Red Hat Directory Server 12

アクセス制御の管理

アクセス制御手順を使用したパーミッションの設定

Last Updated: 2024-11-23

Red Hat Directory Server 12 アクセス制御の管理

アクセス制御手順を使用したパーミッションの設定

法律上の通知

概要

Red Hat Directory Server で接尾辞およびエントリーに対して特定のアクションを実行できるユーザーを定義する方法を説明します。当該タスクは、アクセス制御手順 (ACI) によって制御されます。さまざまな ACI タイプ、ACI のユースケース、バインドルール、およびエントリーのアクセス権を確認する方法を説明します。

目次

RED HAT DIRECTORY SERVER に関するフィードバックの提供

Red Hat のドキュメントおよび製品に関するご意見をお待ちしております。ドキュメントの改善点があればお知らせください。改善点を報告する場合は、以下のように行います。

- Jira を通じて Red Hat Directory Server ドキュメントに関するフィードバックを送信する場合 (アカウントが必要):
  1. Red Hat Issue Tracker にアクセスしてください。
  2. **Summary** フィールドにわかりやすいタイトルを入力します。
  3. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも追加してください。
  4. ダイアログの下部にある **Create** をクリックします。
- Jira を通じて Red Hat Directory Server 製品に関するフィードバックを送信する場合 (アカウントが必要):
  1. Red Hat Issue Tracker にアクセスしてください。
  2. **Create Issue** ページで、**Next** をクリックします。
  3. **Summary** フィールドに入力します。
  4. **Component** フィールドでコンポーネントを選択します。
  5. **Description** フィールドに以下の内容を入力します。
     a. 選択したコンポーネントのバージョン番号。
     b. 問題を再現するための手順、または改善のための提案。
  6. **Create** をクリックします。

第1章 アクセス制御手順の管理

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

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

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

重要

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

1.1. ACI 配置

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

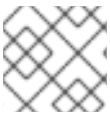
- ACI を含むエントリー (子エントリーがない場合) にのみ適用されます。たとえば、クライアントが **uid=user_name,ou=People,dc=example,dc=com** オブジェクトへのアクセスを必要とし、ACI が **dc=example,dc=com** にのみ設定されており、子エントリーには設定されていない場合は、この ACI のみが適用されます。

  注記

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

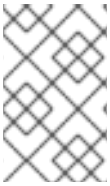
- ACI を含むエントリーと、(子エントリーがある場合は) その下のすべてのエントリーへ。これにより、サーバーが指定のエントリーに対するアクセスパーミッションを評価すると、リクエストされたディレクトリー接尾辞と、エントリー自体の ACI との間のすべてのエントリーについて ACI を検証します。
  たとえば、ACI は **dc=example,dc=com** および **ou=People,dc=example,dc=com** エントリーに設定されます。ACI が設定されていない **uid=user_name,ou=People,dc=example,dc=com** オブジェクト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



注記

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

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



注記

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

1.2. ACI の構造

aci 属性は以下の構文を使用します。

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

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

パーミッションとバインドルールのペアはアクセス制御ルールと呼ばれます。

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

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

1.3. ACI 評価

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

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

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

注記

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

1.4. ACI の制限

アクセス制御手順 (ACI) を設定する場合は、以下の制限が適用されます。

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

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

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

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

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

1.6. ACI の表示、追加、削除、および更新

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

ACI の表示

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

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

ACI の追加

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

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

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

ACI の削除

ACI を削除するには、以下を実行します。

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

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

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

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

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

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

ACI の更新

ACI を更新するには、以下を実行します。

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

1.7. ACI ターゲットの定義

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

ACI では、以下の強調表示された部分がターゲットルールになります。

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

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

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

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

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

1.7.1. ターゲットルールの構文

ターゲットルールの一般的な構文は、以下のとおりです。

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

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

警告

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

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

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

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

1.7.2. ディレクトリーエントリーのターゲット

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

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

注記

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

例1.1 target キーワードの使用

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

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

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

target キーワードでのワイルドカードの使用

* ワイルドカード文字ターゲットに複数のエントリーを使用できます。

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

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

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

例1.2 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

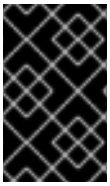
Directory Server は、DN の接尾辞部分でのワイルドカードをサポートしません。たとえば、ディレクトリーの接尾辞が **dc=example,dc=com** の場合は、**(target = "ldap:///dc=*.com")** などのように、この接尾辞でワイルドカード付きのターゲットは使用できません。

1.7.3. ターゲット属性

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

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

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



重要

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

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

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

式に設定された属性はスキーマに定義する必要があります。

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

例1.3 targetattr キーワードの使用

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

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

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

targetattr キーワードでのワイルドカードの使用

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

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

警告

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

1.7.4. LDAP フィルターを使用したエントリーと属性の対象

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

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

フィルター式は、標準の LDAP 検索フィルターです。

例1.4 targetfilter キーワードの使用

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

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

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

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

注記

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

targetfilter キーワードでのワイルドカードの使用

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

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

1.7.5. LDAP フィルターを使用した属性値のターゲット

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

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

- 1 つの属性とフィルターの組み合わせが含まれる操作の場合:

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

- 複数の属性とフィルターの組み合わせのある操作の場合:

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

- 複数の属性とフィルターを組み合わせた 2 つの操作の場合。

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

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

以下では、フィルターを一致させる方法を説明します。

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

例1.5 targattrfilters キーワードの使用

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

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

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

1.7.6. ソースおよび宛先 DN のターゲット

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

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

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

例1.6 target_from および target_to キーワードの使用

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

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

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

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

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

1.8. ターゲットルールの高度な使用方法

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

1.8.1. グループの作成およびメンテナンスへのパーミッションの委譲

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

例1.7 グループの作成およびメンテナンスへのパーミッションの委譲

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

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

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

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

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

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

1.8.2. エントリーと属性の両方をターゲットに設定

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

例1.8 エントリーと属性の両方をターゲットに設定

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

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

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

1.8.3. フィルターに一致するエントリーの個別属性のターゲット設定

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

例1.9 フィルターに一致するエントリーの個別属性のターゲット設定

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

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

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

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

1.8.4. 単一ディレクトリーエントリーのターゲット設定

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

例1.10 単一ディレクトリーエントリーのターゲット設定

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

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

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

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

重要

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

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

1.9. ACI パーミッションの定義

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

ACI では、以下の強調表示された部分はパーミッションルールになります。

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

1.9.1. パーミッションルールの構文

パーミッションルールの一般的な構文は、以下のとおりです。

permission (**rights**)

