Red Hat Directory Server 12

管理访问控制

使用访问控制指令配置权限

Last Updated: 2024-05-09

Red Hat Directory Server 12 管理访问控制

使用访问控制指令配置权限

法律通告

摘要

了解如何定义哪些用户可以对 Red Hat Directory Server 中的后缀和条目执行特定操作。这些任务由访问控制指令(ACI)控制。了解不同的 ACI 类型、ACI 用例、绑定规则和检查条目访问权限的方法。

目录

对红帽文档提供反馈

我们感谢您对文档提供反馈信息。请让我们了解如何改进文档。要做到这一点：

- 要通过 JIRA 提交反馈（需要帐户）：
  1. 登录到 Jira 网站。
  2. 在顶部导航栏中点 **Create**
  3. 在 **Summary** 字段中输入描述性标题。
  4. 在 **Description** 字段中输入您对改进的建议。包括到文档相关部分的链接。
  5. 点对话框底部的 **Create**。
- 要通过 Bugzilla 提交反馈（需要帐户）：
  1. 进入 Bugzilla 网站。
  2. 在 Component 中选择 **Documentation**。
  3. 在 **Description** 中输入您要提供的信息。包括文档相关部分的链接。
  4. 点 **Submit Bug**。

第 1 章 管理访问控制指令

当 Directory 服务器收到请求时，它使用用户在绑定操作中定义的验证信息，以及目录中定义的访问控制指令(ACI)，以允许或拒绝对所请求的条目或属性的访问。服务器可以允许或拒绝操作权限，如 **读取、写入、搜索和 比较**。授予用户的权限级别取决于提供的身份验证信息。

Directory Server 中的访问控制允许您在适用于 ACI 时设置精确的规则：

- 对于整个目录、子树或特定条目
- 对于特定用户，属于特定组或角色的所有用户，或 目录中的所有用户
- 对于特定位置，如 IP 地址、IP 范围或 DNS 名称。
  请注意，负载均衡器可能会影响特定于位置的规则。

重要

难以阅读和理解复杂的 ACI。您可以编写多个简单规则来实现同样的效果，而不是一个复杂的 ACI。但是，有大量 ACI 也会增加 ACI 处理的成本。

1.1. ACI 放置

目录服务器在目录条目中的多值 **aci** operational 属性中存储访问控制指令(ACI)。要设置 ACI，请在对应的目录条目中添加 **aci** 属性。目录服务器应用 ACI：

- 如果没有任何子条目，则 仅限包含 ACI 的条目。例如，如果客户端需要访问 **uid=user_name,ou=People,dc=example,dc=com** 对象，并且 ACI 在 **dc=example,dc=com** 上仅应用于任何子条目，则仅应用此 ACI。

  注意

  带有 **添加权限** 的 ACI 也适用于将来创建的子条目。

- 指向包含 ACI 及其下面所有条目的条目（如果具有子条目）。直接，当服务器评估访问权限到任何给定条目时，它会验证所请求和目录后缀之间的每个条目的 ACI，以及条目本身的 ACI。
  例如，在 **dc=example,dc=com** 和 **ou=People,dc=example,dc=com** 条目上设置 ACI：如果客户端希望访问 **uid=user_name,ou=People,dc=example,dc=com** 对象，没有 ACI 设置，Directory 服务器会首先验证 **ou=People,dc=example,dc=com** 条目上的 ACI。如果此 ACI 授予访问权限，评估将停止并授予访问权限。如果没有，Directory 服务器会在 **ou=People,dc=example,dc=com** 上验证 ACI。如果此 ACI 成功授权客户端，它可以访问对象。

注意

rootDSE 条目中设置的 ACI 仅适用于此条目。

在条目上创建的 ACI 可以不直接应用到该条目，而是将其设置为以下子树中的一些或全部条目。这种方法的优势在于，可以将常规 ACI 放入目录树中，使其对位于树里的条目产生影响。例如，可以在 **organizationalUnit** 条目或本地性条目中创建包含 **inetOrgPerson** 对象类的 ACI。

注意

通过将常规规则放置到高级别分支点，将 ACI 的数量减小到目录树中。要限制更具体规则的范围，请尽可能将它们放入 leaf 条目。

1.2. ACI 的结构

aci 属性使用以下语法：

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

- **target_rule** 指定条目、属性或一组用于控制访问的条目和属性。
- **版本 3.0** 是一个所需字符串，用于标识访问控制指令(ACI)版本。
- **ACL "*ACL 名称*"** 设置描述 ACI 的名称或字符串。
- **permission_rule** 可设置允许或拒绝权限，如 **read** 或 **write**。
- **bind_rules** 指定在绑定以允许或拒绝访问过程中必须匹配哪些规则。

权限和绑定规则对称为访问控制规则。

要有效地为给定目标设置多个访问控制，您可以为每个目标设置多个访问控制规则：

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

1.3. ACI 评估

要评估特定条目的访问权限，服务器会在条目本身上创建一个访问控制指令(ACI)列表，并将父条目备份到存储在目录服务器中的顶级条目。ACI 针对特定实例在所有数据库中评估，但不在不同的实例之间。

目录服务器根据 ACI 的语义评估 ACI 列表，而不是在目录树中的放置上评估。这意味着，与目录树的根目录接近的 ACI 不优先于目录树所处处的 ACI。

在 Directory Server 中，ACI 中的 **拒绝** 权限优先于 **允许** 权限。例如，如果在目录的根目录中拒绝写入权限，无论其他 ACI 是否授予了此权限，任何用户都不能写入该目录。要为 目录授予特定用户写入权限，您必须为原始拒绝规则添加例外，以便用户在该目录中进行写入。

注意

若要改进 ACI，请使用精细的 **允许规则** 而不是 **拒绝规则**。

1.4. ACI 的限制

当您设置访问控制指令(ACI)时，会有以下限制：

- 如果您的目录数据库在多个服务器间分布，则以下限制适用于您可以在 ACI 中使用的关键字：
  - 使用 **groupdn** 关键字分组条目的 ACI 必须位于与组条目相同的服务器上。
    如果组是动态的，则组的所有成员都必须在服务器上有一个条目。静态组的成员条目可以位于远程服务器中。
  - 使用 **roledn** 关键字，基于角色定义的 ACI 必须位于与角色定义条目相同的服务器上。设计为具有该角色的每个条目也必须位于同一服务器上。

  但是，您可以使用 **userattr** 关键字，将目标条目中存储的值与绑定用户条目中存储的值匹配。在这种情况下，即使 bind 用户在存储 ACI 的服务器上没有条目，访问通常会被评估。

- 您可以在以下 ACI 关键字中使用虚拟属性，如 Service(CoS)属性：
  - **targetfilter**
  - **targattrfilters**
  - **userattr**
- 仅在本地服务器上评估访问控制规则。例如，如果您在 ACI 关键字中的 LDAP URL 中指定服务器的主机名，该 URL 将被忽略。

1.5. DIRECTORY 服务器如何在复制拓扑中处理 ACI

访问控制指令(ACI)存储在条目 **的辅助** 属性中。因此，如果包含 ACI 的条目是复制数据库的一部分，则 ACI 将被复制。

ACI 始终在解析传入 LDAP 请求的服务器上评估。当消费者服务器收到更新请求时，它将在评估请求是否可以在供应商上服务请求前返回到供应商服务器。

1.6. 显示、添加、删除和更新 ACI

您可以使用 **ldapsearch** 实用程序搜索，使用 **ldapmodify** 程序来添加、删除和更新访问控制说明(ACI)。

显示 ACI：

例如，要显示在 **dc=example,dc=com** 和 sub-entries 中设置的 ACIs，请输入：

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

添加 ACI

例如，要将 ACI 添加到 **ou=People,dc=example,dc=com** 条目中，请输入：

```
# 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";)
```

删除 ACI

删除 ACI：

- 如果条目上只设置了 **aci** 属性，或者您想要从条目中删除所有 ACI:

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

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

- 如果条目上存在多个 ACI，并且要删除特定的 ACI，请指定准确的 ACI：

```
# 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";)
```

更新 ACI

更新 ACI：

- 删除现有的 ACI。
- 使用更新的设置添加新的 ACI。

1.7. 定义 ACI 目标

访问控制指令(ACI)中的目标规则定义目录服务器应用 ACI 的条目。如果您没有设置目标，ACI 将应用到包含 **aci** 属性和以下条目的条目。

在 ACI 中，以下突出显示的部分是目标规则：

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

对于复杂的 ACI，Directory 服务器支持多个目标规则，它们带有 ACI 中不同关键字：

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

如果您指定多个目标规则，则顺序不相关。请注意，您只能在 ACI 中使用以下关键字：

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

1.7.1. 目标规则的语法

目标规则的一般语法是：

```
(keyword comparison_operator "expression")
```

- **关键字** ：设置目标的类型。
- **comparison_operator**: 有效值为 **=** 和 **!=**，并指明目标是表达式中指定的对象。

