一致します。
2. サブジェクトの **[$dn]** は、**dc=subdomain1,dc=hostedCompany1** に置き換えられます。

   この場合の結果は、**groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany1,dc=example,dc=com"** になります。バインド DN が対象のグループのメンバーである場合は、一致するプロセスが停止し、ACI が評価されます。一致しない場合は、プロセスが続行します。
3. サブジェクトの **[$dn]** は、**dc=hostedCompany1** に置き換えられます。

   この場合の結果は、**groupdn="ldap:///cn=DomainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com"** になります。この場合、バインド DN がそのグループのメンバーではない場合、ACI は評価されません。これがメンバーの場合には、ACI が評価されます。

[$dn] は、マクロの利点は、ドメインレベルの管理者にディレクトリーツリー内の **すべての** サブドメインにアクセスを付与する柔軟な方法を提供することです。したがって、ドメイン間の階層関係を表現するのに便利です。

たとえば、以下の ACI について考えてみましょう。

```
aci: (target="ldap:///ou=*, ($dn),dc=example,dc=com")
    (targetattr="*")(targetfilter=(objectClass=nsManagedDomain))
    (version 3.0; acl "Domain access"; allow (read,search)
    groupdn="ldap:///cn=DomainAdmins,ou=Groups,[$dn],dc=example,dc=com";)
```

cn=DomainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com のメンバーに、**dc=hostedCompany1** の下にあるすべてのサブドメインへのアクセス権限を付与します。そのため、該当グループに属する管理者は **ou=people,dc=subdomain1.1,dc=subdomain1** などのサブツリーにアクセスできます。

ただし、同時に **cn=DomainAdmins,ou=Groups,dc=subdomain1.1** のメンバーは、**ou=people,dc=hostedCompany1** および **ou=people,dc=subdomain1,dc=hostedCompany1** ノードへのアクセスが拒否されます。

18.14.2.3. Macro Matching for ($attr.attrName)

($attr.attrName) マクロは、DN のサブジェクト部分に常に使用されます。たとえば、以下の **roledn** を定義します。

```
roledn = "ldap:///cn=DomainAdmins,($attr.ou)"
```

ここで、サーバーが、以下のエントリーでターゲットに設定された LDAP 操作を受け取ることを想定しています。

```
dn: cn=Jane Doe,ou=People,dc=HostedCompany1,dc=example,dc=com
cn: Jane Doe
sn: Doe
ou: Engineering,dc=HostedCompany1,dc=example,dc=com
...
```

ACI の **roledn** 部分を評価するため、サーバーはターゲットエントリーに保存されている ***ou*** 属性を確認し、この属性の値をマクロを展開します。そのため、この例では **roledn** は以下のように展開されます。

```
roledn = "ldap:///cn=DomainAdmins,ou=Engineering,dc=HostedCompany1,dc=example,dc=com"
```

次に、Directory Server は通常の ACI 評価アルゴリズムに従って ACI を評価します。

属性が多値を持つ場合、それぞれの値を使ってマクロをデプロイメントし、一致した最初の値を使用します。以下に例を示します。

```
dn: cn=Jane Doe,ou=People,dc=HostedCompany1,dc=example,dc=com
cn: Jane Doe
sn: Doe
ou: Engineering,dc=HostedCompany1,dc=example,dc=com
ou: People,dc=HostedCompany1,dc=example,dc=com...
```

この場合、Directory Server が ACI を評価すると、以下の拡張された式で論理 OR が実行します。

```
roledn = "ldap:///cn=DomainAdmins,ou=Engineering,dc=HostedCompany1,dc=example,dc=com"

roledn = "ldap:///cn=DomainAdmins,ou=People,dc=HostedCompany1,dc=example,dc=com"
```

18.15. DIRECTORY MANAGER でのアクセス制御の設定

未制約の管理ユーザーがあると、メンテナンスパースペクティブからは妥当になります。Directory Manager では、メンテナンスタスクを実行し、インシデントへの対応に高いレベルのアクセスが必要です。

ただし、Directory Manager ユーザーの権限により、ある程度のアクセス制御は、root ユーザーとして、承認されていないアクセスや攻撃を阻止することを推奨します。

通常のアクセス制御ルールはディレクトリーツリーに適用されます。Directory Manager は通常のユーザーエントリーではないため、(通常の) ACI を Directory Manager ユーザーに適用できません。ACI は、特別なプラグイン設定エントリーを介して適用されます。

18.15.1. Directory Manager アカウントのアクセス制御

通常のアクセス制御ルールは、Directory Manager ユーザーには適用されません。Directory Manager ユーザーの権限は Directory Server にハードコーディングされており、バインドルールには使用できません。

Directory Manager のアクセス制御は、**RootDN アクセス制御プラグイン** を介して実装されます。このプラグインは Directory Server 設定に適用されるため、Directory Manager エントリーにはアクセス制御ルールを適用できます。

プラグインは標準の ACL を定義しません。これには、ターゲット (Directory Manager エントリー) および許可される権限 (すべて) などの一部の情報が暗に示されています。RootDN アクセス制御プラグインの目的は、Directory Manager が実行可能な **もの** を制限しません。そのため、場所または時間をもとに、(有効な認証情報でも) Directory Manager としてログインできるユーザーを制限することで、セキュリティーレベルを提供することが目的です。

このため、Directory Manager の ACI はバインドルールのみを設定します。

- 時間ベースのアクセス制御 (例: 8a.m. から 5p.m)(0800 から 1700)、および曜日のアクセス制御により、アクセスは明示的に定義された日数でのみ許可されます。これは「曜日の特定の日におけるアクセスの定義」および「特定の時刻におけるアクセスの定義」に類似しています。
- 指定した IP アドレス、ドメイン、またはサブネットのみが明示的に許可または拒否される IP アドレスルール。これは、「特定の IP アドレスまたは範囲からのアクセスの定義」に類似しています。
- ホストアクセスルール。指定されたホスト名、ドメイン名、またはサブドメインのみが明示的に許可または拒否されます。これは、「特定のホストまたはドメインからアクセスの定義」に類似しています。

他のアクセス制御ルールと同様、deny ルールは allow ルールよりも優先されます。

重要

Directory Manager が常に適切なレベルのアクセスを許可されていることを確認してください。Directory Manager は、オフタイム (ユーザーの負荷が少ない時) や障害対応のためにメンテナンス作業を行う必要があります。その場合、時間や曜日ベースのアクセス制御ルールを厳しく設定すると、Directory Manager がディレクトリーを適切に管理できなくなる可能性があります。

18.15.2. RootDN アクセス制御プラグインの設定

ルート DN アクセス制御ルールはデフォルトで無効になっています。**RootDN Access Control** プラグインを有効にし、適切なアクセス制御ルールを設定します。

注記

Directory Manager には 1 つのアクセス制御ルールがあり、プラグインエントリーには、ディレクトリー全体のすべてのアクセスに適用されます。

1. **RootDN Access Control** プラグインを有効にします。

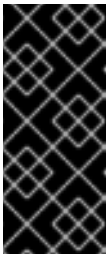
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin root-dn enable
Plugin 'RootDN Access Control' enabled
...
```

2. アクセス制御命令にバインドルールを設定します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin root-dn set --open-
time=0600 --close-time=2100 --allow-host="*.example.com" --deny-
host="*.remote.example.com"
```

以下のパラメーターを設定できます。

- 時間ベースのアクセス制御の場合は ***--open-time*** および ***--close-time***。
- 日ベースのアクセス制御の場合は ***--days-allowed***。
- ホストベースのアクセス制御用の ***--allow-host***、***--deny-host***、***--allow-ip***、および ***--deny-ip***これらはすべての多値の属性であり、ワイルドカードを使用して IP 範囲またはドメインを許可または拒否することができます。



重要

拒否ルールの優先度は、許可ルールより高く設定されます。たとえば、***--allow-host*** パラメーターが ***.example.com** に設定され、***--deny-host*** が ***.front-office.example.com** に設定されている場合は、Directory Manager が阻止されているため、**front-office.example.com** サブドメインのすべてのホストからのアクセスが阻止されます。

3. Directory Server を再起動します。

```
# dsctl instance_name restart
```

第19章 ヘルスチェック機能を使用した問題の特定

この **dsctl healthcheck** コマンドは、潜在的な問題について Directory Server インスタンスを分析し、解決するためのソリューションを推奨します。

以下の表は、ヘルスチェック機能の実行チェックを示しています。

表19.1 チェックの概要

コンポーネント	重大度	結果コード	説明
バックエンド	低	DSBLE0003	データベースは初期化されませんでした。データベースは作成されましたが、データベースが空です。
バックエンド	中	DSBLE0001	バックエンドのマッピングツリーエントリーが設定にありません。
コンフィグ	低	DSCLE0001	高解像度のタイムスタンプが無効になります。
コンフィグ	高	DSVIRTLE0001	仮想属性が誤ってインデックス化されています。ロールまたは CoS (Class of Service) 定義で使用されるインデックス化された属性により、検索結果が破損する可能性があります。
オペレーティングシステム	中	DSPERMLE0001	**/etc/resolve.conf** ファイルに設定されているパーミッションは、**0644** とは異なります。
オペレーティングシステム	高	DSDSLE0001	ディスク領域不足
オペレーティングシステム	高	DSPERMLE0002	**/etc/dirsrv/slapd-*instance_name*/pin.txt** および **/etc/dirsrv/slapd-instance_name/pwdfile.txt** ファイルに設定された権限は、**0400** とは異なります。
プラグイン	低	DSRILE0001	更新の遅延は **Referential Integrity** プラグインに設定されます。これにより、レプリケーションの問題が発生する可能性があります。
プラグイン	高	DSRILE0002	**Referential Integrity** プラグインにはインデックスがありません。プラグインは、インデックス化されていない場合にすべての削除操作に対して特定の属性をクエリーします。これにより、インデックスのない検索結果が検出されにくくなったり、CPU 使用率が高くなったりします。
レプリケーション	低	DSREPLLE0002	競合エントリーがデータベースに存在します。

コンポーネント	重大度	結果コード	説明
レプリケーション	低	DSSKEWLE0001	レプリケーションタイムのずれが 6 時間より大きく、12 時間より小さくなっています。
レプリケーション	中	DSCLLE0001	changelog のトリミングは無効になっています。この場合、changelog は制限なしで増加します。
レプリケーション	中	DSREPLLE0004	ヘルスチェックは、レプリケーションのステータスを取得できませんでした。
レプリケーション	中	DSREPLLE0003	トポロジーは同期されていませんが、レプリケーションは機能しています。
レプリケーション	中	DSREPLLE0005	リモートレプリカには到達できません。
レプリケーション	中	DSSKEWLE0002	レプリケーションタイムのずれが 12 時間より大きく、24 時間より小さくなっています。
レプリケーション	高	DSREPLLE0001	トポロジーは同期されておらず、レプリケーションは機能しません。
レプリケーション	高	DSSKEWLE0003	レプリケーションタイムのずれが 24 時間より大きい。レプリケーションセッションが破損する可能性がありました。
セキュリティー	中	DSELE0001	最小の TLS バージョンは TLS 1.2 未満の値に設定されます。
セキュリティー	高	DSCLE0002	弱いパスワードストレージスキームが設定されている。
サーバー	高	DSBLE0002	ヘルスチェックは、バックエンドのクエリーに失敗しました。
TLS 証明書	中	DSCERTLE0001	サーバー証明書は次の 30 日以内に有効期限が切れます。
TLS 証明書	高	DSCERTLE0002	サーバー証明書の有効期限が切れている。

19.1. DIRECTORY SERVER ヘルスチェックの実行

ヘルスチェックを実行するには、以下を入力します。

```
# dsctl instance_name healthcheck
Beginning lint report, this could take a while ...
Checking Backends ...
Checking Config ...
```

```
Checking Encryption ...
Checking FSChecks ...
Checking ReferentialIntegrityPlugin ...
Checking MonitorDiskSpace ...
Checking Replica ...
Checking Changelog5 ...
Checking NssSsl ...
Healthcheck complete.
1 Issue found!  Generating report ...
```

例19.1 ヘルスチェックの予想されるレポート

以下は、ヘルスチェックレポートの例です。

```
[1] DS Lint Error: DSELE0001
--------------------------------------------------------------------------------
Severity: MEDIUM
Affects:
 -- cn=encryption,cn=config

Details:
-----------
This Directory Server may not be using strong TLS protocol versions. TLS1.0 is known to
have a number of issues with the protocol. Please see:

https://tools.ietf.org/html/rfc7457

It is advised you set this value to the maximum possible.

Resolution:
-----------
There are two options for setting the TLS minimum version allowed.  You,
can set "sslVersionMin" in "cn=encryption,cn=config" to a version greater than "TLS1.0"
You can also use 'dsconf' to set this value.  Here is an example:

    # dsconf slapd-instance_name security set --tls-protocol-min=TLS1.2

You must restart the Directory Server for this change to take effect.

Or, you can set the system wide crypto policy to FUTURE which will use a higher TLS
minimum version, but doing this affects the entire system:

    # update-crypto-policies --set FUTURE


===== End Of Report (1 Issue found) =====
```

JSON 形式で出力を表示するには、***--json*** パラメーターをコマンドに渡します。

```
# dsctl --json instance_name healthcheck
```

例19.2 JSON 形式のヘルスチェックの可能なレポート

以下は、JSON 形式のヘルスチェックレポートの例です。


```
[
    {
        "dsle": "DSELE0001",
        "severity": "MEDIUM",
        "items": [
            "cn=encryption,cn=config"
        ],
        "detail": "This Directory Server may not be using strong TLS protocol versions. TLS1.0 is known to\nhave a number of issues with the protocol. Please see:\n\nhttps://tools.ietf.org/html/rfc7457\n\nIt is advised you set this value to the maximum possible.",
        "fix": "There are two options for setting the TLS minimum version allowed.  You,\ncan set \"sslVersionMin\" in \"cn=encryption,cn=config\" to a version greater than \"TLS1.0\"\nYou can also use 'dsconf' to set this value.  Here is an example:\n\n    # dsconf slapd-instance_name security set --tls-protocol-min=TLS1.2\n\nYou must restart the Directory Server for this change to take effect.\n\nOr, you can set the system wide crypto policy to FUTURE which will use a higher TLS\nminimum version, but doing this affects the entire system:\n\n    # update-crypto-policies --set FUTURE"
    }
]
```

第20章 ユーザー認証の管理

ユーザーが Red Hat Directory Server に接続すると、最初にユーザーが認証されます。次に、ディレクトリーは認証時に確立されたアイデンティティーに応じてアクセス権限およびリソース制限をユーザーに付与します。

この章では、ディレクトリーのパスワードとアカウントのロックアウトポリシーの設定、ディレクトリーへのアクセスを拒否するユーザーグループの設定、バインド DN に応じてユーザーが利用できるシステムリソースの制限など、ユーザーを管理するためのタスクを説明します。

20.1. ユーザーパスワードの設定

このエントリーを使用して、***userPassword*** 属性があり、アクティブではない場合にのみディレクトリーにバインドできます。ユーザーパスワードはディレクトリーに格納されるため、**ldapmodify** ユーティリティーを使用する場合など、LDAP 操作でユーザーパスワードを設定またはリセットできます。

管理者がユーザーのパスワードを変更すると、Directory Server は、ユーザーのエントリーの ***pwdReset*** 操作属性を **true** に設定します。アプリケーションはこの属性を使用して、管理者がユーザーのパスワードをリセットしたかどうかを識別できます。

ディレクトリーエントリーの作成または変更に関する情報は、3章*ディレクトリーエントリーの管理*を参照してください。ユーザーアカウントを非アクティブにする方法は、「ユーザーおよびロールの手動による非アクティブ化」を参照してください。

「パスワード管理者の設定」に説明されているパスワード管理者のみが、事前にハッシュ化されたパスワードを追加できます。これらのユーザーは、パスワードポリシーに違反させることもできます。

警告

パスワード管理者アカウントまたは **Directory Manager** (root DN) を使用してパスワードを設定すると、パスワードポリシーは回避され、検証されません。これらのアカウントは、通常のユーザーパスワードの管理には使用しないでください。パスワードポリシーの回避が必要なパスワード管理タスクの実行にのみ使用します。

20.2. パスワード管理者の設定

Directory Manager は、*パスワード管理者* ロールをユーザーまたはグループに追加できます。アクセス制御命令 (ACI) は設定する必要があるため、グループで、すべてのパスワード管理者を管理する単一の ACI セットのみを許可することが推奨されます。パスワード管理者は、以下を含むユーザーパスワード操作を実行できます。

- ユーザーがパスワードの変更を強制する
- パスワードポリシーで定義されている異なるストレージスキームへのユーザーのパスワードの変更
- パスワード構文チェックを回避
- ハッシュ化されたパスワードを追加します。

「ユーザーパスワードの設定」 で説明されているように、通常のパスワードの更新は、**userPassword** 属性のみを更新するパーミッションを持つデータベース内の既存のロールで実行することが推奨されます。Red Hat は、このような通常のタスクにパスワード管理者アカウントを使用することは推奨されません。

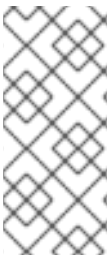
ユーザーまたはグループをパスワード管理者として指定できます。

- ローカルポリシーで、次を実行します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp set
ou=people,dc=example,dc=com --pwdadmin
"cn=password_admins,ou=groups,dc=example,dc=com"
```

- グローバルポリシーで、次を実行します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy set --pwdadmin
"cn=password_admins,ou=groups,dc=example,dc=com"
```



注記

cn=config エントリーの下に新しい ***passwordAdminSkipInfoUpdate: on/off*** 設定を追加すると、パスワード管理者が実行するパスワードの更新を詳細に制御できます。この設定を有効にすると、パスワードの更新は ***passwordHistory***、***passwordExpirationTime***、***passwordRetryCount***、***pwdReset***、***passwordExpWarned*** などの特定の属性を更新しません。

20.3. 外部に保存されたパスワードの変更

ほとんどのパスワードは **ldapmodify** 操作で変更できますが、通常の LDAP 操作で変更することはできないパスワードがあります。これらのパスワードは、SASL アプリケーションに保存されているパスワードなど、Directory Server 外に保存できます。これらのパスワードは、**パスワード変更拡張操作** で変更できます。

Directory Server は RFC 3062 で定義されているパスワード変更操作をサポートするため、ユーザーは標準に準拠した状態で適切なクライアントを使用してパスワードを変更できます。**dsidm** ユーティリティーは、指定されたユーザーのパスワードの変更を渡します。

```
# dsidm ldap://server.example.com -D bind_dn -W -b dc=example,dc=com account
change_password user newPassword oldPassword
```

重要

パスワード操作はセキュアな接続 (SASL、TLS、または STARTTLS) に対して実行する必要があります。LDAP クライアントツールでセキュアな接続を使用する方法は、「証明書を使用した認証」を参照してください。

パラメーターの詳細は、**dsidm *instance_name* account change_password --help** コマンドの出力を参照してください。

セキュアでないポート上でコマンドを実行する STARTTLS を使用するには、**-Z** オプションと標準の LDAP ポート番号を指定して **dsidm** を実行します。パスワード拡張変更操作の形式は以下のとおりです。

```
# dsidm ldap://server.example.com -Z bind_dn -W -b dc=example,dc=com account
change_password user newPassword oldPassword
```

注記

STARTTLS 接続を機能させるには、「証明書を使用した認証」 で説明されているように、TLS 環境変数を設定する必要があります。

-Z オプションを使用して、強制的に接続を成功させます。

エントリーのパスワードを変更するには、他の操作と同様に **dsidm** を実行します。アカウントがバインド DN に指定されたものと同じでも、バインド DN を指定する必要があります。以下に例を示します。

```
# dsidm ldap://server.example.com -Z bind_dn -W -b dc=example,dc=com account
change_password user newPassword oldPassword
```

アクセス制御はパスワードの変更操作に対して適用されます。バインド DN に指定のパスワードを変更する権限がない場合、操作は **Insufficient rights** エラーを出力して失敗します。

20.4. パスワードポリシーの管理

パスワードポリシーは、一定レベルのセキュリティーを強制することで、パスワードを使用するリスクを最小限に抑えることができます。たとえば、パスワードポリシーでは、以下を定義できます。

- ユーザーはスケジュールに応じてパスワードを変更する必要があります。
- ユーザーは、簡単ではないパスワードを提供する必要があります。
- パスワード構文は、特定の複雑な要件を満たす必要があります。

警告

パスワード管理者アカウントまたは **Directory Manager** (root DN) を使用してパスワードを設定すると、パスワードポリシーは回避され、検証されません。これらのアカウントは、通常のユーザーパスワードの管理には使用しないでください。パスワードポリシーの回避が必要なパスワード管理タスクの実行にのみ使用します。

Directory Server は、きめ細かなパスワードポリシーをサポートしており、パスワードポリシーは、ディレクトリー全体 (**global** パスワードポリシー)、特定のサブツリー (**subtree-level** または **local** パスワードポリシー)、または特定のユーザー (**user-level** または **local** パスワードポリシー) に適用することができます。

ユーザーアカウントに適用される完全なパスワードポリシーは、以下の要素で設定されます。

- **パスワードポリシーチェックのタイプまたはレベル。** この情報は、サーバーがグローバルパスワードポリシーまたはローカル (サブツリー/ユーザーレベル) パスワードポリシーを確認および有効にするかどうかを示します。

パスワードポリシーは、一般的なものから特定のものまで、逆ピラミッド型になっています。グローバルパスワードポリシーは、サブツリーレベルのパスワードポリシーに置き換えられました。これは、ユーザーレベルのパスワードポリシーに置き換えられます。エントリーに対して強制されるパスワードポリシーは1つだけで、パスワードポリシーは追加されません。つまり、特定の属性がグローバルレベルまたはサブツリーレベルのポリシーで設定されていて、ユーザーレベルのパスワードポリシーで設定されていない場合、アクティブで適用されるポリシーはユーザーレベルのポリシーであるため、ログインが試みられてもその属性はユーザーに使用されません。

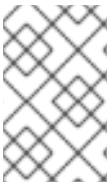
- **パスワードの追加および変更の情報。**パスワード情報には、パスワードの構文およびパスワード履歴の詳細が含まれます。
- **バインド情報**バインド情報には、許可された猶予期間の数、パスワードエージング属性、およびバインド失敗の追跡が含まれます。

注記

パスワードポリシーを設定したら、アカウントのロックアウトポリシーを設定すると、ユーザーパスワードを潜在的な脅威から保護できます。アカウントのロックアウトは、ユーザーのパスワードを繰り返し推測することで、ディレクトリーに分割を試行するハッカーから保護されます。

20.4.1. グローバルパスワードポリシーの設定

デフォルトでは、グローバルパスワードポリシー設定は無効になっています。本セクションでは、グローバルパスワードポリシーを設定する例を紹介します。



注記

パスワードポリシーを設定したら、アカウントロックアウトポリシーを設定します。詳細については、「パスワードベースのアカウントロックアウトポリシーの設定」 を参照してください。

20.4.1.1. コマンドラインを使用したグローバルパスワードポリシーの設定

dsconf ユーティリティーを使用して、グローバルパスワードポリシー設定を表示および編集します。

1. 現在の設定を表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy get
Global Password Policy: cn=config
--------------------------------
passwordstoragescheme: PBKDF2_SHA256
passwordChange: on
passwordMustChange: off
passwordHistory: off
passwordInHistory: 6
...
```

2. パスワードポリシー設定を調整します。たとえば、パスワード構文の確認を有効にしてパスワードの最小の長さを **12** 文字に設定するには、次のコマンドを実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy set --
pwdchecksyntax=on --pwdmintokenlen=12
```

使用可能な設定の完全なリストを表示するには、次のように入力します。

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

3. パスワードポリシーを有効にします。

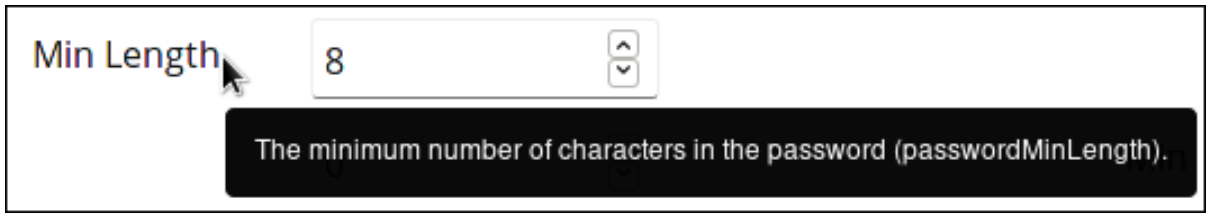
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy set --pwdlockout on
```

20.4.1.2. Web コンソールを使用したグローバルパスワードポリシーの設定

Web コンソールを使用してグローバルパスワードポリシーを設定するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** メニューを開きます。
4. **Password Policies** メニューで、**Global Policy** を選択します。
5. グローバルパスワードポリシーを設定します。次のカテゴリーのパラメーターを設定できます。
   - パスワードストレージスキームなどの一般的な設定
   - パスワード失効回数など、パスワードの有効期限設定。
   - アカウントのロックアウト設定 (何回ログインに失敗したらアカウントをロックするかなど)。
   - パスワード構文の設定 (パスワードの最小長など)。

   ツールヒントとパラメーターの **cn=config** エントリーで対応する属性名を表示するには、設定の上にマウスのカーソルを合わせます。詳細は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 のパラメーターの説明を参照してください。



6. **Save** をクリックします。

20.4.2. ローカルパスワードポリシーの使用

ローカルパスワードポリシーは、ディレクトリー全体の設定を定義するグローバルパスワードポリシーとは対照的に、特定のユーザーまたはサブツリーに対するポリシーです。

きめ細かいパスワードポリシーでパスワード構文が設定されていない場合、***nsslapd-pwpolicy-inherit-global*** パラメーターがオンになっていればグローバルポリシーから構文を継承できます。

--pwpinheritglobal オプションが定義されている場合、***passwordchecksyntax*** オプションはローカルポリシーでオフに、グローバルポリシーでオンに設定され、次の属性をグローバルポリシーからローカルポリシーに継承できます。

- ***passwordchecksyntax***
- ***passwordminlength***
- ***passwordmindigits***
- ***passwordminalphas***
- ***passwordminuppers***
- ***passwordminlowers***
- ***passwordminspecials***
- ***passwordmin8bit***
- ***passwordmaxrepeats***
- ***passwordmincategories***
- ***passwordmintokenlength***

20.4.2.1. Directory Server がローカルパスワードポリシーエントリーを保存する場所

dsconf localpwp adduser または **dsconf localpwp addsubtree** コマンドを使用する場合、Directory Server はポリシー属性を保存するエントリーを自動的に作成します。

- サブツリー (例: **ou=people,dc=example,dc=com**) では、以下のエントリーが追加されます。
  - サブツリーとそのすべての子について、さまざまなパスワードポリシー関連のエントリーを保持するためのサブツリーレベルのコンテナーエントリー (***nsPwPolicyContainer***)。以下に例を示します。

    ```
    dn: cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
    objectClass: top
    objectClass: nsContainer
    cn: nsPwPolicyContainer
    ```

  - サブツリーに固有のすべてのパスワードポリシー属性を保持するための実際のパスワードポリシー仕様エントリー (***nsPwPolicyEntry***) です。以下に例を示します。

    ```
    dn: cn="cn=nsPwPolicyEntry,ou=people,dc=example,dc=com",
     cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
    objectclass: top
    objectclass: extensibleObject
    objectclass: ldapsubentry
    objectclass: passwordpolicy
    ```

  - 上記の (***nsPwPolicyEntry***) エントリーを指定する ***pwdpolicysubentry*** 値を持つ CoS テンプレートエントリー (***nsPwTemplateEntry***)。以下に例を示します。

    ```
    dn: cn="cn=nsPwTemplateEntry,ou=people,dc=example,dc=com",
    ```

```
 cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: costemplate
objectclass: ldapsubentry
cosPriority: 1
pwdpolicysubentry: cn="cn=nsPwPolicyEntry,ou=people,dc=example,dc=com",
    cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
```

- サブツリーレベルでの CoS 仕様エントリー。以下に例を示します。

```
dn: cn=newpwdpolicy_cos,ou=people,dc=example,dc=com
objectclass: top
objectclass: LDAPsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=cn=nsPwTemplateEntry\,ou=people\,dc=example,dc=com,
 cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
cosAttribute: pwdpolicysubentry default operational
```

- ユーザーの場合 (例: **uid=*user_name*,ou=people,dc=example,dc=com**)、以下のエントリーが追加されます。
  - ユーザーとそのすべての子について、さまざまなパスワードポリシー関連のエントリーを保持するための親レベルのコンテナーエントリー (***nsPwPolicyContainer***)。以下に例を示します。

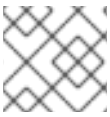
```
dn: cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
objectClass: top
objectClass: nsContainer
cn: nsPwPolicyContainer
```

  - ユーザーに固有のパスワードポリシー属性を保持するための実際のパスワードポリシー仕様エントリー (***nsPwPolicyEntry***) です。以下に例を示します。

```
dn: cn="cn=nsPwPolicyEntry,uid=user_name,ou=people,dc=example,dc=com",
 cn=nsPwPolicyContainer,ou=people,dc=example,dc=com
objectclass: top
objectclass: extensibleObject
objectclass: ldapsubentry
objectclass: passwordpolicy
```

20.4.2.2. ローカルパスワードポリシーの設定

ローカルパスワードポリシーを設定するには、以下を実行します。



注記

現在、コマンドラインを使用してのみローカルパスワードポリシーを設定できます。

1. サブツリーまたはユーザーエントリーにローカルパスワードポリシーがすでに存在しているかどうかを確認します。以下に例を示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp get
"ou=People,dc=example,dc=com"
Error: The policy wasn't set up for the target dn entry or it is invalid
```

ローカルポリシーが存在しない場合は、これを作成します。

- サブツリーパスワードポリシーを作成するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp addsubtree
"ou=People,dc=example,dc=com"
```

- ユーザーパスワードポリシーを作成するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp adduser
"uid=user_name,ou=People,dc=example,dc=com"
```

重要

新しいローカルポリシーを作成すると、上記のコマンドは、**cn=config** エントリーの ***nsslapd-pwpolicy-local*** パラメーターを **on** に自動的に設定します。

ローカルパスワードポリシーを有効にできない場合は、手動でパラメーターを **off** に設定します。

```
dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy set --
pwdlocal off
```

2. ローカルポリシー属性を設定します。たとえば、パスワードの有効期限を有効にし、パスワードの最大期間を 14 日 (**1209600** 秒) に設定するには、以下を実行します。

- サブツリーパスワードポリシー:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp set --
pwdexpire=on --pwdmaxage=1209600 "ou=People,dc=example,dc=com"
```

- ユーザーパスワードポリシー:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp set --
pwdexpire=on --pwdmaxage=1209600
"uid=user_name,ou=People,dc=example,dc=com"
```

使用可能な設定の完全なリストを表示するには、次のように入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp set --help
```

20.5. 一時パスワードルールの設定

Directory Server のパスワードポリシーは、ユーザーアカウントの一時パスワードの設定をサポートしています。ユーザーに一時パスワードを割り当てると、Directory Server は、ユーザーがパスワードを変更するまで、パスワード変更以外の操作を拒否します。

一時パスワードの機能は次のとおりです。

- **cn=Directory Manager** アカウントのみが一時パスワードを割り当てることができます。
- Directory Server は、攻撃者によるパスワードの調査を防ぐために、認証の試行を一定の回数だけ許可します。
- Directory Server は、設定直後に一時パスワードを使用できないように、指定された遅延時間後に認証の試行を許可します。
- Directory Server は、ユーザーが一時パスワードを使用またはリセットしなかった場合に一時パスワードが期限切れになるように、指定された期間のみ認証の試行を許可します。
- 認証が成功した場合、Directory Server は他の操作を実行する前に、ユーザーによるパスワードのリセットを要求します。

デフォルトでは、一時パスワードルールは無効になります。一時パスワードルールは、グローバルまたはローカルパスワードポリシーで設定できます。

20.5.1. グローバルパスワードポリシーでの一時的なパスワードルールの有効化

Directory Server インスタンス全体に対して一時パスワード機能を有効にするには、次の手順を実行します。

1. 管理者がパスワードをリセットした場合にユーザーによるパスワードの変更を必須にします。
2. グローバルパスワードポリシーで一時パスワード機能を設定します。

管理者がユーザーの ***userPassword*** 属性を更新し、***passwordMustChange*** 属性を **on** に設定すると、Directory Server は一時パスワード規則を適用します。

手順

1. 管理者によるパスワードリセット後のユーザーによるパスワード変更を必須に設定します。

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

2. グローバルパスワードポリシーで一時パスワードルールを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy set --pwptprmaxuse
5 --pwptprdelayexpireat 3600 --pwptprdelayvalidfrom 60
```