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

例1.11 パーミッションの定義

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

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

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

1.9.2. パーミッションルールのユーザー権限

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

表1.1 ユーザーの権利

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

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

1.9.3. LDAP 操作に必要な権限

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

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

    - 属性タイプで **compare** パーミッションを付与します。

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

1.10. ACI バインドルールの定義

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

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

ACI では、以下の強調表示された部分はバインドルールになります。

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

1.10.1. バインドルールの構文

バインドルールの一般的な構文は以下のとおりです。

> **keyword comparison_operator "expression"**

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

1.10.2. ユーザーベースのアクセスの定義

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

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

式の DN を以下のように設定します。

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

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

注記

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

userdn キーワードでの DN の使用

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

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

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

例1.12 userdn キーワードでの DN の使用

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

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

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

LDAP フィルターで userdn キーワードの使用

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

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

注記

LDAP フィルターは * ワイルドカードをサポートします。

例1.13 LDAP フィルターで userdn キーワードの使用

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

```
```

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

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

匿名アクセスの付与

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

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

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

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

例1.14 匿名アクセスの付与

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

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

認証済みユーザーへのアクセスの付与

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

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

例1.15 認証済みユーザーへのアクセスの付与

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

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

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

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

ユーザーが空のエントリーにアクセスできるようにする

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

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

例1.16 ユーザーが空のエントリーにアクセスできるようにする

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

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

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

ユーザーの子エントリーへのアクセス設定

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

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

例1.17 ユーザーの子エントリーへのアクセス設定

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

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

1.10.3. グループベースのアクセスの定義

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

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

- member
- uniqueMember
- memberURL
- memberCertificateDescription

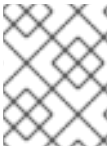
groupdn キーワードでルールをバインドするには、以下の構文を使用します。

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

式の識別名 (DN) を次のように設定します。

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

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



注記

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

groupdn キーワードでの DN の使用

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

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

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

例1.18 groupdn キーワードでの DN の使用

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

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

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

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

LDAP フィルターで groupdn キーワードの使用

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

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

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

注記

LDAP フィルターは * ワイルドカードをサポートします。

例1.19 LDAP フィルターで groupdn キーワードの使用

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

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

1.10.4. 値の一致に基づくアクセスの定義

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

userattr キーワードは、以下の構文を使用します。

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

詳細は、以下を参照してください。

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

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

重要

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

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

USERDN バインドタイプの使用

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

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

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

例1.20 USERDN バインドタイプの使用

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

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

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

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

GROUPDN バインドタイプの使用

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

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

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

例1.21 GROUPDN バインドタイプの使用

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

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

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

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

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

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

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

ROLEDN バインドタイプの使用

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

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

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

例1.22 ROLEDN バインドタイプの使用

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

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

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

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

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

SELFDN バインドタイプの使用

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

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

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

例1.23 SELFDN バインドタイプの使用

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

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

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

LDAPURL バインドタイプの使用

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

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

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

例1.24 LDAPURL バインドタイプの使用

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

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

継承による userattr キーワードの使用

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

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

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

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

以下に例を示します。

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

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

例1.25 継承による userattr キーワードの使用

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

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

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

1.10.5. 特定の IP アドレスまたは範囲からのアクセスの定義

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

ip キーワードでルールをバインドするには、以下の構文を使用します。

```
ip comparison_operator "IP_address_or_range"
```

例1.26 バインドルールでの IPv4 アドレス範囲の使用

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

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

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

例1.27 バインドルールでの IPv6 アドレス範囲の使用

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

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

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

1.10.6. 特定のホストまたはドメインからアクセスの定義

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

警告

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

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

dns キーワードでルールをバインドするには、以下の構文を使用します。

```
dns comparison_operator "host_name_or_domain_name"
```

例1.28 特定のホストからのアクセスの定義

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

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

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

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

例1.29 特定のドメインからアクセスの定義

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

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

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

1.10.7. 接続に一定レベルのセキュリティーの要求

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

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

ssf キーワードでルールをバインドするには、以下の構文を使用します。

```
ssf comparison_operator key_strength
```

以下の比較演算子を使用できます。

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

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

例1.30 接続に一定レベルのセキュリティーの要求

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

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

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

1.10.8. 曜日の特定の日におけるアクセスの定義

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

注記

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

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

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

例1.31 特定の曜日にアクセスの付与

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

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

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

1.10.9. 特定の時刻におけるアクセスの定義

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

注記

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

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

```
timeofday comparison_operator "time"
```

以下の比較演算子を使用できます。

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

重要

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

例1.32 特定の時刻におけるアクセスの定義

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

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

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

1.10.10. 認証方法に基づいたアクセスの定義

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

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

```
authmethod comparison_operator "authentication_method"
```

以下の認証方法を設定できます。

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

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

例1.33 EXTERNAL SASL 認証方法を使用した接続でのみアクセスのみの有効化

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

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

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

1.10.11. ロールに基づくアクセスの定義

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

注記

Red Hat は、ロールの代わりにグループを使用することを推奨します。

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

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

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

例1.34 ロールに基づくアクセスの定義

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

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

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

1.10.12. ブール演算子を使用したバインドルールの組み合わせ

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

バインドルールとブール演算子を組み合わせた構文は以下の通りです。

> **bind_rule_1 boolean_operator bind_rule_2**...