警告

出于安全考虑，红帽建议不要使用 **!=** 运算符，因为它允许指定对所有其他条目或属性的操作。例如：

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

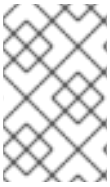
前面的示例允许用户设置、更新或删除任何属性，除了设置了 ACI 的 Distinguished Name(DN)下的 **userPassword** 属性除外。不过，这也可让用户添加额外的 **aci** 属性，允许对此属性进行写入访问权限。

- **表达式** ：设置目标，且必须使用引号括起。表达式本身取决于您使用的关键字。

1.7.2. 定位目录条目

要根据可辨识的名称(DN)和其下面的条目控制访问，请使用访问控制指令(ACI)中的 **target** 关键字。使用 **target** 关键字的目标规则将 DN 用作表达式：

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



注意

您必须在目标的 DN 或更高级别的 DN 上使用 **target** 关键字设置 ACI。例如，如果目标 ou=People,dc=example,dc=com，您必须在 ou=People,dc=example,dc=com 或 dc=example,dc=com 中设置 ACI。

例 1.1. 使用 target 关键字

要启用存储在 ou=People,dc=example,dc=com 条目中的用户，以搜索和显示他们自己的条目中的所有属性：

```
# 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");)
```

使用 target 关键字的通配符

您可以使用 * 通配符目标多个条目。

以下目标规则示例匹配 **ou=People,dc=example,dc=com** 中所有条目，其 uid 属性设置为以字母 **a** 开头的值：

-

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

根据通配符的位置，该规则不仅适用于属性值，而且也适用于完整的 DN。因此，您可以使用通配符替换 DN 的部分。

例 1.2. 使用通配符作为目录条目

以下规则以 **dc=example,dc=com** 树中的所有条目为目标，其具有匹配 **uid** 属性，而不仅存储在 **dc=example,dc=com** 条目本身中的条目：

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

以上目标规则与多个条目匹配，例如：

- **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

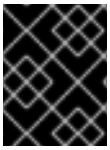
目录服务器不支持 DN 的后缀部分的通配符。例如，如果您的目录的后缀为 **dc=example,dc=com**，则无法使用此后缀带有通配符的目标，如 **(target = "ldap:///dc=*.com")**。

1.7.3. 目标属性

要将访问控制指令(ACI)中的访问权限限制为某些属性，请使用 **targetattr** 关键字。例如，这个关键字定义：

- 在读取操作中，将向客户端返回哪些属性
- 在搜索操作中，搜索哪些属性
- 在写入操作中，可以将哪些属性写入对象
- 在 add 操作中，创建新对象时可以添加哪些属性

在某些情况下，您可以通过将其他目标关键字与 **targetattr** 结合使用，使用 **targetattr** 关键字来保护 ACI。请参阅 目标规则 的高级用法。



重要

在 **read** 和 **search** 操作中，默认目标为 no 属性。没有 **targetattr** 关键字的 ACI 仅适用于影响完整条目（如 **add** 或 **delete** ）的权限。

要分隔使用 **targetattr** 关键字的目标规则中的多个属性，请使用 || ：

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

表达式中设置的属性必须在架构中定义。

表达式中指定的属性适用于创建 ACI 的条目以及如果未通过进一步目标规则限制的所有条目。

例 1.3. 使用 targetattr 关键字

要启用存储在 **dc=example,dc=com** 中的用户，且所有子尝试更新他们自己的条目中的 **userPassword** 属性，请输入：

```
# 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");)
```

使用 targetattr 关键字的通配符

使用 * 通配符字符，您可以针对所有属性：

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

警告

出于安全考虑，不要将通配符与 **targetattr** 一起使用，因为它允许访问所有属性，包括操作属性。例如，如果用户可以添加或修改所有属性，用户可以创建额外的 ACI 并增加自己的权限。

1.7.4. 使用 LDAP 过滤器目标条目和属性

要针对与特定条件匹配的一组条目，请使用 **targetfilter** 关键字及 LDAP 过滤器：

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

过滤器表达式是标准 LDAP 搜索过滤器。

例 1.4. 使用 targetfilter 关键字

要为 **cn=Human Resources,dc=example,dc=com** 组的成员授予权限，以修改将 department 属性设为 **Engineering** 或 **Sales** 的所有条目：

```
# 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 关键字针对整个条目。如果您将其与 **targetattr** 关键字合并，则访问控制指令(ACI)只适用于目标条目的子集。请参阅目标与过滤器匹配的条目的特定属性。

注意

当目标在目录中分发的条目和属性时，使用 LDAP 过滤器很有用。但是，结果有时不可预测，因为过滤器不会直接命名管理访问的对象。当添加或删除属性时，由过滤 ACI 的目标条目集合可能会改变。因此，如果您在 ACI 中使用 LDAP 过滤器，请验证它们是否在同一过滤器中针对正确的条目和属性，例如在 **ldapsearch** 操作中。

使用 targetfilter 关键字的通配符

targetfilter 关键字支持与标准 LDAP 过滤器类似。例如，要将值以 **adm** 开头的所有 uid 属性为目标，请使用：

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

1.7.5. 使用 LDAP 过滤器目标属性值

您可以使用访问控制来针对特定属性值。这意味着，如果该属性的值满足访问控制指令(ACI)中定义的条件，可以向属性授予或拒绝权限。基于属性值授予或拒绝访问权限的 ACI 称为基于值的 ACI。这只适用于 **ADD** 和 **DEL** 操作。您不能根据特定值限制搜索权利。

要创建基于值的 ACI，请使用以下语法使用 **targattrfilters** 关键字：

- 对于带有单个属性和过滤组合的一个操作：

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

- 对于带有多个属性和过滤组合的一个操作：

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

- 对于两个操作，各自具有多个属性并过滤组合：

  ```
  (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 ")
  ```

在前面的语法示例中，您可以设置操作 **来添加** 或 **del**。**attribute:filter** 组合设置过滤器和过滤器所应用到的属性。

下面描述了过滤器如何匹配：

- 当创建条目和过滤器应用到新条目中的属性时，该属性的每个实例都必须与过滤器匹配。
- 当删除条目和过滤器应用到条目中的属性时，该属性的每个实例还必须与过滤器匹配。

- 当修改条目和操作时，操作会添加一个属性，则应用到该属性的 **add** 过滤器必须匹配。
- 如果操作删除某个属性，则应用到该属性的 **del** 过滤器必须匹配。如果条目中已存在某个属性的单独值已被替换，则 **add** 和 **del** 过滤器都必须匹配。

例 1.5. 使用 targattrfilters 关键字

要创建 ACI，允许用户在其自己的条目中添加任何角色，但 **Admin** 角色除外，以及添加 电话 属性（只要值以 **123** 前缀开头），请输入：

```
# 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. 目标源和目标 DN

在某些情况下，管理员希望允许用户移动目录条目。在访问控制指令(ACI)中使用 **target_from** 和 **target_to** 关键字，您可以在不启用用户的情况下指定操作的源和目的地：

- 要从 ACI 中设定的不同源移动条目。
- 要将条目移动到 ACI 中设定的不同目的地。
- 从源 Distinguished Name(DN)中删除现有条目。
- 将新条目添加到目标 DN 中：

例 1.6. 使用 target_from 和 target_to 关键字

要启用 **uid=user,dc=example,dc=com** 帐户，将用户帐户从 **cn=staging,dc=example,dc=com** 条目移到 **cn=Person,dc=example,dc=com**，请输入：

```
# 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 仅适用于定义它们的子树。在上例中，ACI 仅应用到 **dc=example,dc=com** 子树。

如果没有设置 **target_from** 或 **target_to** 关键字，则 ACI 与任何源或目的地匹配。

1.8. 目标规则的高级使用

通过组合多个关键字，您可以创建复杂的目标规则。本节提供了目标规则的高级用法示例。

1.8.1. 委派权限以创建和维护组

在某些情况下，管理员希望将权限委派给其他帐户或组。通过组合目标关键字，您可以创建解决此请求的安全访问控制指令(ACI)。

例 1.7. 委派权限以创建和维护组

要启用 uid=user,ou=People,dc=example,dc=com" 帐户以在 ou=groups,dc=example,dc=com 条目中创建和更新组：

```
# 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");)
```

为安全起见，前面的示例会添加某些限制。**uid=test,ou=People,dc=example,dc=com** 用户：

- 可以创建必须包含 **top** 和 **groupOfUniqueNames** 对象类的对象。
- 无法添加额外的对象类，如 **account**。例如，如果您使用 Directory 服务器帐户进行本地身份验证，这可以防止使用无效用户 ID 创建新用户，如 **root** 用户的 **0**。

targetfilter 规则确保 ACI 条目只适用于包含 **groupofuniquenames** 对象类和 **targetattrfilter** 规则的条目，确保无法添加其他对象类。

1.8.2. 将条目和属性都作为目标

目标 根据可分辨的名称(DN)控制访问权限。但是，如果您将其与通配符和 **targetattr** 关键字结合使用，您可以同时处理条目和属性。

例 1.8. 将条目和属性都作为目标

要启用 **uid=user,ou=People,dc=example,dc.com** 用户，在 **dc=example,dc=com** 子树中读取和搜索所有组织单元中的组的成员：