この例では、以下のように設定されています。

- ***--pwptprmaxuse*** オプションは、ユーザーが一時パスワードを **5** に使用できる最大試行回数を設定します。
- ***--pwptprdelayexpireat*** オプションは、一時パスワードが **3600** 秒 (1 時間) で期限切れになるまでの時間を設定します。
- ***--pwptprdelayvalidfrom*** オプションは、管理者がユーザーのパスワードをリセットした後に、***--pwptprdelayexpireat*** に設定した時間が **60** 秒を起動するように設定します。

検証

- 一時パスワードルールを保存する属性を表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy get | grep -i TPR
passwordTPRMaxUse: 5
passwordTPRDelayExpireAt: 3600
passwordTPRDelayValidFrom: 60
```

20.5.2. ローカルパスワードポリシーでの一時的なパスワードルールの有効化

特定のユーザーまたはサブツリーの一時パスワード機能を有効にするには、管理者によるパスワードリセット時のユーザーによるパスワード変更を必須にして、ローカルパスワードポリシーで機能を設定します。

管理者がユーザーの ***userPassword*** 属性を更新し、***passwordMustChange*** 属性を **on** に設定すると、ユーザーが以下の場合に、Directory Server は一時パスワード規則を適用します。

- ローカルパスワードポリシーが有効になっている
- ローカルパスワードポリシーが有効になっているサブツリーに保存されている

手順

1. 管理者によるパスワードリセット後のユーザーによるパスワード変更を必須に設定します。

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

2. 一時パスワードルール設定を設定します。

   - サブツリーの場合:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp addsubtree --
pwptprmaxuse 5 --pwptprdelayexpireat 3600 --pwptprdelayvalidfrom 60
ou=People,dc=example,dc=com
```

   - ユーザーの場合:

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp adduser --
pwptprmaxuse 5 --pwptprdelayexpireat 3600 --pwptprdelayvalidfrom 60
uid=example,ou=People,dc=example,dc=com
```

   ローカルパスワードポリシーは、存在するエントリーにのみ設定できることに注意してください。

   この例では、以下のようになります。

   - ***--pwptprmaxuse*** オプションは、ユーザーが一時パスワードを **5** に使用できる最大試行回数を設定します。
   - ***--pwptprdelayexpireat*** オプションは、一時パスワードが **3600** 秒 (1 時間) で期限切れになるまでの時間を設定します。
   - ***-pwptprdelayvalidfrom*** オプションは、管理者がユーザーのパスワードをリセットした後に、***--pwptprdelayexpireat*** に設定した時間が **60** 秒を起動するように設定します。

検証

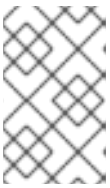
- 識別名 (DN) のローカルパスワードポリシーを表示します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com localpwp get
distinguished_name | grep -i TPR
passwordTPRMaxUse: 5
passwordTPRDelayExpireAt: 3600
passwordTPRDelayValidFrom: 60
```

20.6. パスワードの有効期限コントロールの概要

ユーザーが有効なパスワードを使用して Directory Server に認証し、パスワードの期限が切れたり、リセットする必要がある場合、サーバーは以下の LDAP コントロールをクライアントに送信します。

- 期限切れのコントロール (**2.16.840.1.113730.3.4.4**): パスワードの有効期限が切れていることを示しています。Directory Server は以下の状況でこの制御を送信します。
  - パスワードの失効していますが、猶予のあるログインがなくなりました。サーバーは、**Error 49** メッセージでバインドを拒否します。
  - パスワードは失効していますが、自動ログインは引き続き利用できます。バインドが許可されます。
  - **cn=config** エントリーで ***passwordMustChange*** を有効にし、管理者が変更した後にパスワードをリセットする必要があります。バインドが許可されますが、パスワードの変更以外の後続の操作では、**Error 53** メッセージが表示されます。
- 期限切れコントロール (**2.16.840.1.113730.3.4.5**): パスワードが間もなく期限切れになることを示します。Directory Server は以下の状況でこの制御を送信します。
  - このパスワードは、**cn=config** エントリーの ***passwordWarning*** 属性に設定されたパスワード警告期間内に期限切れとなります。
  - **cn=config** エントリーの ***passwordSendExpiringTime*** 属性でパスワードポリシー設定オプションが有効になっている場合は、パスワードが警告期間内にいるかどうかにかかわらず、期限切れ制御が常に返されます。
- バインド応答制御 (**1.3.6.1.4.1.42.2.27.8.5.1**): この制御には、期限切れ間近のパスワードや、もうすぐ期限切れになるパスワードの状態に関する詳細な情報が含まれています。



注記

Directory Server は、クライアントが要求した場合に限りバインド応答制御を送信します。たとえば、**ldapsearch** を使用する場合は、***-e ppolicy*** パラメーターをコマンドに渡してバインド応答制御を要求する必要があります。

例20.1 Query でのバインド処理コントロールの要求

-e ppolicy パラメーターを **ldapsearch** コマンドに渡すなどしてバインド応答制御を要求する場合、サーバーはアカウントの有効期限に関する詳細情報を返します。以下に例を示します。

```
# ldapsearch -D "uid=user_name,dc=example,dc=com" -xLLL -W \
    -b "dc=example,dc=com" -e ppolicy
ldap_bind: Success (0); Password expired (Password expired, 1 grace logins remain)
```

20.7. DIRECTORY MANAGER パスワードの管理

Directory Manager は特権データベース管理者であり、Linux の **root** ユーザーと類似しています。Directory Manager のエントリーと、対応するパスワードは、インスタンスのインストール時に設定されます。

Directory Manager のデフォルトの識別名 (DN) は **cn=Directory Manager** です。

20.7.1. Directory Manager パスワードのリセット

Directory Manager のパスワードを紛失した場合は、リセットします。

1. Directory Server インスタンスを停止します。

```
# dsctl instance_name stop
```

2. 新しいパスワードハッシュを生成します。以下に例を示します。

```
# pwdhash -D /etc/dirsrv/slapd-instance_name password
{PBKDF2_SHA256}AAAgABU0bKhyjY53NcxY33ueoPjOUWtl4iyYN5uW...
```

Directory Server 設定へのパスを指定すると、***nsslapd-rootpwstoragescheme*** 属性に設定されたパスワードストレージスキームが自動的に使用され、新しいパスワードを暗号化します。

3. **/etc/dirsrv/slapd-*instance_name*/dse.ldif** ファイルを編集し、***nsslapd-rootpw*** 属性を直前の手順で表示された値に設定します。

```
nsslapd-rootpw: {PBKDF2_SHA256}AAAgABU0bKhyjY53NcxY33ueoPjOUWtl4iyYN5uW...
```

4. Directory Server インスタンスを起動します。

```
# dsctl instance_name start
```

20.7.2. Directory Manager パスワードの変更

本セクションでは、Directory Manager アカウントのパスワードを変更する方法を説明します。

20.7.2.1. コマンドラインを使用した Directory Manager パスワードの変更

以下のオプションのいずれかを使用して、新しいパスワードを設定します。

重要

暗号化された接続を使用してパスワードのみを設定します。暗号化されていない接続を使用すると、パスワードをネットワークに公開できます。サーバーが暗号化された接続に対応していない場合は、Web コンソールを使用して Directory Manager のパスワードを更新してください。「Web コンソールを使用した Directory Manager パスワードの変更」を参照してください。

- ***nsslapd-rootpw*** パラメーターを、Directory Server が自動的に暗号化する平文値に設定するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldaps://server.example.com config replace nsslapd-rootpw=password
```

警告

パスワードに中括弧 (**{}**) を使用しないでください。Directory Server はパスワードを **{password-storage-scheme}hashed_password** 形式で保存します。サーバーは、中括弧内の文字をパスワードストレージスキームとして解釈します。文字列が無効なストレージスキームであるか、パスワードが正しくハッシュ化されない場合、Directory Manager はサーバーに接続できません。

- パスワードを手動で暗号化し、***nsslapd-rootpw*** パラメーターに設定するには、以下を実行します。

1. 新しいパスワードハッシュを生成します。以下に例を示します。

```
# pwdhash -D /etc/dirsrv/slapd-instance_name password
{PBKDF2_SHA256}AAAgAMwPYIhEkQozTagoX6RGG5E7d6/6oOJ8TVty...
```

Directory Server 設定へのパスを指定すると、***nsslapd-rootpwstoragescheme*** 属性に設定されたパスワードストレージスキームが自動的に使用され、新しいパスワードを暗号化します。

2. セキュアな接続 (STARTTLS) を使用して、前のステップで表示される値に ***nsslapd-rootpw*** 属性を設定します。

```
# dsconf -D "cn=Directory Manager" ldaps://server.example.com config replace nsslapd-rootpw="{PBKDF2_SHA256}AAAgAMwPYIhEkQozTagoX6RGG5E7d6/6oOJ8TVty..."
```

20.7.2.2. Web コンソールを使用した Directory Manager パスワードの変更

管理者として、パスワードを変更するには、以下の手順を実施します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。

2. インスタンスを選択します。

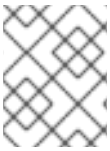
3. **Server Settings** メニューを開き、**Server Settings** を選択します。
4. **Directory Manager** タブを開きます。
5. **Directory Manager Password** フィールドおよび **Confirm Password** フィールドに新規パスワードを入力します。
6. 必要に応じて、異なるパスワードストレージスキームを設定します。
7. **Save** をクリックします。

20.7.3. Directory Manager パスワードストレージスキームの変更

パスワードストレージスキームでは、Directory Server がどのアルゴリズムをパスワードハッシュに使用するかを指定します。コマンドラインを使用してストレージスキームを変更するには、サーバーは暗号化された接続に対応している必要があります。サーバーが暗号化された接続に対応していない場合は、Web コンソールを使用してストレージスキームを設定します。「Web コンソールを使用した Directory Manager パスワードストレージスキームの変更」を参照してください。

Directory Manager (***nsslapd-rootpwstoragescheme***) のストレージスキームは、ユーザーパスワードの暗号化に使用されるスキームと異なる場合があります (***nsslapd-pwstoragescheme***)。

サポート対象のパスワードストレージスキームのリストは、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の該当するセクションを参照してください。



注記

Directory Manager のパスワードストレージスキームを変更する場合は、そのパスワードもリセットする必要があります。既存のパスワードは再暗号化できません。

20.7.3.1. コマンドラインを使用した Directory Manager パスワードストレージスキームの変更

サーバーが暗号化された接続に対応している場合は、以下の手順に従ってパスワードストレージのスキームを変更します。

1. 新しいストレージスキームを使用する新しいパスワードハッシュを生成します。以下に例を示します。

```
# pwdhash -s PBKDF2_SHA256 password
{PBKDF2_SHA256}AAAgAMwPYIhEkQozTagoX6RGG5E7d6/6oOJ8TVty...
```

2. ***nsslapd-rootpwstoragescheme*** 属性をストレージスキームに設定し、***nsslapd-rootpw*** 属性をセキュアな接続 (STARTTLS) を使用して、前の手順で表示した値に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-
rootpwstoragescheme=PBKDF2_SHA256 nsslapd-
rootpw="{PBKDF2_SHA256}AAAgAMwPYIhEkQozTagoX6RGG5E7d6/6oOJ8TVty..."
```

20.7.3.2. Web コンソールを使用した Directory Manager パスワードストレージスキームの変更

Web コンソールを使用してパスワードを変更するには、以下の手順を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** メニューを開き、**Server Settings** を選択します。
4. **Directory Manager** タブを開きます。
5. パスワードストレージスキームを設定します。
6. Directory Server は、新しいストレージスキームを使用して現在のパスワードを再暗号化することができません。したがって、**Directory Manager Password** フィールドおよび **Confirm Password** フィールドに新規パスワードを入力します。
7. **Save Configuration** をクリックします。

20.7.4. Directory Manager DN の変更

管理者として、Directory Manager DN を **cn=New Directory Manager** に変更するには、以下の手順を実施します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-
rootdn="cn=New Directory Manager"
```

Directory Server は、コマンドラインを使用して Directory Manager DN の変更のみをサポートすることに注意してください。

20.8. パスワードなしのアクセスについてのアカウント可用性の確認

多くの場合は、Directory Server がユーザーアカウントの認証情報を返すため、クライアントは実際にそのユーザーとしてバインド (またはバインドを試行) します。また、バインドの試行には、ユーザー認証情報 (通常はパスワードまたは証明書) が必要です。Directory Server は、認証されていないバインドおよび匿名バインドを許可しますが、いずれのバインドもユーザーアカウント情報をすべて返しません。

クライアントが他の操作を行うために、ユーザーアカウントに関する情報、特にそのアカウントの認証を許可するかどうかの情報を必要とする場合がありますが、クライアントは Directory Server にそのユーザーアカウントの認証情報を持っていないか、使用しています。基本的に、クライアントは、ユーザーアカウント情報 (アカウントにパスワードがある場合は、パスワードの有効期限情報を含む) を取得するために、認証情報なしで認証されたバインド操作を実行する必要があります。

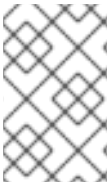
Account Usability Extension Control を渡すと、**ldapsearch** でこれを行うことができます。このコントロールは、あたかも特定のユーザーに対して認証済みのバインド操作を行い、そのユーザーのアカウントステータスを返すかのように動作しますが、実際にはサーバーにバインドすることはありません。これにより、クライアントはそのアカウントがログインに使用できるかどうかを判断し、そのアカウント情報を PAM などの別のアプリケーションに渡すことができます。

たとえば、Account Usability Extension Control を使用すると、システムが Directory Server を ID バックエンドとして使用でき、認証操作が Directory Server の外部で実行されるスマートカードや SSH 鍵などのパスワードなしの認証方法を使用できます。

20.8.1. アカウントのユーザービリティー拡張制御を使用したエントリーの検索

Account Usability Extension Control は **ldapsearch** の拡張機能です。アカウントのステータスと、その

アカウントのパスワードポリシーに関する情報を提供する、返された各エントリーの追加行を返します。次に、クライアントまたはアプリケーションはそのステータスを使用して、そのユーザーアカウントの Directory Server 外で行われた認証の試行を評価できます。基本的に、この制御では、認証操作なしにユーザーが認証を許可するかどうかを制御します。



注記

Directory Server が使用する OpenLDAP ツールは、Account Usability Extension Control に対応していません。OpenDS などの他の LDAP ユーティリティーや、制御をサポートする他のクライアントを使用できます。

例えば、OpenDS ツールを使用する場合、コントロールは、コントロール OID (1.3.6.1.4.1.42.2.27.9.5.8) を持つ **-J**、または **accountusability:true** フラグを使用して指定することができます。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b "dc=example,dc=com"
-s sub -J "accountusability:true" "(objectclass=*)"
# Account Usability Response Control
# The account is usable
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
...
```

これは、特定のエントリーに対して実行することもできます。

```
# ldapsearch -D "cn=Directory Manager" -W -p 389 -h server.example.com -b
"uid=bjensen,ou=people,dc=example,dc=com" -s base -J "accountusability:true" "(objectclass=*)"
# Account Usability Response Control
# The account is usable
dn: uid=bjensen,ou=people,dc=example,dc=com
...
```

注記

デフォルトでは、Directory Manager のみが Account Usability Extension Control を使用できます。他のユーザーが Account Usability Extension Control を使用できるようにするには、**cn=features** でサポート対象のコントロールエントリー上の ACI に設定します。「アカウントのユーザービリティー検索の対象を変更」を参照してください。

このコントロールは、アカウントの実際のステータスや、ユーザーアカウントのパスワードポリシー設定 (ユーザーにパスワードがある場合) に応じて異なるメッセージを返します。

表20.1 アカウント信頼性制御の結果メッセージ

アカウントのステータス	コントロール結果メッセージ
有効なパスワードがあるアクティブなアカウント	The account is usable
パスワードが設定されていないアクティブなアカウント	The account is usable

アカウントのステータス	コントロール結果メッセージ
期限切れのパスワード	Password expired
アカウントのパスワードポリシーが変更される	Password expired
アカウントがロックされ、ロックアウト期間がありません	Password reset
アカウントがロックされ、ロックアウト期間があります	**Time** (in seconds) for automatic unlock of the account
最初のログイン時にアカウントのパスワードをリセットする必要があります	Password reset
パスワードの期限が切れており、猶予期間ログインが許可されます	Password expired and **X** grace login is allowed
パスワードの有効期限が切れ、猶予ログイン回数がなくなっています。	Password expired
パスワードの有効期限が切れます (期限切れの警告)。	Password will expire in **X** number of seconds

20.8.2. アカウントのユーザービリティー検索の対象を変更

デフォルトでは、Directory Manager のみが Account Usability Extension Control を使用できます。他のユーザーは、サポートされるコントロールエントリーに適切な ACI を設定することで、Account Usability Extension Control を使用できます。コントロールエントリーは、Account Usability Extension Control OID (1.3.6.1.4.1.42.2.27.9.5.8) に対して名前が付けられます。

たとえば、**cn=Administrators,ou=groups,dc=example,dc=com** グループのメンバーが、全ユーザーの Account Usability Extension Control を読み取れるようにするには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -x

dn: oid=1.3.6.1.4.1.42.2.27.9.5.8,cn=features,cn=config
changetype: modify
add: aci
aci: (targetattr = "*")(version 3.0; acl "Account Usable"; allow (read)(groupdn =
"ldap:///cn=Administrators,ou=groups,dc=example,dc=com");)
```

20.9. パスワードベースのアカウントロックアウトポリシーの設定

アカウントのロックアウトは、ユーザーのパスワードを繰り返し推測することで、ディレクトリーに分割を試行するハッカーから保護されます。パスワードポリシーを設定して、特定のユーザーが指定した数のバインドの試行後にディレクトリーからロックされるようにできます。

20.9.1. コマンドラインを使用したアカウントロックアウトポリシーの設定

dsconf pwpolicy set コマンドを使用して、アカウントロックアウトポリシーを設定します。例えば、ロックアウトポリシーを有効にして、ログインに 4 回失敗するとアカウントがロックされるように設定するには、次のようにします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com pwpolicy set --pwdlockout on --pwdmaxfailures=4
```

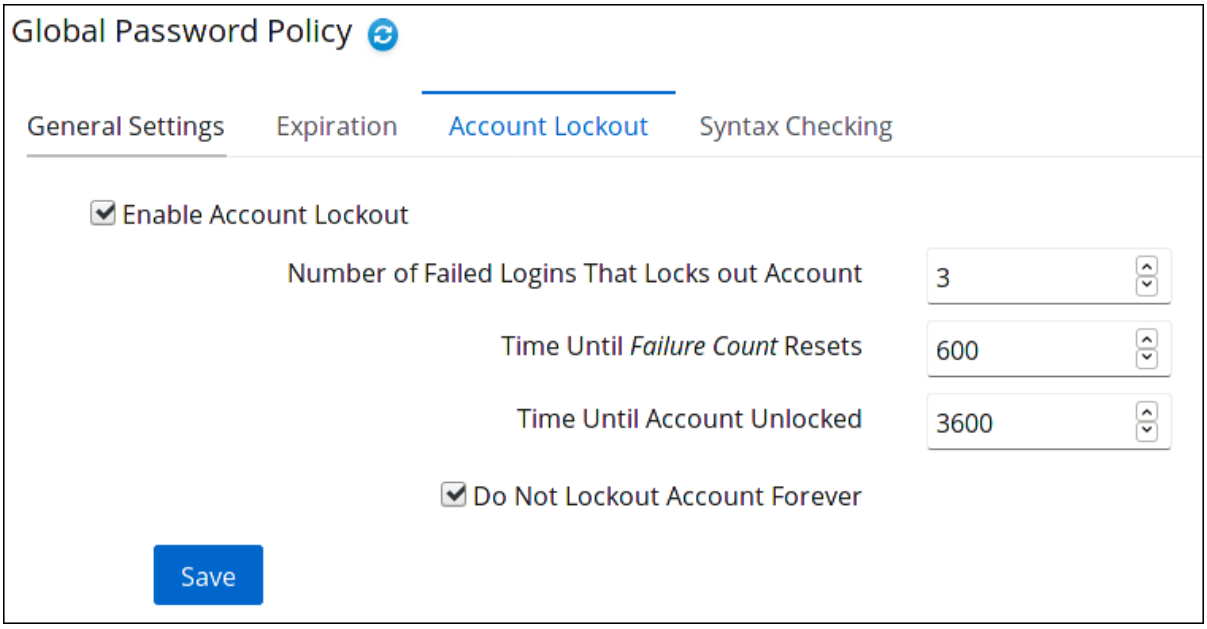
以下のパラメーターは、アカウントパスワードポリシーを制御します。

- ***--pwdlockout***: アカウントロックアウト機能を有効または無効にするには、このパラメーターを **on** または **off** に設定します。
- ***--pwdunlock***: ロックアウトの期間後にアカウントのロックを解除するには、このパラメーターを **on** に設定します。
- ***--pwdlockoutduration***: アカウントがロックされる秒数を設定します。
- ***--pwdmaxfailures***: アカウントがロックされるまでの、失敗したパスワードの最大試行回数を設定します。
- ***--pwdresetfailcount***: Directory Server がアカウントの失敗したログイン数をリセットするまでの秒数を設定します。

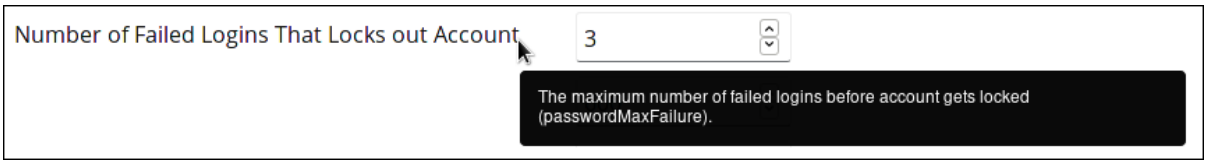
20.9.2. Web コンソールを使用したアカウントロックアウトポリシーの設定

Web コンソールを使用してアカウントロックアウトポリシーを設定するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Database** タブを開き、**Global Password Policy** を選択します。
4. **Account Lockout** タブでは、**Enable Account Lockout** 設定を有効にしてパラメーターを設定します。以下に例を示します。



ツールヒントとパラメーターの **cn=config** エントリーで対応する属性名を表示するには、設定の上にマウスのカーソルを合わせます。詳細は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 のパラメーターの説明を参照してください。



5. **Save** をクリックします。

20.9.3. レガシーパスワードロックアウト動作の無効化

パスワードの最大失敗 (***passwordMaxFailure***) に達すると、解釈する方法は複数あります。これは、サーバーが最後の失敗を全体の失敗数にどのようにカウントするかによります。

LDAP クライアントの従来の動作は、制限に達した **後に** 障害が発生した場合を想定することです。つまり、失敗回数を 3 回に設定すると、4 回目の失敗でロックアウトされます。これは、4 回目の試みが成功した場合、技術的には失敗の限界に達していたとしても、ユーザーは正常に認証できることを意味します。これは、カウントの **n+1** です。

LDAP クライアントは、最大失敗制限を増やして、最後の失敗試行を最終的な試行としてカウントすることを期待します。そのため、障害の上限が 3 に設定されている場合、3 番目の障害によりアカウントはロックされます。4 番目の試行では、正しい認証情報を使用しても失敗します。これはカウントの **n** です。

最初のシナリオ (試行回数が超過した場合にのみアカウントがロックされる) は過去の動作であるため、これは従来のパスワードポリシーの動作と考えられます。Directory Server では、このポリシーはデフォルトで有効になっているため、障害数が **n+1** の場合のみアカウントがロックされます。このレガシー動作を無効にして、新しい LDAP クライアントが予想される際にエラー (LDAP_CONSTRAINT_VIOLATION) を受け取るようにすることができます。これは ***passwordLegacyPolicy*** パラメーターで設定されます。

レガシーパスワードのロックアウト動作を無効にするには、次のコマンドを実行します。

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

20.10. 時間ベースのアカウントロックアウトポリシーの設定

認証に失敗した場合にアカウントをロックする以外にも、アカウントの非アクティブ化やアカウントのエイジに基づいてアカウントロックアウトポリシーを定義する方法があります。アカウントポリシープラグインは、**相対** 時間設定を使用してアカウントをロックする必要があるかどうかを判断します。

注記

サービスのロールまたはクラスは、**絶対** アカウント時間に基づいてアカウントを非アクティブにするのに使用できます。たとえば、特定の日付の前に作成されたすべてのアカウントで CoS を作成できます。

アカウントポリシープラグインには、3 つの設定エントリーが必要です。

- プラグイン自体の設定エントリー。これにより、そのサーバーに設定されたすべてのアカウントポリシーに使用されるグローバル値が設定されます。

- アカウントポリシー設定エントリー。このエントリーはユーザーディレクトリー内にあり、基本的にはユーザーアカウントエントリーに参照および適用されるテンプレートとなります。
- アカウントポリシーエントリーを適用するエントリー。ユーザーアカウントは、直接アカウントポリシーを参照したり、CoS またはロールを使用してアカウントポリシーを自動的にユーザーアカウントのセットに適用することができます。

注記

アカウントポリシーは、***acctPolicySubentry*** 属性を介して適用されます。この属性はユーザーアカウントに直接追加できますが、この属性は単値になります。つまり、そのアカウントにはアカウントポリシーを 1 つだけ適用できます。

これはほとんどの場合で問題ありません。しかし、現実的には、2 つのアカウントポリシーを作成することができます。1 つはアカウントの非アクティブ化のため、もう 1 つは年齢に基づくアカウントの失効のためです。

CoS を使用してアカウントポリシーを適用すると、複数のアカウントポリシーをアカウントに使用できます。

20.10.1. アカウントポリシープラグインの構文

Account Policy プラグイン自体には 2 つの設定属性のみがあります。

- **nsslapd-pluginEnabled**: プラグインが有効かどうかを設定します。この属性はデフォルトで **off** です。
- **nsslapd-pluginarg0**: プラグイン設定ディレクトリーの DN を参照します。設定エントリーは、通常プラグイン自体の子エントリーです (例: **cn=config,cn=Account Policy Plugin,cn=plugins,cn=config**)。

そのため、アカウントポリシーは 2 つの部分で定義されます。

- **nsslapd-pluginarg0** 属性で特定されたプラグイン設定エントリー。これにより、アカウントポリシー設定エントリーの特定やユーザーアカウントエントリーの管理に使用するプラグインのグローバル設定が設定されます。これらの設定はサーバー全体に適用されます。

  設定エントリー属性は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 の 『アカウントポリシープラグインの属性』 セクションで説明されています。

- アカウントポリシーの設定エントリー。これは、アカウントポリシーに特定の値を設定するテンプレートエントリーとよく似ています。ユーザーアカウントは、直接または CoS エントリーを介して、このアカウントポリシーエントリーを参照します。

  アカウントポリシーとユーザーエントリーの属性については、以下の表で説明されています。

表20.2 アカウントポリシーエントリーおよびユーザーエントリーの属性

属性	定義	設定またはユーザーエントリー
accountpolicy (オブジェクトクラス)	アカウントの無効化または期限切れポリシーのテンプレートエントリーを定義します。	設定

属性	定義	設定またはユーザーエントリー
accountInactivityLimit (属性)	アカウントの最終ログイン時刻から、非アクティブ時にアカウントがロックされるまでの時間を秒単位で設定します。	設定
acctPolicySubentry (属性)	アカウントのポリシー (具体的には、アカウントロックアウトポリシー) に属するエントリーを指定します。この属性の値は、エントリーに適用されるアカウントポリシーの DN を参照します。	ユーザー
createTimestamp (操作属性)	エントリーが最初に作成された日時が含まれます。	ユーザー
lastLoginTime (操作属性)	指定のアカウントがディレクトリーに対して認証された最終時刻のタイムスタンプが含まれます。	ユーザー

詳細は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 の属性の説明を参照してください。

20.10.2. アカウントアクティビティーとアカウントの有効期限

Account Policy プラグインを使用すると、以下を設定できます。

- アカウントの有効期限: アカウントの作成後に一定時間無効になります。
- アカウントの非アクティブ化: 最後にログインに成功してから一定時間が経過すると、アカウントが無効になります。これにより、未使用のアカウントを自動的に無効にできます。

無効にされたアカウントはログインできなくなります。

Account Policy プラグインを設定するには、以下を実行します。

1. アカウントポリシープラグインを有効にします。

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

2. プラグイン設定エントリーを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin account-policy set --
config-entry="cn=config,cn=Account Policy Plugin,cn=plugins,cn=config"
```

3. プラグイン設定エントリーを作成します。

- アカウントポリシーで CoS またはロールを使用するには、***alwaysRecordLogin*** の値を **yes** に設定します。これは、***acctPolicySubentry*** 属性がない場合でも、すべてのエントリーにログイン時間が記録されることを意味します。

- アカウントポリシー評価に使用するプライマリー属性を ***stateAttrName*** の値として設定します。アカウントの停止状態の場合は、**lastLoginTime** 属性を使用します。単純なアカウントの有効期限の場合は、**createTimestamp** 属性を使用します。
- ***altStateAttrName*** にセカンダリー属性を設定できます。これは、***stateAttrName*** で定義されたプライマリー属性が存在しない場合にチェックできます。属性を指定していない場合は、デフォルト値の **createTimestamp** が使用されます。

警告

プライマリー属性の値が **lastLoginTime** と ***altStateAttrName*** で **createTimestamp** に設定されていると、既存の環境のユーザーは、アカウントに **lastLoginTime** 属性がなく、設定した非アクティブ期間よりも ***createTimestamp*** が古い場合に、既存の環境のユーザーは自動的にロックされます。

この状況に対処するには、代替属性を **1.1** に設定します。これは、代替として属性を使用しないことを明示します。***lastLoginTime*** 属性は、ユーザーが次回ログインした後に自動的に作成されます。

- アカウントポリシーを適用するエントリーを表示するのに使用する属性を設定します (**acctPolicySubentry**)。
- 実際のタイムアウト期間の設定に使用されるアカウントポリシーの属性を秒単位で設定します (**accountInactivityLimit**)。

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

4. サーバーを再起動して、新しいプラグイン設定を読み込みます。

```
# dsctl instance_name restart
```

5. アカウントポリシーを定義します。

```
# ldapadd -a -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=Account Inactivation Policy,dc=example,dc=com

objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: accountpolicy
accountInactivityLimit: 2592000
cn: Account Inactivation Policy
```

6. サービステンプレートエントリーのクラスを作成します。

```
# ldapadd -a -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=TempltCoS,dc=example,dc=com

objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: cosTemplate
acctPolicySubentry: cn=Account Inactivation Policy,dc=example,dc=com
```

アカウントポリシーは、CoS を使用する代わりに、ユーザーエントリーで直接定義できます。しかし、CoS を使用することで、複数のエントリーに対して確実にアカウントポリシーを適用および更新することができ、1 つのエントリーに複数のポリシーを適用することができます。

7. サービス定義エントリーのクラスを作成します。CoS の管理エントリーはアカウントポリシー属性 ***acctPolicySubentry*** です。この例では、CoS をディレクトリーツリー全体に適用します。

```
# ldapadd -a -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=DefnCoS,dc=example,dc=com

objectClass: top
objectClass: ldapsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=TempltCoS,dc=example,dc=com
cosAttribute: acctPolicySubentry default operational-default
```

20.10.3. パスワード失効後の特定期間のアカウントの無効化

Directory Server では、パスワードの期限が切れてから一定期間アカウントを無効にするアカウントポリシーを設定できます。無効にしたアカウントはログインできなくなります。

この設定を設定するには、「アカウントアクティビティーとアカウントの有効期限」の手順に従います。ただし、プラグイン設定エントリーを設定する場合は、代わりに以下の設定を使用してください。

```
dn: cn=config,cn=Account Policy Plugin,cn=plugins,cn=config

objectClass: top
objectClass: extensibleObject
cn: config
alwaysrecordlogin: yes
stateAttrName: non_existent_attribute
altStateAttrName: passwordExpirationTime
specattrname: acctPolicySubentry
limitattrname: accountInactivityLimit
```

この設定では、***stateAttrName*** パラメーターにダミー値を使用します。したがって、***altStateAttrName*** パラメーターで設定した **passwordExpirationTime** 属性のみが、アカウントの期限が切れるタイミングを算出するために使用されます。

また、ユーザーエントリーの ***lastLoginTime*** 属性に最後に成功したログインの時間を記録するため、以下を設定します。