> **例1.35 ブール演算子を使用したバインドルールの組み合わせ**
>
> **cn=Administrators,ou=Groups,dc=example,com** および **cn=Operators,ou=Groups,dc=example,com] group can [command]`read** の両方のグループのメンバーであるユーザーが、**ou=People,dc=example,dc=com** のエントリーを、**search**、**add**、**update**、および **delete** できるように設定するには、次のコマンドを実行します。
>
> ```
> # ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x
>
> dn: ou=People,dc=example,dc=com
> changetype: modify
> add: aci
> aci: (target="ldap:///ou=People,dc=example,dc=com") (version 3.0;
>  acl "Allow members of administrators and operators group to manage users";
>  allow (read, search, add, write, delete)
>  groupdn = "ldap:///cn=Administrators,ou=Groups,dc=example,com" AND
>  groupdn = "ldap:///cn=Operators,ou=Groups,dc=example,com";)
> ```

Directory Server によるブール値演算子の評価方法

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

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

  > (**bind_rule_1**) OR (**bind_rule_2**)

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

  > (**bind_rule_1**) OR ((**bind_rule_2**) AND (**bind_rule_3**))

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

  > (**bind_rule_1**) AND NOT (**bind_rule_2**)

  AND および **OR** 演算子には優先順位がありません。

第2章 マクロアクセス制御命令の使用

マクロアクセス制御命令 (ACI) を使用すると、LDAP エントリーの識別名 (DN) またはその一部を対象とするアクセスを自動化し、ACI の数を減らすことができます。

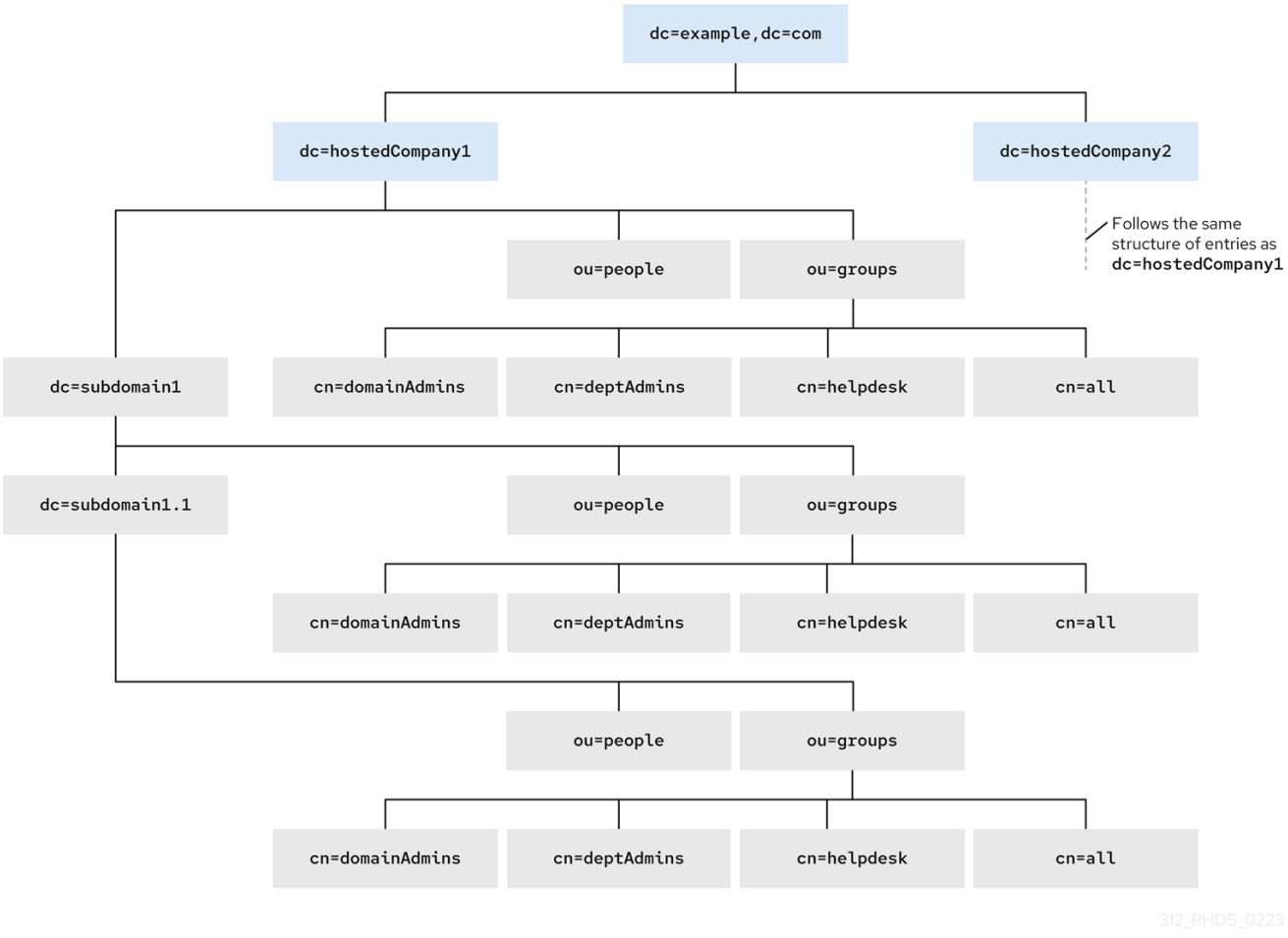
2.1. マクロアクセス制御命令の例

次の図は、サフィックスとして **dc=hostedCompany1,dc=example,dc=com** および **dc=hostedCompany2,dc=example,dc=com** と、サブドメインの繰り返しパターンを持つディレクトリーツリーを示しています。各サブドメインの構造は、**ou=groups**、**ou=people** エントリーと同じです。ディレクトリーツリーは、マクロアクセス制御命令 (ACI) を使用して、ACI の総数を減らします。

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、次の ACI は **dc=hostedCompany1,dc=example,dc=com** ノードにあり、**DomainAdmins** グループに対する読み取りおよび検索権限を、そのツリー内の任意のエントリーに付与します。

```
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";)
```

図2.1 マクロ ACI ディレクトリーツリーの例



以下の ACI は、**groupdn** キーワードの DN の異なる部分を示しています。

- **dc=hostedCompany1,dc=example,dc=com** ノードには、次の ACI が含まれています。

```
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";)
```

- **dc=subdomain1,dc=hostedCompany1,dc=example,dc=com** ノードには、次の ACI が含まれています。

```
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";)
```