```
# 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. 针对与过滤器匹配的条目的特定属性

如果您在两个目标规则中组合 **targetattr** 和 **targetfilter** 关键字，您可以针对与过滤器匹配的条目的特定属性。

例 1.9. 针对与过滤器匹配的条目的特定属性

要允许 **cn=Engineering Admins,dc=example,dc=com** 组的成员修改 **jpegPhoto** 和 **manager** 属性，并将 **department** 属性设置为 **Engineering**，请输入：

```
# 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. 以单一目录条目为目标

要针对单个目录条目，请组合 **targetattr** 和 **targetfilter** 关键字。

例 1.10. 以单一目录条目为目标

要启用 uid=user,ou=People,dc=example,dc=com 用户，以便在 ou=Engineering,dc=example,dc=com 条目中读取和搜索 ou=Engineering,dc=example,dc=com 条目：

```
# 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");)
```

要使前面的示例仅限制为目标，**ou=Engineering,dc=example,dc=com** 条目，**ou=Engineering,dc=example,dc=com** 中的 sub-entries in ou=Engineering,dc=com 不得将 **ou** =Engineering 设为 **Engineering**。

重要

如果您的目录结构更改，这些类型的 ACI 可能会失败。

或者，您可以创建一个与绑定请求中用户输入匹配的绑定规则，以及存储在目标条目中的属性值。请参阅根据值匹配，请参阅定义访问。

1.9. 定义 ACI 权限

权限规则定义了与访问控制指令(ACI)关联的权限，以及是否允许或拒绝访问。

在 ACI 中，以下突出显示的部分是权限规则：

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

1.9.1. 权限规则的语法

权限规则的一般语法是：

> *permission* (*rights*)

- **权限** ：如果访问控制指令(ACI)允许或拒绝权限，请设置。
- **权限** ：设置 ACI 允许或拒绝的权限。请参阅 权限规则 中的用户权限。

> **例 1.11. 定义权限**
>
> 要启用存储在 **ou=People,dc=example,dc=com** 条目中的用户，以搜索和显示他们自己的条目中的所有属性：
>
> ```
> # 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. 权限规则中的用户权限

权限规则的权限定义了授予或拒绝什么操作。在 ACI 中，您可以设置以下一个或多个权利：

表 1.1. 用户权限

right	描述
读取	设定用户是否可以读取目录数据。这个权限只适用于 LDAP 中的搜索操作。
write	通过添加、修改或删除属性来设置用户是否可以修改条目。此权限适用于 LDAP **中的修改和 修改操作**。
add	设置用户是否可以创建条目。这个权限只适用于 LDAP 中的 **add** 操作。
delete	设置用户是否可以删除条目。这个权限只适用于 LDAP **中的** 删除操作。

right	描述
search	设置用户是否可以搜索目录数据。要查看搜索结果中返回的数据，请分配 **search** 和 **read** 权限。这个权限只适用于 LDAP 中的搜索操作。
compare	设定用户是否可以将提供的数据与目录中存储的数据进行比较。对于 **比较** 权利，目录会返回成功或失败消息以响应静默，但用户无法看到 entry 或 属性的值。这个权限只适用于 LDAP 中比较操作。
selfwrite	设置用户是否可以向组群添加或删除自己的可分辨名称(DN)。右侧仅用于组管理。
proxy	设定指定的 DN 是否可以使用另一个条目权限访问目标。**代理** 正确在 ACL 的范围内获得，而作为所授予的用户或组可以作为任何 Directory Server 用户运行命令。您不能限制特定用户的 **代理** 权限。为安全起见，请设置在目录最目标级别使用代理的 ACI。
all	设置所有权利，除了 **代理** 之外。

1.9.3. LDAP 操作所需的权限

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

- 添加一个条目：
  - 为要添加的条目授予 **add** 权限。
  - 为条目中的每个属性值授予 **写入权限**。默认情况下会授予这个权利，但可以使用 **targattrfilters** 关键字对其进行限制。
- 删除条目：
  - 为要删除的条目授予 **delete** 权限。
  - 为条目中的每个属性值授予 **写入权限**。默认情况下会授予这个权利，但可以使用 **targattrfilters** 关键字对其进行限制。
- 修改条目中的属性：
  - 授予属性类型 **的写入权限**。
  - 授予每个属性类型的值 **的写入权限**。默认情况下会授予这个权利，但可以使用 **targattrfilters** 关键字对其进行限制。
- 修改条目的 RDN：
  - 授予条目 **的写入权限**。
  - 在新的 RDN 中使用的属性类型上授予 **写入权限**。
  - 如果要授予删除旧 RDN 的权利，请为旧 RDN 中使用的属性类型授予 **写入权限**。
  - 为新的 RDN 中使用的属性值授予 **写入权限**。默认情况下会授予这个权利，但可以使用 **targattrfilters** 关键字对其进行限制。

- 比较属性值：
  - 授予属性类型 **的比较** 权限。
- 搜索条目：
  - 为搜索过滤器中使用的每个属性类型授予**搜索**权限。
  - 授予条目中使用的属性类型 **的读取权限**。

1.10. 定义 ACI 绑定规则

访问控制指令(ACI)中的绑定规则定义必须满足所需的绑定参数，以便目录服务器应用 ACI。例如，您可以基于以下方法设置绑定规则：

- DNS
- 组成员资格或分配角色
- 条目要从中绑定的位置
- 在绑定过程中必须使用的验证类型
- 绑定发生的时间或天

在 ACI 中，以下突出显示的部分是绑定规则：

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

1.10.1. 绑定规则的语法

绑定规则的一般语法是：

> *keyword comparison_operator* "*expression*"

- **关键字** ：设置 bind 操作的类型。
- **comparison_operator**: 有效值为 **=** 和 **!=**，并指明目标是表达式中指定的对象。如果关键字支持额外的比较运算符，则相应的部分将提到它。
- **表达式** ：设置表达式，且必须使用引号括起。表达式本身取决于您使用的关键字。

1.10.2. 定义基于用户的访问

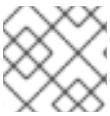
userdn 关键字允许您根据一个或多个 DN 授予或拒绝访问，并使用以下语法：

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

将表达式中的 DN 设置为：

- DN：请参阅 使用带有 userdn 关键字的 DN。
- LDAP 过滤器：请参阅在 LDAP 过滤器中使用 userdn 关键字。
- **任何人都** 别名：请参阅 授予匿名访问。

- **all** 别名：请参阅 授予经过身份验证的用户访问权限。
- **自我** 别名： 请参阅启用用户来访问他们自己的条目。
- **父** 别名： 请参阅设置用户 的子条目访问权限。



注意

不要在 LDAP URL 中指定主机名或端口号。URL 始终适用于本地服务器。

使用 userdn 关键字的 DN

将 **userdn** 关键字设置为可分辨名称(DN)，将 ACI 仅应用到匹配的条目。要匹配多个条目，请在 DN 中使用 * 通配符。

将 **userdn** 关键字与 DN 搭配使用必须匹配以下语法：

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

例 1.12. 使用 userdn 关键字的 DN

要启用 **uid=admin,ou=People,dc=example,dc=com** 用户，以读取 **ou=People,dc=example,dc=com** 条目中所有其他用户的 **manager** 属性：

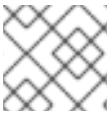
```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com -x

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";)
```

将 userdn 关键字与 LDAP 过滤器一起使用

如果要动态允许或拒绝用户的权限，请使用 **userdn** 关键字和 LDAP 过滤器：

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



注意

LDAP 过滤器支持 * 通配符。

例 1.13. 将 userdn 关键字与 LDAP 过滤器一起使用

要启用将 department 属性设为 **human Resources** 的用户，以更新 **ou=People,dc=example,dc=com** 条目中的用户的 **homePostalAddress** 属性：

```
# 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)";)
```

授予匿名访问

在某些情况下，管理员希望配置对目录中的数据的匿名访问。匿名访问意味着可以通过提供以下内容来绑定到目录：

- 没有绑定 DN 和密码
- 有效的绑定 DN 和密码

要配置匿名访问，请使用 **ldap:///anyone** 表达式在绑定规则中使用 **userdn** 关键字：

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

例 1.14. 授予匿名访问

要启用没有身份验证的任何人，请在 **ou=People,dc=example,dc=com** 条目中读取和搜索 **sn**、**givenName** 和 Tel **Number** 属性：

```
# 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")
```

授予经过身份验证的用户访问权限

在某些情况下，管理员希望向能够成功绑定到目录服务器的任何用户授予权限，但匿名绑定除外。要配置此功能，请在绑定规则中使用 **ldap:///all** 表达式以及 **userdn** 关键字：

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

例 1.15. 授予经过身份验证的用户访问权限

要启用经过身份验证的用户，在 **ou=example,ou=groups,dc=example,dc=com** 组中添加或删除自身作为成员：

```
# 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")
```

允许用户访问自己的条目

要设置允许或拒绝用户对自己条目的访问的 ACI，请在 bind 规则中使用 **ldap:///self** 表达式以及 bind 规则中的 **userdn** 关键字：

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

例 1.16. 允许用户访问自己的条目

要在 **ou=People,dc=example,dc=com** 条目中启用用户，以更新自己的 **userPassword** 属性：