-

```
dn: cn=config,cn=Account Policy Plugin,cn=plugins,cn=config

alwaysRecordLoginAttr: lastLoginTime
```

この設定では、ユーザーの***passwordExpirationTime*** 属性と***accountInactivityLimit*** パラメーターの値に設定されている時間の合計が過去になった場合は、アカウントが自動的に無効になります。この設定では、ユーザーの ***passwordExpirationTime*** 属性と ***accountInactivityLimit*** パラメーターの値の合計が、***alwaysRecordLoginAttr*** 属性が最後に更新されてからの時間を超えた場合は、アカウントが自動的に無効になります。

20.10.4. ロックアウトポリシーを設定しないログイン時間の追跡

アカウントポリシープラグインを使用して、有効期限や非アクティブ期間を設定 **しなくても**、ユーザーのログイン時間を追跡することもできます。この場合、アカウントポリシープラグインは ***lastLoginTime*** 属性をユーザーエントリーに追加するために使用されますが、他のポリシールールを設定する必要はありません。

その場合には、アカウントポリシープラグインを通常どおりに設定して、ログイン時間を追跡します。ただし、追跡中のログイン情報に作用する CoS は作成しないでください。

1. アカウントポリシープラグインを有効にします。

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

2. ログイン時間を記録するプラグイン設定エントリーを作成します。

   - すべてのエントリーにログイン時間が記録されるように、***alwaysRecordLogin*** の値を yes に設定します。
   - ***lastLoginTime*** 属性をアカウントポリシー (***stateattrname***) に使用する属性として設定します。
   - アカウントポリシーを適用するエントリーを表示するのに使用する属性を設定します (***acctPolicySubentry***)。
   - 実際のタイムアウト期間 (秒単位) を設定するのに使用されるアカウントポリシーの属性を設定します (***accountInactivityLimit***)。

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

3. サーバーを再起動して、新しいプラグイン設定を読み込みます。

```
# dsctl instance_name restart
```

20.10.5. Inactive アカウントのロックの解除

非アクティブ制限に達したためにアカウントがロックされた場合は、以下のいずれかの方法で再アクティベートできます。

- **dsidm** ユーティリティーの使用:

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
account unlock "uid=example"
```

- ***lastLoginTime*** 属性を現在のタイムスタンプに手動でリセット:

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: uid=example,ou=people,dc=example,dc=com
changetype: modify
replace: lastLoginTime
lastLoginTime: 20210901000000Z
```

タイムスタンプに **Z** が追加されているため、***lastLoginTime*** 属性は値を GMT/UTC 時間 (Zulu タイムゾーン) で保存することがわかります。

20.11. アカウントロックアウト属性の複製

アカウントのロックアウトポリシーにより、ログイン試行が失敗した回数を超えると、ユーザー ID が Directory Server にアクセスできなくなります。これにより、ハッカーやその他の悪意のあるユーザーは、パスワードを推測することで Directory Server に不正にアクセスできなくなります。パスワードポリシーはローカルに設定され、通常、アカウントロックアウト属性は各レプリカに対してローカルになります。つまり、アカウントのロックアウト回数に達するまでは、あるレプリカにログインを試み、すぐに別のレプリカで再試行することができるのです。それを防ぐには、アカウントのロックアウト回数に関連する属性をエントリーに複製し、1 つのサプライヤーでログイン試行に失敗した場合に、悪意のあるユーザーが設定内のすべてのサプライヤーとコンシューマーのレプリカからロックアウトされるようにします。

デフォルトでは、他のパスワード属性がある場合でも、3 つのパスワードポリシー属性は複製されません。これらの属性は、ログインの失敗およびロックアウト期間に関連します。

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

20.11.1. アカウントロックアウトおよびレプリケーションの管理

パスワードとアカウントのロックアウトポリシーの適用は、複製された環境では若干異なります。

- パスワードポリシーはデータサプライヤーで実施されます。
- アカウントロックアウトは、レプリケーションに参加するすべてのサーバーに適用されます。

ディレクトリー内のパスワードポリシー情報の一部は、自動的に複製されます。

- ***passwordMinAge*** および ***passwordMaxAge***
- ***passwordExp***
- ***passwordWarning***

ただし、設定情報はローカルに保持され、複製されません。この情報には、パスワード構文とパスワード変更の履歴が含まれます。アカウントロックアウトカウンターおよび層は、特にレプリケーション用に設定されていない限り複製されません。

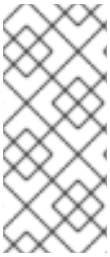
複製された環境でパスワードポリシーを設定する場合は、これらの要素が有効であることを確認し、パスワードポリシーとアカウントロックアウト設定が一貫して実行されるようにします。

- パスワードの有効期限が迫っていることを示すサーバーからの警告は、すべてのレプリカで発行されます。この情報は、各サーバーにローカルに保存されるため、ユーザーが複数のレプリカにバインドされた場合は、同じ警告が複数回発行されます。また、ユーザーがパスワードを変更した場合は、その情報がレプリカに反映されるまでに時間がかかることがあります。ユーザーがパスワードを変更してすぐに再バインドすると、レプリカが変更を登録するまでバインドに失敗することがあります。

- サプライヤーやレプリカなど、すべてのサーバーで同じバインド動作が発生する必要があります。各サーバーで同じパスワードポリシー設定情報を作成してください。

- アカウントロックアウトカウンターは、マルチサプライヤー環境で期待どおりに機能しない可能性があります。アカウントのロックアウトカウンターは、デフォルトでは複製されません (ただし、設定は可能です)。アカウントのロックアウト属性がまったく複製されない場合、あるユーザーがあるサーバーからロックアウトされていても、別のサーバーには正常にバインドできる可能性があります (または、あるサーバーではロックが解除されていても、別のサーバーではブロックされている場合もあります)。アカウントロックアウト属性が複製されると、アカウントのロックアウトの変更と、その変更が他のサーバーに伝播されるときにラグが発生することがあります。これはレプリケーションのスケジュールにより異なります。

- レプリケーション用に作成されるエントリー (例: サーバーアイデンティティー) には有効期限のないパスワードが必要です。これらの特別なユーザーに有効期限のないパスワードがあることを確認するには、エントリーに ***passwordExpirationTime*** 属性を追加し、その値を **20380119031407Z** 有効な範囲内に) 指定します。



注記

パスワードポリシーが有効になり、***alwaysRecordLogin*** パラメーターが **yes** に設定されている場合、***lastLoginTime*** 属性の値は、サプライヤーと読み取り専用レプリカで異なる場合があります。たとえば、ユーザーが読み取り専用のレプリカにログインすると、***lastLoginTime*** 属性はローカルに更新されますが、値はサプライヤーサーバーに複製されません。

20.11.2. パスワードポリシー属性を複製する Directory Server の設定

特別なコア設定属性は、パスワードポリシーの操作属性が複製されるかどうかを制御します。これは、コンシューマー Directory Server 設定で有効になっている ***passwordIsGlobalPolicy*** 属性で、コンシューマーがパスワードポリシーの操作属性を受け入れるようにします。

デフォルトでは、この属性は **off** に設定されます。

これらの属性を複製できるようにするには、コンシューマーで ***passwordIsGlobalPolicy*** 設定パラメーターを変更します。

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

この値を **on** に変更すると、***passwordRetryCount***、***retryCountResetTime***、および ***accountUnlockTime*** が複製されます。

20.11.3. パスワードポリシー属性に対する一部レプリケーションの設定

passwordIsGlobalPolicy 属性を設定すると、コンシューマーがそれらの属性への更新を受け取ることができるように、レプリケーションのコンシューマーに影響します。パスワードポリシー属性がサプラ

イヤーによって実際に複製されるかどうかを制御するには、特定のエントリー属性が複製されるものを制御する一部レプリケーションを使用します。

パスワードポリシー属性を複製する必要がある場合は (デフォルトでは設定) 一部レプリカ合意にこれらの属性が含まれていることを確認してください。

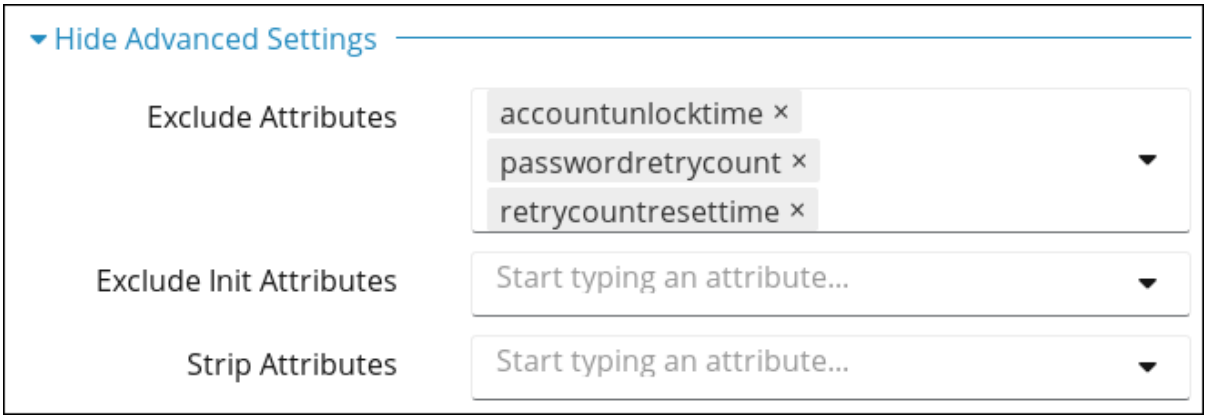
コンシューマーで ***passwordIsGlobalPolicy*** 属性が **off** に設定されているため、パスワードポリシー属性を複製する必要がない場合は、一部レプリケーション (「一部レプリケーションを使用した属性のサブセットの複製」で説明) を使用してサプライヤーでレプリケーションを強制し、それらの属性をレプリカ合意から明確に除外します。

レプリケーションの設定に関する詳細は、以下を参照してください。

- 「単一サプライヤーレプリケーション」
- 「マルチサプライヤーのレプリケーション」
- 「カスケードレプリケーション」

上記のリンクでレプリカ合意を作成する場合は、一部レプリケーションを設定します。

1. サプライヤーにレプリカ合意を設定する場合は、**Show Advanced Settings** をクリックします。
2. **Exclude Attributes** フィールドに ***passwordRetryCount*** 属性、***retryCountResetTime*** 属性、および ***accountUnlockTime*** 属性の名前を入力します。



3. レプリカ合意の設定を終了します。

20.12. 異なるタイプのバインドの有効化

エンティティーが Directory Server にログインするかアクセスするたびにディレクトリーに **バインド** されます。バインド操作にはさまざまな種類があり、バインドの方法に応じたもの (シンプルバインドやオートバインドなど) や、ディレクトリーにバインドするユーザーのアイデンティティーに応じたもの (匿名バインドや未認証バインド) があります。

以下のセクションでは、バインドのセキュリティーを高めたり (「セキュアなバインドの要求」)、バインド操作を効率化したり (「自動バインドの設定」など) するための設定パラメーターを紹介します。

20.12.1. セキュアなバインドの要求

単純なバインドは、エンティティーが単純なバインド DN とパスワードの組み合わせを使用して Directory Server に対して認証される場合です。コマンドラインからパスワードを直接送信するのでは

なく、パスワードファイルを使用することは可能ですが、いずれの方法でもネットワーク経由で平文のパスワードを送受信する必要があります。これでは、接続を盗聴された場合に、パスワードが脆弱になってしまいます。

セキュアな接続 (TLS または STARTTLS) で単純なバインドを行うことが必要になる場合があります。これにより、バインド操作で送信される平文のパスワードを実質的に暗号化できます。(SASL 認証や証明書ベースの認証など、簡易バインドの代わりに使用することも可能です。)

重要

通常ユーザーは、サーバーおよび LDAP 操作にログインすると、単純なバインドにセキュアな接続を要求することで、サーバー間の接続に影響があります。たとえば、レプリケーション、同期、データベースチェーンはすべて、サーバー間で単純なバインドを使用できます。

nsslapd-require-secure-binds 属性が有効にになっている場合は、レプリカ合意、同期合意、およびチェーン設定がセキュアな接続を指定するようにしてください。それ以外の場合、これらの操作は失敗します。

注記

バインド操作のセキュアな接続を **認証バインド** にのみ適用する必要があります。パスワードのないバインド操作 (匿名および認証されていないバインド) は、標準の接続を引き継ぐことができます。

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

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

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

20.12.2. 匿名バインドの無効化

ユーザー名またはパスワードを指定せずに Directory Server への接続を試みると、これは **匿名バインド** になります。匿名バインドは、ユーザーが最初にディレクトリーに対して認証を行う必要がないため、電話番号や電子メールアドレスをディレクトリーで確認するような、一般的な検索および読み取り操作を簡素化します。

注記

デフォルトでは、匿名バインドは検索操作および読み取り操作に対して許可 (on) されます。これにより、ユーザーおよびグループのエントリーに加えて、root DSE などの設定エントリーを含む **通常のディレクトリーエントリー** にアクセスすることができます。別のオプション **rootdse** により、匿名検索および root DSE 自体への読み取りアクセスが許可されますが、他のすべてのディレクトリーエントリーへのアクセスを制限します。

ただし、匿名バインドにはリスクがあります。機密情報へのアクセスを制限したり、変更や削除などのアクションを許可しないように、適切な ACI を導入する必要があります。さらに、匿名バインドは、サービス拒否攻撃や、悪意のあるユーザーがサーバーへのアクセスを取得するのに使用できます。

「匿名アクセスの付与」は、ACI を設定して匿名ユーザーがアクセスするものを制御する例があり、「匿名バインドでのリソース制限の設定」には、匿名ユーザーのリソース制限の設定に関する情報があります。

このオプションで十分なレベルのセキュリティーが提供されない場合は、匿名バインドを完全に無効にできます。

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

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

2. インスタンスを再起動します。

```
# dsctl instance_name restart
```

注記

匿名バインドが無効の場合、ユーザーは RDN を使用してログインできません。これは、ログインのために完全な DN を提供する必要があります。

さらに、匿名バインドを無効にする場合は、認証されていないバインドも自動的に無効になります。

20.12.3. 認証されていないバインドの許可

認証されていないバインドは、ユーザーが空のパスワードを提供する Directory Server への接続です。Directory Server では、デフォルト設定を使用すると、セキュリティー上の理由から、このシナリオのアクセスを拒否します。

```
# ldapsearch -w "" -p 389 -h server.example.com -b "dc=example,dc=com" \
    -s sub -x "(objectclass=*)"

ldap_bind: Server is unwilling to perform (53)
 additional info: Unauthenticated binds are not allowed
```

警告

Red Hat は、認証されていないバインドを有効にしないことを推奨します。この認証方法により、Directory Manager を含むアカウントとしてパスワードを指定せずにユーザーがバインドできます。バインド後、ユーザーはバインドに使用されるアカウントのパーミッションを持つすべてのデータにアクセスできます。

セキュアでない非認証バインドを有効にするには、***nsslapd-allow-unauthenticated-binds*** 設定オプションを **on** に設定します。

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

20.12.4. 自動バインドの設定

Autobind は、ローカルの UNIX 認証情報に基づいて Directory Server に接続する方法です。これは、ディレクトリー自体に保存されたアイデンティティーにマッピングされます。autobind は、以下の 2 つの部分で設定されます。

autobind を設定する前に、まず LDAPI が有効であることを確認してください。次に、(「**Autobind** 機能の設定」に) autobind マッピングを設定します。

20.12.4.1. Autobind および LDAPI の概要

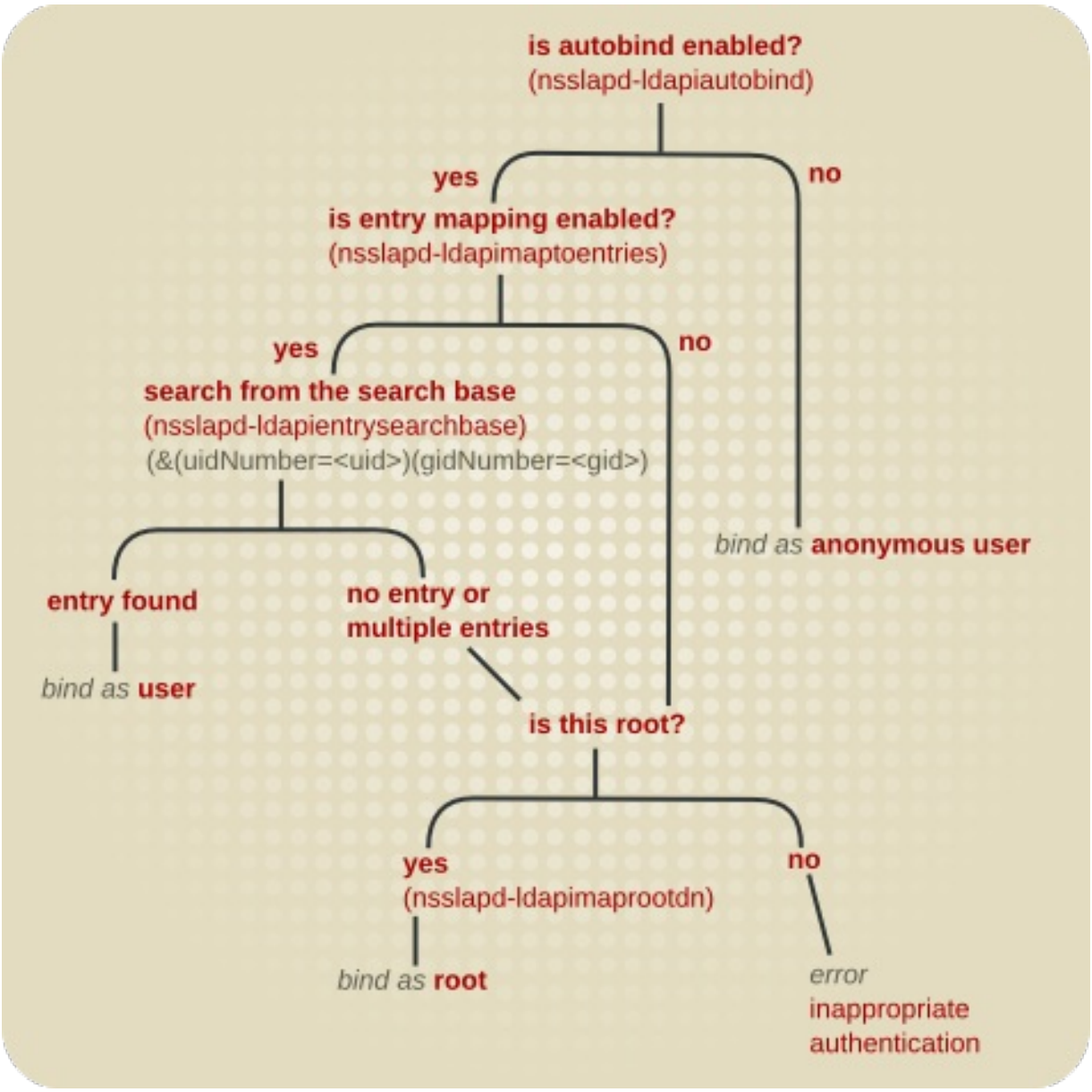
IPC (Inter-process communication) は、Unix マシン上のプロセスやネットワークを区別して相互に直接通信する方法です。**LDAPI** は、これらの IPC 接続で LDAP 接続を実行する方法です。つまり、LDAP 操作は Unix ソケット上で実行できます。これらの接続は、通常の LDAP 接続よりもはるかに高速で、より安全です。

Directory Server はこの LDAPI 接続を使用して、ユーザーがすぐに Directory Server にバインドしたり、Unix ソケットを介した接続をサポートするツールを使用して Directory Server にアクセスできるようにします。autobind は、Unix ユーザーの **uid:gid** を使用して、そのユーザーを Directory Server のエントリーにマッピングし、そのユーザーのアクセスを許可します。

autobind では、3 つのディレクトリーエントリーへのマッピングを許可します。

- Unix ユーザーが 1 つのユーザーエントリーに一致した場合はユーザーエントリー
- Unix ユーザーが **root** または ***nsslapd-ldapimaprootdn*** で定義されているスーパーユーザーである場合は Directory Manager

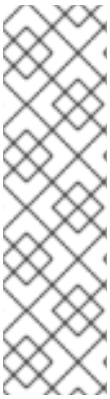
図20.1 自動バインド接続パス



特別な自動バインドユーザーのエントリーは、特別な自動バインド接尾辞の下 (一般ユーザーのサブツリー外) にあります。この下のエントリーは、ユーザーおよびグループの ID 番号で識別されます。

gidNumber=*gid*+uidNumber*uid*, *autobindsuffix*

自動バインドが有効になっていないが LDAPI の場合は、他のバインド認証情報を指定しない限り、Unix ユーザーは Directory Server に匿名でバインドされます。



注記

自動バインドを使用すると、バインドユーザー名とパスワードを指定したり、他の SASL 認証メカニズムを使用したりせずに、クライアントが Directory Server に要求を送信できます。LDAP 標準によると、要求でバインド情報が指定されていない場合、サーバーは要求を匿名バインドとして処理します。何らかのバインド情報を必要とする規格に準拠するため、自動バインドを使用するクライアントは SASL/EXTERNAL で要求を送信する必要があります。

SASL の設定に関する詳細は、「SASL Identity マッピングの設定」 を参照してください。

20.12.4.2. Autobind 機能の設定

Autobind 機能を有効にすると、Directory Server への匿名アクセスのみが許可されます。ただし、Linux ユーザーを Directory Server エントリーにマッピングするように設定すると、**root** ユーザーを Directory Manager にマップすることもできます。

1. ***nsslapd-ldapiautobind*** パラメーターが有効化されていることを確認します。これはデフォルトです。

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

2. ***nsslapd-ldapiautobind*** パラメーターが **off** に設定されている場合は、有効にします。

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

3. ユーザーエントリーをマッピングするには、以下のように設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-ldapimaptoentries=on nsslapd-ldapiuidnumbertype=uidNumber nsslapd-ldapigidnumbertype=gidNumber nsslapd-ldapientrysearchbase=ou=People,dc=example,dc=com
```

   - **nsslapd-ldapimaptoentries=on** は、エントリーマッピングを有効にします。
   - **nsslapd-ldapiuidnumbertype=*uidNumber*** は、Unix UID 番号が含まれる Directory Server の属性を設定します。
   - **nsslapd-ldapigidnumbertype=*gidNumber*** は、Unix GID 番号が含まれる Directory Server の属性を設定します。
   - **nsslapd-ldapientrysearchbase=*ou=People,dc=example,dc=com*** は、ユーザーエントリーを検索する DN を設定します。

4. 必要に応じて、Red Hat Enterprise Linux の **root** ユーザーを Directory Server の **cn=Directory Manager** アカウントにマッピングするには、以下を実行します。

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

5. インスタンスを再起動します。

```
# dsctl instance_name restart
```

20.13. パススルー認証の使用

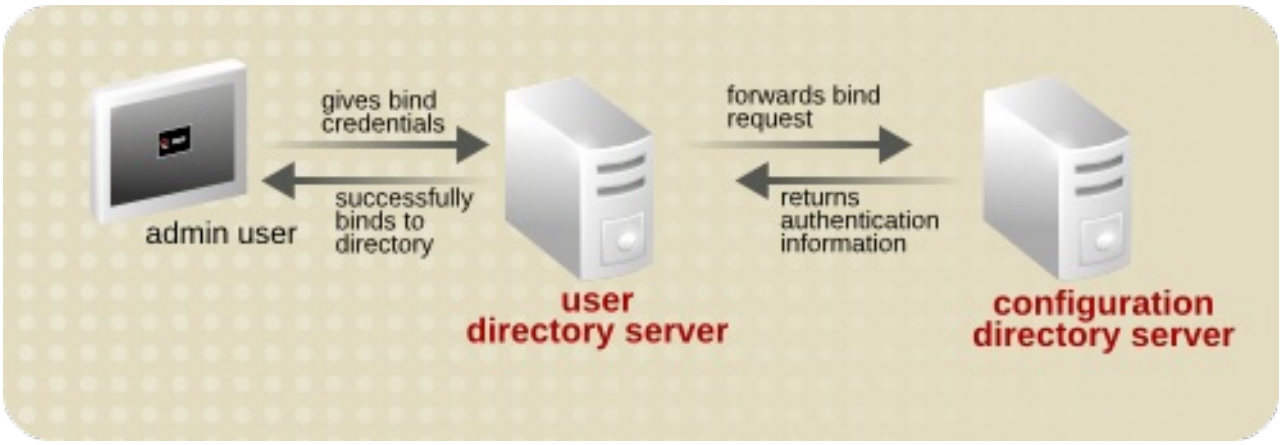
パススルー認証 (PTA) は、ある Red Hat Directory Server インスタンスがバインド要求を認証できるメカニズムです。パススルー認証は PTA プラグインを介して実装されます。有効にすると、Directory Server インスタンスは、ローカルデータベースに保管されないエントリーの単純なバインド操作 (パスワードベース) を受け入れます。

Directory Server は、PTA を使用して、Directory Server の別のインスタンスでユーザーおよび設定ディレクトリーを管理します。

最初のインスタンスは、別の Directory Server へのバインド要求を通過する PTA Directory Server として機能します。2 つ目のインスタンスは、認証ディレクトリーとして機能します。これは、エントリーが含まれるサーバーであり、要求しているクライアントのバインド認証情報を検証します。

パススルーサブツリー は、PTA ディレクトリーに **存在しない** サブツリーです。ユーザーのバインド DN にこのサブツリーが含まれる場合、ユーザーの認証情報が認証ディレクトリーに渡されます。

図20.2 簡易的な PAM パススルー認証プロセス



パススルー認証が機能する仕組みを以下に示します。

1. 設定 Directory Server (認証ディレクトリー) がマシン A にインストールされています。設定ディレクトリーには、認証するユーザーエントリーを含む接尾辞 **o=RedHat** など) が常に含まれます。この例では、サーバー名は **authdir.example.com** です。
2. ユーザー Directory Server (PTA ディレクトリー) がマシン B にインストールされます。ユーザーディレクトリーは、**dc=example,dc=com** などのルート接尾辞を保存します。この例では、サーバー名は **userdir.example.com** です。
3. 以下のコマンドを使用して **userdir.example.com** にプラグインを設定します。

```
# dsconf -D "cn=Directory Manager" ldap://userdir.example.com plugin pass-through-auth enable
# dsconf -D "cn=Directory Manager" ldap://userdir.example.com plugin pass-through-auth url add "ldap://authdir.example.com/o=RedHat"
```

4. **userdir.example.com** で Directory Server を再起動します。
5. ユーザーディレクトリーは、**o=RedHat** が含まれる DN を持つエントリーのすべてのバインド要求を設定ディレクトリー **authdir.example.com** に送信するように設定されるようになりました。
6. ユーザーディレクトリーでは、任意のユーザーが **o=RedHat** からバインドできるようになります。

20.13.1. PTA プラグインの構文

PTA プラグインの設定情報は、必要な PTA 構文を使用して PTA ディレクトリーの **cn=Pass Through Authentication**、**cn=plugins,cn=config** エントリー (認証ディレクトリーへのバインド要求をパススルーするように設定されたユーザーディレクトリー) に指定されます。

以下のコマンドを使用して、パススルー認証 URL を管理します。

- パススルー認証 URL を追加するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin pass-through-auth url add URL
```

- パススルー認証 URL を変更するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin pass-through-auth url modify old_URL new_URL
```

- パススルー認証 URL を削除するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin pass-through-auth url delete URL
```

PTA プラグイン構文の変数コンポーネントは、表20.3「PTA プラグインのパラメーター」 で説明されています。

注記

LDAP URL (**ldap|ldaps://authDS/subtree**) は、1 つの空白で任意のパラメーター (**maxconns、maxops、timeout、ldver、connlifetime、startTLS**) から分離する必要があります。任意のパラメーターのいずれかが定義される場合は、デフォルト値のみが使用されている場合でも、それらをすべて定義する必要があります。

「複数の認証用 Directory Server の指定」 にあるように、***nsslapd-pluginarg*** 属性接尾辞を 1 つずつ増やすことで、いくつかの認証ディレクトリーまたはサブツリーを指定できます。以下に例を示します。

```
nsslapd-pluginarg0: LDAP URL for the first server
nsslapd-pluginarg1: LDAP URL for the second server
nsslapd-pluginarg2: LDAP URL for the third server
...
```

任意のパラメーターは、構文で表示される順序で以下の表で説明されています。

表20.3 PTA プラグインのパラメーター

変数	定義
state	プラグインを有効または無効にするかどうかを定義します。使用できる値は **on** または **off** です。
ldap|ldaps	2 つの Directory Server 間の通信に TLS を使用するかどうかを定義します。詳細は、「セキュアな接続を使用するようにサーバーを設定」を参照してください。

変数	定義
authDS	認証ディレクトリーのホスト名。Directory Server のポート番号は、コロンとポート番号を追加して指定できます。たとえば、**ldap://dirserver.example.com:389/** です。ポート番号が指定されていない場合、PTA サーバーは標準ポートのいずれかを使用して接続を試みます。 URL に **ldap://** が指定されている場合のポート 389。 URL に **ldaps://** が指定されている場合のポート 636。 詳細は、「認証する Directory Server の指定」を参照してください。
subtree	**パススルーサブツリー。**PTA Directory Server は、このサブツリーに DN を持つすべてのクライアントから、認証する Directory Server にバインド要求を渡します。詳細は、「パススルーサブツリーの指定」を参照してください。このサブツリーは、このサーバーに存在させることはできません。
maxconns	**任意**。PTA ディレクトリーの最大接続数は、認証ディレクトリーに対して同時に開くことができます。デフォルトは **3** です。詳細は、「オプションパラメーターの設定」を参照してください。
maxops	**任意**。PTA ディレクトリーは単一接続内の認証ディレクトリーに送信できる同時操作の最大数 (通常はバインド要求)。デフォルトは **5** です。詳細は、「オプションパラメーターの設定」を参照してください。
timeout	**任意**。PTA ディレクトリーが、認証用ディレクトリーサーバーからの応答を待つ時間を秒単位で指定します。このタイムアウトを超えると、サーバーはエラーをクライアントに返します。デフォルトは **300** 秒 (5 分) です。ゼロ (**0**) を指定すると、時間制限をかけないことを示します。詳細は、「オプションパラメーターの設定」を参照してください。
ldver	**任意**。認証用ディレクトリーへの接続に使用される LDAP プロトコルのバージョン。Directory Server は LDAP バージョン 2 および 3 をサポートします。デフォルトはバージョン 3 です。Red Hat は、古くなり、非推奨になる LDAPv2 を使用 **しない** ことを強く推奨します。詳細は、「オプションパラメーターの設定」を参照してください。
connlifetime	**任意**。接続を使用できる制限時間を秒単位で指定します。この時間が経過した後にクライアントからバインド要求が開始すると、サーバーは接続を閉じ、認証するディレクトリーへの新しい接続を開きます。バインド要求が開始し、ディレクトリーが接続寿命を超えたと判断しない限り、サーバーは接続を閉じません。このオプションを指定しない場合、または 1 つのホストのみが記載されている場合は、接続の有効期間は実行されません。2 つ以上のホストがリストされている場合、デフォルトは **300** 秒 (5 分) です。詳細は、「オプションパラメーターの設定」を参照してください。

変数	定義
startTLS	**任意**。認証用ディレクトリーへの接続に STARTTLS を使用するかどうかを示すフラグ。STARTTLS は標準ポート上でセキュアな接続を確立するため、LDAPS の代わりに LDAP を使用して接続するのに便利です。TLS サーバーと CA 証明書の両方がサーバーで使用できる必要があります。 デフォルトは **0** (off) です。STARTTLS を有効にするには、**1** に設定します。STARTTLS を使用するには、LDAP URL は **ldaps:** ではなく **ldap:** を使用する必要があります。 詳細は、「オプションパラメーターの設定」を参照してください。

20.13.2. PTA プラグインの設定

PTA 設定を変更するには、以下を実行します。

1. **dsconf plugin pass-through-auth** を使用して、**cn=Pass Through Authentication,cn=plugins,cn=config** エントリーを変更します。
2. Directory Server を再起動します。

PTA プラグインパラメーターを設定する前に、Directory Server に PTA プラグインエントリーが必要です。このエントリーが存在しない場合は、「PTA プラグインの構文」 で説明されているように、適切な構文で作成します。

注記

ユーザーと設定ディレクトリーがそのディレクトリーの別のインスタンスにインストールされている場合は、PTA プラグインエントリーがユーザーディレクトリーの設定に自動的に追加され、有効になります。

本セクションでは、以下のセクションでプラグインを設定する方法を説明します。

- 「セキュアな接続を使用するようにサーバーを設定」
- 「認証する Directory Server の指定」
- 「パススルーサブツリーの指定」
- 「オプションパラメーターの設定」

20.13.2.1. セキュアな接続を使用するようにサーバーを設定

PTA ディレクトリーは、PTA ディレクトリーの LDAP URL に LDAPS を指定して、TLS 経由で認証ディレクトリーと通信するように設定できます。以下に例を示します。