- **dc=hostedCompany2,dc=example,dc=com** ノードには、次の ACI が含まれています。

```
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";)
```

- **dc=subdomain1,dc=hostedCompany2,dc=example,dc=com** ノードには、次の ACI が含まれています。

```
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";)
```

マクロを使用して、繰り返しパターンの複数の ACI を置き換えます。たとえば、上記の ACI を 1 つに減らすには、次のマクロを使用します。

```
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";)
```

2.2. マクロアクセス制御命令の構文

マクロアクセス制御命令 (ACI) には、DN または DN の一部を置き換える次のタイプの式が含まれます。

- **($dn)**,
- **[$dn]**,
- **($attr.attrName)**。この場合の **attrName** は、ターゲットエントリーの一部である属性を表します。

ACI キーワードは、ACI の対象であるバインド認証情報を提供します。サブジェクトは、ACI が適用される場所を決定します。

表2.1 ACI キーワードのマクロ

マクロ	ACI キーワード	説明
($dn)	**target, targetfilter, userdn, roledn, groupdn, userattr**	サブジェクトのマッチングと直接置換。**target** または **targetfilter** に一致し、一致した値を **userdn**、**groupdn**、または **userattr** に置き換えます。
[$dn]	**targetfilter, userdn, roledn, groupdn, userattr**	サブジェクトのサブツリーで機能する複数の RDN の置き換え。
($attr.attrName)	**userdn, roledn, groupdn, userattr**	ターゲットエントリーからサブジェクトへの **attributeName** 属性値の置換。

マクロを使用する場合は、**($dn)** マクロを含むターゲットを定義する必要があることに注意してください。**($dn)** マクロと **($attr.attrName)** マクロを組み合わせることができます。

2.3. ($DN) マクロの例

($dn) マクロは、置換値を LDAP リクエストからのエントリーと比較します。たとえば、LDAP リクエストは次のエントリーを対象としています。

```
cn=all,ou=groups,dc=subdomain1,dc=hostedCompany1,dc=example,dc=com
```

ACI は、次のターゲットを定義します。

```
(target="ldap:///ou=groups,($dn),dc=example,dc=com")
```

この例では、**($dn)** マクロは **dc=subdomain1,dc=hostedCompany1** と一致します。

ACI のサブジェクトが **($dn)** マクロを使用する場合、ターゲットに一致するサブストリングによりサブジェクトが展開されます。

```
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";)
```

ACI は次のように展開されます。

```
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";)
```

マクロが展開された後、Red Hat Directory Server は通常のプロセスに従って ACI を評価し、アクセスが許可されているか判断します。

2.4. [$DN] マクロの例

[$dn] マクロは、ターゲットとなるソースの DN を複数回確認します。このマクロは、一致するまで、確認するたびに左端の RDN コンポーネントを削除します。

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 リクエストと、以下の ACI があるとします。

```
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";)
```

マクロは次のように展開されます。

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。結果が一致しない場合、処理は続行され、左端部分が削除されます。
3. サブジェクトの [$dn] は **dc=hostedCompany1** です。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。バインド DN がそのグループのメンバーでない場合、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] マクロは、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へのアクセスを許可します。これは、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";)
```

この ACI は、**cn=domainAdmins,ou=groups,dc=hostedCompany1,dc=example,dc=com** のメンバーに、**dc=hostedCompany1** 下にあるすべてのサブドメインへのアクセスを許可します。そのグループのメンバーである管理者は、**ou=people,dc=subdomain1.1,dc=subdomain1** のようなサブツリーにアクセスできます。ただし、**cn=domainAdmins,ou=groups,dc=subdomain1.1** のメンバーに **ou=people,dc=hostedCompany1** および **ou=people,dc=subdomain1,dc=hostedCompany1** ノードへのアクセスはありません。

2.5. ($ATTR.ATTRNAME) マクロの例

($attr.attrName) マクロは、必ず DN の一部として使用します。たとえば、以下の **roledn** を定義します。

```
roledn = "ldap:///cn=DomainAdmins,($attr.ou),dc=HostedCompany1,dc=example,dc=com"
```

サーバーが、以下のエントリーを対象とする LDAP 操作を受信すると仮定します。

```
dn: cn=Jane Doe,ou=People,dc=HostedCompany1,dc=example,dc=com
cn: Jane Doe
```

```
sn: Doe
ou: Engineering...
```

ACI の **roledn** 部分を評価するために、サーバーはターゲットエントリーの **ou** 属性を確認し、その値を使用してマクロを展開します。**roledn** は次のように展開されます。

```
roledn = "ldap:///cn=DomainAdmins,ou=Engineering,dc=HostedCompany1,dc=example,dc=com"
```

Red Hat 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。

属性に複数の値がある場合、RHDS は各値を使用してマクロを展開し、展開されたマクロと最初に一致した値を使用します。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
ou: People...
```

Red Hat Directory Server が ACI を評価するとき、展開された次の式に対して論理 **OR** を実行します。

```
roledn = "ldap:///cn=DomainAdmins,ou=Engineering,dc=HostedCompany1,dc=example,dc=com"

roledn = "ldap:///cn=DomainAdmins,ou=People,dc=HostedCompany1,dc=example,dc=com"
```

第3章 LDAP ブラウザーでのアクセス制御命令の管理

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を説明します。

3.1. 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 Browser** 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。
- 親エントリーが Red Hat Directory Server に存在する。

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。

3.2. 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。

- 親エントリーが Red Hat Directory Server に存在する。

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。

3.3. 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。
- 親エントリーが Red Hat Directory Server に存在する。