```
# 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")
```

设置用户子条目的访问权限

要指定只有其绑定 DN 是目标条目的父项时，才会授予用户或拒绝对条目的访问权限，请在绑定规则中使用 **userdn** 关键字的 **self:///parent** 表达式：

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

例 1.17. 设置用户子条目的访问权限

要启用 **cn=user,ou=People,dc=example,dc=com** 用户来更新自己的子条目的 **manager** 属性，如 **cn=example,cn=user,ou=People,dc=example,dc=com**:

```
# 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. 定义基于组的访问

基于组的访问控制指令(ACI)可让您通过在组中添加或删除用户来管理访问权限。要配置基于组成员资格的 ACI，请使用 **groupdn** 关键字。如果用户是一个或多个指定组的成员，则 ACI 将匹配。

当使用 **groupdn** 关键字时，Directory 服务器会根据以下属性验证组成员资格：

- 成员

- uniqueMember
- memberURL
- memberCertificateDescription

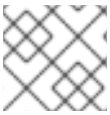
使用 **groupdn** 关键字绑定规则使用以下语法：

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

将表达式中的可分辨名称(DN)设置为：

- A DN.请参阅 使用带有 groupdn 关键字的 DN。
- LDAP 过滤器。请参阅使用 groupdn 关键字与 LDAP 过滤器

如果您在一个绑定规则中设置多个 DN，如果经过身份验证的用户是这些组的成员，则 Directory 服务器会应用 ACI。要将用户设置为多个组的成员，请使用多个 **groupdn** 关键字，并使用 Boolean **和** operator 合并它们。详情请参阅 使用布尔值 Operator 组合绑定规则。



注意

不要在 LDAP URL 中指定主机名或端口号。URL 始终适用于本地服务器。

使用带有 groupdn 关键字的 DN

要将 ACI 应用到组的成员，请将 **groupdn** 关键字设置为组的 DN。

groupdn 关键字设置为 DN 使用以下语法：

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

例 1.18. 使用带有组关键字的 DN

要启用 **cn=example,ou=Groups,dc=example,dc=com** 组的成员搜索并读取 **ou=People,dc=example,dc=com** 条目的 manager 属性：

```
# 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";)
```

使用 groupdn 关键字及 LDAP 过滤器

使用带有 **groupdn** 关键字的 LDAP 过滤器，您可以定义经过身份验证的用户必须是过滤器搜索返回的其中一个组的成员，以匹配 ACI。

带有 LDAP 过滤器的 **groupdn** 关键字使用以下语法：

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

注意

LDAP 过滤器支持 * 通配符。

例 1.19. 将 groupdn 关键字与 LDAP 过滤器一起使用

要启用 **dc=example,dc=com** 和 subtrees 中的组成员，将 **manager** 属性设为 **example**，更新 **ou=People,dc=example,dc=com** 中条目的 **homePostalAddress**。

```
# 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. 基于值匹配定义访问权限

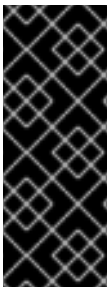
在绑定规则中使用 **userattr** 关键字来指定在用于绑定到目录和目标条目的条目之间必须匹配哪个属性。

userattr 关键字使用以下语法：

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

如需了解更多详细信息，请参阅：

- 使用 USERDN 绑定类型
- 使用 GROUPDN 绑定类型
- 使用 ROLEDN 绑定类型
- 使用 SELFDN 绑定类型
- 使用 LDAPURL 绑定类型
- 使用带有继承的用户attr 关键字



重要

默认情况下，Directory 服务器评估对创建条目的访问权限。但是，为了避免同一级别上用户对象，在使用 **userattr** 关键字时，Directory 服务器不会向条目授予设置访问控制指令 (ACI) 的 **add** 权限。要配置此行为，请结合使用 **userattr** 关键字和 **parent** 关键字，并在级别 0 上额外授予权限。

有关继承的详情，请参阅 基于值匹配定义访问权限。

使用 USERDN 绑定类型

要在绑定用户可分辨名称(DN)匹配属性中存储的 DN 时应用 ACI，请使用 **USERDN** 绑定类型。

userattr 关键字及 **USERDN** 绑定类型需要以下语法：

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

例 1.20. 使用 USERDN 绑定类型

为经理经理所有权限授予自己同事的 电话 属性：

```
# 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";)
```

如果执行 **ou=People,dc=example,dc=example,dc=com** 的一个条目中的操作的用户的 DN 与存在这个条目中的 **manager** 属性中的 DN 匹配，则上一 ACI 被评估为 true。

使用 GROUPDN 绑定类型

要在绑定用户 DN 是属性中设置的组成员时应用 ACI，请使用 **GROUPDN** 绑定类型。

带有 **GROUPDN** 绑定类型的 **userattr** 关键字需要以下语法：

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

例 1.21. 使用 GROUPDN 绑定类型

要授予用户删除他们自己在 **ou=Social Committee,ou=Groups,dc=example,dc=com** 条目下拥有的组条目的权限：

```
# 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";)
```

如果执行该操作的用户的 DN 是 **owner** 属性中指定的组的成员，则前面的 ACI 被评估为 true。

指定的组可以是动态组，组的 DN 可以是数据库的任意后缀。但是，对服务器进行此类 ACI 的评估非常密集型。

如果您使用与目标条目相同的后缀下的静态组，请使用以下表达式来提高性能：

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

使用 ROLEDN 绑定类型

要在绑定用户属于属性中指定的角色时应用 ACI，请使用 **ROLEDN** 绑定类型。

带有 **ROLEDN** 绑定类型的 **userattr** 关键字需要以下语法：

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

例 1.22. 使用 ROLEDN 绑定类型

要使用 **cn=Administrators,dc=example,dc=com** 角色启用用户，以搜索和读取 **ou=People,dc=example,dc=com** 条目的 **manager** 属性：

```
# 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;)
```

指定角色可以位于数据库中的任何后缀。如果您也正在使用过滤的角色，对 ACI 的评估会在服务器上使用大量资源。

如果您使用静态角色定义，且角色条目与目标条目相同，则使用以下表达式来提高性能：

使用 SELFDN 绑定类型

SELFDN 绑定类型允许您在条目的单值属性中设置绑定用户的 DN 时授予权限。

带有 **SELFDN** 绑定类型的 **userattr** 关键字需要以下语法：

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

例 1.23. 使用 SELFDN 绑定类型

要让用户添加 **ipatokenuniqueid=*,cn=otp,dc=example,dc=com** 条目，这些条目在 **ipatokenOwner** 属性中设置的 bind 用户 DN：

```
# 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";)
```

使用 LDAPURL 绑定类型

要在绑定 DN 与目标条目属性中指定的过滤器匹配时应用 ACL，请使用 **LDAPURL** 绑定类型。

带有 **LDAPURL** 绑定类型的 **userattr** 关键字需要以下语法：

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

例 1.24. 使用 LDAPURL 绑定类型

要为包含 **aciurl** 属性设为 **ldap:///ou=People,dc=example,dc=com?one?one?(uid=user*)的用户** 对象授予读取和搜索权限：

```
# 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);)
```

使用带有继承的用户attr 关键字

当您使用 **userattr** 关键字将用于绑定目标条目的条目关联时，ACI 仅适用于指定的目标，而不应用到下面的条目。在某些情况下，管理员希望在目标条目下扩展 ACI 的几个级别的应用程序。这可以通过使用 **parent** 关键字并指定应该继承 ACI 的目标下方的级别数量。

使用带有 **parent** 关键字 **的用户attr** 关键字时，其语法如下：

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

- **inheritance_level**: Comma- separated list，它指示目标中有多少级别继承 ACI。您可以在目标条目下面包含五个级别（**0**、**1**、**2**、**3**、**4**）。零(**0**)表示目标条目。
- **attribute_name** ：以 **userattr** 或 **groupattr** 关键字为目标的属性。
- **bind_type_or_attribute_value**:设置属性值或绑定类型，如 **USERDN**。

例如：

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

如果绑定 DN 与目标条目的 manager 属性匹配，则此绑定规则被评估为 true。当绑定规则被评估为 true 时，授予的权限适用于目标条目，并立即应用到目标条目。

例 1.25. 使用带有继承的用户attr 关键字

要让用户读取和搜索 **cn=Profiles,dc=example,dc=com** 条目，其中用户的 DN 在 **owner** 属性中设置，以及包含 **cn=mail,cn=Profiles,dc=example,dc=com** 和 **cn=news,cn=Profiles,dc=example,dc=com** 的第一级子条目：

```
# 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. 定义来自特定 IP 地址或范围的访问

绑定规则中的 **ip** 关键字可让您授予或拒绝来自特定 IP 地址或 IP 地址的访问。

带有 **ip** 关键字的绑定规则使用以下语法：

```
ip comparison_operator "IP_address_or_range"
```

例 1.26. 在绑定规则中使用 IPv4 地址范围

要拒绝从 **192.0.2.0/24** 网络到 **dc=example,dc=com** 条目的访问：

```
# 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. 在绑定规则中使用 IPv6 地址范围

要拒绝从 **2001:db8::/64** 网络到 **dc=example,dc=com** 条目的访问：