```
nsslapd-pluginarg0: ldaps://ldap.example.com:636/o=example
```

20.13.2.2. 認証する Directory Server の指定

認証用ディレクトリーには、クライアントがバインドしようとしているエントリーのバインド認証情報が含まれます。PTA ディレクトリーは、認証ディレクトリーとして定義されたホストにバインド要求を渡します。認証する Directory Server を指定するには、表20.3「PTA プラグインのパラメーター」 で説

明されているように、PTA ディレクトリーの LDAP URL の **authDS** を、認証するディレクトリーのホスト名に置き換えます。

1. **dsconf plugin pass-through-auth** コマンドを使用して PTA プラグインエントリーを編集します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin pass-through-auth add
ldap://server.example.com/o=example
```

必要に応じて、ポート番号を含めます。ポート番号が指定されていない場合、PTA Directory Server は **ldap://** の標準ポート (389) または **ldaps://** のセキュアなポート (636) を使用して接続を試みます。

PTA Directory Server と認証する Directory Server との間の接続が破損するか、接続を開始できない場合は、PTA Directory Server が、指定された次のサーバー (存在する場合) に要求を送信します。最初の Directory Server が利用できない場合にフェイルオーバーを提供するために、必要に応じて認証する複数の Directory Server を指定できます。認証するすべての Directory Server が ***nsslapd-pluginarg0*** 属性に設定されます。

認証する複数の Directory Server は、以下の形式で、スペース区切りの **host:port** ペアのリストで記述されます。

```
ldap|ldaps://host1:port1 host2:port2/subtree
```

2. サービスを再起動します。

```
# dsctl instance_name restart
```

20.13.2.3. パススルーサブツリーの指定

PTA ディレクトリーは、パススルーサブツリーで定義された DN を持つすべてのクライアントからの認証要求を、バインド要求を渡します。サブツリーを指定するには、PTA ディレクトリーの LDAP URL の **subtree** パラメーターを置き換えて指定します。

パススルーのサブツリーは PTA ディレクトリーに存在すべきではありません。そうすると、PTA ディレクトリーは自分のディレクトリーの内容を使用してバインド要求を解決しようとするため、バインドが失敗します。

1. **dsconf plugin pass-through-auth** コマンドを使用して、LDIF ファイルをディレクトリーにインポートします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin pass-through-auth add
ldap://server.example.com/o=example
```

この構文の変数コンポーネントの詳細は、表20.3「PTA プラグインのパラメーター」 を参照してください。

2. サービスを再起動します。

```
# dsctl instance_name restart
```

20.13.2.4. オプションパラメーターの設定

PTA 接続の制御に使用する追加のパラメーターは LDAP URL で設定できます。

```
ldap|ldaps://authDS/subtree maxconns, maxops, timeout, ldver, connlifetime, startTLS
```

- PTA Directory Server が認証ディレクトリーに対して同時に開くことができる最大の接続数で、PTA の構文では **maxconns** で表されます。デフォルト値は **3** です。
- PTA Directory Server が単一の接続内の認証する Directory Server に同時に送信できるバインド要求の最大数。PTA 構文では、このパラメーターは **maxops** です。デフォルト値は **5** です。
- PTA Directory Server が認証する Directory Server からの応答を待つ時間制限。PTA 構文では、このパラメーターは **timeout** です。デフォルト値は **300** 秒 (5 分) です。
- 認証する Directory Server への接続に使用する PTA Directory Server の LDAP プロトコルのバージョン。PTA 構文では、このパラメーターは **ldver** です。デフォルトは **LDAPv3** です。
- 接続を使用できる制限時間を秒単位で指定します。この時間を過ぎてからクライアントがバインドリクエストを開始すると、そのサーバーは接続を閉じ、認証する Directory Server への新しい接続を開きます。バインド要求が開始し、サーバーがタイムアウトを超えたかどうかを判別しない限り、サーバーは接続を閉じません。このオプションが指定されていない場合や、認証する Directory Server が **authDS** パラメーターに記載されている場合に限り、時間制限が適用されません。2 つ以上のホストがリストされている場合、デフォルトは **300** 秒 (5 分) です。PTA 構文では、このパラメーターは **connlifetime** になります。
- 接続に STARTTLS を使用するかどうか。STARTTLS は標準の LDAP ポートでセキュアな接続を作成します。STARTTLS については、サーバーおよび CA 証明書がインストールされている必要がありますが、TLS で実行する必要はありません。

  デフォルトは **0** で、STARTTLS がオフになっていることを意味します。STARTTLS を有効にするには、**1** に設定します。STARTTLS を使用するには、LDAP URL は **ldaps:** ではなく **ldap:** を使用する必要があります。

1. **dsconf plugin pass-through-auth** コマンドを使用して、プラグインエントリーを編集します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin pass-through-auth add
   ldap://server.example.com/o=example 3,5,300,3,300,0
   ```

   (この例では、各オプションのパラメーターはデフォルト値に設定されます。) **subtree** パラメーターと任意のパラメーターの間にスペースがあることを確認してください。

   注記

   これらのパラメーターは任意ですが、いずれかのパラメーターが定義されている場合は、デフォルト値を使用していてもすべてのパラメーターを定義する必要があります。

2. サービスを再起動します。

   ```
   # dsctl instance_name restart
   ```

20.13.3. PTA プラグイン構文の例

本セクションでは、**dse.ldif** ファイルの PTA プラグイン構文の以下の例を説明します。

- 「Directory Server と 1 つのサブツリーの指定」
- 「複数の認証用 Directory Server の指定」
- 「1 つの Directory Server と複数のサブツリーを指定」
- 「デフォルト以外のパラメーター値の使用」
- 「認証する異なる Directory Server の異なる任意のパラメーターおよびサブツリーの指定」

20.13.3.1. Directory Server と 1 つのサブツリーの指定

この例では、PTA プラグインを設定して、オプションの変数の全デフォルトを受け入れるように設定します。この設定により、PTA Directory Server は **o=example** サブツリーへのすべてのバインド要求に対して、認証する Directory Server に接続します。認証する Directory Server のホスト名は **configdir.example.com** です。

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: ldap://configdir.example.com/o=example
...
```

20.13.3.2. 複数の認証用 Directory Server の指定

PTA Directory Server と認証する Directory Server との間の接続が破損するか、接続を開始できない場合は、PTA Directory Server が、指定された次のサーバー (存在する場合) に要求を送信します。最初の Directory Server が利用できない場合にフェイルオーバーを提供するために、必要に応じて認証する複数の Directory Server を指定できます。認証するすべての Directory Server が ***nsslapd-pluginarg0*** 属性に設定されます。認証する複数の Directory Server は、**host:port** ペアの空白区切りリストにリスト表示されます。以下に例を示します。

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: ldap://configdir.example.com:389 config2dir.example.com:1389/o=example
...
```

注記

nsslapd-pluginarg0 属性は、認証する Directory Server を設定します。追加の ***nsslapd-pluginargN*** 属性は、使用する PTA プラグインの追加 **接尾辞** を設定できますが、追加の **ホスト** ではありません。

20.13.3.3. 1 つの Directory Server と複数のサブツリーを指定

以下の例では、PTA Directory Server が複数のサブツリーのバインド要求をパススルーするように設定します (パラメーターのデフォルトを使用)。

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
```

```
nsslapd-pluginarg0: ldap://configdir.example.com/o=example
nsslapd-pluginarg1: ldap://configdir.example.com/dc=example,dc=com
...
```

20.13.3.4. デフォルト以外のパラメーター値の使用

この例では、最大接続数パラメーター **maxconns** のみに、デフォルト以外の値 (**10**) を使用しています。その他のパラメーターはデフォルト値に設定されます。ただし、1つのパラメーターが指定されているため、構文ですべてのパラメーターを明示的に定義する必要があります。

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0: ldap://configdir.example.com/o=example 10,5,300,3,300,1
...
```

20.13.3.5. 認証する異なる Directory Server の異なる任意のパラメーターおよびサブツリーの指定

認証する Directory Server ごとに異なるパススルーサブツリーと任意のパラメーター値を指定するには、複数の LDAP URL/任意のパラメーターペアを設定します。LDAP URL/任意のパラメーターペアは、以下のように単一スペースで区切ります。

```
dn: cn=Pass Through Authentication,cn=plugins,cn=config
...
nsslapd-pluginEnabled: on
nsslapd-pluginarg0:ldap://configdir.example.com/o=example 10,15,30,3,600,0
nsslapd-pluginarg1:ldap://config2dir.example.com/dc=example,dc=com 7,7,300,3,300,1
...
```

20.14. 認証に ACTIVE DIRECTORY 形式のユーザー名の使用

Directory Server に接続する場合は、**uid=*user_name*,ou=People,dc=example,dc=com** などのユーザーの識別名 (DN) を指定して認証する必要があります。ただし、DN は記憶しにくくなります。**AD DN** プラグインを有効にして設定する場合には、DN ではなく ***user_name*** や ***user_name@domain*** などの Active Directory 形式のユーザー名を使用できます。

プラグインを有効にし、ユーザーは DN 形式ではないユーザー名を使用してディレクトリーに接続すると、Directory Server はプラグインの設定に基づいて DN を検索します。検索で DN が1つ返される場合、Directory Server はこの DN を使用して認証を行います。DN がない場合や複数の DN が返された場合は、認証に失敗します。

注記

コマンドラインで **AD DN** プラグインのみを有効して設定できます。

example.com をデフォルトドメインとして使用するようにプラグインを有効にして設定するには、以下を行います。

1. **cn=addn,cn=plugins,cn=config** プラグインエントリーを追加し、デフォルトドメインを設定します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=addn,cn=plugins,cn=config
changetype: add
objectClass: top
objectClass: nsSlapdPlugin
objectClass: extensibleObject
cn: addn
nsslapd-pluginPath: libaddn-plugin
nsslapd-pluginInitfunc: addn_init
nsslapd-pluginType: preoperation
nsslapd-pluginEnabled: on
nsslapd-pluginId: addn
nsslapd-pluginVendor: 389 Project
nsslapd-pluginVersion: 1.3.6.0
nsslapd-pluginDescription: Allow AD DN style bind names to LDAP
addn_default_domain: example.com
```

プラグインエントリーで必要な ***addn_default_domain*** パラメーターにより、デフォルトのドメインが設定されます。認証時に指定されたユーザー名にドメイン名が含まれていない場合、プラグインはこのドメインを追加します。

2. デフォルトドメインの設定エントリーを追加します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
dn: cn=example.com,cn=addn,cn=plugins,cn=config
changetype: add
objectClass: top
objectClass: extensibleObject
cn: example.com
addn_base: ou=People,dc=example,dc=com
addn_filter: (&(objectClass=account)(uid=%s))
```

この例で使用されるパラメーターの詳細については、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 に記載される各パラメーターの説明を参照してください。

警告

デフォルトドメインに、少なくとも設定エントリーを追加する必要があります。エントリーが見つからないと、Directory Server は起動できません。

3. オプションでは、前のステップで説明されているように、追加のドメイン設定を作成して、複数のドメイン名をサポートすることができます。各ドメイン設定は、異なる検索ベースおよびフィルターを使用できます。

4. Directory Server インスタンスを再起動します。

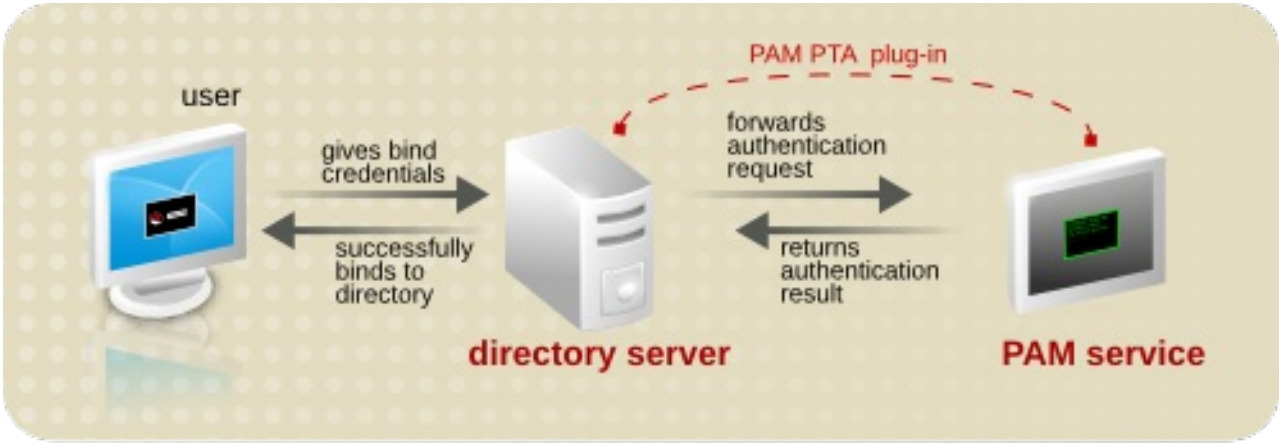
```
# dsctl instance_name restart
```

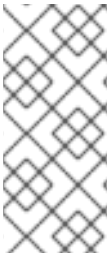
20.15. パススルー認証での PAM の使用

多くのシステムでは、Unix ユーザーおよび Linux ユーザー用の認証メカニズムがすでに含まれています。最も一般的な認証フレームワークの 1 つは、**プラグ可能な認証モジュール** (PAM) です。多くのネットワークではすでに既存の認証サービスを利用できるため、管理者はこれらのサービスを引き続き使用したいことがあります。PAM モジュールは、Directory Server に対して LDAP クライアントに既存の認証ストアを使用するように指示するように設定できます。

Red Hat Directory Server における PAM パススルー認証は、PAM パススルー認証プラグインを使用します。これにより、Directory Server は PAM サービスと通信して LDAP クライアントを認証できます。

図20.3 PAM パススルー認証プロセス





注記

PAM パススルー認証は、適切なマッピング方法 (ENTRY) が使用されていることを前提に、ユーザーを認証する際にアカウントの無効化と連動します。ただし、PAM パススルー認証では、パスワードは Directory Server ではなく PAM モジュールで設定され保存されるため、グローバルまたはローカルに設定されたパスワードポリシーに照らしてパスワードを検証することはできません。

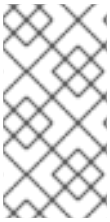
20.15.1. PAM パススルー認証設定オプション

PAM パススルー認証は、PAM パススルー認証プラグインコンテナーエントリーの下にある子エントリーで設定されます。複数の PAM パススルー認証ポリシーがあり、接尾辞内の異なる接尾辞やエントリーに適用されます。

PAM パススルー用に設定できるエリアはいくつかあります。

- PAM パススルー認証プラグインによって制御される接尾辞。ここでは、除外する接尾辞および含める接尾辞と、欠落した接尾辞の処理方法を説明します。
- 認証設定のターゲットである、設定された接尾辞内の個々のエントリー。デフォルトでは、接尾辞内のすべてのエントリーが認証スコープに含まれますが、複数の異なる PAM パススルー認証プラグインインスタンスを設定し、異なるユーザーに異なるプラグイン設定を適用することが可能です。
- PAM 属性マッピング。Directory Server に提示された認証情報は、何らかの方法で LDAP エントリーにマッピングされ、さらに PAM サービスの認証情報に戻される必要があります。これは、マッピングメソッドを定義し、任意で認証情報と一致させるために使用する LDAP 属性を定義します。

- TLS 接続の使用や、使用する PAM サービス、および PAM 認証に失敗した場合の LDAP 認証へのフォールバックなど、一般的な設定。



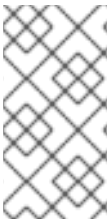
注記

PAM パススルー認証プラグインの複数の設定インスタンスが存在する可能性があります。PAM パススルー認証プラグインのインスタンスは、***pamFilter*** 属性を使用して、プラグインで使用する特定のエントリーを検索するよう LDAP フィルターを設定することで、ユーザーエントリーのサブセットに適用できます。

設定可能な属性の一覧は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の 『PAM パススルー認証プラグインの属性』 セクションを参照してください。

20.15.1.1. PAMPTA のターゲットとなる接尾辞の指定

PAM PTA プラグインは、明示的に除外されない限り、デフォルトですべての接尾辞にグローバルに適用されます。接尾辞を除外して組み込むと、ディレクトリーのエリアが LDAP 認証ではなく PAM 認証を使用する場合に役立ちます。



注記

PAM パススルー認証エントリーのターゲットは、任意のサブツリーではなく接尾辞でなければなりません。「接尾辞の作成および維持」 で説明されているように、**userRoot** に関連するルート接尾辞 **dc=example,dc=com** に関連する **cn=config** など、特定のバックエンドデータベースに関連付けられるサブツリーです。

pamExcludeSuffix 属性は接尾辞を除外します。デフォルトでは、設定サブツリー (**cn=config**) のみが除外されます。別の方法では、PAM PTA プラグインは ***pamIncludeSuffix*** 属性の接尾辞に適用することもできます。これらの属性はいずれも多値で設定されます。

include 属性が設定されている場合、他の接尾辞はすべて自動的に除外されます。同様に、除外属性が設定されている場合、他のすべての接尾辞は自動的に含まれます。

```
pamExcludeSuffix: cn=config
```

pamIncludeSuffix を使用すると、指定した接尾辞のみが含まれ、その他は自動的に除外されます。この属性は多値であるため、接尾辞を明示的にリスト表示することで、PAM 評価に複数の接尾辞を追加できます。

```
pamIncludeSuffix: ou=Engineering,dc=example,dc=com
pamIncludeSuffix: ou=QE,dc=example,dc=com
```

pamMissingSuffix 属性は、指定された接尾辞 (include または exclude) が存在しない場合に失敗を処理する方法をサーバーに指示します。**IGNORE** に設定すると、接尾辞が存在しない場合は、プラグインはその接尾辞を省略し、次の試行を試みます。

```
pamMissingSuffix: IGNORE
pamIncludeSuffix: ou=Engineering,dc=example,dc=com
pamIncludeSuffix: ou=Not Real,dc=example,dc=com
```

20.15.1.2. 異なるエントリーへの異なる PAM パススルー認証設定の適用

デフォルトでは、PAM パススルー認証ポリシーは指定の接尾辞内のすべてのエントリーに適用されます。ただし、***pamFilter***属性で LDAP フィルターを指定することができます。これは、PAM パススルー認証ポリシーを適用する接尾辞内の特定のエントリーを識別します。

これは、複数の PAM パススルー認証ポリシーを使用して、異なる PAM 設定やマッピング方法を異なるユーザータイプに適用する場合に便利です。

20.15.1.3. PAM PTA マッピングの設定

LDAP アイデンティティーを PAM アイデンティティーに接続する方法が必要です。最初に定義するのは、エントリーのマッピングに使用する **方法** です。DN、RDN、および ENTRY の 3 つのオプションがあります。ENTRY はエントリーでユーザー定義の属性を使用します。

複数のマッピング方法を、スペースで区切って順番に並べて指定することができます。プラグインは、認証が成功するまで、またはリストの最後に到達するまで、リスト表示される順序で各マッピングメソッドの使用を試行します。

例えば、このマッピング方法では、まず RDN メソッドをマッピングし、そのメソッドの順に ENTRY、次に DN がマッピングされます。

```
pamIDMapMethod: RDN ENTRY DN
```

異なるマッピングメソッドが、表20.4「PAM 認証のマッピングメソッド」 に一覧表示されます。

注記

Directory Server ユーザーアカウントは、ENTRY マッピング方法を使用してのみ検証されます。RDN または DN では、アカウントが非アクティブの Directory Server ユーザーでも、サーバーに正常にバインドされます。

表20.4 PAM 認証のマッピングメソッド

マッピング	説明
RDN	このメソッドは、バインド DN の左端にある RDN から値を使用します。このメソッドのマッピングは、Directory Server で定義されます。指定がない場合は、これがデフォルトのマッピングメソッドになります。
ENTRY	このメソッドは、バインド DN エントリーのユーザー定義の属性から PAM アイデンティティーの値をプルします。identity 属性は ***pamIDAttr*** 属性で定義されます。例: **pamIDAttr: customPamUid**
DN	このメソッドは、バインド DN からの完全な識別名を使用します。このメソッドのマッピングは、Directory Server で定義されます。

20.15.1.4. 汎用 PAM PTA 設定の設定

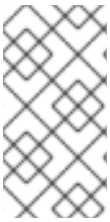
PAM 認証には、一般的な 3 つの設定を設定できます。

- PAM (**pamService**) に送信するサービス名。これは、**/etc/pam.d** で使用する設定ファイルの名前です。
- セキュアな接続 (**pamSecure**) を必要とするかどうか。

- PAM 認証に失敗した場合の LDAP 認証にフォールバックするかどうか (**pamFallback**)

```
pamFallback: false
pamSecure: false
pamService: ldapserver
```

20.15.2. PAM パススルー認証の設定

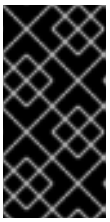


注記

Pluggable Authentication Module (PAM) パススルー認証には、複数の設定インスタンスが存在する場合があります。PAM パススルー認証のインスタンスは、***pamFilter*** 属性を使用して、プラグインで使用する特定のエントリーを検索するよう LDAP フィルターを設定することで、ユーザーエントリーのサブセットに適用できます。

PAM パススルー認証は、コマンドラインで設定されます。

1. PAM サービスが完全に設定されていることを確認してください。
2. **pam_fprintd.so** モジュールを PAM 設定ファイルから削除します。



重要

pam_fprintd.so モジュールは、PAM パススルー認証プラグイン設定の ***pamService*** 属性によって参照される設定ファイルにすることはできません。PAM の **fprintd** モジュールを使用すると、Directory Server は最大ファイル記述子制限に到達し、Directory Server プロセスが中止する可能性があります。

3. PAM パススルーの認証プラグインを有効にします。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin set "PAM Pass
Through Auth" --enabled on
```

4. PAM パススルー認証設定エントリーを作成します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com plugin pass-through-auth
pam-config "Admin PAM PTA Config" add --exclude-suffix="cn=config" --
id_map_method="RDN ENTRY" --id-attr="customPamUid" --
filter="(manager=uid=example_user,ou=people,dc=example,dc=com pamFallback: FALSE" -
-secure="TRUE" --service="ldapserver"
```

5. インスタンスを再起動します。

```
# dsctl instance_name restart
```

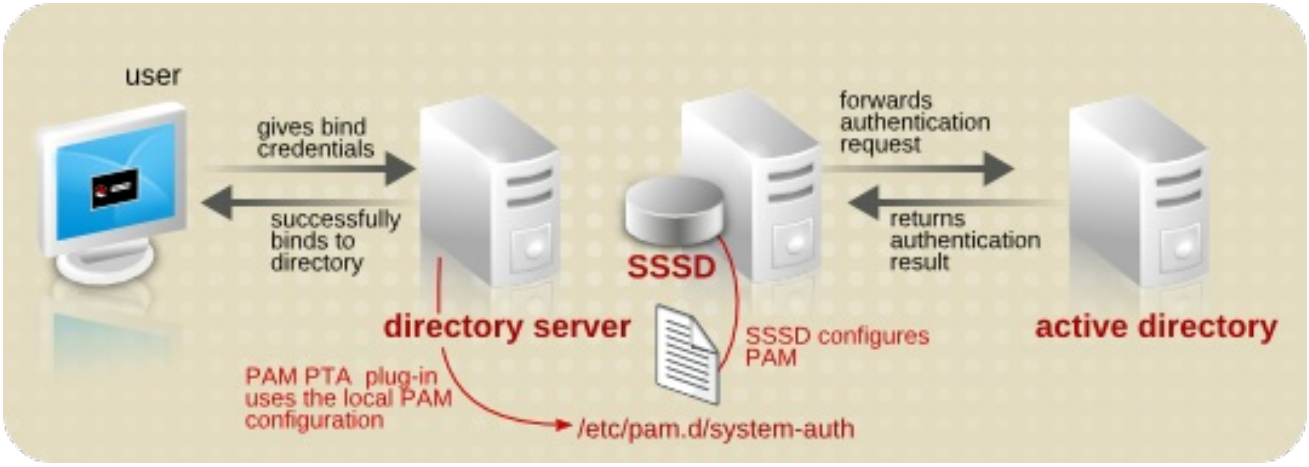
20.15.3. Active Directory をバックエンドとして PAM パススルー認証の使用

PAM パススルー認証では、Directory Server から PAM サービスに認証情報を転送します。1 つのオプションとして、Directory Server 専用の PAM モジュールを設定できます。また、インフラストラクチャーによっては、より再現性が高く便利な方法として、SSSD (System Security Services Daemon) を

使用して PAM を設定する方法もあります。SSSD はさまざまなアイデンティティーストアを使用できるため、Active Directory などの認証情報を提供するのに多くの異なるサーバーやサービスを使用できます。

SSSD を介してパススルー認証を使用することはサービスのデイジーチェーンです。PAM PTA プラグインは通常通りに設定されます。使用する特定の PAM サービスファイルを指します。このサービスファイルは SSSD によって管理され、SSSD は複数のプロバイダーであっても必要なアイデンティティープロバイダーに接続するように設定されます。

図20.4 SSSD による PAM パススルー認証



Active Directory で PAM パススルー認証を設定するには、以下を行います。

1. Active Directory サーバーを ID プロバイダーの 1 つとして使用するように SSSD を設定します。

   この設定については、『RHEL システムと Windows Active Directory の直接統合 』 の 『SSSD を使用した RHEL システムと AD の直接接続』 セクションを参照してください。

2. 次のように PAM パススルー認証プラグインを有効にします。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin set "PAM Pass Through Auth" --enabled on
   ```

3. 次のように、PAM パススルー認証設定エントリーを作成します。

   ```
   # dsconf -D "cn=Directory Manager" ldap://server.example.com plugin pass-through-auth pam-config "AD PAM PTA Config" add --id_map_method="ENTRY" --id-attr="uid" --service="login" --include-suffix="ou=people,dc=example,dc=com" --missing-suffix="ERROR"
   ```

   この例では、***uid LDAP*** 属性を Active Directory に渡すユーザー名として使用し、この設定を people OU に対してのみ有効にしています。

4. Directory Server インスタンスを再起動して、設定をロードします。

   ```
   # dsctl instance_name restart
   ```

20.16. ユーザーおよびロールの手動による非アクティブ化

1 つのユーザーアカウントまたはアカウントのセットを一時的に非アクティブにできます。アカウントが非アクティブになると、ユーザーがディレクトリーにバインドできません。認証操作は失敗します。

ユーザーおよびロールは、動作している属性 ***nsAccountLock*** を使用して非アクティベートされます。エントリーに値が **true** の ***nsAccountLock*** 属性が含まれる場合、サーバーはバインドを拒否します。

同じ手順を使用して、ユーザーとロールを非アクティブ化します。ただし、ロールが非アクティブになると、ロールエントリー自体ではなく **ロールのメンバー** が非アクティブになります。一般的なロールの詳細、およびロールとアクセスコントロールの関係については、*8章 エントリーの編成とグループ化*を参照してください。

警告

データベースのルートエントリー (ルートまたは部分接尾辞に対応するエントリー) は非アクティブにできません。*3章 ディレクトリーエントリーの管理*には、root またはサブ接尾辞のエントリーの作成に関する情報があり、*2章 ディレクトリーデータベースの設定*には、root およびサブ接尾辞の作成に関する情報があります。

20.16.1. アカウントまたはロールのステータスの表示

以下のステータスを表示するには、次を実行します。

- アカウントで、次を入力します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
account entry-status "uid=user_name,ou=People,dc=example,dc=com"
Entry DN: uid=user_name,ou=People,dc=example,dc=com
Entry Creation Date: 20200813085535Z (2020-08-13 08:55:35)
Entry Modification Date: 20200813085535Z (2020-08-13 08:55:35)
Entry State: activated
```

オプション: **-V** オプションをコマンドに渡して、追加情報を表示します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
account entry-status "uid=user_name,ou=People,dc=example,dc=com" -V
Entry DN: uid=user_name,ou=People,dc=example,dc=com
Entry Creation Date: 20200824160645Z (2020-08-24 16:06:45)
Entry Modification Date: 20200824160645Z (2020-08-24 16:06:45)
Entry Last Login Date: 20200824160645Z (2020-08-24 16:06:45)
Entry Time Until Inactive: 2 seconds (2020-08-24 16:07:45)
Entry State: activated
```

上記の出力は、出力の最後の 2 行で表されるアクティブなアカウントの例です。非アクティブなアカウントは、以下のような出力を提供します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
account entry-status "uid=user_name,ou=People,dc=example,dc=com" -V
Entry DN: uid=user_name,ou=People,dc=example,dc=com
Entry Creation Date: 20200824160645Z (2020-08-24 16:06:45)
Entry Modification Date: 20200824160645Z (2020-08-24 16:06:45)
Entry Last Login Date: 20200824160645Z (2020-08-24 16:06:45)
Entry Time Since Inactive: 3 seconds (2020-08-24 16:07:45)
Entry State: inactivity limit exceeded
```

- ロールには、以下を入力します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com" role
entry-status "cn=Marketing,ou=People,dc=example,dc=com"
Entry DN: cn=Marketing,ou=people,dc=example,dc=com
Entry State: activated
```

エントリーの代わりにサブツリーのステータスを表示するには、**entry-status** オプションの代わりに **subtree-status** を使用します。**subtree-status** オプションを使用する場合は、フィルター (**-f**) と検索範囲 (**-s**) を指定して結果を絞り込むことができます。また、**-i** オプションを使用して検索を絞り込み、非アクティブアカウントのみを返すか、**-o *date*** オプションを使用して、指定した日付までに非アクティブになるアカウントのみを返したりすることができます。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com" account
account "ou=People,dc=example,dc=com"  -f "(uid=*)" -V -o "2020-08-25T14:30:30"
```

*YYYY-MM-DD*T*HH:MM:SS* 形式で日付を指定します。

20.16.2. コマンドラインを使用したユーザーおよびロールの非アクティブ化およびアクティブ化

非アクティブにするには、次のコマンドを実行します。

- ユーザーアカウントで、以下を入力します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
account lock "uid=user_name,ou=People,dc=example,dc=com
```

- ロールには、以下を入力します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com" role
lock "cn=Marketing,ou=People,dc=example,dc=com
```

有効にするには、次のコマンドを実行します。

- ユーザーアカウントで、以下を入力します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com"
account unlock "uid=user_name,ou=People,dc=example,dc=com
```

- ロールには、以下を入力します。

```
# dsidm -D "cn=Directory Manager" ldap://server.example.com -b "dc=example,dc=com" role
unlock "cn=Marketing,ou=People,dc=example,dc=com
```

第21章 サーバーおよびデータベースアクティビティーの監視

本章では、データベースおよび Red Hat Directory Server ログの監視を説明します。SNMP を使用して Directory Server を監視する方法は、「SNMP を使用した Directory Server の監視」 を参照してください。

21.1. DIRECTORY SERVER ログファイルの種類

Directory Server は以下のログタイプを提供します。

- アクセスログ: クライアント接続および Directory Server インスタンスへの接続試行に関する情報が含まれます。このログタイプは、デフォルトで有効になります。
- エラーログ: エラーや、通常の操作中のディレクトリーエクスペリエンスに関する詳細なエラーメッセージが含まれます。このログタイプは、デフォルトで有効になります。

警告

Directory Server がエラーログに書き込みに失敗した場合、サーバーはエラーメッセージを **Syslog** サービスに送信し、終了します。このログタイプは、デフォルトで有効になります。

- 監査ログ: 各データベースとサーバー設定に加えられた変更を録画します。このログはデフォルトでは有効になっていません。
- 監査失敗ログ: レコードで監査ログが失敗しました。このログはデフォルトでは有効になっていません。

21.2. ログファイルの表示

コマンドラインおよび Web コンソールを使用して Directory Server ログファイルを表示できます。

21.2.1. コマンドラインでログファイルの表示

コマンドラインを使用してログファイルを表示するには、**less**、**more**、**cat** などの、Red Hat Enterprise Linux に含まれるユーティリティーを使用します。以下に例を示します。