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。

第4章 パスワードベースのアカウントロックアウトポリシーの設定

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

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

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

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

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

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

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

前提条件

- アカウントロックアウトポリシーを設定している。

手順

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

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

検証

1. **passwordmaxfailure** 設定の値を表示します。

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

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

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

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

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

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

関連情報

- コマンドラインでパスワードベースのアカウントロックアウトポリシーの設定

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

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

重要

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

手順

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

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

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

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

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

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

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

検証

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

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

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

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

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

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

関連情報

関連情報

- レガシーパスワードポリシーの設定

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

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

重要

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

前提条件

- Web コンソールでインスタンスにログインしている。

手順

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

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

   この設定は、Web コンソールでは利用できません。

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

検証

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

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

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

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

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

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

関連情報

- レガシーパスワードポリシーの設定

第5章 時間ベースのアカウントロックアウトポリシーの設定

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

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

5.1. 最後に成功したログインで一定時間アカウントを自動的に無効にする

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

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

手順

1. アカウントポリシープラグインを有効にします。

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

2. プラグイン設定エントリーを設定します。

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

コマンドは、以下のオプションを使用します。

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

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

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

4. アカウント非アクティブ化ポリシーエントリーを作成します。

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

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

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

5. CoS テンプレートエントリーを作成します。

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

dn: cn=TemplateCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: cosTemplate
acctPolicySubentry: cn=Account Inactivation Policy,dc=example,dc=com
```

このテンプレートエントリーは、アカウントの非アクティブ化ポリシーを参照します。

6. CoS 定義エントリーを作成します。

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

dn: cn=DefinitionCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=TemplateCoS,dc=example,dc=com
cosAttribute: acctPolicySubentry default operational-default
```

この定義エントリーは CoS テンプレートエントリーを参照し、**acctPolicySubentry** 属性が各ユーザーエントリーに表示され、値が **cn=Account Inactivation Policy,dc=example,dc=com** 設定されます。

検証

1. **lastLoginTime** 属性を、設定した非アクティブ時間よりも古い値に設定します。

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

dn: uid=example,ou=People,dc=example,dc=com
changetype: modify
replace: lastLoginTime
lastLoginTime: 20210101000000Z
```

2. このユーザーとしてディレクトリーに接続してみてください。

```
# ldapsearch -H ldap://server.example.com -x -D
"uid=example,ou=People,dc=example,dc=com" -W -b "dc=example,dc=com"
ldap_bind: Constraint violation (19)
 additional info: Account inactivity limit exceeded. Contact system administrator to reset.
```

Directory Server がアクセスを拒否してこのエラーを返した場合、アカウントの非アクティブが機能します。

関連情報

- 非アクティブ制限に達したアカウントを再度有効にする

5.2. アカウントを作成してから一定時間、アカウントを自動的に無効にする

次の手順に従って、**dc=example,dc=com** エントリーのアカウントが管理者が作成してから 60 日後に期限切れになるように設定します。

たとえば、アカウントの有効期限機能を使用して、外部ワーカーのアカウントが作成されてから一定時間ロックされるようにします。

手順

1. アカウントポリシープラグインを有効にします。

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

2. プラグイン設定エントリーを設定します。

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

コマンドは、以下のオプションを使用します。

- **--always-record-login yes**: ログイン時間のログを有効にします。これは、**acctPolicySubentry** 属性が設定されていない場合でも、サービスクラス (CoS) またはアカウントポリシーを持つロールを使用するために必要です。

- **--state-attr createTimestamp**: アカウントポリシープラグインが **createTimestamp** 属性の値を使用して、アカウントの有効期限が切れているかどうかを計算するように設定します。
- **--alt-state-attr 1.1**: 代替属性を使用してプライマリー属性が存在しないかどうかをチェックすることを無効にします。
- **--spec-attr acctPolicySubentry**: **acctPolicySubentry** 属性が設定されているエントリーにポリシーを適用するように Directory Server を設定します。この属性は、CoS エントリーで設定します。
- **--limit-attr accountInactivityLimit**: アカウントの有効期限ポリシーエントリーの **accountInactivityLimit** 属性に最大経過時間を保存するように設定します。

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

4. アカウントの有効期限ポリシーエントリーを作成します。

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

dn: cn=Account Expiration Policy,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: accountpolicy
accountInactivityLimit: 5184000
cn: Account Expiration Policy
```

accountInactivityLimit 属性の値は、アカウントが作成されてから **5184000** 秒 (60 日) で期限切れになるように設定します。

5. CoS テンプレートエントリーを作成します。

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

dn: cn=TemplateCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: cosTemplate
acctPolicySubentry: cn=Account Expiration Policy,dc=example,dc=com
```

このテンプレートエントリーは、アカウントの有効期限ポリシーを参照します。

6. CoS 定義エントリーを作成します。

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

dn: cn=DefinitionCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectclass: cosSuperDefinition
```

```
objectclass: cosPointerDefinition
cosTemplateDn: cn=TemplateCoS,dc=example,dc=com
cosAttribute: acctPolicySubentry default operational-default
```

この定義エントリーは CoS テンプレートエントリーを参照し、**acctPolicySubentry** 属性が各ユーザーエントリーに表示され、値が **cn=Account Expiration Policy,dc=example,dc=com** 設定されます。

検証

- **createTimestamp** 属性が 60 日以上前の値に設定されている **dc=example,dc=com** エントリーに格納されているユーザーとしてディレクトリーに接続してみてください。

```
# ldapsearch -H ldap://server.example.com -x -D "uid=example,dc=example,dc=com" -W -b "dc=example,dc=com"
ldap_bind: Constraint violation (19)
 additional info: Account inactivity limit exceeded. Contact system administrator to reset.
```

Directory Server がアクセスを拒否してこのエラーを返した場合、アカウントの有効期限が機能します。

関連情報

- 非アクティブ制限に達したアカウントを再度有効にする

5.3. パスワードの有効期限が切れてから一定時間アカウントを自動的に無効にする

この手順に従って、28 日を超えてパスワードを変更しない **dc=example,dc=com** エントリーの下のユーザーを非アクティブ化する時間ベースのロックアウトポリシーを設定します。

前提条件

- ユーザーは、エントリーに **passwordExpirationTime** 属性を設定する必要があります。

手順

1. パスワードの有効期限機能を有効にします。

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

2. アカウントポリシープラグインを有効にします。

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

3. プラグイン設定エントリーを設定します。

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

```
non_existent_attribute --alt-state-attr passwordExpirationTime --spec-attr
acctPolicySubentry --limit-attr accountInactivityLimit
```