```
# 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. 定义特定主机或域的访问

绑定规则中的 **dns** 关键字允许您授予或拒绝来自特定主机或域的访问。

警告

如果目录服务器无法使用 DNS 解析连接 IP 地址到其完全限定域名(FQDN)，服务器不会应用使用此客户端的 **dns** 绑定规则的访问控制指令(ACI)。

如果无法使用 DNS 解析客户端 IP 地址，请使用 **ip** 关键字和 IP 地址。请参阅 定义来自特定 IP 地址或范围的访问权限。

使用 **dns** 关键字绑定规则使用以下语法：

```
dns comparison_operator "host_name_or_domain_name"
```

例 1.28. 定义特定主机的访问

要拒绝从 client.example.com 主机到 dc=example,dc=com 条目的访问：

```
# 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. 定义特定域的访问

要拒绝 example.com 域内所有主机对 dc=example,dc=com 条目的访问：

```
# 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. 在连接中需要一定级别的安全性

连接的安全性由其安全强号(SSF)决定，它设定了处理操作所需的最低关键强度。在绑定规则中使用 **ssf** 关键字，您可以设置连接必须使用一定级别的安全性。这可让您强制操作，例如密码更改，通过加密连接来执行。

任何操作的 SSF 的值是 TLS 连接和 SASL 绑定之间的高值。这意味着，如果服务器被配置为通过 TLS 运行，并且为 SASL/GSSAPI 配置复制协议，则操作的 SSF 是哪些可用加密类型更为安全。

使用 **ssf** 关键字绑定规则使用以下语法：

-

```
ssf comparison_operator key_strength
```

您可以使用以下比较运算符：

- **=**（等同于）
- **!**（不等于）
- **&** lt;（不超过）
- **&** gt;（多于）
- **abrt**（不等于或等于）
- **>**=（等于或等于）

如果将 **key_strength** 参数设置为 **0,** 则 LDAP 操作不需要安全操作。

例 1.30. 在连接中需要一定级别的安全性

要配置 dc=example,dc=com 条目中的用户，只能在 SSF 为 128 或更高版本时更新他们的 userPassword 属性：

```
# 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. 定义特定星期几的访问

绑定规则中的 **dayofweek** 关键字允许您根据星期几授予或拒绝访问。

注意

目录服务器使用服务器上的时间来评估访问控制指令(ACI)；而不是客户端上的时间。

使用 **dayofweek** 关键字绑定规则使用以下语法：

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

例 1.31. 授予对特定星期天的访问权限

要拒绝 **uid=user,ou=People,dc=example,dc=com** 用户条目的访问，以绑定到 Saturdays 和 Sundays 中的服务器：

```
# 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. 在特定时间定义访问

绑定规则中的 **timeofday** 关键字允许您根据当日来授予或拒绝访问。

注意

目录服务器使用服务器上的时间来评估访问控制指令(ACI)；而不是客户端上的时间。

使用 **timeofday** 关键字绑定规则使用以下语法：

```
timeofday comparison_operator "time"
```

您可以使用以下比较运算符：

- **=**（等同于）
- **!**（不等于）
- **&** lt;（不超过）
- **&** gt;（多于）
- **abrt**（不等于或等于）
- **>**=（等于或等于）

重要

timeofday 关键字要求您以 24 小时格式指定时间。

例 1.32. 在一天的特定时间定义访问

要拒绝 **uid=user,ou=People,dc=example,dc=com** 用户条目的访问，以便在 6pm 和 0am 之间绑定到服务器：

```
# 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. 根据验证方法定义访问权限

bind 规则中的 **authmethod** 关键字会设置客户端在连接到服务器时必须使用的身份验证方法，以应用访问控制指令(ACI)。

使用 **authmethod** 关键字绑定规则使用以下语法：

> authmethod *comparison_operator* "*authentication_method*"

您可以设置以下验证方法：

- **none** ：不需要身份验证，代表匿名访问。这是默认值。
- **simple** ：客户端必须提供要绑定到目录的用户名和密码。
- **SSL** ：客户端必须在数据库、智能卡或其他设备中使用 TLS 证书绑定到目录。有关基于证书的身份验证的详情，请参考 基于验证方法定义访问权限。
- **SASL** ：客户端必须通过简单验证和安全层(SASL)连接绑定到 目录。当您在绑定规则中使用这个验证方法时，还指定 SASL 机制，如 **EXTERNAL**。

> **例 1.33. 仅对使用 EXTERNAL SASL 身份验证方法的连接启用访问**
>
> 如果连接没有使用基于证书的验证方法或 SASL，则拒绝对服务器的访问：
>
> ```
> # 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. 基于角色定义访问权限

绑定规则中的 **roledn** 关键字允许您授予或拒绝对拥有一个或多个角色集的用户的访问。

注意

红帽建议使用组而不是角色。

使用 **roledn** 关键字绑定规则使用以下语法：

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

如果可分辨的名称(DN)包含逗号，请使用反斜杠转义逗号。

> **例 1.34. 基于角色定义访问权限**
>
> 要启用在 **nsRole** 属性中设置的 **cn=Human Resources、ou=People,dc=example,dc=com** 角色的用户，以搜索并读取 **ou=People,dc=example,dc=com** 中条目的 **manager** 属性：

```
# 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. 使用布尔值运算符组合绑定规则

在创建复杂的绑定规则时，可以使用 **AND**、**OR** 和 **NOT** 布尔值运算符组合多个关键字。

绑定规则与布尔值运算符组合有以下语法：

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

例 1.35. 使用布尔值运算符组合绑定规则

要配置是 **cn=Administrators,ou=Groups,dc=example,com** 和 **cn=Operators,ou=Groups,dc=example,com] 组的成员 [command]'read , search,add,update, delete** entries in **ou=People,dc=example,dc=com**:

```
# 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";)
```