```
# less /var/log/dirsrv/slapd-instance_name/access
```

ログファイルの場所を表示するには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config get nsslapd-accesslog
nsslapd-errorlog nsslapd-auditlog nsslapd-auditfaillog

nsslapd-accesslog: /var/log/dirsrv/slapd-instance_name/access
nsslapd-errorlog: /var/log/dirsrv/slapd-instance_name/errors
nsslapd-auditlog: /var/log/dirsrv/slapd-instance_name/audit
nsslapd-auditfaillog: /var/log/dirsrv/slapd-instance_name/audit-failure
```

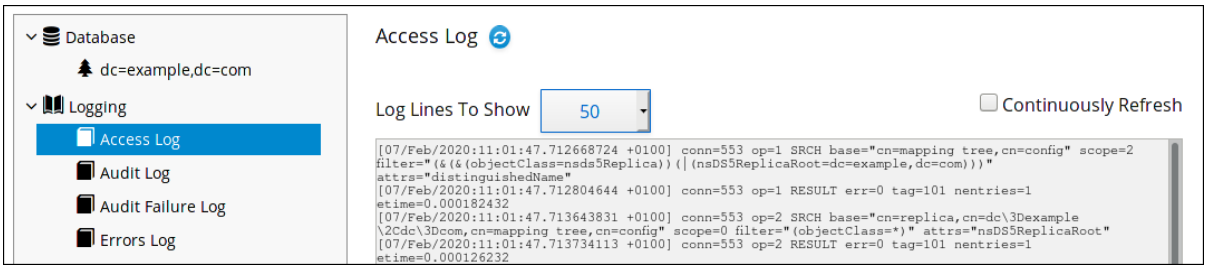
注記

ログタイプのロギングが有効になっていない場合は、対応するログファイルが存在しません。

21.2.2. Web コンソールを使用したログファイルの表示

Directory Server ログファイルを表示するには、次のコマンドを実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Monitoring** メニューを開きます。
4. **Logging** メニューを開き、表示するログファイルを選択します。



5. 必要に応じて、以下の設定をログファイルビューアーに適用することができます。
   - **Log Lines To Show** フィールドに表示される行数を設定します。
   - **継続的なリフレッシュ** を選択して、新しいログエントリーを自動的に表示できるようにします。
6. **Refresh** ボタンをクリックして変更を適用します。

21.3. ログファイルの設定

すべてのタイプのログファイルについて、ログの **作成** ポリシーおよびログ **削除** ポリシーを設定する必要があります。ログ作成ポリシーは、新規ログファイルの起動時に設定され、古いログファイルが削除される際にログ削除ポリシーが設定されます。

21.3.1. ログの有効化または無効化

アクセスおよびエラーロギングはデフォルトで有効になっています。ただし、監査および監査の失敗ロギングはデフォルトで無効になっています。

注記

アクセスログを無効にすると、ディレクトリーへの 2000 回のアクセスごとに約 1 メガバイトのログファイルが増加するため、特定のシナリオで有用です。ただし、アクセスログをオフにする前に、この情報で問題のトラブルシューティングを行うことができます。

21.3.1.1. コマンドラインを使用したロギングの有効化または無効化

dsconf config replace コマンドを使用して、Directory Server のロギング機能を制御する **cn=config** サブツリーのパラメーターを変更します。

- アクセスログ: ***nsslapd-accesslog-logging-enabled***
- エラーログ: ***nsslapd-errorlog-logging-enabled***
- 監査ログ: ***nsslapd-auditlog-logging-enabled***
- 監査失敗ログ: ***nsslapd-auditfaillog-logging-enabled***

詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の該当するセクションを参照してください。

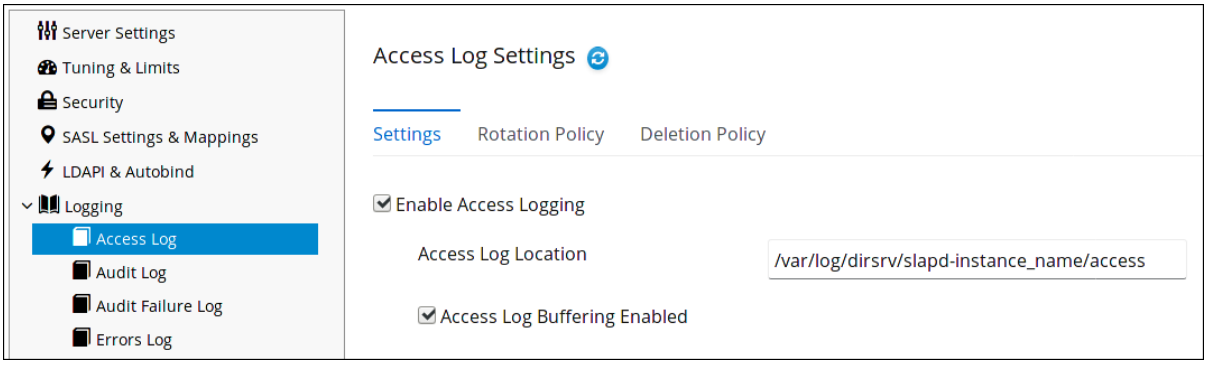
たとえば、監査ロギングを有効にするには以下を入力します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-auditlog-logging-enabled=on
```

21.3.1.2. Web コンソールを使用したロギングの有効化または無効化

Web コンソールでロギングを有効または無効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings** メニューを開き、**Logging** エントリーで設定するログタイプを選択します。



4. 選択したログタイプのロギング機能を有効または無効にします。
5. オプションで、ログローテーションポリシーまたはログ削除ポリシーなどを定義する追加のパラメーターを設定します。
6. **Save** をクリックします。

21.3.2. プラグイン固有のロギングの設定

デバッグのために、プラグインが実行する操作についてアクセスおよび監査ロギングを有効にできます。詳細は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 の対応するセクションに記載されている **nsslapd-logAccess** および **nsslapd-logAudit** パラメーターの説明を参照してください。

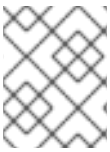
21.3.3. 高解像度のログタイムスタンプの無効化

Directory Server は、デフォルト設定を使用して、ナノ秒の精度でエントリーをログに記録します。

```
[27/May/2016:17:52:04.754335904 -0500] schemareload - Schema validation passed.
[27/May/2016:17:52:04.894255328 -0500] schemareload - Schema reload task finished.
```

高解像度のログタイムスタンプを無効にするには、以下を実行します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-logging-hr-timestamps-enabled=off
```



注記

高解像度のログのタイムスタンプを無効にするオプションは非推奨で、将来のリリースで削除される予定です。

高解像度のログのタイムスタンプを無効にすると、Directory Server は秒単位の精度でしかログを記録しなくなります。

```
[27/May/2016:17:52:04 -0500] schemareload - Schema validation passed.
[27/May/2016:17:52:04 -0500] schemareload - Schema reload task finished.
```

21.3.4. ログファイルのローテーションポリシーの定義

現在のログファイルを定期的にアーカイブして新しいファイルを作成するには、ログファイルのローテーションポリシーを設定します。コマンドラインまたは Web コンソールを使用して、**cn=config** サブツリーの設定を更新できます。

以下の設定パラメーターを設定して、ログファイルのローテーションポリシーを制御できます。

アクセスモード

アクセスモードでは、新規に作成されたログファイルにファイル権限を設定します。

- アクセスログ: ***nsslapd-accesslog-mode***
- エラーログ: ***nsslapd-errorlog-mode***
- 監査ログ: ***nsslapd-auditlog-mode***
- 監査失敗ログ: ***nsslapd-auditfaillog-mode***

ログの最大数

保持するログファイルの最大数を設定します。ファイル数に達すると、Directory Server は新しいログファイルを作成する前に、最も古いログファイルを削除します。

- アクセスログ: ***nsslapd-accesslog-maxlogsperdir***
- エラーログ: ***nsslapd-errorlog-maxlogsperdir***
- 監査ログ: ***nsslapd-auditlog-maxlogsperdir***
- 監査失敗ログ: ***nsslapd-auditfaillog-maxlogsperdir***

各ログのファイルサイズ

ログファイルがローテーションされるまでの最大サイズをメガバイト単位で設定します。

- アクセスログ: ***nsslapd-accesslog-maxlogsize***
- エラーログ: ***nsslapd-errorlog-maxlogsize***
- 監査ログ: ***nsslapd-auditlog-maxlogsize***
- 監査失敗ログ: ***nsslapd-auditfaillog-maxlogsize***

毎回ログの作成

ログファイルの最大期間を設定します。

- ***nsslapd-accesslog-logrotationtime*** および ***nsslapd-accesslog-logrotationtimeunit***
- ***nsslapd-errorlog-logrotationtime*** および ***nsslapd-errorlog-logrotationtimeunit***
- ***nsslapd-auditlog-logrotationtime*** および ***nsslapd-auditlog-logrotationtimeunit***
- ***nsslapd-auditfaillog-logrotationtime*** および ***nsslapd-auditfaillog-logrotationtimeunit***

さらに、以下のパラメーターを使用してログファイルがローテーションされるまでの時間を設定することもできます。

- ***nsslapd-accesslog-logrotationsynchour*** および ***nsslapd-accesslog-logrotationsyncmin***
- ***nsslapd-errorlog-logrotationsynchour*** および ***nsslapd-errorlog-logrotationsyncmin***
- ***nsslapd-auditlog-logrotationsynchour*** および ***nsslapd-auditlog-logrotationsyncmin***
- ***nsslapd-auditfaillog-logrotationsynchour*** および ***nsslapd-auditfaillog-logrotationsyncmin***

詳細は、『Red Hat Directory Server 設定、コマンド、およびファイルリファレンス』 の対応するセクションに記載されているパラメーターの説明を参照してください。

各ログファイルは、ログファイルのアーカイブまたは交換を容易にするため、サーバーのバージョン、ホスト名、およびポートを識別するタイトルで始まります。以下に例を示します。

```
389-Directory/1.4.0.11 B2018.197.1151
server.example.com:389 (/etc/dirsrv/slapd-instance)
```

21.3.4.1. コマンドラインを使用したログファイルローテーションポリシーの定義

dsconf config replace コマンドを使用して、Directory Server のロギング機能を制御するパラメーターを変更します。たとえば、エラーログの場合は、アクセスモード **600** を設定して最大 **2** を維持し、ログファイルのサイズを **100** MB あるいは **5 日** ごとにローテーションするには、次のコマンドを実行します。

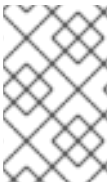
```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-errorlog-mode=600 nsslapd-errorlog-maxlogsperdir=2 nsslapd-errorlog-maxlogsize=100 nsslapd-errorlog-logrotationtime=5 nsslapd-errorlog-logrotationtimeunit=day
```

21.3.4.2. Web コンソールを使用したログファイルローテーションポリシーの定義

「Web コンソールを使用したロギングの有効化または無効化」を参照してください。

21.3.5. ログファイルの削除ポリシーの定義

ディレクトリーサーバーは、**削除ポリシー** を設定すると、アーカイブされた古いログファイルを自動的に削除します。



注記

ログファイルのローテーションポリシーが設定されている場合に限り、ログファイルの削除ポリシーを設定できます。Directory Server は、ログローテーション時に削除ポリシーを適用します。

以下の設定パラメーターを設定して、ログファイルの削除ポリシーを制御できます。

ログサイズの合計

すべてのアクセス、エラー、監査、または監査失敗ログファイルのサイズが設定された値を越えると、最も古いログファイルが自動的に削除されます。

- アクセスログ: ***nsslapd-accesslog-logmaxdiskspace***
- エラーログ: ***nsslapd-errorlog-logmaxdiskspace***
- 監査ログ: ***nsslapd-auditlog-logmaxdiskspace***
- 監査ログ: ***nsslapd-auditfaillog-logmaxdiskspace***

空きディスク領域がより少ない

空きディスク容量がこの値に達すると、最も古いアーカイブファイルが自動的に削除されます。

- アクセスログ: ***nsslapd-accesslog-logminfreediskspace***
- エラーログ: ***nsslapd-errorlog-logminfreediskspace***
- 監査ログ: ***nsslapd-auditlog-logminfreediskspace***
- 監査ログ: ***nsslapd-auditfaillog-logminfreediskspace***

指定した時間よりもファイルが古い場合

ログファイルが設定された時間よりも古い場合は、これが自動的に削除されます。

- アクセスログ: ***nsslapd-accesslog-logexpirationtime*** および ***nsslapd-accesslog-logexpirationtimeunit***
- エラーログ: ***nsslapd-errorlog-logminfreediskspace*** および ***nsslapd-errorlog-logexpirationtimeunit***
- 監査ログ: ***nsslapd-auditlog-logminfreediskspace*** および ***nsslapd-auditlog-logexpirationtimeunit***
- 監査ログ: ***nsslapd-auditfaillog-logminfreediskspace*** および ***nsslapd-auditfaillog-logexpirationtimeunit***

詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の該当するセクションを参照してください。

21.3.5.1. コマンドラインを使用したログ削除ポリシーの設定

dsconf config replace コマンドを使用して、Directory Server のロギング機能を制御するパラメーターを変更します。たとえば、すべてのアクセスログファイルの合計サイズが **500** MB 増加した場合に、最も古いアクセスログファイルを自動的に削除するには、次のように実行します。

```
dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-accesslog-logmaxdiskspace=500
```

21.3.5.2. Web コンソールを使用したログ削除ポリシーの設定

「Web コンソールを使用したロギングの有効化または無効化」を参照してください。

21.3.6. 手動ログファイルローテーション

Directory Server は、3 つのすべてのログの自動ログファイルのローテーションをサポートします。ただし、自動ログファイルの作成や削除ポリシーが設定されていない場合には、ログファイルを手動でローテーションすることができます。デフォルトでは、アクセス、エラー、監査、および監査失敗のログファイルは、以下の場所にあります。

```
/var/log/dirsrv/slapd-instance
```

ログファイルを手動でローテーションするには、以下を実行します。

1. インスタンスを停止します。

   ```
   # dsctl instance_name stop
   ```

2. 古いログファイルが今後の参照で利用できるように、ローテーションされるログファイルを移動するか名前を変更します。

3. インスタンスを起動します。

   ```
   # dsctl instance_name restart
   ```

21.3.7. ログレベルの設定

アクセスとエラーログはいずれも、設定されるログレベルに応じて、さまざまな情報を記録できます。

以下の設定パラメーターを設定して、以下のログレベルを制御できます。

- アクセスログ: ***nsslapd-accesslog-level***
- エラーログ: ***nsslapd-errorlog-level***

詳細情報およびサポートされるログレベルのリストは、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の該当するセクションを参照してください。

注記

デフォルトからログレベルを変更すると、ログファイルが急速に増大する可能性があります。Red Hat は、Red Hat のテクニカルサポートからの要請がない限り、デフォルト値を変更しないことを推奨します。

21.3.7.1. コマンドラインを使用したログレベルの設定

dsconf config replace コマンドを使用してログレベルを設定します。

たとえば、検索フィルターロギング (**32**) および設定ファイル処理 (**64**) を有効にするには、***nsslapd-errorlog-level*** パラメーターを **96** (32 + 64) に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-errorlog-level=96
```

たとえば、内部アクセス操作ロギング (**4**) および接続、操作、および結果のロギング (**256**) を有効にするには、***nsslapd-accesslog-level*** パラメーターを **260** (4 + 256) に設定します。

```
# dsconf -D "cn=Directory Manager" ldap://server.example.com config replace nsslapd-accesslog-level=260
```

21.3.7.2. Web コンソールを使用したログレベルの設定

Web コンソールを使用してアクセスおよびエラーログレベルを設定するには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. 設定するには、以下を実行します。
   - アクセスログレベル:
     1. **Server Settings → Logging → Access Log** メニューを開きます。
     2. **Access Logging Levels** セクションでログレベルを選択します。以下に例を示します。

	Logging Level
☑	Default Logging
☐	Internal Operations
☐	Entry Access and Referrals

   - エラーログレベル:
     1. **Server Settings → Logging → Error Log** メニューを開きます。

2. **Error Logging Levels** セクションでログレベルを選択します。以下に例を示します。

	Logging Level
☐	Trace Function Calls
☐	Packet Handling
☐	Heavy Trace Output
☐	Connection Management
☐	Packets Sent & Received
☐	Search Filter Processing
☐	Config File Processing
☐	Access Control List Processing
☐	Log Entry Parsing
☐	Housekeeping
☐	Replication
☐	Entry Cache
☐	Plugin
☐	Access Control Summary

4. **Save** をクリックします。

21.3.7.3. 内部操作のロギング

複数の操作により、Directory Server で追加の内部操作が発生します。たとえば、ユーザーがエントリーを削除した場合、サーバーはエントリーの検索や、ユーザーが所属していたグループの更新など、いくつかの内部処理を行います。このセクションでは、内部操作ログエントリーの形式を説明します。ログレベルの設定に関する詳細は、「ログレベルの設定」 を参照してください。

Directory Server は、内部操作ロギングの以下のような形式を提供します。

サーバー開始の内部操作

サーバーが開始した内部操作ログエントリーの例:

```
[14/Jan/2021:09:45:25.814158882 -0400] conn=Internal(0) op=0(0)(0) MOD dn="cn=uniqueid
```

```
generator,cn=config"
[14/Jan/2021:09:45:25.822103183 -0400] conn=Internal(0) op=0(0)(0) RESULT err=0 tag=48
nentries=0 etime=0.0007968796
```

このタイプのログエントリーの場合:

- ***conn*** フィールドは、**Internal** に続いて **(0)** が設定されます。
- ***op*** フィールドは **0(0)(*nesting_level*)** に設定されます。server-initiated 内部操作の場合、操作 ID と内部操作 ID の両方は常に **0** になります。入れ子ではないログエントリーの場合、ネストレベルは **0** になります。

クライアントで開始される内部操作

クライアントが開始した内部操作ログエントリーの例:

```
[14/Jan/2021:09:45:14.382918693 -0400] conn=5 (Internal) op=15(1)(0) SRCH
base="cn=config,cn=userroot,cn=ldbm database,cn=plugins,cn=config" scope=1
filter="objectclass=vlvsearch" attrs=ALL
[14/Jan/2021:09:45:14.383191380 -0400] conn=5 (Internal) op=15(1)(0) RESULT err=0 tag=48
nentries=0 etime=0.0000295419
[14/Jan/2021:09:45:14.383216269 -0400] conn=5 (Internal) op=15(2)(0) SRCH
base="cn=config,cn=example,cn=ldbm database,cn=plugins,cn=config" scope=1
filter="objectclass=vlvsearch" attrs=ALL
[14/Jan/2021:09:45:14.383449419 -0400] conn=5 (Internal) op=15(2)(0) RESULT err=0
```

このタイプのログエントリーの場合:

- ***conn*** フィールドはクライアント接続 ID に続いて文字列 **(Internal)** に設定されます。
- ***op*** フィールドで、操作 ID の後に **(*internal_operation_ID*)(*nesting_level*)** が含まれます。内部の操作 ID は変わることがあり、ネスト化していないログエントリーは **0** になります。

nsslapd-plugin-logging パラメーターが **on** に設定され、内部操作のロギングが有効になっている場合、Directory Server はプラグインの内部操作の追加をログに記録します。

例21.1 プラグインロギングが有効になっている内部操作ログエントリー

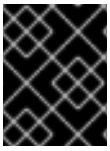
uid=user,dc=example,dc=com エントリーを削除し、**Referential Integrity** プラグインが **example** グループからこのエントリーを自動的に削除すると、サーバーログは以下のようになります。

```
[time_stamp] conn=2 op=37 DEL dn="uid=user,dc=example,dc=com"
[time_stamp] conn=2 (Internal) op=37(1) SRCH base="uid=user,dc=example,dc=com" scope=0
filter="(|(objectclass=*)(objectclass=ldapsubentry))" attrs=ALL
[time_stamp] conn=2 (Internal) op=37(1) RESULT err=0 tag=48 nentries=1 etime=0.0000129148
[time_stamp] conn=2 (Internal) op=37(2) SRCH base="dc=example,dc=com" scope=2 filter="
(member=uid=user,dc=example,dc=com)" attrs="member"
[time_stamp] conn=2 (Internal) op=37(2) RESULT err=0 tag=48 nentries=0 etime=0.0000123162
[time_stamp] conn=2 (Internal) op=37(3) SRCH base="dc=example,dc=com" scope=2 filter="
(uniquemember=uid=user,dc=example,dc=com)" attrs="uniquemember"
[time_stamp] conn=2 (Internal) op=37(3) RESULT err=0 tag=48 nentries=1 etime=0.0000128104
[time_stamp] conn=2 (Internal) op=37(4) MOD dn="cn=example,dc=example,dc=com"
[time_stamp] conn=2 (Internal) op=37(5) SRCH base="cn=example,dc=example,dc=com"
scope=0 filter="(|(objectclass=*)(objectclass=ldapsubentry))" attrs=ALL
[time_stamp] conn=2 (Internal) op=37(5) RESULT err=0 tag=48 nentries=1 etime=0.0000130685
```

```
[time_stamp] conn=2 (Internal) op=37(4) RESULT err=0 tag=48 nentries=0 etime=0.0005217545
[time_stamp] conn=2 (Internal) op=37(6) SRCH base="dc=example,dc=com" scope=2 filter="
(owner=uid=user,dc=example,dc=com)" attrs="owner"
[time_stamp] conn=2 (Internal) op=37(6) RESULT err=0 tag=48 nentries=0 etime=0.0000137656
[time_stamp] conn=2 (Internal) op=37(7) SRCH base="dc=example,dc=com" scope=2 filter="
(seeAlso=uid=user,dc=example,dc=com)" attrs="seeAlso"
[time_stamp] conn=2 (Internal) op=37(7) RESULT err=0 tag=48 nentries=0 etime=0.0000066978
[time_stamp] conn=2 (Internal) op=37(8) SRCH base="o=example" scope=2 filter="
(member=uid=user,dc=example,dc=com)" attrs="member"
[time_stamp] conn=2 (Internal) op=37(8) RESULT err=0 tag=48 nentries=0 etime=0.0000063316
[time_stamp] conn=2 (Internal) op=37(9) SRCH base="o=example" scope=2 filter="
(uniquemember=uid=user,dc=example,dc=com)" attrs="uniquemember"
[time_stamp] conn=2 (Internal) op=37(9) RESULT err=0 tag=48 nentries=0 etime=0.0000048634
[time_stamp] conn=2 (Internal) op=37(10) SRCH base="o=example" scope=2 filter="
(owner=uid=user,dc=example,dc=com)" attrs="owner"
[time_stamp] conn=2 (Internal) op=37(10) RESULT err=0 tag=48 nentries=0 etime=0.0000048854
[time_stamp] conn=2 (Internal) op=37(11) SRCH base="o=example" scope=2 filter="
(seeAlso=uid=user,dc=example,dc=com)" attrs="seeAlso"
[time_stamp] conn=2 (Internal) op=37(11) RESULT err=0 tag=48 nentries=0 etime=0.0000046522
[time_stamp] conn=2 op=37 RESULT err=0 tag=107 nentries=0 etime=0.0010297858
```

21.3.8. デバッグ用のアクセスログバッファーの無効化

デバッグの目的で、デフォルトで有効になっているアクセスログバッファーを無効にできます。アクセスログのバッファーが無効になっていると、Directory Server はログエントリーをディスクに直接書き込みます。



重要

通常の操作環境では、アクセスログを無効にしないでください。バッファーを無効にすると、特に負荷が大きい場合に、Directory Server のパフォーマンスが低下します。

21.3.8.1. コマンドラインを使用したアクセスログバッファーの無効化

コマンドラインを使用してアクセスログバッファーを無効にするには、以下を実行します。

- ***nsslapd-accesslog-logbuffering*** パラメーターを **off** に設定します。

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

21.3.8.2. Web コンソールを使用したアクセスログバッファーの無効化

Web コンソールを使用してアクセスログバッファーを無効にするには、以下を実行します。

1. Web コンソールで Directory Server ユーザーインターフェイスを開きます。「Web コンソールを使用した Directory Server へのログイン」を参照してください。
2. インスタンスを選択します。
3. **Server Settings → Logging → Access Log** を開きます。
4. **Disable Access Log Buffering** を選択します。

5. **Save Configuration** をクリックします。

21.4. アクセスログ統計の取得

logconv.pl スクリプトはアクセスログを解析し、サーバーで実行するさまざまなユーザーおよび操作に関するサマリー情報を返します。

最も簡単な方法は、スクリプトはアクセスログ (単数または複数) を解析するだけです。

```
# logconv.pl /relative/path/to/accessLog
```

このスクリプトはワイルドカードを受け入れて複数のアクセスログを解析できます。これはログローテーションが使用される場合に役立ちます。

```
# logconv.pl /var/log/dirsrv/slapd-instance/access*
```

logconv.pl のさまざまなオプションは、man ページと 『設定、コマンド、およびファイルリファレンス』 で説明されています。

logconv.pl は、アクセスログから一般的な使用状況を引き出すために、いくつかの異なる方法があります。

最も簡単な方法としては、**logconv.pl** が、総操作数、総接続数、各操作タイプごとのカウント、永続的な検索などの拡張操作のカウント、およびバインド情報のリストを表示します。

```
# logconv.pl /var/log/dirsrv/slapd-instance/access
Access Log Analyzer 8.2
Command: logconv.pl /var/log/dirsrv/slapd-instance/access
Processing 1 Access Log(s)...

[001] /var/log/dirsrv/slapd-instance/access size (bytes):       77532

Total Log Lines Analysed:  527

Start of Logs:    14/Oct/2017:16:15:22.452909568
End of Logs:      14/Oct/2017:16:39:50.157790196

Processed Log Time:  0 Hours, 24 Minutes, 27.704877056 Seconds

Restarts:                    10
Secure Protocol Versions:
  - TLS1.2 client bound as uid=user_name,ou=people,o=example.com (11 connections)
  - TLS1.2 128-bit AES; client CN=CA Subsystem,O=example.com; issuer CN=Certificate
Authority,O=example.com (11 connections)
  - TLS1.2 128-bit AES-GCM (2 connections)
  - TLS1.2 128-bit AES (3 connections)

Peak Concurrent Connections:  38
Total Operations:             4771
Total Results:                4653
Overall Performance:          97.5%

Total Connections:            249          (0.17/sec)  (10.18/min)
 - LDAP Connections:          107          (0.07/sec)  (4.37/min)
 - LDAPI Connections:         128          (0.09/sec)  (5.23/min)
```

```
 - LDAPS Connections:          14           (0.01/sec)  (0.57/min)
 - StartTLS Extended Ops:      2            (0.00/sec)  (0.08/min)

Searches:                    2963          (2.02/sec)  (121.13/min)
Modifications:               649           (0.44/sec)  (26.53/min)
Adds:                        785           (0.53/sec)  (32.09/min)
Deletes:                     10            (0.01/sec)  (0.41/min)
Mod RDNs:                    6             (0.00/sec)  (0.25/min)
Compares:                    0             (0.00/sec)  (0.00/min)
Binds:                       324           (0.22/sec)  (13.25/min)

Proxied Auth Operations:     0
Persistent Searches:         17
Internal Operations:         0
Entry Operations:            0
Extended Operations:         4
Abandoned Requests:          0
Smart Referrals Received:    0

VLV Operations:              30
VLV Unindexed Searches:      0
VLV Unindexed Components:    20
SORT Operations:             22

Entire Search Base Queries:  12
Paged Searches:              2
Unindexed Searches:          0
Unindexed Components:        149

FDs Taken:                   249
FDs Returned:                212
Highest FD Taken:            107

Broken Pipes:                0
Connections Reset By Peer:   0
Resource Unavailable:        0
Max BER Size Exceeded:       0

Binds:                       324
Unbinds:                     155
--------------------------------
 - LDAP v2 Binds:            41
 - LDAP v3 Binds:            180
 - AUTOBINDs(LDAPI):         103
 - SSL Client Binds:         0
 - Failed SSL Client Binds:  0
 - SASL Binds:               134
   - EXTERNAL: 114
   - GSSAPI: 20
 - Directory Manager Binds:  10
 - Anonymous Binds:          1

Cleaning up temp files...
Done.
```

操作と接続のサマリー情報に加えて、サーバーへのすべての接続のより詳細なサマリー情報を提供しま

す。この情報には、サーバーへの接続に使用された最も一般的な IP アドレス、ログインに最も失敗した DN、サーバーへのアクセスに使用された合計バインド DN 数、最も一般的なエラーコードやリターンコードなどが含まれます。

その他の接続サマリーは単一オプションとして渡されます。たとえば、サーバー (**b**) への接続に使用する DN 数と、サーバーによって返された合計接続コード (**c**) を **-bc** として渡します。

```
# logconv.pl -bc /var/log/dirsrv/slapd-instance/access
...
----- Total Connection Codes -----

U1            3    Cleanly Closed Connections
B1            1    Bad Ber Tag Encountered

----- Top 20 Bind DN's -----

Number of Unique Bind DN's: 212

1801           cn=Directory Manager
1297           Anonymous Binds
311           uid=jsmith,ou=people...
87           uid=bjensen,ou=peopl...
85           uid=mreynolds,ou=peo...
69           uid=jrockford,ou=peo...
55           uid=sspencer,ou=peop...
...
```

データは、特定の開始時間 (**-S**) 以降、特定の終了時間 (**-E**) 以降、または範囲内からエントリーに制限することができます。開始時間と終了時間が設定されると、**logconv.pl** が最初に指定の時間範囲を出力し、次にその期間の概要を出力します。

```
# logconv.pl -S "[01/Jul/2016:16:11:47.000000000 -0400]" -E "[01/Jul/2016:17:23:08.999999999 -0400]" /var/log/dirsrv/slapd-instance/access
...
----------- Access Log Output ------------

Start of Logs:    01/Jul/2016:16:11:47
End of Logs:      01/Jul/2016:17:23:08
...
```

開始時間と終了の期間は、合計サマリー数の生成に使用されるデータの時間制限のみを設定します。それでも、集計または合計の数が表示されます。Directory Server への接続や操作のパターンを把握するために、1 分ごと (**-M**) や 1 秒ごと (**-m**) のカウントデータを出力することができます。この場合、データは時間単位で、指定した CSV 出力ファイルに出力されます。

```
# logconv.pl -m|-M outputFile accessLogFile
```

以下に例を示します。

```
# logconv.pl -M /home/output/statsPerMin.txt /var/log/dirsrv/slapd-instance/access*
```

-M|-m オプションは、**-S** 引数および **-E** 引数と共に使用することで、特定の期間内の分単位または秒単位のカウントを取得することもできます。

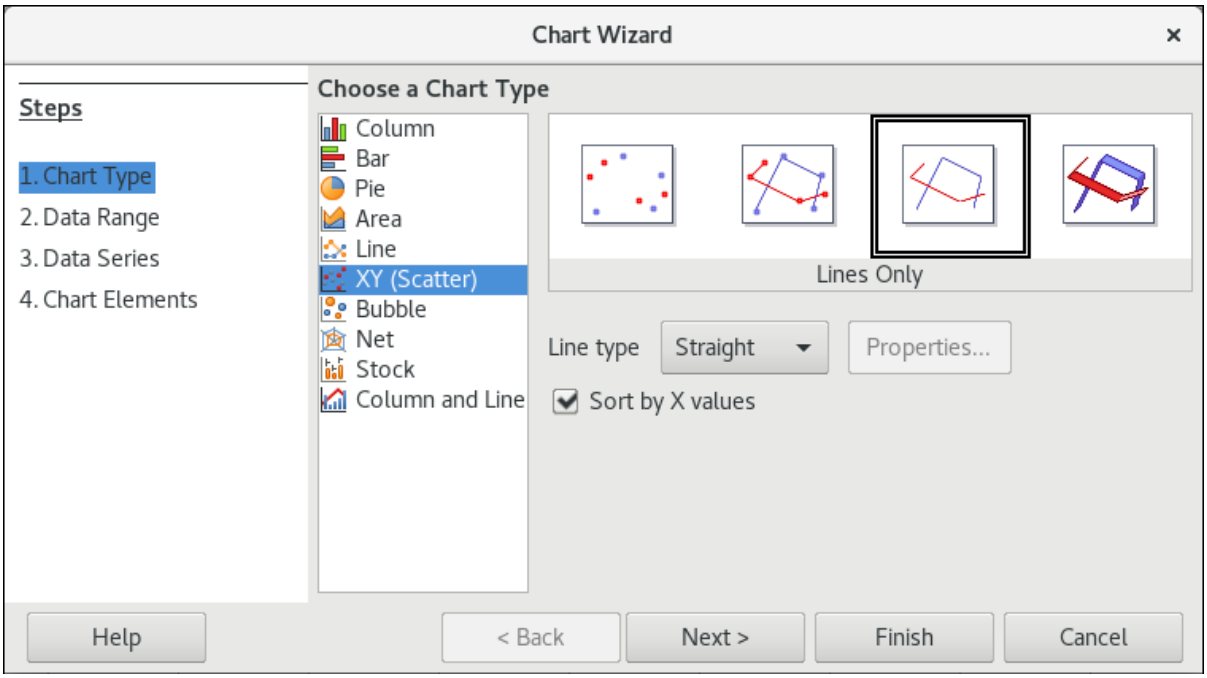
ファイルの各行は、分または秒の 1 つの時間単位を表し、その期間の合計数を示しています。CSV ファイル (1 分ごとの統計および 1 秒ごと統計の両方) には、以下のコラムが順番に含まれます。