コマンドは、以下のオプションを使用します。

- **--always-record-login yes**: ログイン時間のログを有効にします。これは、**acctPolicySubentry** 属性が設定されていない場合でも、サービスクラス (CoS) またはアカウントポリシーを持つロールを使用するために必要です。
- **--always-record-login-attr lastLoginTime**: アカウントポリシープラグインがユーザーの **lastLoginTime** 属性に最終ログイン時刻を保存するように設定します。
- **--state-attr non_existent_attribute**: アカウントポリシーの評価に使用されるプライマリー時間属性を、存在しないダミー属性名に設定します。
- **--alt-state-attr `passwordExpirationTime**: チェックする代替属性として **passwordExpirationTime** 属性を使用するようにプラグインを設定します。
- **--spec-attr acctPolicySubentry**: **acctPolicySubentry** 属性が設定されているエントリーにポリシーを適用するように Directory Server を設定します。この属性は、CoS エントリーで設定します。
- **--limit-attr accountInactivityLimit**: アカウントポリシーエントリーの **accountInactivityLimit** 属性に、最後にパスワードを変更した後にアカウントが非アクティブ化された時刻を保存するように設定します。

4. インスタンスを再起動します。

```
# dsctl instance_name restart
```

5. アカウント非アクティブ化ポリシーエントリーを作成します。

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

dn: cn=Account Inactivation Policy,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: accountpolicy
accountInactivityLimit: 2419200
cn: Account Inactivation Policy
```

accountInactivityLimit 属性の値は、パスワードが変更されてから **2419200** 秒 (28 日) 後に Directory Server がアカウントを非アクティブ化するように設定します。

6. CoS テンプレートエントリーを作成します。

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

dn: cn=TemplateCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectClass: extensibleObject
objectClass: cosTemplate
acctPolicySubentry: cn=Account Inactivation Policy,dc=example,dc=com
```

このテンプレートエントリーは、アカウントの非アクティブ化ポリシーを参照します。

7. CoS 定義エントリーを作成します。

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

dn: cn=DefinitionCoS,dc=example,dc=com
objectClass: top
objectClass: ldapsubentry
objectclass: cosSuperDefinition
objectclass: cosPointerDefinition
cosTemplateDn: cn=TemplateCoS,dc=example,dc=com
cosAttribute: acctPolicySubentry default operational-default
```

この定義エントリーは CoS テンプレートエントリーを参照し、**acctPolicySubentry** 属性が各ユーザーエントリーに表示され、値が **cn=Account Inactivation Policy,dc=example,dc=com** 設定されます。

検証

1. ユーザーの **passwordExpirationTime** 属性を、設定した非アクティブ時間よりも古い値に設定します。

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

dn: uid=example,ou=People,dc=example,dc=com
changetype: modify
replace: passwordExpirationTime
passwordExpirationTime: 20210101000000Z
```

2. このユーザーとしてディレクトリーに接続してみてください。

```
# ldapsearch -H ldap://server.example.com -x -D
"uid=example,ou=People,dc=example,dc=com" -W -b "dc=example,dc=com"
ldap_bind: Constraint violation (19)
 additional info: Account inactivity limit exceeded. Contact system administrator to reset.
```

Directory Server がアクセスを拒否してこのエラーを返した場合、アカウントの非アクティブが機能します。

関連情報

- 非アクティブ制限に達したアカウントを再度有効にする

5.4. アカウントのステータス (アクティブかどうか) とパスワードの有効期限の両方でアカウントを自動的に無効にする

checkAllStateAttrs 設定を使用してユーザーの認証時に、アカウントのステータス (アクティブかどうか)、またパスワードの有効期限の両方を適用できます。デフォルトでは、プラグイン設定エントリーに **checkAllStateAttrs** が存在しない場合、またはこのパラメーターを **no** に設定した場合、プラグインは状態属性 **lastLoginTime** をチェックします。属性がエントリーに存在しない場合には、プラグインは別の状態属性をチェックします。

プラグインが **passwordExpirationtime** 属性に基づいて有効期限を処理するようにする場合は、メイン

の状態属性を存在しない属性に設定し、別の状態属性を **passwordExpirationtime** に設定できます。このパラメーターを有効にすると、主な状態属性をチェックし、アカウントに問題がある場合は、別の状態属性を確認します。

これは、passwordExpirationtime がアクティブではない期間の制限を超えると、アカウントポリシープラグインがアカウントを完全に無効にするという点で、パスワードポリシーのパスワード有効期限とは異なります。パスワードポリシーの有効期限が切れても、ユーザーは引き続きログインしてパスワードを変更できます。アカウントポリシープラグインはユーザーの操作を完全にブロックするため、管理者はアカウントをリセットする必要があります。

手順

1. プラグイン設定エントリーを作成し、設定を有効にします。

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

2. サーバーを再起動して、新しいプラグイン設定を読み込みます。

```
# dsctl instance_name restart
```

警告

checkAllStateAttrs 設定は、別の state 属性が **passwordExpiratontime** に設定されている場合にのみ機能するように設計されています。これを **createTimestamp** に設定すると、予想外の結果が発生し、エントリーがロックアウトされる可能性があります。

第6章 非アクティブ制限に達したアカウントを再度有効にする

Directory Server が非アクティブ制限に達したためにアカウントを非アクティブ化した場合、管理者はアカウントを再度有効にすることができます。

6.1. アカウントポリシープラグインによって非アクティブ化されたアカウントを再度有効にする

dsconf account unlock コマンドを使用するか、非アクティブ化されたユーザーの **lastLoginTime** 属性を手動で更新することにより、アカウントを再度有効にすることができます。

前提条件

- 非アクティブ化されたユーザーアカウント。

手順

- 次のいずれかの方法を使用して、アカウントを再アクティブ化します。
  - **dsconf account unlock** コマンドの使用:

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

  - ユーザーの **lastLoginTime** 属性を最近のタイムスタンプに設定するには、次のようにします。

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

    dn: uid=example,ou=People,dc=example,dc=com
    changetype: modify
    replace: lastLoginTime
    lastLoginTime: 20210901000000Z
    ```