目录服务器评估布尔值运算符的方式

目录服务器使用以下规则评估布尔值运算符：

- 所有表达式（从左到右）
  在以下示例中，首先评估 **bind_rule_1** ：

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

- 从左边到最外围的父表达式首先。
  在以下示例中，先评估 **bind_rule_2**，再评估 **bind_rule_3** 秒：

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

- **不早于 AND** 或 **OR** 运算符。
  在以下示例中，首先评估 *bind_rule_2* ：

  -

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

AND 和 **OR** 运算符没有优先级顺序。

第 2 章 使用宏访问控制说明

宏访问控制指令(ACI)可让您自动执行 LDAP 条目可区分名称(DN)或其部分并减少 ACI 的数量。

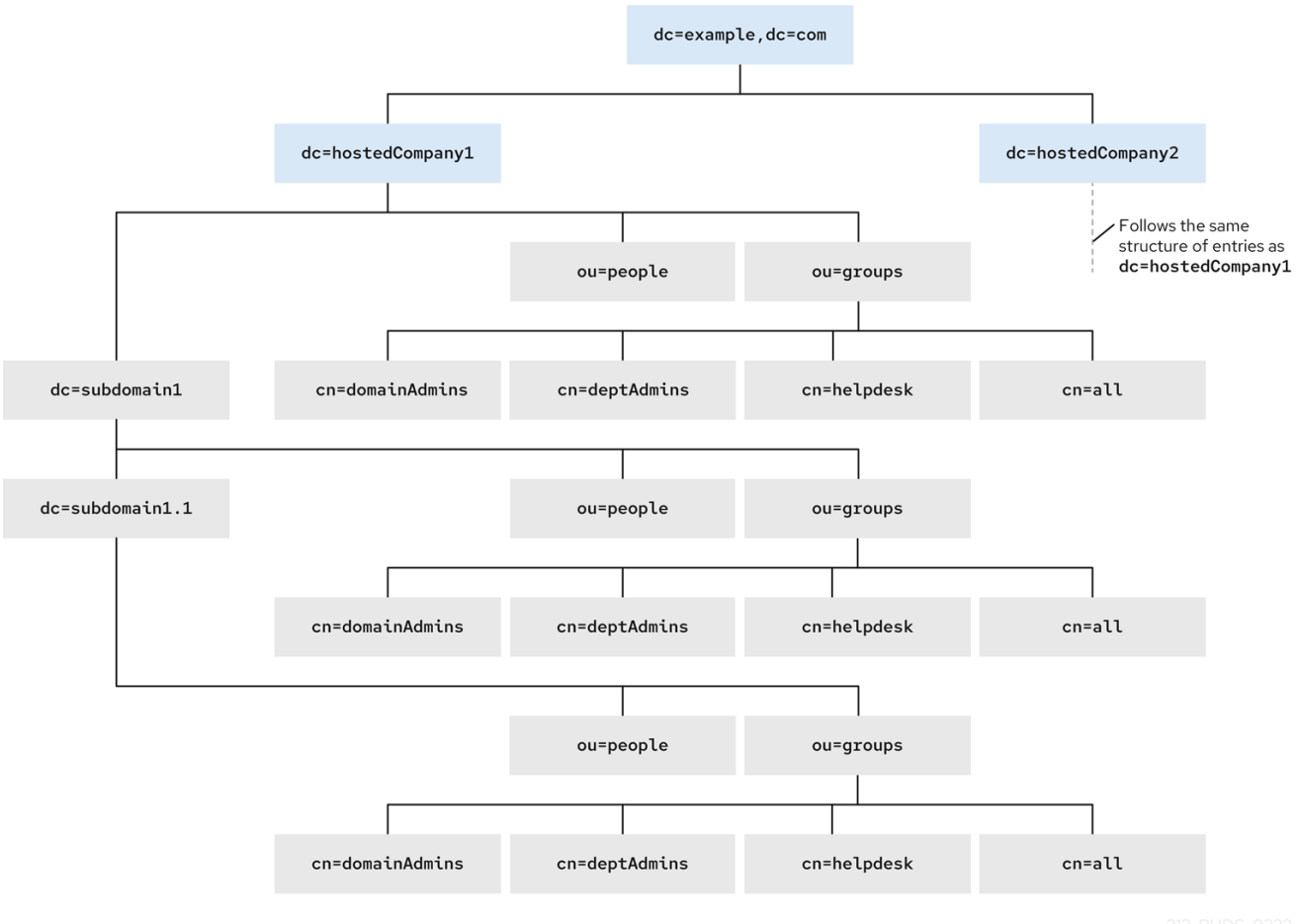
2.1. 宏访问控制指令示例

下图显示了带有后缀 **dc=hostedCompany1,dc=example,dc=com** 和 **dc=hostedCompany2,dc=example,dc=com** 的目录树，其中包含子域的重复模式。每个子域具有与 **ou=groups**,**ou=people** 条目相同的结构。目录树使用宏访问控制指令(ACI)来减少 ACI 总数。

在目录树中应用的 ACI 也具有重复模式。例如，以下 ACI 位于 **dc=hostedCompany1,dc=example,dc=com** 节点上，并将 **DomainAdmins** 组的访问权限授予该树中的任何条目：

```
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";)
```

图 2.1. 宏 ACI 示例的目录树



以下 ACI 在 **groupdn** 关键字中显示 DN 的不同部分：

- **dc=hostedCompany1,dc=example,dc=com** 节点包含以下 ACI：

```
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** 节点包含以下 ACI：

```
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** 节点包含以下 ACI：

```
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** 节点包含以下 ACI：

```
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";)
```

使用宏替换重复模式的多个 ACI。例如，要将以上 ACI 减少为一，请使用以下宏：

```
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. 宏访问控制指令语法

宏访问控制指令(ACI)包括以下类型的表达式来替换 DN 或 DN 的一部分：

- **($DN)**,
- **[$DN]**,
- **($attr.*attrName*)**，其中 *attrName* 代表一个属性，这是目标条目的一部分。

ACI 关键字提供绑定凭证，这些凭证是 ACI 的主题。主题决定 ACI 应用的位置。

表 2.1. ACI 关键字的宏

Macro	ACI 关键字	描述

Macro	ACI 关键字	描述
($DN)	**target, targetfilter, userdn, roledn, groupdn, userattr**	在主体中匹配和直接替换。它将匹配 **target** 或 **targetfilter**，并将匹配的值替换为 **userdn**、**groupdn** 或 **userattr**。
[$DN]	**targetfilter, userdn, roledn, groupdn, userattr**	在主题的子树中替换多个 RDN。
($attr.*attrName*)	**userdn, roledn, groupdn, userattr**	将 **attributeName** 属性值替换为主题。

请注意，如果您使用任何宏，您必须定义包含 **($dn)** 宏的目标。您可以组合 **($dn)** 和 **($attr.*attrName*)** 宏。

2.3. ($DN)宏示例

($dn) 宏将替换值与 LDAP 请求中的条目进行比较。例如，LDAP 请求以条目为目标：

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

ACI 定义以下目标：

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

($dn) 宏与本例中的 **dc=subdomain1,dc=hostedCompany1** 匹配。

当 ACI 的主题使用 **($dn)** 宏时，与目标匹配的子字符串会展开主题：

```
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 会展开，如下所示：

```
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";)
```

扩展宏后，红帽目录服务器会根据正常过程评估 ACI，以确定是否授予访问权限。

2.4. [$DN] 宏示例

[$dn] 宏检查目标源的 DN 多次。这个宏会丢弃每个迭代最多的 RDN 组件，直到找到匹配项。

例如，您在 **cn=all,ou=groups,dc=subdomain1,dc=hostedCompany1,dc=example,dc=com** 子树中使用目标的 LDAP 请求：

```
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";)
```

宏展开，如下所示：

1. 目标中的 **($dn)** 与 **dc=subdomain1,dc=hostedCompany1** 匹配。
2. 主体中的 **[$dn]** 的替换为 **dc=subdomain1,dc=hostedCompany1**。
   结果为
   groupdn="ldap:///cn=domainAdmins,ou=Groups,dc=subdomain1,dc=hostedCompany1,dc=example,dc=com"。如果绑定 DN 是该组的成员，则匹配的进程会停止，并且评估 ACI。如果结果不匹配，该过程将继续并丢弃最左边的部分。
3. 主题中的 [$dn] 是 **dc=hostedCompany1**。
   结果为
   groupdn="ldap:///cn=domainAdmins,ou=Groups,dc=hostedCompany1,dc=example,dc=com"。如果绑定 DN 不是该组的成员，则不会评估 ACI。如果是一个成员，则评估 ACI。

[$dn] 宏将对域级别管理员的访问权限授予目录树中的所有子域。它可用于表达域之间的分层关系。例如，请考虑以下 ACI：

```
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";)
```

此 ACI 将 **cn=domainAdmins,ou=groups,dc=hostedCompany1,dc=example,dc=com** 的访问权限授予 **dc=hostedCompany1** 下的所有子域。属于该组成员的管理员可以访问 **ou=body,dc=subdomain1.1,dc=subdomain1** 等子树。但是 **cn=domainAdmins,ou=groups,dc=subdomain1.1** 的成员无法访问 **ou=body,dc=hostedCompany1** 和 **ou=body,dc=subdomain1,dc=hostedCompany1** 节点。

2.5. ($ATTR.ATTRNAME)宏示例

您始终使用 **($attr.*attrName*)** 宏作为 DN 的一部分。例如，定义 **以下角色dn** ：

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

假设服务器收到以下条目的目标 LDAP 操作：

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

要评估 ACI 的 **roledn** 部分，服务器会在目标条目中查看 **ou** 属性，并使用此属性的值来扩展宏。**roledn** 扩展如下：

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

Red Hat Directory Server 根据正常的 ACI 评估算法评估 ACI。

如果属性有多个值，RHDS 使用每个值来扩展宏，并使用第一个成功与扩展宏匹配的值。例如：

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

当 Red Hat Directory 服务器评估 ACI 时，它会在以下扩展表达式上执行逻辑 **OR** ：

```
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 在 LDAP 浏览器中管理访问控制指令

这组指令可让您使用 web 控制台中的 LDAP 浏览器向导来管理访问控制指令(ACI)的基础知识。

3.1. 在 LDAP 浏览器中创建访问控制指令

您可以使用 web 控制台中的 **LDAP 浏览器** 为红帽目录服务器(RHDS)条目创建并添加访问控制指令(ACI)。

先决条件

- 访问 Web 控制台。
- Red Hat Directory Server 中存在父条目。

流程

1. 登录到 Web 控制台并点 **Red Hat Directory Server**。
2. Web 控制台加载 **Red Hat Directory Server** 界面后，单击 **LDAP 浏览器**。
3. 选择 LDAP 条目并点击 Options 菜单。
4. 在下拉菜单中选择 ACIs。
5. 要使用 LDAP 浏览器向导创建 ACI，有两个选项：
   a. 点 **Add ACI Wizard** 使用向导创建 ACI。继续下一步。
   b. 点 **Add ACI** manually，在文本字段中指定指令，然后点 **Save ACI**。
6. 按照向导中的步骤，在完成每个步骤后点 **Next** 按钮。
7. 要创建 ACI，请查看向导生成的数据，然后点 **Add ACI**。
8. 要关闭向导窗口，点 **Finish** 按钮。

验证

- 验证新的 ACI 出现在 **Manage ACIs** 窗口中。

3.2. 在 LDAP 浏览器中编辑访问控制指令

您可以使用 web 控制台中的 **LDAP 浏览器 管理 ACI 窗口来编辑 Red Hat Directory Server 条目的访问控制指令(ACI)**。

先决条件

- 访问 Web 控制台。
- Red Hat Directory Server 中存在父条目。

流程

1. 登录到 Web 控制台并点 **Red Hat Directory Server**。
2. Web 控制台加载 **Red Hat Directory Server** 界面后，单击 **LDAP 浏览器**。
3. 选择 LDAP 条目并点击 Options 菜单。
4. 从下拉菜单中选择 ACIs。
5. 点 Options 菜单并选择 **Edit ACI**。
6. 修改文本字段中的指令并点 **Save ACI**。