```
Time,time_t,Results,Search,Add,Mod,Modrdn,Delete,Abandon,Connections,SSL Conns,Bind,Anon Bind,Unbind,Unindexed
```

CSV ファイルは、LibreOffice Calc などのスプレッドシートプログラムや、その他多くのビジネスアプリケーションで操作できます。CSV データをインポートし、チャートまたは他のメトリックを生成する手順は、アプリケーション自体によって異なります。

たとえば、LibreOffice Calc でチャートを作成するには、次のコマンドを実行します。

1. CSV ファイルを開きます。
2. **Insert** メニューをクリックし、**Chart** を選択します。
3. **Chart Type** エリアで、チャートタイプを **XY (Scatter)** に設定します。
   a. サブタイプを行のみに設定します。
   b. X 値でソートするオプションを選択します。



4. 他の画面のデフォルト (特に、データ系列を列で使用すること、最初の行と最初の列をラベルとして設定すること) を受け入れて、チャートを作成します。

21.5. シャットダウンのローカルディスクの監視

『Red Hat Directory Server パフォーマンスチューニングガイド』 の 『適切なシャットダウンに対するローカルディスクの監視』 セクションを参照してください。

21.6. サーバーアクティビティーの監視

『Red Hat Directory Server パフォーマンスチューニングガイド』 の 『サーバーアクティビティーの監視』 セクションを参照してください。

21.7. データベースアクティビティーの監視

『Red Hat Directory Server パフォーマンスチューニングガイド』 の 『データベースアクティビティーの監視』 セクションを参照してください。

21.8. データベースリンクアクティビティーの監視

『Red Hat Directory Server パフォーマンスチューニングガイド』 の 『データベースのリンクアクティビティーの監視』 セクションを参照してください。

21.9. カウンターの有効化および無効化

nsslapd-counters パラメーターにより、実行するカウンターが有効になります。ただし、カウンターの実行はパフォーマンスに影響する可能性があるため、カウンターをオフにすることもできます。カウンターがオフの場合、カウンターの値はすべてゼロ (0) になります。

デフォルトでは、カウンターはすでに有効になっています。パフォーマンスカウンターを有効または無効にするには、**ldapmodify** を使用します。たとえば、無効にするには、次のコマンドを実行します。

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

21.10. SNMP を使用した DIRECTORY SERVER の監視

21章 *サーバーおよびデータベースアクティビティーの監視*で説明しているサーバーおよびデータベースアクティビティー監視のログ設定は、Directory Server に固有のものです。また、SNMP (Simple Network Management Protocol) を使用して Directory Server を監視することもできます。SNMP は、ネットワーク活動を監視するのに使用する管理プロトコルで、さまざまな機器をリアルタイムに監視することができます。

Directory Server は、AgentX サブエージェントを使用して SNMP を監視できます。SNMP 監視は、バインド情報、サーバーで実行される操作、キャッシュ情報など、Directory Server に関する有用な情報を収集します。Directory Server SNMP サブエージェントは SNMP トラップをサポートし、サーバーインスタンスの実行状態の変更に関する通知を送信します。

21.10.1. SNMP の概要

SNMP は広く普及しているため、相互運用性があります。このような相互運用性と、SNMP がさまざまなデバイスクラスに固有の多くのジョブを引き受けることができるという事実により、SNMP はグローバルネットワークの制御と監視のための理想的な標準メカニズムとなっています。SNMP により、ネットワーク管理者はすべてのネットワーク監視活動を統合することができ、Directory Server の監視もその一部となります。

SNMP は、ネットワークアクティビティーに関するデータを交換するために使用されます。SNMP では、ユーザーがネットワークをリモートで管理する管理デバイスとネットワーク管理アプリケーション (NMS) の間でデータが伝送されます。管理デバイスは、ホスト、ルーター、Directory Server などの SNMP を実行するすべてです。NMS は通常、1 つ以上のネットワーク管理アプリケーションがインストールされた強力なワークステーションです。ネットワーク管理アプリケーションは、管理している機器の情報、どの機器が稼働または停止しているのか、どのエラーメッセージをどれだけ受け取ったのか、などをグラフィカルに表示します。

NMS と管理デバイスに関する情報は、サブエージェントと **マスターエージェント** の 2 種類のエージェントを使用して転送されます。サブエージェントは、マネージドデバイスに関する情報を収集し、情報をマスターエージェントに渡します。Directory Server にはサブエージェントがあります。マスター

エージェントは、さまざまなサブエージェントと NMS との間で報を交換します。マスターエージェントは、通常、リモートマシンで実行できるものの、通信するサブエージェントと同じホストマシンで実行します。

問い合わせ可能な SNMP 属性 (変数とも呼ばれる) の値は、マネージド機器に保持され、必要に応じて NMS に報告されます。各変数は **管理オブジェクト** と呼ばれ、エージェントがアクセスして NMS に送信できるものです。すべての管理オブジェクトは、ツリーのような階層を持つデータベースである管理情報ベース (MIB) で定義されます。階層の最上位には、ネットワークに関する最も一般的な情報が含まれます。その下の各ブランチはより具体的で、個別のネットワーク領域を扱っています。

SNMP は、プロトコルデータユニット (PDU) の形式でネットワーク情報を交換します。PDU には、マネージドデバイスに保存されている変数に関する情報が含まれます。これらの変数は、管理オブジェクトとも呼ばれ、必要に応じて NMS にレポートされる値とタイトルを持ちます。NMS とマネージド機器の間の通信は、NMS が更新情報を送信したり、情報を要求したり、マネージドオブジェクトがサーバーのシャットダウンや起動時に **トラップ** と呼ばれる通知や警告を送信することで行われます。

21.10.2. SNMP サポートの有効化および無効化

デフォルトでは、SNMP プロトコルが Directory Server で有効になり、サブエージェントの設定後にこれを使用できます。

インスタンスで SNMP を有効または無効にするには、***nsSNMPEnabled*** パラメーターを **on** または **off** に設定します。たとえば、Directory Server インスタンスで SNMP を無効にするには、次のコマンドを実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=SNMP,cn=config
changetype: modify
replace: nsSNMPEnabled
nsSNMPEnabled: on
```

21.10.3. SNMP を使用してインスタンスを識別するためのパラメーターの設定

Directory Server は、SNMP を使用してインスタンスの特定に役立つ以下の属性を提供します。

- ***nsSNMPOrganization***
- ***nsSNMPLocation***
- ***nsSNMPContact***
- ***nsSNMPDescription***

パラメーターの詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 の 『cn=SNMP』 セクションを参照してください。

たとえば、***nsSNMPLocation*** パラメーターを **Munich, Germany** に設定するには、以下を実行します。

```
# ldapmodify -D "cn=Directory Manager" -W -p 389 -h server.example.com -x

dn: cn=SNMP,cn=config
```

```
changetype: modify
replace: nsSNMPLocation
nsSNMPLocation: Munich, Germany
```

21.10.4. Directory Server の SNMP Agent の設定

SNMP プロトコルを使用して Directory Server から情報をクエリーするには、SNMP エージェントを設定します。

1. 389-ds-base-snmp パッケージおよび net-snmp パッケージをインストールします。

```
# yum install 389-ds-base-snmp net-snmp
```

2. SNMP マスターエージェントを設定するには、**/etc/snmp/snmpd.conf** ファイルを編集し、以下のエントリーを追加してエージェントの拡張性 (AgentX) プロトコルを有効にします。

```
master agentx
```

   AgentX プロトコルの詳細は RFC 2741 を参照してください。

3. SNMP サブエージェントを設定するには、**/etc/dirsrv/config/ldap-agent.conf** ファイルを編集し、監視する各 Directory Server インスタンスに server パラメーターを追加します。以下に例を示します。

```
server slapd-instance_name
```

4. 必要に応じて、SNMP ユーザーアカウントを作成します。

   a. **snmpd** サービスを停止します。

```
# systemctl stop snmpd
```

   b. SNMP ユーザーアカウントを作成します。以下に例を示します。

```
# net-snmp-create-v3-user -A authentication_password -a SHA \
    -X private_password -x AES user_name
```

   コマンドで使用されるパラメーターの詳細は、net-snmp-create-v3-user(1) の man ページを参照してください。

   c. **snmpd** サービスを起動します。

```
# systemctl start snmpd
```

5. 必要に応じて、Directory Server 記述プロパティーを設定します。詳細については、「SNMP を使用してインスタンスを識別するためのパラメーターの設定」 を参照してください。

6. **dirsrv-snmp** サービスを起動します。

```
# systemctl start dirsrv-snmp
```

7. 必要に応じて、設定を確認するには、以下を実行します。

a. net-snmp-utils パッケージをインストールします。

```
# yum install net-snmp-utils
```

b. Directory Server オブジェクト識別子 (OID) をクエリーします。以下に例を示します。

```
# snmpwalk -v3 -u user_name -M /usr/share/snmp/mibs:/usr/share/dirsrv/mibs/ \
    -l AuthPriv -m +RHDS-MIB -A authentication_password -a SHA
    -X private_password -x AES server.example.com .1.3.6.1.4.1.2312.6.1.1
```

21.10.5. SNMP トラップの設定

SNMP トラップは基本的に、監視対象のサーバーによって問題が発生した場合に通知をトリガーするしきい値です。トラップを使用するには、マスターエージェントを設定して、トラップを許可し、それらの操作を行うように設定する必要があります。たとえば、トラップは、Directory Server インスタンスの管理者が停止するメール通知をトリガーできます。

サブエージェントは、トラップをマスターエージェントに送信するだけです。マスターエージェントとトラップハンドラーは、使用している SNMP マスターエージェントのドキュメントに従って設定する必要があります。

トラップは **Entity Table** からの情報に付随します。これには、名前やバージョン番号などの Directory Server インスタンスに固有の情報が含まれます。**Entity Table** は、「エンティティーテーブル」 で説明されています。つまり、マスターエージェントがトラップを受けたときに取るアクションは、あるインスタンスでは ***dsEntityContact*** 変数に定義された電子メールアドレスに電子メールを送信する一方で、別のインスタンスでは ***dsEntityContact*** 変数に定義されたページャー番号に通知を送信するなど、柔軟に対応することができます。

サブエージェントでサポートされるトラップは 2 つあります。

- **DirectoryServerDown.** このトラップは、サブエージェントが Directory Server が実行されていないことを検出するたびに生成されます。このトラップは、Directory Server インスタンスの説明、バージョン、物理的な場所、および連絡先情報と共に送信されます。詳細は、***dsEntityDescr*** 変数、***dsEntityVers*** 変数、***dsEntityLocation*** 変数、および ***dsEntityContact*** 変数を参照してください。

- **DirectoryServerStart.** このトラップは、サブエージェントが Directory Server が起動または再起動していることを検出すると常に生成されます。このトラップは、Directory Server インスタンスの説明、バージョン、物理的な場所、および連絡先情報と共に送信されます。詳細は、***dsEntityDescr*** 変数、***dsEntityVers*** 変数、***dsEntityLocation*** 変数、および ***dsEntityContact*** 変数を参照してください。

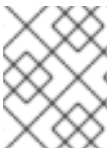
21.10.6. 管理情報ベースの使用

Directory Server の MIB は、**/usr/share/dirsrv/mibs** ディレクトリーに保存されている **redhat-directory.mib** と呼ばれるファイルです。この MIB には、そのディレクトリーのネットワーク管理に関する変数の定義が含まれます。これらの変数は、管理オブジェクトと呼ばれます。ディレクトリー MIB および Net-SNMP を使用すると、ネットワーク上の他の全デバイスと同様にディレクトリーを監視できます。MIB を使用する方法は、「Directory Server の SNMP Agent の設定」 を参照してください。

クライアントツールは、Directory Server MIB を読み込み、以下のセクションに記載されている変数名を使用する必要があります。

ディレクトリー MIB を使用すると、管理者は SNMP を使用してディレクトリーの管理情報を確認し、リアルタイムでサーバーを監視できるようになります。ディレクトリー MIB は、管理オブジェクトの 4 つの異なるテーブルに分類されています。

- 「操作表」
- 「エントリー表」
- 「エンティティーテーブル」
- 「対話表」



注記

SNMP に監視される Directory Server 属性はすべて、32 ビットシステムであってもカウンターに 64 ビット整数を使用します。

21.10.6.1. 操作表

Operations Table は、Directory Server のアクセス、操作、およびエラーに関する統計情報を提供します。表21.1「操作テーブル: 管理オブジェクトと説明」 **redhat-directory.mib** ファイルの **Operations Table** に格納されている管理オブジェクトについて説明します。

表21.1 操作テーブル: 管理オブジェクトと説明

管理オブジェクト	説明
dsAnonymousBinds	サーバーの起動以降、ディレクトリーへの匿名バインド数。
dsUnauthBinds	サーバーの起動以降、ディレクトリーへの認証されていないバインド数。
dsSimpleAuthBinds	サーバーが起動してから、単純な認証方法 (パスワード保護など) で確立された、ディレクトリーのバインド数。
dsStrongAuthBinds	サーバーの起動してから、強力な認証方法 (TLS や、Kerberos のような SASL メカニズムなど) を使用して確立されたディレクトリーへのバインドの数。
dsBindSecurityErrors	サーバーの起動以降に、認証失敗または無効な認証情報により、ディレクトリーで拒否されたバインド要求の数。
dsInOps	サーバーの起動以降、別のディレクトリーからこのディレクトリーに転送される操作の数。
dsReadOps	アプリケーションが起動してからこのディレクトリーによる読み取り操作の数。LDAP は検索操作を使用して間接的に読み取り操作を実装するため、このオブジェクトの値は常に **0** になります。
dsCompareOps	サーバーの起動時にこのディレクトリーによる比較操作の数。
dsAddEntryOps	サーバーの起動時にこのディレクトリーによる追加操作の数。

管理オブジェクト	説明
dsRemoveEntryOps	サーバーの起動以降、このディレクトリーがサービス化された削除操作の数。
dsModifyEntryOps	サーバーの起動以降、このディレクトリーがサービス化された変更操作の数。
dsModifyRDNOps	サーバー起動以降、このディレクトリーが処理する RDN 操作の数。
dsListOps	サーバーの起動時にこのディレクトリーによるリスト操作の数。LDAP は検索操作を使用して間接的にリスト操作を実装するため、このオブジェクトの値は常に **0** になります。
dsSearchOps	サーバーの起動以降、このディレクトリーで処理された検索操作の合計数。
dsOneLevelSearchOps	サーバー起動以降、このディレクトリーが処理する 1 レベルの検索操作の数。
dsWholeSubtreeSearchOps	サーバー起動以降、このディレクトリーが指定したサブツリー検索操作全体の数。
dsReferrals	サーバー起動からクライアント要求に対応して、このディレクトリーが返す参照数。
dsSecurityErrors	セキュリティー要件を満たしていないこのディレクトリーに転送される操作の数。
dsErrors	エラー (セキュリティーエラーや参照エラー以外) のためにサービスを提供できなかった要求の数です。エラーには、名前エラー、更新エラー、属性エラー、およびサービスエラーなどがあります。部分的に設定されたリクエストはエラーとしてカウントされません。

21.10.6.2. エントリー表

Entries Table は、ディレクトリーエントリーの内容に関する情報を提供します。表21.2「エントリーテーブル: 管理オブジェクトと説明」 は、 **redhat-directory.mib** ファイルの **Entries Table** に保存されている管理オブジェクトを説明します。

表21.2 エントリーテーブル: 管理オブジェクトと説明

管理オブジェクト	説明
dsCopyEntries	このディレクトリーのコピーが含まれるディレクトリーエントリーの数。このオブジェクトの値は、常に **0** になります (現在実行された更新は実行されていません)。
dsCacheEntries	ディレクトリーにキャッシュされたエントリーの数。

管理オブジェクト	説明
dsCacheHits	アプリケーションが起動してからローカルに保持されたキャッシュから指定された操作数。

21.10.6.3. エンティティーテーブル

Entity Table には、Directory Server インスタンスに関する識別情報が含まれます。「SNMP を使用してインスタンスを識別するためのパラメーターの設定」 で説明されているように、**Entity Table** の値は **cn=SNMP,cn=config** エントリーに設定されます。

表21.3「エントリーの表: 管理オブジェクトおよび説明」は、**redhat-directory.mib** ファイルの **Entity Table** に保存されている管理オブジェクトを説明します。

表21.3 エントリーの表: 管理オブジェクトおよび説明

管理オブジェクト	説明
dsEntityDescr	Directory Server インスタンスの説明セット。
dsEntityVers	Directory Server インスタンスの Directory Server のバージョン番号。
dsEntityOrg	Directory Server インスタンスに対応する組織。
dsEntityLocation	Directory Server インスタンスの物理的な場所。
dsEntityContact	Directory Server インスタンス担当する担当者の名前と連絡先情報。
dsEntityName	Directory Server インスタンスの名前。

21.10.6.4. 対話表

注記

Interaction Table はサブエージェントではサポート **されていません**。サブエージェントはテーブルをクエリーできますが、有効なデータで更新されることはありません。

表21.4「対話テーブル: 管理オブジェクトと説明」は、**redhat-directory.mib** ファイルの **Interaction Table** に保存されている管理オブジェクトを説明します。

表21.4 対話テーブル: 管理オブジェクトと説明

管理オブジェクト	説明
dsIntTable	表の各行で、監視される Directory Server とそれぞれのピア Directory Server との対話の履歴に関連する詳細が表示されます。

管理オブジェクト	説明
dsIntEntry	Directory Server と相手の Directory Server との相互作用の詳細を示すエントリー。
dsIntIndex	**applIndex** と共に一意の鍵の一部であり、(**applIndex** で参照される) Directory Server と相手の Directory Server との間の (試行された) 相互作用に関する有用な情報を含む概念的な行を特定するためのものです。
dsName	このエントリーが属するピア Directory Server の識別名 (DN)。
dsTimeOfCreation	この行が作成された場合の **sysUpTime** の値。ネットワーク管理サブシステムが初期化される前にエントリーが作成されると、このオブジェクトにはゼロの値が含まれます。
dsTimeOfLastAttempt	この Directory Server に対する接続最終試行時の **sysUpTime** の値。ネットワーク管理サブシステムが初期化される前に最後の試行が行われた場合、このオブジェクトにはゼロの値が含まれます。
dsTimeOfLastSuccess	この Directory Server に問い合わせた最後の試行時の **sysUpTime** の値。このエントリーは、成功した試行がない場合や、最後に成功した試行がネットワーク管理サブシステムの初期化前に行われた場合には、0 の値になります。
dsFailuresSinceLastSuccess	この Directory Server への初回連絡の試行に成功した後の失敗回数。試行に成功しなかった場合、このカウンターには、このエントリーが作成されてからの失敗回数が格納されます。
dsFailures	このエントリーの作成からの累積的な障害。
dsSuccesses	このエントリーの作成以降、累積成功。
dsURL	Directory Server アプリケーションの URL。

第22章 高可用性および障害復旧計画の作成

Directory Server デプロイメントを効率的に実行する場合は、その最悪のケースシナリオに計画されています。この章では、災害復旧計画を作成するための一般的な原則を説明し、災害復旧に役立つ Directory Server の機能を紹介します。

障害復旧 は、ある種の壊滅的な障害が発生した場合に、ある動作環境から別の動作環境へのスムーズな移行を計画および実行する方法です。Directory Server の災害復旧計画は、大規模な事業継続計画の一部である場合もあれば、ディレクトリーサービスの中断に特化した独立した計画である場合もあります。

注記

本章では、障害復旧に関する非常に一般的な概念を説明します。

障害復旧は非常に複雑で、詳細固有の内容になります。Red Hat Directory Server のような機密性の高いサービスやミッションクリティカルなサービスに対する障害復旧の設計、保守、テストには、専門サービスの利用を検討してください。

22.1. 潜在的なシナリオの特定

最初のステップでは、発生する可能性のある問題、サービスへの影響、および実行すべき応答を特定します。『Red Hat Directory Server Deployment Guide』では、管理者が既存のインフラストラクチャーと提案するインフラストラクチャーのサイトサーベイを行い、どのようなディレクトリーを設計するかを決定しました。災害対策についても同様で、表22.1「障害シナリオおよび応答」のように、データインフラストラクチャーがどこにあるかを特定し、そのコンポーネントが失われた場合の影響を判断し、理想的な対応策を検討します。

表22.1障害シナリオおよび応答

シナリオ	インフラストラクチャーへの影響	理想的な応答
データの破損	ソフトウェアやハードウェア障害 (または悪意のある攻撃経由) を介して、1つのサイトまたは1つサーバーのデータが破損する可能性があります。その破損したサーバーがマルチサプライヤーレプリケーションのサプライヤーである場合、破損はすぐにデプロイメント全体に伝播してしまいます。	破損していないデータの最新のバックアップにアクセスできる、分離されたサーバーを用意する必要があります。問題が検出された場合は、通常のインフラストラクチャーでのレプリケーションを中断し、このサーバーをオンラインにして、適切なデータでサプライヤーを再初期化することができます。
自然な障害およびその他の大量イベント	自然災害は、長期間の停電だけでなく、オフィスやデータセンター全体を停止させる可能性があります。	ディレクトリー操作は、同じデータを使用して、別の物理の場所にあるミラーリングされたサイトに転送できます。
サーバーまたはマシンの損失	1つのマシンが失敗する可能性があります。	同じデータを持つ別のマシンは、失われたマシンがあることを想定できます。

22.2. ロールオーバーの種類の定義

災害復旧は、あるシステムから別のシステムに移行するプロセスであり、可能な限りサービスを中断するプロセスです。これは **ロールオーバー** と呼ばれ、ロールオーバーを行う方法は 3 つあります。

- **ホットロール** オーバーは、インフラストラクチャーが別のサイトで完全にミラーリングされ、バックアップサイトは常にプライマリーサイトで最新の状態であることを意味します。これには、プライマリーからバックアップに操作を切り替えるための調整のみが必要になります。
- **ウォーム** ロールオーバーとは、バックアップサイトのすべての要素 (適切なネットワーク接続、必要なすべてのアプリケーションとハードウェア) は整っているが、システムがアクティブに稼働していない、または必ずしも設定されていない状態を指します。これには、マシンを設定し、システムの実行に追加の時間が必要になる場合があります。
- **コールド** ロールオーバーとは、サイトは利用可能だが、それをセットアップするためのリソースがすぐには得られないことを意味します。

ロールオーバーの種類における明らかな違いは、バックアップサイトの設定に必要な時間と費用です。ホットサイトおよびウォームサイトは、立ち上げや運営にかかる初期費用が高くなります。

計画している特定の障害シナリオに応じて、ロールオーバータイプの組み合わせを使用できます。たとえば、1 台のサーバーが失われた場合のロールオーバー計画では、Directory Server インスタンスの仮想マシンコピーを作成して保持し、数分以内にオンラインにすることで、簡単かつ比較的安価にホットロールオーバーを利用することができます。仮想マシンを別の施設やネットワークで維持する必要もありません。一方、コールドロールオーバーは、データセンターやオフィス全体の損失を想定して計画することができます。

ロールオーバーのプロセスを、災害シナリオの深刻さ、予算と利用可能なリソース、問題発生の可能性に合わせます。

22.3. 障害復旧における便利な DIRECTORY SERVER 機能の特定

復旧で最も難しいのはハードウェアではなく、サーバー内のデータの信頼できるコピーを得ることです。障害復旧用にデータコピーを準備する優れたツールとして、Directory Server には 3 つの機能があります。

- データベースのバックアップおよびバックアップの定期的な検証
- マルチサプライヤーのレプリケーション、チェーン、データベースのバックアップ、および名前付きパイプスクリプトでサーバーの監視
- チェーン

また、名前付きパイプスクリプトや他の Directory Server のパフォーマンスカウンターを使用してサーバーを監視することで、特定の重要なイベントを発見し、迅速に対応することができます。

22.3.1. 災害リカバリー用のディレクトリーデータのバックアップ

障害復旧で最も便利なツールは、ディレクトリーインスタンスのバックアップを頻繁に行うことです。アーカイブは、プライマリーデータセンターとは異なる場所で、コールドバックアップの場所で物理メディアに保存できます。

バックアップは、cron ジョブを使用して定期的に実行するように自動化できます。たとえば、**ldap://server.example.com** インスタンスを毎日 22:00 (10pm) に作成するには、以下のコマンドを実行します。

```
0 22 * * 1    /usr/sbin/dsconf -D "cn=Directory Manager" ldap://server.example.com backup create
```

dsconf backup create コマンドは、最初にサーバーを停止せずにディレクトリーデータをバックアップします。

注記

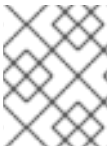
Red Hat は、マルチサプライヤーレプリケーション環境のすべてのサーバー上でデータのバックアップを行うことを推奨します。

「Directory Server のバックアップ」では、ディレクトリーデータベースとディレクトリー設定 (**dse.ldif** ファイル) の両方のバックアップが対象です。

22.3.2. 高可用性のためのマルチサプライヤーレプリケーション

マルチサプライヤーレプリケーションは、1台のサーバーや、場合によってはオフィスや部門全体における損失に関する最善の防御策です。少数のサーバーがデータサプライヤーとなる一方で、複数のサーバーがすべて同じデータを保持しており、1つのレプリケーション環境に数十台のサプライヤーとハブが存在する可能性があります。これにより、複数のサーバーがオフラインであっても、クライアントが情報にアクセスできる状態を維持します。

レプリケーションは、データをサーバーにコピーしたり、より迅速に交換するのに使用できます。



注記

レプリケーションを介して伝播されるデータの破損を保護するには、データベースを頻繁にバックアップします。

レプリケーション設定により、プライマリーサプライヤーがアクセスできない場合に、書き込み操作はフェイルオーバーサーバーと呼ばれます。つまり、サーバーがオフラインであっても、書き込み操作はクライアントの観点から通常通り続行できます。

例22.1 マルチサプライヤーレプリケーションのシナリオ

レプリケーションは、いくつかのシナリオで障害復旧のための汎用的なツールです。

- 単一のサーバー障害の場合、そのインスタンスに保存されているすべてのデータには、他のサーバーからアクセスと取得が可能です。
- オフィス全体やコロケーション施設が失われた場合は、まったく別の物理的な場所にサーバーをミラーリングすることができます (Directory Server の広域レプリケーションの性能が役立ちます)。最低限の努力で、新しいサーバーをオンラインにすることなく、トラフィックは複製されたサイトにリダイレクトされます。

レプリケーションの設定については、15章 *レプリケーションの管理*で説明しています。

22.3.3. 高可用性のデータベースチェーン

チェーン は、クライアントが要求を1台のサーバーに送信し、その要求が自動的に別のサーバーに転送されて処理されるという設定です。データベースリンク (またはチェーン) に複数のサーバーを設定して、1つのサーバーが利用できない場合に自動フェイルオーバーを行うことができます。

例22.2 連鎖のシナリオ

チェーンがフェイルオーバーサーバーのリストと組み合わせると、クライアントトラフィックはオフライン時に単一サーバー (またはサーバーのグループも) から自動的にリダイレクトできます。これは復旧には役立ちませんが、プライマリーサーバーからバックアップサーバーへの移行を管理するのに役立ちます。

チェーンデータベースについては、「データベースリンクの作成および維持」で説明しています。

22.4. リカバリープロセスの定義

障害復旧を支援するツールは数多くありますが、効果的な復旧プロセスは、あらゆるシナリオで何をすべきかを明確に定義したプランを持つことに集約されます。少なくとも 2 つの項目を明確に特定する必要があります。

- 障害を示すシグナル。明白なもの (大規模な停電、ネットワーク損失、または火災) もありますが、定義する必要があるものもあります。たとえば、バックアップサーバーをオンラインにする必要があるというメッセージは何ですか。
- 障害に応答する人および方法。災害が発生したら、誰が責任を持って行動しますか。これらのイベントについて通知する方法。何を期待されていますか。

重要

- 災害復旧計画を出力したものをサイト外に保管します。
- 障害復旧計画を定期的にテストし、設定とインフラストラクチャーの変更後にテストします。

22.5. 基本的な例: リカバリーの実行

管理者 (John Smith) は、そのディレクトリーデプロイメントの障害復旧計画を作成する必要があります。Example Corp. は、サンフランシスコ、ダラス、アーリントンの 3 箇所にオフィスを構えています。各サイトには 10 台のサーバーがあり、ローカルで相互にレプリケーションを行い、各サイトの 1 台のサーバーが他の 2 つのサイトの別のサーバーにレプリケーションを行います。

各サイトのディレクトリーには、ビジネスに不可欠な顧客データや、人事データが保存されています。請求などの操作を実行するために、データにいくつかの外部アプリケーションにアクセスする必要があります。

John Smith の最初の手順は、サイト Survey を実行することです。彼が探しているのは、ディレクトリーの使用状況 (アクセスするクライアントやサイト全体のトラフィック負荷)、現在の資産、そして取得が必要な資産の 3 つです。これは、Red Hat Directory Server のデプロイ時に実行する初期サイトサーベイと似ています。

次の手順では、障害のシナリオを特定します。3 つのサイトのうち 2 つ (サンフランシスコとダラス) は、自然災害に対して非常に脆弱です。3 つのサイトはすべて、電源やインターネットアクセスの停止など、通常の中断が生じる可能性がありました。また、各サイトのサプライヤーは独自のローカルデータをするため、各サイトはサーバーインスタンスまたはマシンの損失に対して脆弱です。

John Smith は、障害復旧計画を 3 つの部分に分類します。

- プラン A では、Directory Server における 1 つのインスタンスの損失について扱います。

- プラン B は、何らかのデータの破損または攻撃を扱います。
- プラン C では、オフィス全体が失われた場合を扱います。

プラン A と B の場合、John Smith はホットリカバリーを使用して、1 つのインスタンスからバックアップに機能を即座に切り替えることにしました。各サーバーは毎日バックアップされ、cron ジョブを使用してアーカイブをコピーし、次にアーカイブが仮想マシンでコピーされ、復元されます。仮想マシンは別のサブネットに保管されますが、そのピアがオフラインになるとすぐに切り替えることができます。John Smith は、簡単な SNMP トラップを使用して、各 Directory Server インスタンスの可用性を追跡します。

プラン C はより広範囲です。サイト間のレプリケーションとローカルのバックアップに加えて、彼は各サイトのバックアップの物理的なコピーを、ローカルのインスタンスごとに、週に一度、他の 2 つのコロケーション施設に郵送することにしました。また、仮想マシンを使用してサイト全体をリストアするために、十分なインターネットアクセスとソフトウェアライセンスを備えた予備のサーバーを、他の異なるコロケーション施設の 1 つに置いています。彼は、IT スタッフのほとんどがいる Arlington を一次復旧拠点とし、次にサンフランシスコ、最後にダラスと、人員の分布に基づいて指定しています。すべてのイベントにおいて、3 つのサイトの IT 管理者に通知され、管理者は仮想マシンのセットアップ、物理的なバックアップからの Directory Server インスタンスのリストア、クライアントトラフィックの再ルーティングなどの責任を負います。

John Smith は、新しいハードウェアやアプリケーションの変更を考慮して、四半期ごとに計画を見直し、更新する予定です。年に一度、3 つのサイトすべてが、Disaster Plan C の手順に従って、他の 2 つのサイトのリカバリーとデプロイメントの手順を実行しなければなりません。

第23章 テストエントリーの作成

dsctl ldifgen コマンドは、異なるタイプのテストエントリーを持つ LDIF ファイルを作成します。たとえば、この LDIF ファイルを使用して、テストインスタンスまたはサブツリーを設定すると、サンプルエントリーで Directory Server のパフォーマンスをテストできます。