検証

- 再アクティブ化したユーザーとして認証します。たとえば、次の検索を実行します。

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

  ユーザーが正常に認証できる場合、アカウントは再度アクティブ化されました。

第7章 ロックアウトポリシーを設定せずに最終ログイン時間を追跡する

アカウントポリシープラグインを使用すると、有効期限や非アクティブ期間を設定せずに、ユーザーのログイン時間を追跡できます。この場合、プラグインは **lastLoginTime** 属性をユーザーエントリーに追加します。

7.1. 最終ログイン時刻を記録するようにアカウントポリシープラグインを設定する

この手順に従って、ユーザーエントリーの **lastLoginTime** 属性にユーザーの最終ログイン時刻を記録します。

手順

1. アカウントポリシープラグインを有効にします。

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

2. ログイン時間を記録するプラグイン設定エントリーを作成します。

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

   コマンドは、以下のオプションを使用します。

   - **--always-record-login yes**: ログイン時間のログ記録を有効にします。
   - **--state-attr lastLoginTime**: アカウントポリシープラグインがユーザーの **lastLoginTime** 属性に最終ログイン時刻を保存するように設定します。

3. インスタンスを再起動します。

```
# dsctl instance_name restart
```

検証

1. ユーザーとして Directory Server にログインします。たとえば、次の検索を実行します。

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

2. 前の手順で使用したユーザーの **lastLoginTime** 属性を表示します。

```
# ldapsearch -H ldap://server.example.com -x -D "cn=Directory Manager" -W -b "uid=example,ou=people,dc=example,dc=com" lastLoginTime
...
dn: uid=example,ou=People,dc=example,dc=com
lastLoginTime: 20210913091435Z
```

lastLoginTime 属性が存在し、Directory Server がその値を更新した場合、最終ログイン時刻の記録が機能します。

第8章 GET EFFECTIVE RIGHTS 検索を使用したエントリーのアクセス権の管理

管理者は、特定のエントリー内の属性に対してユーザーが持つアクセス権を検索および制御できます。

Get effective rights (GER) は、ディレクトリー検索を拡張して、指定したエントリーに対してユーザーが持っているアクセス権を表示する方法です。次の権限を指定できます。

- Read
- Write と Self-write
- 検索
- 追加
- 削除

エントリーに対する有効な権限を確認すると、次のような場合に役立ちます。

- GER コマンドを使用して、ディレクトリーのアクセス制御命令をより適切に編成できます。あるユーザーグループが閲覧または編集できる内容を、別のユーザーグループと比較して制限しなければならない状況は頻繁に発生します。たとえば、**QA Managers** グループのメンバーは、**manager** や **salary** などの属性を検索して読み取る権限を持っていますが、それを変更または削除する権限を持っているのは **HR Group** のメンバーだけです。他にも方法はありますが、ユーザーまたはグループの実効権限を確認することで、管理者が適切なアクセス制御を設定しているか確認できます。
- GER コマンドを使用して、個人のエントリーで表示または変更できる属性を確認できます。たとえば、ユーザーは **homePostalAddress** や **cn** などの属性にアクセスできるはずですが、**manager** 属性や **salary** 属性に対しては読み取り権限しか持っていない場合があります。

getEffectiveRights 検索では、次のエンティティーが使用されます。

- **リクエスター**。これは、**getEffectiveRights** 検索が操作を発行したときに認証されたエントリーです。
- 権限を評価する **サブジェクト**。これは、GER コントロールで認可 **DN** として定義されます。
- **ターゲット**。リクエストの検索ベース、検索フィルター、属性リストで定義します。

8.1. GET EFFECTIVE RIGHTS 検索のパーミッション

Get Effective Rights (GER) 検索では、すべてのエントリーが持つことができる、以下のアクセス権が表示されます。

- エントリーに対する権限である **上位権限**。そのアクセス権は、**ユーザー A** が **ユーザー B** のエントリーに対して実行できる操作の種類を示します。
- **第 2 位権限** は、**ユーザー A** が特定の属性に対して持つ権利を示します。**ユーザー A** は、同じエントリー内の異なる属性に対して異なる権限を持っている場合があります。ユーザーが持つアクセス制御は、そのエントリーに対して有効な権限です。

以下に例を示します。

```
entryLevelRights: vadn
attributeLevelRights: givenName:rscWO, sn:rscW, objectClass:rsc, uid:rsc, cn:rscW
```

GER 検索には、エントリーと属性に対する次のアクセス権があります。

表8.1 エントリーの権限

Permissions	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8.2 属性権

Permissions	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。

8.2. GET EFFECTIVE RIGHTS 検索の形式

Get effective rights (GER) は拡張ディレクトリー検索です。これを使用するには、 **ldapsearch** コマンドで **-E** オプションを Lightweight Directory Access Protocol (LDAP) コントロールに渡す必要があります。以下に例を示します。

```
# ldapsearch -x -D bind_dn -W -p server_port -h server_hostname -b base_DN -E
[!]1.3.6.1.4.1.42.2.27.9.5.2=:GER_subject (searchFilter) attributeList
```

- **-b** は、GER サブジェクトを検索でき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、または結果として 1 つのエントリーのみ返される場合、その特定のエントリーに対してリクエスターが持つ権限が結果に示されます。複数のエントリーがフィルターに一致する場合、その検索では一致するすべてのエントリーと、各エントリーに対するリクエスターの権限が返されます。

- **1.3.6.1.4.1.42.2.27.9.5.2** オプションは、GER コントロールのオブジェクト ID です。感嘆符 (**!**) は、サーバーがこのコントロール (**!**) 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、何も返さないかを定義します。
- **GER_subject** は、確認対象の権限を持つユーザーです。**GER_subject** を空白 (**dn:**) のままにすると、匿名ユーザーの権限を結果として取得できます。
- オプションの **attributeList** を使用すると、GER の結果は指定された属性またはオブジェクトクラス (**mail** 属性など) に限定されます。
- アスタリスク (*) 記号を使用すると、すべての属性が返されます。
- プラス (**+**) 記号を使用すると、操作属性が返されます。

GER オプションは、**ldapsearch** の結果に追加情報が追加され、特定のユーザーが持つ権限が示されます。その GER サブジェクトユーザーは、追加オプション **-D** を使用して、自分のエントリーに対する権限を要求できます。