验证

- 在 **Manage ACIs** 窗口中，展开您修改的 ACI 并观察您的更改。

3.3. 在 LDAP 浏览器中删除访问控制指令

您可以使用 web 控制台中的 **LDAP 浏览器** 删除 Red Hat Directory Server 条目的访问控制指令(ACI)。

先决条件

- 访问 Web 控制台。
- Red Hat Directory Server 中存在父条目。

流程

1. 登录到 Web 控制台并点 **Red Hat Directory Server**。
2. Web 控制台加载 **Red Hat Directory Server** 界面后，单击 **LDAP 浏览器**。
3. 选择 LDAP 条目并点击 Options 菜单。
4. 从下拉菜单中选择 **ACIs** 打开 **Manage ACIs** 窗口。
5. 点您要删除的 ACI 的 Node 选项图标，然后选择 **Remove ACI**。
6. 选择 **Yes, I'm sure** 复选框，然后点 **Delete ACI** 按钮。

验证

- 在 **Manage ACIs** 窗口中，验证您删除的 ACI 不再出现在 ACI 列表中。

第 4 章 配置基于密码的帐户锁定策略

基于密码的帐户锁定策略可防止攻击者重复尝试猜测用户的密码。您可以配置帐户锁定策略，以便在指定失败尝试绑定后锁定用户帐户。

如果配置了基于密码的帐户锁定策略，Directory 服务器会在以下用户条目的属性中维护锁定信息：

- **passwordRetryCount** ：存储绑定尝试失败的数量。如果稍后用户成功绑定到目录，目录服务器会重置该值，而不是 **retryCountResetTime** 中的时间。用户第一次绑定失败时会出现此属性。
- **retryCountResetTime**: 保存重置 **passwordRetryCount** 属性的时间。用户第一次绑定失败时会出现此属性。
- **accountUnlockTime** ：存储用户帐户被解锁的时间。此属性在帐户第一次锁定后存在。

4.1. 配置在达到或超过配置的最大尝试时是否锁定帐户

当 Directory 服务器在登录失败时锁定帐户时，管理员可以配置以下行为之一：

- 如果超过限制，服务器会锁定帐户。例如，如果将限制设定为 3 次，则发生锁定在第四次尝试后 (**n+1**)。这也意味着，如果第四个尝试成功，Directory 服务器不会锁定帐户。
  默认情况下，Directory 服务器会使用传统 LDAP 客户端通常预期的旧密码策略。
- 如果达到限制，服务器会锁定帐户。例如，如果限制设定为 3 次，服务器会在第三个尝试后锁定帐户(**n**)。
  现代 LDAP 客户端通常会预期此行为。

这个步骤描述了如何禁用旧的密码策略。更改策略后，Directory 服务器会阻止登录尝试达到配置的限制的用户。

先决条件

- 已配置了帐户锁定策略。

流程

- 要禁用旧的密码策略并在达到限制时锁定帐户，请输入：

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

验证

1. 显示 **passwordmaxfailure** 设置的值：

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

2. 尝试绑定使用无效密码超过 **passwordmaxfailure** 中设置的值：

```
# 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.
```

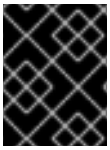
禁用旧的密码后，Directory 服务器会在第二次尝试后锁定帐户，进一步尝试会通过 **ldap_bind: Constraint violation(19)** 错误阻止。

其他资源

- 使用命令行配置基于密码的帐户锁定策略

4.2. 使用命令行配置基于密码的帐户锁定策略

要阻止带有无效密码的登录重复绑定尝试，请配置基于密码的帐户锁定策略。



重要

Directory 服务器在达到时或超过配置的最大尝试时是否锁定帐户取决于传统密码策略设置。

流程

1. 可选：识别旧的密码策略是启用还是禁用：

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

2. 启用密码锁定策略并将最大失败数设置为 **2**：

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

启用旧的密码策略后，Directory 服务器会在第三个失败尝试绑定后锁定帐户（**--pwdmaxfailures** 参数 + 1）。

dsconf pwpolicy set 命令支持以下参数：

- **--pwdlockout** ：启用或禁用帐户锁定功能。默认： **off**。
- **--pwdmaxfailures** ：设置在 Directory 服务器锁定帐户前允许的最大尝试失败尝试数。默认： **3**。
  请注意，如果启用了旧密码策略设置，这个锁定会在以后尝试。默认： **3**。

- **--pwdresetfailcount** ：在 Directory 服务器重置用户条目中的 **passwordRetryCount** 属性之前设置时间（以秒为单位）。默认值： **600** 秒（10 分钟）。
- **--pwdlockoutduration**: 设置帐户的锁定时间（以秒为单位）。如果您将 **--pwdunlock** 参数设置为 **off**，则忽略此参数。默认值： **3600** 秒（1 小时）。
- **--pwdunlock** ：启用或禁用锁定帐户在一定时间后是否应解锁，或者保持禁用状态，直到管理员手动解锁它们。默认： **上的**。

验证

- 尝试绑定使用无效的密码两次，超过您在 **--pwdmaxfailures** 参数中设置的值：

```
# 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.
```

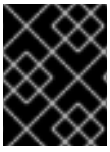
启用旧的密码后，Directory 服务器会在超过限制后锁定帐户，并使用 **ldap_bind: Constraint violation(19)** 错误阻止进一步尝试。

其他资源

- 配置旧的密码策略

4.3. 使用 WEB 控制台配置基于密码的帐户锁定策略

要阻止带有无效密码的登录重复绑定尝试，请配置基于密码的帐户锁定策略。



重要

Directory 服务器在达到时或超过配置的最大尝试时是否锁定帐户取决于传统密码策略设置。

先决条件

- 在 web 控制台中登录到实例。

流程

1. 可选：识别旧的密码策略是启用还是禁用：

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

   此设置在 web 控制台中不可用。

2. 导航到 **Database → Password Policies → Global Policy → Account Lockout**。

3. 选择 **"启用帐户锁定** "。

4. 配置锁定设置：

   - **锁定帐户的失败登录数量** ：在 Directory 服务器锁定帐户前设置允许的尝试的最大失败尝试数。
   - time **Until Failure Count Resets** ：在 Directory Server 重置用户条目中的 **passwordRetryCount** 属性之前设置时间（以秒为单位）。
   - **时间不锁定** ：设置帐户锁定的时间（以秒为单位）。如果您永久禁用 **Do Not Lockout Account**，则此参数将被忽略。
   - **不要锁定帐户** ：启用或禁用锁定帐户在一定时间后是否应解锁，或者保持禁用状态，直到管理员手动解锁它们。

5. 点 **Save**。

验证

- 尝试绑定，使用无效密码两次超过您在 **Number of Failed Logins 中设定的值，即锁定帐户** ：

```
# 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.
```

启用旧的密码后，Directory 服务器会在超过限制后锁定帐户，并使用 **ldap_bind: Constraint violation(19)** 错误阻止进一步尝试。

其他资源

- 配置旧的密码策略

第 5 章 配置基于时间的帐户锁定策略

您可以使用 Account Policy 插件配置不同的基于时间的锁定策略，例如：

- 自动禁用帐户最后一次成功登录的时间一定时间
- 在创建帐户后自动禁用帐户
- 在密码过期后自动禁用帐户一定时间
- 在帐户不活跃和密码过期时自动禁用帐户

5.1. 在最后成功登录后自动禁用帐户一定时间

按照以下步骤配置基于时间的锁定策略，该策略会在 **dc=example,dc=com** 条目下的用户中激活，且不能为超过 21 天登录。

此帐户不活动功能，例如，如果员工留有公司，并且管理员忘记删除帐户，而目录服务器在一定时间后激活帐户。

流程

1. 启用帐户策略插件：

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

2. 配置插件配置条目：

```
# 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
```

这个命令使用以下选项：

- **--always-record-login yes** ：启用登录时间的日志记录。这需要使用带帐户策略的类服务(CoS)或角色，即使它没有设置 **acctPolicySubentry** 属性。
- **--state-attr lastLoginTime** ：配置帐户策略插件会在用户的 **lastLoginTime** 属性中存储最后一次登录时间。
- **--Alt-state-attr 1.1** ：禁用使用替代属性来检查主属性是否不存在。默认情况下，Directory 服务器使用 **createTimestamp** 属性作为替代方案。但是，这会导致如果其帐户没有设置 **lastLoginTime** 属性并且 **createTimestamp** 比配置的不活跃周期更旧，Directory 服务器会自动记录现有用户。禁用 alternative 属性会导致 Directory 服务器在下次登录时自动将 **lastLoginTime** 属性添加到用户条目中。
- **--spec-attr acctPolicySubentry**: Configures Directory Server 应用策略到设置了 **acctPolicySubentry** 属性的条目。您可以在 CoS 条目中配置此属性。
- **--limit-attr accountInactivityLimit**: Configures that the **accountInactivityLimit** 属性 in inivation policy entry store the inactive time.