以下のエントリータイプの引数のいずれかを **dsctl ldifgen** に渡すことができます。

- **users**: ユーザーエントリーが含まれる LDIF ファイルを作成します。
- **groups**: 静的グループおよびメンバーエントリーが含まれる LDIF ファイルを作成します。
- **cos-def**: 従来のポインターまたは間接的な Class of Service (CoS) 定義が含まれる LDIF ファイルを作成します。
- **cos-template**: CoS テンプレートが含まれる LDIF ファイルを作成します。
- **roles**: 管理されたロールエントリー、フィルターが設定されたロールエントリーー、または間接ロールエントリーが含まれる LDIF ファイルを作成します。
- **mod-load**: 変更操作が含まれる LDIF ファイルを作成します。**ldapmodify** ユーティリティーを使用してこのファイルをインポートします。
- **nested**: カスケードツリーやフラクタルツリーのように重く入れ子になったエントリーを含む LDIF ファイルを作成します。

注記

dsctl ldifgen コマンドは LDIF ファイルのみを作成します。ファイルを Directory Server インスタンスに読み込むには、以下を使用します。

- **mod-load** オプションを使用して LDIF ファイルを作成した後の **ldapmodify** ユーティリティー
- その他のすべてのオプションの **ldapadd** ユーティリティー

ネスト されたエントリータイプを除き、コマンドラインオプションを指定しないと、**dsctl ldifgen** コマンドはインタラクティブモードを使用します。

```
# dsctl instance_name ldifgen entry_type
```

23.1. サンプルユーザーエントリーを使用した LDIF ファイルの作成

dsctl ldifgen users コマンドを使用して、サンプルユーザーエントリーのある LDIF ファイルを作成します。たとえば、**dc=example,dc=com** 接尾辞に 100,000 人の一般ユーザーを追加する **/tmp/users.ldif** という名前の LDIF ファイルを作成するには、次のように入力します。

```
# dsctl instance_name ldifgen users --suffix "dc=example,dc=com" --number 100000 --generic --ldif-file=/tmp/users.ldif
```

このコマンドは、以下の組織単位 (OU) を作成し、ユーザーをこれらの OU にランダムに割り当てることに注意してください。

- **ou=accounting**

- **ou=product development**
- **ou=product testing**
- **ou=human resources**
- **ou=payroll**
- **ou=people**
- **ou=groups**

詳細と、LDIF ファイルの作成に使用できるその他のオプションについては、以下を入力します。

```
# dsctl instance_name ldifgen users --help
```

23.2. グループエントリーの例を使用した LDIF ファイルの作成

dsctl ldifgen groups コマンドを使用して、サンプルグループエントリーのある LDIF ファイルを作成します。たとえば、500 グループを **ou=groups,dc=example,dc=com** エントリーに追加し、各グループに **100** メンバーが含まれる **/tmp/groups.ldif** という名前の LDIF ファイルを作成するには、次のコマンドを実行します。

```
# dsctl instance_name ldifgen groups --number 500 --suffix "dc=example,dc=com" --parent
"ou=groups,dc=example,dc=com" --num-members 100 --create-members --member-parent
"ou=People,dc=example,dc=com" --ldif-file /tmp/group.ldif example
```

このコマンドは、LDIF 文を作成して、**ou=People,dc=example,dc=com** にユーザーエントリーを追加することに注意してください。

重要

大規模なグループを作成し、**ldapmodif** ユーティリティーを使用してグループを追加しようとすると、BER (Basic Encoding Rules) サイズ制限を超えており、インポートに失敗します。この場合は、**cn=config** エントリーの ***nsslapd-maxbersize*** パラメーターの値を増やします。

詳細と、LDIF ファイルの作成後に使用できるその他のオプションは、以下を入力します。

```
# dsctl instance_name ldifgen groups --help
```

23.3. COS 定義の例を使用した LDIF ファイルの作成

dsctl ldifgen cos-def コマンドを使用して、Class of Service (CoS) 定義を持つ LDIF ファイルを作成します。たとえば、従来の CoS 定義を **ou=cos definitions,dc=example,dc=com** エントリーに追加する **/tmp/cos-definition.ldif** という名前の LDIF ファイルを作成するには、次のコマンドを実行します。

```
# dsctl instance_name ldifgen cos-def Postal_Def --type classic --parent "ou=cos
definitions,dc=example,dc=com" --cos-specifier businessCatagory --cos-template
"cn=sales,cn=classicCoS,dc=example,dc=com" --cos-attr postalcode telephonenumber --ldif-file
/tmp/cos-definition.ldif
```

この例で使用されるオプションの詳細や、LDIF ファイルの作成を設定する他のオプションの詳細は、次のコマンドを実行します。

```
# dsctl instance_name ldifgen cos-def --help
```

23.4. EXAMPLE MODIFICATION STATEMENTS を使用した LDIF ファイルの作成

dsctl ldifgen mod-load コマンドを使用して、更新操作が含まれる LDIF ファイルを作成します。

```
# dsctl instance_name ldifgen mod-load --parent dc=example,dc=com --num-users 1000 --create-users --mod-users 1000 --add-users 10 --del-users 100 --mod-users 1000 --modrdn-users 100 --mod-attrs cn uid sn --delete-users
```

このコマンドは、以下を行う文が含まれる **/tmp/modifications.ldif** ファイルを作成します。

1. **ADD** 操作 1000 の LDIF ファイルを作成し、ユーザーエントリーを作成します。
2. ***cn*** 属性、***uid*** 属性、および ***sn*** 属性を変更して、すべてのエントリーを変更します。
3. 10 ユーザーエントリーを追加します。
4. 100 個の **MODRDN** 操作を実行します。
5. 100 エントリーの削除
6. 末尾の残りのエントリーをすべて削除します。

詳細と、LDIF ファイルの作成に使用できるその他のオプションについては、以下を入力します。

```
# dsctl instance_name ldifgen mod-load --help
```

23.5. ネストされたサンプルエントリーを持つ LDIF ファイルの作成

dsctl ldifgen nested コマンドを使用して、大きく入れ子になったカスケードフラクタル構造を含む LDIF ファイルを作成します。たとえば、**/tmp/nested.nldif** という名前の LDIF ファイルを作成するには、**dc=example,dc=com** エントリー配下の異なる組織単位 (OU) に合計 600 ユーザーの追加し、各 OU には最大 100 ユーザーが含まれています。

```
# dsctl instance_name ldifgen nested --num-users 600 --node-limit 100 --suffix "dc=example,dc=com"
```

オプションの詳細は、以下を入力します。

```
# dsctl instance_name ldifgen nested --help
```

付録A LDAP クライアントツールの使用

Red Hat Directory Server は、OpenLDAP で提供される LDAP ツール (**ldapsearch** や **ldapmodify** など) を使用します。OpenLDAP ツールオプションは、OpenLDAP man ページ http://www.openldap.org/software/man.cgi で説明されています。

この付録では、これらの LDAP ツールを使用する際の一般的な使用例を紹介しています。

ldapsearch を使用するためのより広範な例は、14章 *ディレクトリーエントリーの検索* に記載されています。**ldapmodify** および **ldapdelete** を使用する例は 3章 *ディレクトリーエントリーの管理* に記載されています。

A.1. 延長操作の実行

Red Hat Directory Server は、特に拡張検索操作など、さまざまな拡張操作をサポートします。拡張操作は、LDAP 操作と共に追加の操作 (get effective rights 検索やサーバー側のソートなど) を渡します。同様に、LDAP クライアントは、多くの拡張操作に対応する可能性があります。

OpenLDAP LDAP ツールは、2 つの方法で拡張された操作をサポートします。すべてのクライアントツール (**ldapmodify**、**ldapsearch** など) は、**-e** オプションまたは **-E** オプションのいずれかを使用して拡張操作を送信します。**-e** 引数は、任意の OpenLDAP クライアントツールと使用でき、パスワードポリシーの処理方法など、操作に関する一般的な指示を送信できます。**-E** は、**ldapsearch** でのみ使用され、GER 検索、ソート、ページ情報などのより有用な制御を渡し、その他情報 (not-explicitly-support 拡張操作など) に関する情報を渡します。

さらに、OpenLDAP には別のツール **ldapexop** があります。これは **ldapsearch -E** を実行するのと同じように、拡張検索操作を行うためだけに使用されます。

ldapsearch の拡張操作の形式は、通常、以下のとおりです。

```
-E extended_operation_type=operation_parameters
```

拡張操作が OpenLDAP ツールで明示的に処理される場合、**extended_operation_type** は、逆参照検索の **deref** やサーバー側ソートの **sss** のようなエイリアスになります。サポートされる拡張操作では、出力がフォーマットされています。GER 検索などの他の拡張操作は、エイリアスではなく OID を使用して渡されるため、**extended_operation_type** は OID になります。サポートされていない操作の場合、ツールはサーバーからの応答を認識しないため、出力は形式化されません。

たとえば、**pg** 拡張操作タイプは、結果を簡易なページの結果に形式化します。

```
# ldapsearch -x -D "cn=Directory Manager" -W -b "ou=Engineers,ou=People,dc=example,dc=com" -E pg=3 "(objectclass=*)" cn

dn: uid=jsmith,ou=Engineers,ou=People,dc=example,dc=com
  cn: John Smith

dn: uid=bjensen,ou=Engineers,ou=People,dc=example,dc=com
  cn: Barbara Jensen

dn: uid=hmartin,ou=Engineers,ou=People,dc=example,dc=com
  cn: Henry Martin

Results are sorted.
next page size (3): 5
```

ldapexop の同じ操作は、ページングされた簡易な結果操作の OID のみを使用して実行できます (ページごとに 3 つの結果)。

```
ldapexop 1.2.840.113556.1.4.319=3
```

ただし、**ldapexop** は **ldapsearch** が実行する検索パラメーターの範囲を受け入れず、柔軟性が低下します。

A.2. エントリーの比較

ldapcompare は、エントリーをチェックして、指定したエントリーに特定値の属性が含まれているかどうかを確認します。たとえば、このチェックは、エントリーに ***sn*** 値が Smith であるかを確認します。

```
# ldapcompare -D "cn=Directory Manager" -W -p 389 -h server.example.com -x sn:smith
uid=bjensen,ou=people,dc=example,dc=com
comparing type: "sn" value: "smith" in entry "uid=bjensen,ou=people,dc=example,dc=com"
compare FALSE

ldapcompare -D "cn=Directory Manager" -W -p 389 -h server.example.com -x sn:smith
uid=jsmith,ou=people,dc=example,dc=com
comparing type: "sn" value: "smith" in entry "uid=jsmith,ou=people,dc=example,dc=com"
compare TRUE
```

compare 属性は以下の 3 つの方法のいずれかで指定できます。

- コマンドラインで直接渡される単一の **attribute:value** 文

  ```
  sn:Smith
  ```

- ***jpegPhoto*** などの属性、または証明書または CRL を検証するために、コマンドラインで渡される単一の **attribute::base64value** ステートメント

  ```
  jpegPhoto:dkdkPDKCDdko0eiofk==
  ```

- 属性の比較値のリストを含むファイルを参照する **attribute:file** 文およびスクリプトは、リストを繰り返し処理します。

  ```
  postalCode:/tmp/codes.txt
  ```

compare 操作自体は、特定のエントリーまたはエントリーのグループに対して実行する必要があります。単一のエントリー DN をコマンドラインで渡すか、**-f** オプションを使用して指定することのできる DN のリストを指定します。

例A.11つの属性値と 1つのエントリーの比較

属性と値の比較と DN の両方はスクリプトで渡されます。

```
ldapcompare -D "cn=Directory Manager" -W -p 389 -h server.example.com -x sn:smith
uid=jsmith,ou=people,dc=example,dc=com
comparing type: "sn" value: "smith" in entry "uid=jsmith,ou=people,dc=example,dc=com"
compare TRUE
```

例A.2 ファイルからのリスト属性値の比較

まず、可能な ***sn*** 値のファイルを作成します。

```
jensen
johnson
johannson
jackson
jorgenson
```

次に、エントリーのリストを作成して、値を比較します。

```
uid=jen200,ou=people,dc=example,dc=com
uid=dsj,ou=people,dc=example,dc=com
uid=matthewjms,ou=people,dc=example,dc=com
uid=john1234,ou=people,dc=example,dc=com
uid=jack.son.1990,ou=people,dc=example,dc=com
```

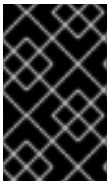
次に、スクリプトを実行します。

```
# ldapcompare -D "cn=Directory Manager" -W -p 389 -h server.example.com -x
sn:/tmp/surnames.txt -f /tmp/names.txt
comparing type: "sn" value: "jensen" in entry "uid=jen200,ou=people,dc=example,dc=com"
compare TRUE
```

A.3. パスワードの変更

ldappasswd コマンドは、新しいユーザー定義のパスワードを設定したり、アカウントの新しいパスワードを生成したりできます。その他の設定 (バインド情報、接続情報、その他のコマンド設定など) は必要なことがあり、OpenLDAP の man ページに記載されています。

```
# ldappasswd -x -D bind_dn -W -p server_port -h server_hostname [-A | -a oldPassword] [-S | -s
newPassword] [user]
```



重要

パスワードの変更操作は、TLS、STARTTLS、SASL などのセキュアな接続で実行する必要があります。LDAP クライアントに TLS を設定する方法は、「証明書を使用した認証」を参照してください。

例A.3 Directory Manager が TLS を介してユーザーのパスワードを変更

Directory Manager は、**uid=tuser1,ou=People,dc=example,dc=com** ユーザーのパスワードを TLS 経由で new_password に変更します。

```
# ldappasswd -D "cn=Directory Manager" -W -ZZ -p 389 -h server.example.com -x -s
new_password "uid=tuser1,ou=People,dc=example,dc=com"
```

例A.4 ユーザーのパスワードを生成する Directory Manager

Directory Manager は、TLS 経由で **uid=tuser2,ou=People,dc=example,dc=com** ユーザーのパスワードを生成します。

```
# ldappasswd -D "cn=Directory Manager" -W -ZZ -p 389 -h server.example.com -x
"uid=tuser2,ou=People,dc=example,dc=com"
```

例A.5 ユーザーが自分のパスワードを変更

ユーザー **tuser3** は、パスワードを TLS 経由で **old_newpassword** から **new_password** に変更します。

```
# ldappasswd -p 389 -h server.example.com -ZZ -x -D
"uid=tuser3,ou=People,dc=example,dc=com" -W -a old_password -s new_password
```

例A.6 DIGEST_MD5 によるユーザー認証および自身のパスワードの変更

ユーザー jsmith が、GSS-API で認証し、パスワードを **new_password** に変更します。

```
# ldappasswd -p 389 -h server.example.com -O noplain,minssf=1,maxbufsize=512 -Y GSSAPI -U
"dn:uid=jsmith,ou=people,dc=example,dc=com" -R EXAMPLE.COM -W -s new_password
```

例A.7 新しいパスワードに対して Kerberos プロンプトにより認証されるユーザー権限

Kerberos によって認証済みのユーザーにより、新しいパスワードが要求されます。これは TLS 上では実行されません。

```
# ldappasswd -p 389 -h server.example.com -O noplain,minssf=1,maxbufsize=512 -I
```

A.4. LDAP URL の生成

LDAP URL は、さまざまな設定エリアおよび操作で使用されます。参照元およびチェーン、レプリケーション、同期、ACI、ならびにインデックスは開始リストで使用します。間違った URL が間違ったサーバーに接続するか、単に操作が失敗する可能性があるため、正確な LDAP URL を構築することは重要です。さらに、すべての OpenLDAP ツールでは、**-H** オプションで、他の接続情報 (ホスト名、ポート、サブツリー、検索ベースなど) の代わりに LDAP URL を渡すことができます。

注記

LDAP URL は、付録C *LDAP URL* で説明されています。

ldapurl コマンドは、以下の 2 つの方法で URL を管理します。

- 指定された LDAP URL をその設定要素に分解
- 指定要素から新しくて有効な LDAP URL を作成

URL を操作するパラメーターは、表A.1「ldapurl パラメーター」に一覧表示されています。パラメーターの一覧は OpenLDAP の man ページにあります。

表A.1 ldapurl パラメーター

オプション	説明
URL の分解の場合	
-H "**URL**"	LDAP URL を渡して要素に分割します。
URL の構築の場合	
-a **attributes**	検索結果で特に返されるコンマ区切りの属性を指定します。
-b **base**	URL の検索ベースまたはサブツリーを設定します。
-f **filter**	使用する検索フィルターを設定します。
-h **hostname**	Directory Server のホスト名を指定します。
-p **port**	Directory Server のポートを指定します。
-S ldap|ldaps|ldapi	**ldap**、**ldaps**、**ldapi** など、接続に使用するプロトコルを指定します。
-s **scope**	検索条件を指定します。

例A.8 LDAP URL の無効化

ldapurl は、**-H** オプションを使用して既存の LDAP URL にフィードし、このツールは neat リストの URL の要素を返します。

```
# ldapurl -H "ldap://:389/dc=example,dc=com?cn,sn?sub?(objectclass=inetorgperson)"
scheme: ldap
port: 389
dn: dc=example,dc=com
selector: cn
selector: sn
scope: sub
filter: (objectclass=inetorgperson)
```

例A.9 LDAP URL の構築

ldapurl の最も便利なアプリケーションは、有効な LDAP URL を手動で構築することです。**ldapurl** を使用すると、URL が有効であることを確認できます。

ldapurl は、検索ベース、スコープ、および属性に対して、すべての LDAP クライアントツールの通

常の接続パラメーターと追加の **ldapsearch** 引数を受け入れますが、このツールは Directory Server インスタンスに接続しないため、バインド情報は必要ありません。接続を受け入れ、検索設定を受け入れ、それらを要素として URL に提供します。

```
ldapurl -a cn,sn -b dc=example,dc=com -s sub -f "(objectclass=inetorgperson)"

ldap://:389/dc=example,dc=com?cn,sn?sub?(objectclass=inetorgperson)
```

付録B LDAP データ交換形式

Red Hat Directory Server (Directory Server) は、LDAP データ交換形式 (LDIF) を使用して、ディレクトリーおよびディレクトリーエントリーをテキスト形式で記述します。LDIF は、初期ディレクトリーデータベースの構築や、多数のエントリーを一度にディレクトリーに追加するために使用されます。さらに、LDIF はディレクトリーエントリーの変更を説明するのにも使用されます。このため、Directory Server のコマンドラインユーティリティーのほとんどは、入力または出力のいずれかに LDIF を使用します。

LDIF はテキストファイル形式であるため、LDIF は実質的には任意の言語を使用して作成できます。すべてのディレクトリーデータは Unicode の UTF-8 エンコーディングを使用して保存されます。そのため、作成される LDIF ファイルも UTF-8 でエンコードする必要があります。

LDIF を使用してディレクトリーエントリーを変更する方法は、3章*ディレクトリーエントリーの管理*を参照してください。

B.1. LDIF ファイルの形式の概要

LDIF は、空白行で区切られた 1 つ以上のディレクトリーエントリーで設定されます。各 LDIF エントリーは、任意のエントリー ID、必須の識別名、複数のオブジェクトクラス、および複数の属性定義で設定されます。

LDIF 形式は、RFC 2849 の 『The LDAP Data Interchange Format (LDIF) 』 で定義されています。Directory Server はこの規格に準拠しています。

LDIF で表されるディレクトリーエントリーの基本的な形式は次のとおりです。

```
dn: distinguished_name
objectClass: object_class
objectClass: object_class
...
 attribute_type[;subtype]:attribute_value
...
```

- すべての LDIF エントリーには、DN と 1 つ以上のオブジェクトクラス定義が必要です。
- エントリーに定義されるオブジェクトクラスで必要な属性を含めます。
- その他の属性およびオブジェクトクラスはすべて任意です。
- オブジェクトクラスおよび属性は任意の順序で指定できます。
- コロン後のスペースは任意です。

表B.1「LDIF フィールド」 では、前の定義で示された LDIF フィールドを説明します。

表B.1 LDIF フィールド

フィールド	定義
[id]	**任意**。エントリー ID を表す正の 10 進数。データベース作成ツールは、この ID を自動的に生成します。この値は独自に追加したり、編集したりしないでください。

フィールド	定義
dn: **distinguished_name**	エントリーの識別名を指定します。
objectClass: **object_class**	このエントリーで使用するオブジェクトクラスを指定します。オブジェクトクラスは、エントリーに使用可能な属性のタイプ (スキーマ) を識別します。標準オブジェクトクラスの一覧は、『Red Hat Directory Server 11 の設定、コマンド、およびファイルリファレンス』 を参照してください。スキーマのカスタマイズに関する情報は、12章 *ディレクトリースキーマの管理*を参照してください。
attribute_type	エントリーで使用する説明的な属性を指定します。属性はスキーマで定義する必要があります。標準属性の一覧は、『Red Hat Directory Server 11 の設定、コマンド、およびファイルリファレンス』 を参照してください。スキーマのカスタマイズに関する情報は、12章 *ディレクトリースキーマの管理*を参照してください。
[**subtype**]	**任意。**サブタイプ、言語、バイナリー、または発音を指定します。このタグを使用して、対応する属性値が表現されている言語や、属性値がバイナリーであるか、属性値の発音であるかを識別します。サポートされるサブタイプタグのリストは、???を参照してください。
attribute_value	属性タイプと使用する属性値を指定します。

注記

ディレクトリーのエントリーへの変更を表す LDIF 構文は、表B.1「LDIF フィールド」 で説明されている構文とは異なります。LDIF を使用してディレクトリーエントリーを変更する方法は、3章 *ディレクトリーエントリーの管理*を参照してください。

B.2. LDIF での行継続

LDIF ファイルでは、行の継続部分をスペース 1 つ分インデントすることで、行を分割して継続する (**折りたたみ** と呼ばれる) ことができます。たとえば、以下の 2 つのステートメントは同じです。

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

dn: cn=some_e
 xample_user,
 dc=example,d
 c=com
```

LDIF 行を分割して継続する必要はありません。ただし、それを行うことで LDIF ファイルの読みやすさが向上することがあります。通常の規則では、LDIF ファイルには 78 列を超えるテキストが含まれません。

B.3. バイナリーデータの表現

JPEG イメージなどのバイナリーデータは、標準の LDIF 表記または base-64 エンコードの 2 つの方法のいずれかを使用して LDIF で表されます。

B.3.1. 標準の LDIF 表記

標準の LDIF 表記は、データがバイナリーであることを示すために、(<) より小さい記号を使用します。以下に例を示します。

```
jpegphoto: < file:/path/to/photo
```

この標準表記では、**ldapmodify -b** パラメーターを指定する必要はありません。ただし、標準の表記法では、LDIF ファイルまたは LDIF 更新ステートメントの先頭に以下の行を追加する必要があります。

```
version: 1
```

以下に例を示します。

```
# ldapmodify -x -D userDN -W

version: 1
dn: cn=Barney Fife,ou=People,dc=example,dc=com
changetype: modify
add: usercertificate
usercertificate;binary: < file: BarneysCert
```

B.3.2. Base-64 でエンコード

バイナリーデータを Base-64 に変換して LDIF ファイルにすることで、イメージから TLS 証明書までさまざまなデータに対応します。ベース 64 でエンコードされたデータは、**::** 記号を使用して識別されます。以下に例を示します。

```
jpegPhoto::encoded_data
```

バイナリーデータの他に、base-64 でエンコードする必要のあるその他の値には以下が含まれます。

- コロン (:) またはスペースで始まる値。
- ASCII 以外のデータを含む値 (新しい行を含む)。

-b パラメーターで **ldif** コマンドラインユーティリティーを使用して、バイナリーデータを LDIF 形式に変換します。

```
# ldif -b attribute_name
```

attribute_name は、バイナリーデータを指定する属性の名前です。バイナリーデータは標準入力から読み取られ、結果は標準出力に書き込まれます。そのため、入力ファイルと出力ファイルの選択にはリダイレクト演算子を使用します。

ldif コマンドラインユーティリティーは入力を取得し、これを正しい行継続性と適切な属性情報で形式化します。**ldif** ユーティリティーは、入力に base-64 エンコーディングが必要かどうかも評価します。以下に例を示します。

```
# ldif -b jpegPhoto <  mark.jpg > out.ldif
```

この例では、JPEG 形式のイメージを含むバイナリーファイルを取得し、属性 ***jpegPhoto*** の LDIF 形式に変換します。出力は **out.ldif** に保存されます。

-b オプションは、**ldif** ユーティリティーが入力全体を単一のバイナリー値として解釈するように指定します。**-b** が存在しない場合は、各行が個別の入力値であると見なされます。

B.4. LDIF を使用したディレクトリーエントリーの指定

多くのエントリーはディレクトリーに保存できます。本セクションでは、ディレクトリーで使用する最も一般的なエントリーの 3 つ (ドメイン、組織単位、および組織の人のエントリー) を 3 つにまとめる方法を説明します。

エントリーに定義されているオブジェクトクラスは、エントリーがドメインまたはドメインコンポーネント、組織単位、組織人、またはその他のタイプのエントリーを表しているかどうかを示すものです。デフォルトでディレクトリーで使用できるオブジェクトクラスの完全一覧と、よく使用される属性の一覧については、『Red Hat Directory Server 11 の設定、コマンド、およびファイルリファレンス』 を参照してください。

B.4.1. ドメインエントリーの指定

ディレクトリーにはドメインエントリーが 1 つ以上含まれます。通常、これはディレクトリーの最初の、つまり一番上のエントリーです。ドメインエントリーは、多くの場合、ディレクトリーの DNS ホストおよびドメイン名に対応します。たとえば、Directory Server ホストが **ldap.example.com** と呼ばれている場合は、ディレクトリーのドメインエントリーは **dc=ldap,dc=example,dc=com** または単に **dc=example,dc=com** という名前になります。

ドメインの定義に使用される LDIF エントリーは以下のようになります。

```
dn: distinguished_name
objectClass: top
objectClass: domain
dc: domain_component_name
 list_of_optional_attributes
...
```

以下は、LDIF 形式のドメインエントリーの例です。

```
dn: dc=example,dc=com
objectclass: top
objectclass: domain
dc: example
description: Fictional example company
```

LDIF フォーマットのドメインエントリーの各要素が 表B.2「ドメインエントリーの LDIF 要素」 に定義されています。

表B.2 ドメインエントリーの LDIF 要素

LDIF 要素	説明
dn: **distinguished_name**	**必須。** エントリーの識別名を指定します。

LDIF 要素	説明
objectClass: top	**必須。top** オブジェクトクラスを指定します。
objectClass: domain	**domain** オブジェクトクラスを指定します。この行は、エントリーをドメインまたはドメインコンポーネントとして定義します。このオブジェクトクラスで利用可能な属性の一覧は、『Red Hat Directory Server 11 Configuration, Command, and File Reference』 を参照してください。-->
dc: **domain_component**	ドメイン名を指定する属性。サーバーは通常、**dc=hostname,dc=domain,dc=toplevel** の形式で接尾辞または命名コンテキストを持つように初期設定時に設定されます。たとえば、**dc=ldap,dc=example,dc=com** です。ドメインエントリーは、**dc: ldap** のように左端の ***dc*** の値を使用する必要があります。接尾辞が **dc=example,dc=com** の場合、***dc*** の値は **dc: example** になります。サーバーが接尾辞を使用するよう設定されていない場合は、**dn: dc=com** のエントリーを作成しないでください。
list_of_attributes	エントリーに保持する任意の属性のリストを指定します。このオブジェクトクラスで利用可能な属性の一覧は、『Red Hat Directory Server 11 Configuration, Command, and File Reference』 を参照してください。

B.4.2. 組織単位エントリーの指定

組織ユニットエントリーは、ディレクトリーツリー内のメジャーブランチポイントまたはサブディレクトリーを表すために使用されます。これらのサブツリーは、企業内の主要な、適度に静的なエンティティーに対応しています。例えば、ユーザーを含むサブツリーや、グループを含むサブツリーなどです。

エントリーに含まれる組織単位属性は、マーケティングやエンジニアリングなど、企業内の主要な組織を表す場合もあります。ただし、このスタイルは推奨されません。Red Hat はフラットなディレクトリーツリーの使用を強く推奨しています。

通常は、ディレクトリーツリー内に、複数の組織単位またはブランチがあります。

組織単位エントリーを定義する LDIF は、以下のように表示される必要があります。

```
dn: distinguished_name
objectClass: top
objectClass: organizationalUnit
ou: organizational_unit_name
 list_of_optional_attributes
...
```

以下は、LDIF 形式の組織単位エントリーの例です。

■

```
dn: ou=people,dc=example,dc=com
objectclass: top
objectclass: organizationalUnit
ou: people
description: Fictional example organizational unit
```

表B.3「組織単位エントリーの LDIF 要素」 は、LDIF 形式の組織単位エントリーの各要素を定義します。

表B.3 組織単位エントリーの LDIF 要素

LDIF 要素	説明
dn: **distinguished_name**	エントリーの識別名を指定します。DN が必要です。DN にコンマがある場合、コンマはバックスラッシュ (\) でエスケープする必要があります (例: **dn: ou=people,dc=example,dc=com**)。
objectClass: top	**必須。top** オブジェクトクラスを指定します。
objectClass: organizationalUnit	**organizationalUnit** オブジェクトクラスを指定します。この行は、エントリーを **organizational unit** として定義します。このオブジェクトクラスで利用可能な属性の一覧は、『Red Hat Directory Server 11 Configuration, Command, and File Reference』 を参照してください。
ou: **organizational_unit_name**	組織単位の名前を指定する属性。
list_of_attributes	エントリーに保持する任意の属性のリストを指定します。このオブジェクトクラスで利用可能な属性の一覧は、『Red Hat Directory Server 11 Configuration, Command, and File Reference』 を参照してください。

B.4.3. 組織の個人エントリーの指定

ディレクトリー内のエントリーの大半は、組織の人を表します。

LDIF では、組織担当者の定義は以下のようになります。

```
dn: distinguished_name
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: common_name
sn: surname
 list_of_optional_attributes
```

以下は、LDIF 形式の組織単位エントリーの例です。