リクエスターが Directory Manager ユーザーではない場合、リクエスターには、リクエスターのエントリーに対して GER サブジェクトが持つ権限のみ表示されます。他のエントリーはすべて、有効な権限に対してアクセス権不足のエラーを返します。

以下は、通常のユーザーが GER 検索を実行する一般的なシナリオです。

- 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限を確認します。

8.3. GET EFFECTIVE RIGHTS 検索の一般的なシナリオ

次の例は、Get Effective Rights 検索をいつ、どのように使用できるかの一般的なシナリオを示しています。

8.3.1. Get Effective Rights 検索の一般的な例

以下は、Get Effective Rights (GER) 検索を使用する必要がある最も一般的なシナリオです。

1. 個人の権利を確認します。ユーザー A が個人エントリーの権限を確認します。たとえば、Ted Morris は自分のエントリーに対して自分が持つ権限を確認します。

例8.1 個人の権限の確認 (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
```

この例では、**-b** オプションにはリクエスターの DN も含まれています。

2. 別のユーザーに対する権限を確認します。たとえば、Ted Morris はマネージャーであり、部下の 1 人である Dave Miller のエントリーを確認する必要があります。

例8.2 別のユーザーに対する権限の確認 (ユーザー A からユーザー B)

```
# 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
```

この例では、Ted Morris は Dave Miller のエントリーに対して、すべての属性の読み取り、検索、比較、変更、削除を実行するための権限を持っています。

3. ディレクトリーマネージャーとして、あるユーザーが別のユーザーのエントリーに対してす持つ権限を確認します。たとえば、ディレクトリーマネージャーは、Jane Smith がマネージャーとして部下である Ted Morris のエントリーに対して持つ権限を確認します。

例8.3 ディレクトリーマネージャーとして、あるユーザーが別のユーザーに対して持つ権限を確認

```
# 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
```

ユーザーが権限を持っていない場合、結果には不十分なアクセス権を示すエラーが表示されます。

例8.4 エントリーに対する権限がない場合

```
# 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
```

4. 自分以外のユーザーが自分のエントリーに対して持つ権限を確認します。たとえば Ted Morris は、Dave Miller が Ted Morris のエントリーに対して持つ権限を確認します。

例8.5 自分以外のユーザーが自分のエントリーに対して持つ権限を確認する場合

```
# 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
```

この例では、Dave Miller は、エントリーの DN を表示し、**ou**、**givenName**、**l**、およびその他の属性の読み取り、検索、比較を実行する権限を持っています。**userPassword** 属性に対する権限はありません。

8.3.2. 存在しない属性に対する Get Effective Rights 検索の例

デフォルトでは、エントリーの属性には値がありません。Get Effective Rights (GER) 検索でアスタリスク (*) を使用すると、そのエントリーで使用できるすべての属性が、エントリーに設定されていない属性も含めて返されます。

例8.6 エントリーのすべての属性に対する権限の確認

```
# 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
```

この例では、**secretary** 属性は設定されていませんが、それでも GER 検索結果で確認できます。

8.3.3. 特定の属性またはオブジェクトクラスに対する Get Effective Rights 検索の例

このセクションの例は、エントリーのオブジェクトクラスに属する特定の属性、属性セット、およびすべての属性に対する権限を検索する方法を示しています。

1. Get Effective Rights (GER) 検索結果に、エントリーの特定の属性が一覧表示されます。以下に例を示します。

例8.7 特定の属性の Get Effective Rights 検索結果

```
# 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
```

```
-
```

2. **attribute@objectClass** 形式を使用した、エントリーのオブジェクトクラスにおける特定の属性の GER 検索。リクエスターは、ディレクトリーマネージャーである必要があります。

例8.8 オブジェクトクラスの特定の属性に対する Get Effective Rights 検索結果

```
# 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
```

***@objectClass** の形式でアスタリスク (*) を使用して、オブジェクトクラスのすべての属性を返すことができます。検索結果には、存在しない属性も含まれます。

8.3.4. 存在しないエントリーの Get Effective Rights 検索の例

この例は、ユーザーのまだ存在しないエントリーに対して、特定のユーザーが持つ権限を確認する方法を示しています。この場合、サーバーによりサブツリー内にテンプレートエントリーが生成され、それに対して Get Effective Rights (GER) を使用できます。存在しないエントリーをチェックする場合、Get Effective Rights (GER) 検索は、指定したオブジェクトクラスを使用して、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。

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。MUST 属性が存在しない場合、サーバーは MAY 属性を使用します。**@objectclass:rdn_attribute** の形式で RDN 値をオブジェクトクラスに渡して指定します。

たとえば、RDN として **uidNumber** を持つ、存在しない POSIX エントリーの **scarter** の権利を確認するには、以下を実行します。

例8.9 存在しないエントリーに対する権限の確認

```
# 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
```

8.3.5. 操作属性の Get Effective Rights 検索の例

ldapsearch コマンドは操作属性を返しません。検索するには、プラス記号 (**+**) を使用します。**+** を使用すると、エントリーに対して使用できる操作属性のみが返されます。

例8.10 操作属性の検索

```
# 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
```

8.3.6. Get Effective Rights 結果とアクセス制御ルールの例

有効なアクセス制御リスト (ACL) は、ユーザーが持つ Get Access Rights (GER) を定義します。

例8.11 アクセス制御リスト

```
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
```

この例では、ACL に **dc=example,dc=com** サブツリーは含まれていません。そのため、GER 検索結果には、ユーザーが **dc=example,dc=com** エントリーに対する権限を持っていないことが示されます。

例8.12 ACL が設定されていない場合の GER 検索結果

```
# 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
```

結果を表示するには、Directory Manager である必要があります。それ以外の場合、結果は空白になります。

8.4. GET EFFECTIVE RIGHT のリターンコード

エラーが発生した場合、Get Effective Rights (GER) 検索結果はエラーコードを返します。次の表では、エラーコードを説明しています。

表8.3 エラーコード

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。重要な式が **true** に設定され、エントリーに有効な権利が存在しない場合。
16	そのような属性はありません。
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。