```
dn: uid=bjensen,ou=people,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Babs Jensen
sn: Jensen
givenname: Babs
uid: bjensen
ou: people
description: Fictional example person
telephoneNumber: 555-5557
userPassword: {SSHA}dkfljlk34r2kljdsfk9
```

表B.4「Person エントリーの LDIF 要素」 は、LDIF 担当者エントリーの各要素を定義します。

表B.4 Person エントリーの LDIF 要素

LDIF 要素	説明
dn: **distinguished_name**	**必須。**エントリーの識別名を指定します。たとえば、**dn: uid=bjensen,ou=people,dc=example,dc=com** です。DN にコンマがある場合は、コンマをバックスラッシュ (\) でエスケープする必要があります。
objectClass: top	**必須。top** オブジェクトクラスを指定します。
objectClass: person	**person** オブジェクトクラスを指定します。多くの LDAP クライアントは、ユーザーや組織のユーザーの検索操作時にこれを必要とするため、このオブジェクトクラス仕様を含める必要があります。
objectClass: organizationalPerson	**organizationalPerson** オブジェクトクラスを指定します。一部の LDAP クライアントは、組織ユーザーの検索操作時に必要であるため、このオブジェクトクラス仕様を含める必要があります。
objectClass: inetOrgPerson	**inetOrgPerson** オブジェクトクラスを指定します。このオブジェクトクラスには属性の範囲が広がるため、このオブジェクトクラスには組織の人エントリーの作成には、**inetOrgPerson** オブジェクトクラスが推奨されます。***uid*** 属性はこのオブジェクトクラスで必要で、このオブジェクトクラスが含まれるエントリーは ***uid*** 属性の値に基づいて名前が付けられます。このオブジェクトクラスで利用可能な属性の一覧は、『Red Hat Directory Server 11 Configuration, Command, and File Reference』 を参照してください。

LDIF 要素	説明
cn: **common_name**	担当者の一般的な名前を指定します。これは、担当者が一般的に使用するフルネームです。たとえば、**cn: Bill Anderson** です。少なくとも 1 つの共通名が必要です。
sn: **姓**	ユーザーの姓を指定します。たとえば、**sn: Anderson** です。姓が必要です。
list_of_attributes	エントリーに保持する任意の属性のリストを指定します。このオブジェクトクラスで利用可能な属性の一覧は、『Red Hat Directory Server 11 Configuration, Command, and File Reference』 を参照してください。

B.5. LDIF を使用したディレクトリーの定義

ディレクトリー全体の内容は、LDIF を使用して定義できます。LDIF の使用は、ディレクトリーに追加するエントリーが多数ある場合に、ディレクトリー作成の効率的な方法です。

LDIF を使用してディレクトリーを作成するには、以下を実行します。

1. LDIF 形式に追加するエントリーを含む ASCII ファイルを作成します。

   各エントリーは、空の行で次のエントリーと区切られていることを確認してください。エントリーの間に 1 行のみを使用し、ファイルの最初の行が空白ではないことを確認します。そうでない場合は、**ldapmodify** ユーティリティーが終了します。詳細は、「LDIF を使用したディレクトリーエントリーの指定」 を参照してください。

2. 各ファイルは、データベースの一番上のエントリー (ルート) から始めます。

   root エントリーは、データベースに含まれる接尾辞または部分接尾辞を表す必要があります。たとえば、データベースに **dc=example,dc=com** の接尾辞がある場合、ディレクトリーの最初のエントリーは **dn: dc=example,dc=com** でなければなりません。

   接尾辞の詳細は、『Red Hat Directory Server の設定、コマンド、およびファイルリファレンス』 に記載されている Suffix パラメーターを参照してください。

3. LDIF ファイルのブランチポイントを表すエントリーが、そのブランチで作成するエントリーの前に配置されていることを確認します。

   たとえば、エントリーをユーザーやグループのサブツリーに置くには、それらのサブツリー内にエントリーを作成する前に、そのサブツリーの分岐点を作成します。

   注記

   LDIF ファイルは順番に読み込まれるため、親エントリーが子エントリーの前にある必要があります。

4. 以下の方法のいずれかを使用して、LDIF ファイルからディレクトリーを作成します。

- **Web コンソールでデータベースの初期化**インポートするデータベースの規模が小さい (10,000 エントリーより少ない) 場合は、この方法を使用します。「Web コンソールを使用したデータのインポート」を参照してください。

警告

このメソッドは破壊的で、接尾辞の既存のデータを削除します。

- **ldif2db または ldif2db.pl コマンドラインユーティリティー。**インポートするデータベースの規模が大きい (10,000 エントリーより多い) 場合は、この方法を使用します。「サーバーがオフライン時のデータのインポート」を参照してください。
  - **ldif2db** サーバーが実行している場合のみ使用できます。
  - **ldif2db.pl** サーバーが実行されている場合のみ使用できます。

警告

このメソッドは破壊的で、接尾辞の既存のデータを削除します。

- **ldapmodify コマンドラインユーティリティーに -a パラメーターをつけたものです。**この方法は、既存のデータベースに新しいサブツリーを追加する場合や、削除してはいけない既存のデータが接尾辞にある場合に使用します。LDIF ファイルからディレクトリーを作成する他の方法とは異なり、**ldapmodify** を使用してサブツリーを追加する前に Directory Server を実行する必要があります。「エントリーの追加」を参照してください。

例B.1 LDIF ファイルの例

この LDIF ファイルには、1 つのドメイン、2 つの組織単位、および 3 つの組織人のエントリーが含まれます。

```
dn: dc=example,dc=com
objectclass: top
objectclass: domain
dc: example
description: Fictional example domain

dn: ou=People,dc=example,dc=com
objectclass: top
objectclass: organizationalUnit
ou: People
description: Fictional example organizational unit
tel: 555-5559

dn: cn=June Rossi,ou=People,dc=example,dc=com
objectClass: top
```

```
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: June Rossi
sn: Rossi
givenName: June
mail: rossi@example.com
userPassword: {sha}KDIE3AL9DK
ou: Accounting
ou: people
telephoneNumber: 2616
roomNumber: 220

dn: cn=Marc Chambers,ou=People,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Marc Chambers
sn: Chambers
givenname: Marc
mail: chambers@example.com
userPassword: {sha}jdl2alem87dlacz1
telephoneNumber: 2652
ou: Manufacturing
ou: People
roomNumber: 167

dn: cn=Robert Wong,ou=People,example.com Corp,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
cn: Robert Wong
cn: Bob Wong
sn: Wong
givenname: Robert
givenname: Bob
mail: bwong@example.com
userPassword: {sha}nn2msx761
telephoneNumber: 2881
roomNumber: 211
ou: Manufacturing
ou: people

dn: ou=Groups,dc=example,dc=com
objectclass: top
objectclass: organizationalUnit
ou: groups
description: Fictional example organizational unit
```

B.6. 複数の言語での情報の保存

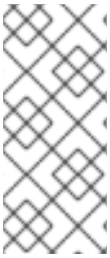
ディレクトリーに単一の言語が含まれる場合は、ディレクトリーに新しいエントリーを追加するための

特別な作業を行う必要はありません。ただし、組織が多国籍である場合は、異なるロケールのユーザーが自分の言語でディレクトリー情報を閲覧できるように、情報を複数の言語で保存する必要があるかもしれません。

ディレクトリー内の情報が複数の言語で表される場合、サーバーは言語タグを属性値に関連付けます。新しいエントリーが追加されると、RDN (相対識別名、命名属性) で使用される属性値は、言語コードなしで指定する必要があります。

複数の言語を 1 つの属性に保存できます。この場合、属性タイプは同じですが、各値には異なる言語コードがあります。

Directory Server がサポートする言語とその関連する言語タグのリストは、「サポート対象のロケール」を参照してください。



注記

言語タグは、文字列がディレクトリー内にどのように保存されるかには影響しません。すべてのオブジェクトクラスおよび属性文字列は UTF-8 を使用して保存されます。ユーザーは、LDIF で使用されるデータを UTF-8 に変換します。ほとんどのオペレーティングシステムが提供する **iconv** または **uconv** コマンドを使用して、ネイティブ文字セットからデータを UTF-8 に変換できます。

たとえば、アメリカとフランスにオフィスを持つ Example Corporation は、社員が母国語でディレクトリー情報を閲覧できるようにしたいと考えています。ディレクトリーエントリーを追加すると、ディレクトリー管理者は英語とフランス語の両方で属性値を指定します。新規従業員 Babs Jensen のディレクトリーエントリーを追加する際に、管理者は以下を行います。

1. 管理者が、フランスの所在地住所値で **street.txt** ファイルを作成します。

```
1 rue de l'Université
```

2. ファイルのコンテンツは UTF-8 に変換されます。

```
# iconv -t UTF-8 -o output.txt street.txt
```

3. 以下の LDIF エントリーは、***streetAddress;lang-fr*** の street アドレス値の UTF-8 値を使用して作成されます。

```
dn: uid=bjensen,ou=people,dc=example,dc=com
objectclass: top
objectclass: person
objectclass: organizationalPerson
name: Babs Jensen
cn: Babs Jensen
sn: Jensen
uid: bjensen
streetAddress: 1 University Street
streetAddress;lang-en: 1 University Street
streetAddress;lang-fr:: AasljdoaAJASI023909jaASJaonasd0ADS
preferredLanguage: fr
```

属性名とサブタイプの後にコロンは、値がバイナリーの base-64 でエンコードされたことを示しています。

推奨言語が英語に設定されている LDAP クライアントを使用して、このディレクトリーエントリーにアクセスすると、**1 University Street** アドレスが表示されます。設定言語がフランス語に設定された LDAP クライアントを持つディレクトリーにアクセスすると、アドレス **1 rue de l'Université** が表示されます。

付録C LDAP URL

LDAP URL は、サイト URL が特定の Web サイトまたは Web ページを特定するのと同様に、Red Hat Directory Server インスタンスを特定します。Directory Server インスタンスの LDAP URL が使用される場合は、以下の 3 つの一般的な時間になります。

- LDAP URL は、Web ベースのクライアントを使用して Directory Server にアクセスする際に特定の Directory Server インスタンスを特定するために使用されます。
- LDAP URL は Directory Server の参照を設定するのに使用されます。
- LDAP URL はアクセス制御の指示を設定するのに使用されます。

注記

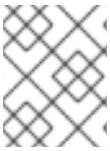
LDAP URL 形式は RFC 4516 で説明されています。http://www.ietf.org/rfc/rfc4516.txt を参照してください。

C.1. LDAP URL のコンポーネント

```
ldap[s]://hostname:port/base_dn?attributes?scope?filter
```

ホスト名の代わりに IPv4 アドレスまたは IPv6 アドレスを使用することもできます。

ldap:// プロトコルは、セキュアでない接続で LDAP サーバーへの接続に使用されます。**ldaps://** プロトコルは、TLS 接続を介して LDAP サーバーに接続するために使用されます。表C.1「LDAP URL コンポーネント」 は、LDAP URL のコンポーネントを一覧表示します。



注記

LDAP URL 形式は RFC 4516 で説明されています。http://www.ietf.org/rfc/rfc4516.txt を参照してください。

表C.1 LDAP URL コンポーネント

コンポーネント	説明
host name	LDAP サーバーの名前、IPv4、または IPv6 アドレス。たとえば、**ldap.example.com**、**192.0.2.90**、または [2001:db8::1] になります。
port	LDAP サーバーのポート番号 (例: **696**)。ポートが指定されていない場合は、標準の LDAP ポート (**389**) または LDAPS ポート (**636**) が使用されます。
base_dn	ディレクトリー内のエントリーの識別名 (DN)。この DN は、検索の開始点であるエントリーを特定します。ベース DN が指定されていない場合、検索はディレクトリーツリーのルートで始まります。
attributes	返される属性。複数の属性を指定するには、コンマを使用して属性を区切ります (例: **cn,mail,telephoneNumber**)。URL に属性が指定されていない場合は、すべての属性が返されます。

コンポーネント	説明
scope	検索の範囲で、以下のいずれかの値になります。 **base** は、URL に指定された識別名 (base_dn) に関する情報のみを取得します。 **one** は、URL に指定された識別名 (base_dn) より 1 レベル下のエントリーに関する情報を取得します。ベースエントリーはこのスコープに含まれません。 **sub** は、URL に指定された識別名 (base_dn) より下のすべてのレベルのエントリーに関する情報を取得します。ベースエントリーはこのスコープに含まれます。 スコープが指定されていない場合、サーバーは **base** 検索を実行します。
filter	指定された検索範囲内のエントリーに適用する検索フィルター。フィルターを指定しないと、サーバーはフィルター **(objectClass=*)** を使用します。

属性、スコープ、およびフィルターコンポーネントは URL の位置によって識別されます。属性が指定されていない場合でも、そのフィールドを区切るために疑問符を含める必要があります。

たとえば、**(sn=Jensen)** を一致させるエントリーの属性をすべて返す **dc=example,dc=com** から開始するサブツリー検索を指定するには、以下の LDAP URL を使用します。

```
ldap://ldap.example.com/dc=example,dc=com??sub?(sn=Jensen)
```

2 つの連続したクエスチョンマーク (**??**) は、属性が指定されていないことを示します。URL には特定の属性が指定されていないため、検索ではすべての属性が返されます。

C.2. 不安全文字のエスケープ

URL の **不安全** 文字はすべてエスケープするか、特殊文字に置き換える必要があります。

たとえば、空白文字は、URL 内で **%20** として表す必要がある不安全な文字です。そのため、識別名 **o=example.com corporation** は **o=example.com%20corporation** としてエンコードされる必要があります。

以下の表は、URL 内で不安全とみなされる文字をリスト表示し、不安全な文字の代わりに使用する関連エスケープ文字を提供します。

不安全な文字	エスケープ文字
space	%20
<	%3c
>	%3e
"	%22

不安全な文字	エスケープ文字
#	%23
%	%25
{	%7b
}	%7d
|	%7c
\	%5c
^	%5e
~	%7e
[	%5b
]	%5d
`	%60

C.3. LDAP URL の例

注記

LDAP URL 形式は RFC 4516 で説明されています。http://www.ietf.org/rfc/rfc4516.txt を参照してください。

例 1

以下の LDAP URL は、識別名 **dc=example,dc=com** のエントリーのベース検索を指定します。

```
ldap://ldap.example.com/dc=example,dc=com
```

- ポート番号が指定されていないため、標準の LDAP ポート番号 (**389**) が使用されます。
- 属性が指定されていないため、検索はすべての属性を返します。
- 検索条件が指定されていないため、検索はベースエントリー **dc=example,dc=com** に制限されます。
- フィルターが指定されていないため、ディレクトリーはデフォルトのフィルター (**objectclass=***) を使用します。

例 2

以下の LDAP URL は、DN **dc=example,dc=com** を使用してエントリーの ***postalAddress*** 属性を取得します。

```
ldap://ldap.example.com/dc=example,dc=com?postalAddress
```

- 検索条件が指定されていないため、検索はベースエントリー **dc=example,dc=com** に制限されます。
- フィルターが指定されていないため、ディレクトリーはデフォルトのフィルター (**objectclass=***) を使用します。

例 3

以下の LDAP URL は、Barbara Jensen のエントリーの ***cn*** 属性、***mail*** 属性、および ***telephoneNumber*** 属性を取得します。

```
ldap://ldap.example.com/cn=Barbara%20Jensen,dc=example,dc=com?cn,mail,telephoneNumber
```

- 検索範囲が指定されていないため、検索はベースエントリー **cn=Barbara Jensen,dc=example,dc=com** に制限されます。
- フィルターが指定されていないため、ディレクトリーはデフォルトのフィルター **(objectclass=*)** を使用します。

例 4

以下の LDAP URL は、姓 **Jensen** を持ち、**dc=example,dc=com** 下のレベルにあるエントリーの検索を指定します。

```
ldap://ldap.example.com/dc=example,dc=com??sub?(sn=Jensen)
```

- 属性が指定されていないため、検索はすべての属性を返します。
- 検索範囲は **sub** であるため、検索にはベースエントリー **dc=example,dc=com** と、ベースエントリー下の全レベルでエントリーが含まれます。

例 5

以下の LDAP URL は、**dc=example,dc=com** 下 1 レベル下にあるすべてのエントリーレベルに対するオブジェクトクラスの検索を指定します。

```
ldap://ldap.example.com/dc=example,dc=com?objectClass?one
```

- 検索範囲は **one** であるため、検索にはベースエントリー **dc=example,dc=com** の 1 レベル下に全エントリーが含まれます。検索範囲にはベースエントリーが含まれません。
- フィルターが指定されていないため、ディレクトリーはデフォルトのフィルター (**objectclass=***) を使用します。

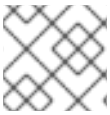
注記

LDAP URL の構文には、認証情報またはパスワードを指定する方法は含まれません。LDAP URL をサポートする LDAP クライアントが認証メカニズムを提供する場合を除き、LDAP URL から開始された検索要求は認証されます。

付録D 国際化

Red Hat Directory Server を使用すると、ユーザーはさまざまな言語でエントリーとそれに関連する属性を保存、管理、および検索できます。国際化されたディレクトリーは、従業員やビジネスパートナーが理解できる言語で必要な情報にすぐにアクセスできる、貴重な企業リソースになり得ます。

Directory Server は、ディレクトリーデータが UTF-8 に保存されるため、デフォルトですべての国際文字セットに対応します。さらに、Directory Server では、検索操作で言語の詳細をもとに、指定したマッチングルールと結合順序を使用できます。



注記

ASCII 文字は属性およびオブジェクトクラス名に必要です。

D.1. ローカルの概要

Directory Server は、**ロケール** を使用して複数の言語をサポートします。ロケールとは、特定の地域、文化、慣習のユーザーがどのようにデータを表示するかについての言語固有の情報を示すもので、ある言語のデータをどのように解釈するか、データをどのようにソートするか、または **照合** するかなどが含まれます。

さらに、ロケール情報は、特定の言語を表すために使用されるコードページを示します。コードページは、オペレーティングシステムがキーボードキーを文字フォント画面表示に関連付けるために使用する内部テーブルです。

具体的には、ロケールは次の 4 つのことを定義します。

- **照合順序。** 照合順序は、特定の言語の文字をどのようにソートするかについて、言語および文化に固有の情報を提供します。アルファベットの文字の順序、アクセントのある文字とアクセントのない文字を比較する方法、文字列を比較するときに無視できる文字があるかどうかなどを識別します。照合順序では、言語が読み取られる方向 (左から右、右から左、または上と下) など、言語に関する文化固有の情報も考慮されます。

- **文字タイプ。** 文字タイプは、アルファベット文字を数字またはその他の文字と区別します。たとえば、ある言語では、パイプ (|) 文字は句読点と見なされますが、他の言語ではアルファベットと見なされます。さらに、大文字への大文字のマッピングを定義します。

- **通貨形式。** 通貨形式は、特定の地域で使用される通貨記号、その記号がその値の前後にあるかどうか、および通貨単位がどのように表されるかを指定します。

- **時刻/日付の形式。** 時刻と日付の形式は、その地域の時刻と日付の慣習的な形式を示しています。日時形式は、日付が **mm/dd/yy** (月、日、年) または **dd/mm/yy** (日、月、年) の形式で、特定の言語の曜日と日付を指定します。たとえば、1996 年 1 月 10 日の日付は、チェコ語では **10. leden 1996**、およびフランス語では **10 janvier 1996** と表示されます。

ロケールは、機械的な言語の違いに加えて、文化的、慣習的、地域的な違いを説明するため、ディレクトリーデータは、ユーザーが理解できる特定の言語に翻訳され、特定の地域のユーザーが期待する方法で表示できます。

D.2. サポート対象のロケール

検索操作など、ロケールを指定する必要があるディレクトリー操作を実行する際に、言語タグや照合順序オブジェクト識別子 (OID) を使用します。

言語タグ は、ISO 標準 639 で定義されているように言語を識別する 2 文字の小文字の言語コードで始

まる文字列です。言語で地域の違いを区別する必要がある場合、言語タグに ISO 標準 3166 に定義されている国コードの 2 文字の文字列が含まれる可能性があります。言語コードおよび国コードは、ハイフンで区切られています。たとえば、イギリス英語のロケールの特定に使用される言語タグは **en-GB** です。

オブジェクト識別子 (OID) は、属性やオブジェクトクラスなどのオブジェクトを一意に識別するために使用される 10 進数の数値です。国際化されたディレクトリーを検索またはインデックス化する OID は、Directory Server でサポートされる特定の照合順序を識別します。たとえば、OID **2.16.840.1.113730.3.3.2.17.1** はフィンランドの照合順序を識別します。

ディレクトリーで国際検索を実行する場合は、言語タグまたは OID を使用して、使用する照合順序を特定します。ただし、国際的なインデックスを設定する場合は OID を使用する必要があります。インデックスの詳細は、13章 *インデックスの管理*を参照してください。

Directory Server でサポートされる言語タグと OID の一覧は、**/etc/dirsrv/config/slapd-collations.conf** ファイルを参照してください。

D.3. サポートされる言語サブタイプ

言語のサブタイプはクライアントで使用することで、検索する特定の値を判断することができます。言語サブタイプの使用に関する詳細は、「国際化されたディレクトリーにおけるエントリーの更新」を参照してください。表D.1「サポートされる言語サブタイプ」 は、Directory Server でサポートされる言語のサブタイプ一覧を表示します。

表D.1 サポートされる言語サブタイプ

言語タグ	言語
af	アフリカーンス語
be	ベラルーシ語
bg	ブルガリア語
ca	カタロニア語
cs	チェコ語
da	デンマーク語
de	ドイツ語
el	ギリシャ語
en	英語
es	スペイン語
eu	バスク語
fi	フィンランド語

言語タグ	言語
fo	フェロー語
fr	フランス語
ga	アイルランド語
gl	ガリシア語
hr	クロアチア語
hu	ハンガリー語
id	インドネシア語
is	アイスランド語
it	イタリア語
ja	日本語
ko	韓国語
nl	オランダ語
no	ノルウェー語
pl	ポーランド語
pt	ポルトガル語
ro	ルーマニア語
ru	ロシア語
sk	スクロバキア語
sl	スロベニア語
sq	アルバニア語
sr	セルビア語
sv	スウェーデン語
tr	トルコ語

言語タグ	言語
uk	ウクライナ語
zh	中国語

D.4. 国際化されたディレクトリーの検索

検索操作の実行時に、Directory Server は、サーバーがサポートする照合順序を持つ任意の言語に基づいて結果をソートすることができます。ディレクトリーでサポートされている照合順序のリストは、「サポート対象のロケール」を参照してください。

注記

国際化された検索を実行するには、LDAPv3 検索が必要です。したがって、**ldapsearch** の呼び出しには LDAPv2 オプションを設定しないでください。

本セクションでは、国際的な属性値を返すためのマッチングルールフィルターの使用に焦点を当てています。一般的な **ldapsearch** 構文の詳細は、「LDAP 検索フィルター」を参照してください。

- 「マッチングルールの形式」
- 「サポートされる検索タイプ」
- 「国際検索の例」

D.4.1. マッチングルールの形式

国際化された検索のマッチングルールのフィルターは、いくつかの方法で表すことができ、どちらを使用するかは好みの問題です。

- 検索のベースとなるロケールの照合順序の OID として。
- 検索のベースとなる照合順序に関連付けられた言語タグとして。
- 照合順序の OID および関係演算子を表す接尾辞として。
- 照合順序に関連付けられた言語タグと、関係演算子を表す接尾辞として。

これらの各オプションの構文は、以下のセクションで説明されています。

- 「マッチングルールでの OID の使用」
- 「マッチングルールに言語タグの使用」
- 「マッチングルールでの OID および Suffix の使用」
- 「一致するルールに対する言語タグと接尾辞の使用」

D.4.1.1. マッチングルールでの OID の使用

Directory Server でサポートされる各ロケールには、関連付けられた照合順序 OID があります。Directory Server がサポートする OID の一覧は、**/etc/dirsrv/config/slapd-collations.conf** ファイルを参照してください。

照合順序 OID は、次のように一致規則フィルターの一致規則部分で使用できます。

```
attr:OID:=(relational_operator value)
```

リレーショナル演算子は文字列の値の部分に含まれ、値はシングルスペースで区切られます。たとえば、スウェーデン語の照合順序で **N4709** 時またはそれ以降のすべての ***departmentNumber*** 属性を検索するには、次のフィルターを使用します。

```
departmentNumber:2.16.840.1.113730.3.3.2.46.1:=>= N4709
```

D.4.1.2. マッチングルールに言語タグの使用

Directory Server でサポートされる各ロケールには、関連する言語タグがあります。Directory Server でサポートされる言語タグの一覧は、**/etc/dirsrv/config/slapd-collations.conf** ファイルを参照してください。

言語タグは、次のようにマッチングルールフィルターのマッチングルール部分で使用できます。

```
attr:language-tag:=(relational_operator value)
```

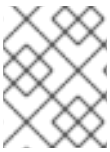
リレーショナル演算子は文字列の値の部分に含まれ、値はシングルスペースで区切られます。たとえば、スペイン語の照合順序を使用して、**estudiante** の値を持つすべての説明属性をディレクトリーで検索するには、次のフィルターを使用します。

```
cn:es:== estudiante
```

D.4.1.3. マッチングルールでの OID および Suffix の使用

関係演算子と値のペアを使用する代わりに、フィルターのマッチングルール部分の OID に特定の演算子を表す接尾辞を追加します。以下のように OID と接尾辞を統合します。

```
attr: OID+suffix:=value
```



注記

この構文は、**mozldap** ユーティリティーでのみサポートされ、**ldapsearch** などの OpenLDAP ユーティリティーではサポートされません。

たとえば、ドイツ語の照合順序で **softwareprodukte** 値を持つ ***businessCategory*** 属性を選択するには、次のフィルターを使用します。

```
businessCategory:2.16.840.1.113730.3.3.2.7.1.3:=softwareprodukte
```

上記の例の **.3** は等価接尾辞です。

Directory Server がサポートする OID の一覧は、**/etc/dirsrv/config/slapd-collations.conf** ファイルを参照してください。リレーショナル演算子とそれらの同等の接尾辞の一覧は、表D.2「検索タイプ、演算子、および接尾辞」 を参照してください。

D.4.1.4. 一致するルールに対する言語タグと接尾辞の使用

関係演算子と値のペアを使用する代わりに、フィルターのマッチングルール部分の言語タグに特定の演算子を表す接尾辞を追加します。以下のように、言語タグと接尾辞を組み合わせます。

```
attr: language-tag+suffix:=value
```

注記

この構文は、**mozldap** ユーティリティーでのみサポートされ、**ldapsearch** などの OpenLDAP ユーティリティーではサポートされません。

たとえば、フランス語の照合順序で **La Salle** の前後に来るすべての名字を検索するには、次のようなフィルターを使用します。

```
sn:fr.4:=La Salle
```

Directory Server でサポートされる言語タグの一覧は、**/etc/dirsrv/config/slapd-collations.conf** ファイルを参照してください。リレーショナル演算子とそれらの同等の接尾辞の一覧は、表D.2「検索タイプ、演算子、および接尾辞」 を参照してください。

D.4.2. サポートされる検索タイプ

Directory Server は、以下のタイプの国際検索をサポートします。

- 等号 (=)
- 部分文字列 (*)
- より大きい (>)
- 以上 (>=)
- より小さい (<)
- 以下 (<=)

近似検索、音声検索、存在検索は英語のみ対応しています。

通常の **ldapsearch** 検索操作と同様に、国際検索は演算子を使用して検索のタイプを定義します。ただし、国際的な検索を行う場合は、検索文字列の値の部分に標準的な演算子 (=、>=、>、<、<=) を使用するか、フィルターのマッチングルールの部分に接尾辞 (ディレクトリー接尾辞と混同しないように) と呼ばれる特殊なタイプの演算子を使用してください。表D.2「検索タイプ、演算子、および接尾辞」 は、各タイプの検索、演算子、および同等の接尾辞をまとめたものです。

表D.2 検索タイプ、演算子、および接尾辞

検索タイプ	演算子	接尾辞
より小さい	<	.1
より小さいか等しい	<=	.2

検索タイプ	演算子	接尾辞
等号	=	.3
より大きいか等しい	>=	.4
より大きい	>	.5
部分文字列	*	.6

D.4.3. 国際検索の例

以下のセクションでは、ディレクトリーデータで国際検索を実行する方法の例を紹介します。それぞれの例では、マッチングルールのフィルター形式がすべて示されているため、形式に慣れて最適なものを選択することができます。

D.4.3.1. less-than の例

小なり演算子 (<) または接尾辞 (**.1**) を使用してロケール固有の検索を実行すると、特定の照合順で指定の属性の前に渡されるすべての属性値が検索されます。

たとえば、スペイン語の照合順序で苗字 **Marquez** の前に来るすべての苗字を検索するには、以下のマッチングルールフィルターのいずれかが有効です。

```
sn:2.16.840.1.113730.3.3.2.15.1:=< Marquez
...
sn:es:=< Marquez
...
sn:2.16.840.1.113730.3.3.2.15.1.1:=Marquez
...
sn:es.1:=Marquez
```

D.4.3.2. less-Than または Equal-to の例

小なりまたは等号演算子 (<=) または接尾辞 (**.2**) を使用してロケール固有の検索を実行すると、特定の照合順で指定の属性またはその前に渡されるすべての属性値が検索されます。

たとえば、ハンガリー語の照合順序で部屋番号 **CZ422** に、またはその前にあるすべての部屋番号を検索するには、次のマッチングルールのフィルターのいずれかが機能します。

```
roomNumber:2.16.840.1.113730.3.3.2.23.1:=<= CZ422
...
roomNumber:hu:=<= CZ422
...
roomNumber:2.16.840.1.113730.3.3.2.23.1.2:=CZ422
...
roomNumber:hu.2:=CZ422
```

D.4.3.3. 等価性の例

等号演算子 (=) や接尾辞 (**.3**) を使用してローカル固有の検索を行うと、特定の照合順序で与えられた属性に一致するすべての属性値が検索されます。

たとえば、ドイツ語の照合順序で値 **softwareprodukte** を持つすべての ***businessCategory*** 属性を検索するには、次のマッチングルールフィルターのいずれかが機能します。

```
businessCategory:2.16.840.1.113730.3.3.2.7.1:==softwareprodukte
...
businessCategory:de:== softwareprodukte
...
businessCategory:2.16.840.1.113730.3.3.2.7.1.3:=softwareprodukte
...
businessCategory:de.3:=softwareprodukte
```

D.4.3.4. より大きいか等しいの例

大なりまたは等号演算子 (**>=**) または接尾辞 (**.4**) を使用してロケール固有の検索を実行すると、特定の照合順で指定の属性またはその後に渡されるすべての属性値が検索されます。

たとえば、フランス語の照合順序で **Québec** に、またはその後に位置するすべての地域を検索するには、次のようなマッチングルールフィルターが有効です。

```
locality:2.16.840.1.113730.3.3.2.18.1:=>= Québec
...
locality:fr:=>= Québec
...
locality:2.16.840.1.113730.3.3.2.18.1.4:=Québec
...
locality:fr.4:=Québec
```

D.4.3.5. より大きいの例

大なり演算子 (>) または接尾辞 (**.5**) を使用してロケール固有の検索を実行すると、特定の照合順で指定の属性またはその前に渡されるすべての属性値が検索されます。

たとえば、チェコ語の照合順序でホスト **schranka4** の後に来るすべてのメールホストを検索するには、次のマッチングルールフィルターのいずれかが機能します。

```
mailHost:2.16.840.1.113730.3.3.2.5.1:=> schranka4
...
mailHost:cs:=> schranka4
...
mailHost:2.16.840.1.113730.3.3.2.5.1.5:=schranka4
...
mailHost:cs.5:=schranka4
```

D.4.3.6. 部分文字列の例

国際的な部分文字列検索を実行すると、指定した照合順序の指定のパターンに一致する値がすべて検索されます。

たとえば、中国語の照合順序で **ming** で終わるすべてのユーザー ID を検索するには、次のマッチングルールフィルターのいずれかを使用します。

```
uid:2.16.840.1.113730.3.3.2.49.1:=* *ming
...
uid:zh:=* *ming
...
uid:2.16.840.1.113730.3.3.2.49.1.6:=* *ming
..
uid:zh.6:=* *ming
```

modifiersName や ***memberOf*** などの DN 値属性を使用する部分文字列検索フィルターは、フィルターに空白文字が 1 つ以上含まれる場合は、エントリーには常に正しく一致しません。

この問題を回避するには、部分文字列ではなくフィルターで DN 全体を使用するか、フィルターの DN 部分文字列が RDN 境界で始まるようにしてください。つまり、DN の **type=** 部分で始まるようにしてください。たとえば、以下のフィルターは使用しないでください。

```
(memberOf=*Domain Administrators*)
```

ただし、以下のいずれかが正常に動作します。

```
(memberOf=cn=Domain Administrators*)
...
(memberOf=cn=Domain Administrators,ou=Groups,dc=example,dc=com)
```

D.5. マッチングルールのトラブルシューティング

国際的な照合順序のマッチングルールは、一貫した動作をしない場合があります。matching-rule 呼び出しの形式が正しく動作しないため、誤った検索結果を生成します。たとえば、以下のルールは動作しません。

```
# ldapsearch -x -p 389 -D "uid=userID,ou=people,dc=example,dc=com" -W -b "dc=example,dc=com"
"sn:2.16.840.1.113730.3.3.2.7.1:=passin"

ldapsearch -x -p 389 -D "uid=userID,ou=people,dc=example,dc=com" -W -b "dc=example,dc=com"
"sn:de:=passin"
```

ただし、以下に記載のルールは機能します (**passin** 値の前に **.3** が必要)。

```
# ldapsearch -x -p 389 -D "uid=userID,ou=people,dc=example,dc=com" -W -b "dc=example,dc=com"
"sn:2.16.840.1.113730.3.3.2.7.1.3:=passin"

ldapsearch -x -p 389 -D "uid=userID,ou=people,dc=example,dc=com" -W -b "dc=example,dc=com"
"sn:de.3:=passin"
```

付録E 更新履歴

改訂番号は本ガイドに関するものであり、Red Hat Directory Server のバージョン番号ではありません。

改訂 11.5-1 Tue May 10 2022 Marc Muehlfeld
Red Hat Directory Server 11.5 版のガイドをリリース

改訂 11.4-1 Tue Nov 09 2021 Marc Muehlfeld
Red Hat Directory Server 11.4 版のガイドをリリース

改訂 11.3-1 Tue May 11 2021 Marc Muehlfeld
Red Hat Directory Server 11.3 版のガイドをリリース

改訂 11.2-1 Tue Nov 03 2020 Marc Muehlfeld
Red Hat Directory Server 11.2 版のガイドをリリース

改訂 11.1-1 Tue Apr 28 2020 Marc Muehlfeld
Red Hat Directory Server 11.1 版のガイドをリリース

改訂 11.0-1 Tue Nov 05 2019 Marc Muehlfeld
Red Hat Directory Server 11.0 版のガイドをリリース