3. 重启实例：

```
# dsctl instance_name restart
```

4. 创建激活策略条目的帐户：

```
# 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 属性中的值配置在最后一次登录后，目录服务器处于激活的 **1814400** 秒（21 天）。

5. 创建 CoS 模板条目：

```
# 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
```

此模板条目引用帐户正在激活策略。

6. 创建 CoS 定义条目：

```
# 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
```

此定义条目引用 CoS 模板条目，并导致 **acctPolicySubentry** 属性出现在每个用户条目中，值设为 **cn=Account Inactivation Policy,dc=example,dc=com**。

验证

1. 将用户的 **lastLoginTime** 属性设置为早于您配置的不活跃时间的值：

```
# 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. 尝试以这个用户身份连接到该目录：

```
# 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.
```

如果目录服务器拒绝访问并返回这个错误，则帐户不活跃工作。

其他资源

- 重新启用达到不活跃限制的帐户

5.2. 在创建帐户后自动禁用帐户一定时间

按照以下步骤配置 **dc=example,dc=com** 条目中的帐户在管理员创建 60 天后过期。

使用帐户到期功能，例如，确保外部 worker 的帐户在创建时间被锁定。

流程

1. 启用帐户策略插件：

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

2. 配置插件配置条目：

```
# 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
```

这个命令使用以下选项：

- **--always-record-login yes** ：启用登录时间的日志记录。这需要使用带帐户策略的类服务(CoS)或角色，即使它没有设置 **acctPolicySubentry** 属性。
- **--state-attr createTimestamp**: Configures that the Account Policy 插件使用 **createTimestamp** 属性的值来计算帐户是否已过期。
- **--Alt-state-attr 1.1** ：禁用使用替代属性来检查主属性是否不存在。
- **--spec-attr acctPolicySubentry**: Configures Directory Server 应用策略到设置了 **acctPolicySubentry** 属性的条目。您可以在 CoS 条目中配置此属性。
- **--limit-attr accountInactivityLimit** ：配置帐户到期策略条目中的 **accountInactivityLimit** 属性存储最大年龄。

3. 重启实例：

```
# dsctl instance_name restart
```

4. 创建帐户过期策略条目：

```
# 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 属性中的值将配置帐户在创建后 **5184000** 秒（60 天）过期。

5. 创建 CoS 模板条目：

```
# 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
```

此模板条目引用帐户到期策略。

6. 创建 CoS 定义条目：

```
# 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
```

此定义条目引用了 CoS 模板条目，并导致 **acctPolicySubentry** 属性出现在每个用户条目中，值设为 **cn=Account Expiration Policy,dc=example,dc=com**。

验证

- 尝试以存储在 **dc=example,dc=com** 条目中的用户的身份连接到该目录，其 **createTimestamp** 属性设置为一个大于 60 天前的值：

```
# 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.
```

如果目录服务器拒绝访问并返回这个错误，则帐户到期可以正常工作。

其他资源

- 重新启用达到不活跃限制的帐户

5.3. 在密码过期后自动禁用帐户一定时间

按照以下步骤配置基于时间的锁定策略，该策略在 **dc=example,dc=com** 条目下的用户中激活，并且不会为超过 28 天更改其密码。

先决条件

- 用户必须在其条目中设置 **passwordExpirationTime** 属性。

流程

1. 启用密码过期功能：

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

2. 启用帐户策略插件：

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

3. 配置插件配置条目：

```
# 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
```

这个命令使用以下选项：

- **--always-record-login yes** ：启用登录时间的日志记录。这需要使用带帐户策略的类服务(CoS)或角色，即使它没有设置 **acctPolicySubentry** 属性。
- **--always-record-login-attr lastLoginTime**: 配置 Account Policy 插件在用户的 **lastLoginTime** 属性中存储最后一次登录的时间。
- **--state-attr *non_existent_attribute*** ：设置用于评估帐户策略到不存在的 dummy 属性的主时间属性。
- **--Alt-state-attr 'passwordExpirationTime** ：配置插件以使用 **passwordExpirationTime** 属性作为要检查的备用属性。
- **--spec-attr acctPolicySubentry**: Configures Directory Server 应用策略到设置了 **acctPolicySubentry** 属性的条目。您可以在 CoS 条目中配置此属性。
- **--limit-attr accountInactivityLimit** ：配置帐户策略条目中的 **accountInactivityLimit** 属性会在帐户上次更改密码后被激活时保存时间。

4. 重启实例：

```
# dsctl instance_name restart
```

5. 创建激活策略条目的帐户：

```
# 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 属性中的值用来控制，在更改密码后，目录服务器使帐户不活跃 **2419200** 秒（28 天）。

6. 创建 CoS 模板条目：

```
# 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
```

此模板条目引用帐户正在激活策略。

7. 创建 CoS 定义条目：

```
# 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
```

此定义条目引用 CoS 模板条目，并导致 **acctPolicySubentry** 属性出现在每个用户条目中，值设为 **cn=Account Inactivation Policy,dc=example,dc=com**。

验证

1. 将用户的 **passwordExpirationTime** 属性设置为比您配置的不活跃时间超过不活跃时间的值：

```
# 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. 尝试以这个用户身份连接到该目录：

```
# 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.
```

如果目录服务器拒绝访问并返回这个错误，则帐户不活跃工作。

其他资源

- 重新启用达到不活跃限制的帐户

5.4. 在帐户不活跃和密码过期时自动禁用帐户

在使用 **checkAllStateAttrs** 设置进行身份验证时，您可以同时应用帐户不活跃和密码过期。默认情况下，当插件配置条目中没有 **checkAllStateAttrs** 时，或者在将此参数设置为 **no** 时，插件会检查 state 属性 **lastLoginTime**。如果条目中没有属性，则插件会检查备用 state 属性。

您可以将 main state 属性设置为不存在的属性，并在您希望插件根据 **passwordExpirationtime** 属性处理过期时将备用 state 属性设置为 **passwordExpirationtime**。当您启用此参数时，它会检查 main state 属性，以及帐户是否微调，然后检查备用状态属性。

这与密码策略的密码过期不同，因为如果密码过期时间超过不活跃的限制，则帐户策略插件会完全禁用帐户。使用密码策略过期时，用户仍然可以登录并更改其密码。帐户策略插件完全阻止用户进行任何操作，管理员必须重置帐户。

流程

1. 创建插件配置条目并启用设置：

```
# 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. 重启服务器以载入新的插件配置：

```
# dsctl instance_name restart
```

警告

checkAllStateAttrs 设置设计为仅在 alternate state 属性设置为 **passwordExpiratontime** 时工作。将它设置为 **createTimestamp** 可能会导致不必要的结果，条目可能会被锁定。

第 6 章 重新启用达到不活跃限制的帐户

如果目录服务器激活帐户，因为它达到不活动限制，管理员可以重新启用该帐户。

6.1. 在帐户策略插件中重新启用帐户

您可以使用 **dsconf 帐户解锁命令重新启用帐户**，也可以通过手动更新被激活的用户的 **lastLoginTime** 属性。

先决条件

- 在激活的用户帐户中。

流程

- 使用以下方法之一重新激活帐户：
  - 使用 **dsconf 帐户解锁** 命令：

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

  - 通过将用户的 **lastLoginTime** 属性设置为最新的时间戳：

```
# 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
```

验证

- 以您重新激活的用户身份进行身份验证。例如，执行搜索：

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

  如果用户成功进行身份验证，则会重新激活帐户。

第 7 章 在不设置锁定策略的情况下跟踪最后一次登录时间

您可以使用 Account Policy 插件跟踪用户登录时间，而无需设置过期时间或不活跃的时间。在这种情况下，插件会将 **lastLoginTime** 属性添加到用户条目中。

7.1. 配置帐户策略插件以记录最后一次登录时间

按照以下步骤，在用户条目的 **lastLoginTime** 属性中记录用户最后一次登录时间。

流程

1. 启用帐户策略插件：

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

2. 创建插件配置条目来记录登录时间：

```
# 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
```

这个命令使用以下选项：

- **--always-record-login yes** ：启用登录时间的日志记录。
- **--state-attr lastLoginTime** ：配置 accounts Policy 插件会在用户的 **lastLoginTime** 属性中存储最后一次登录的时间。

3. 重启实例：

```
# dsctl instance_name restart
```

验证

1. 以 用户身份登录目录服务器。例如，运行搜索：

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

2. 显示您在上一步中使用的用户 **的最后LoginTime** 属性：

```
# 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
```

如果 **lastLoginTime** 属性存在，并且 Directory Server 更新其值，则最后登录时间的记录可以正常工作。

第 8 章 使用 GET EFFECTIVE RIGHTS 搜索检查条目的访问权限

作为管理员，您可以查找和控制用户对特定条目内属性的访问权限。

获得有效的权限 (GER)是扩展目录搜索的方法，以显示用户对指定条目的访问权限。您可以指定以下权限：

- 读
- 编写和自我写入
- 搜索
- 添加
- 删除

在以下情况下，检查条目的有效权限很有用：

- 您可以使用 GER 命令更好地组织目录的访问控制指令。与另一个组相比，通常需要限制一组用户可以查看或编辑的用户。例如，**QA Managers** 组的成员可能有权限搜索并读 **manager** 和 **salary** 等属性，当只有 **HR Group** 的成员有权限修改或删除它们。检查用户或组的有效权限是验证管理员是否设置了适当访问控制的方法。
- 您可以使用 GER 命令查看您可以在个人条目上查看或修改哪些属性。例如，用户应该有权访问 **homePostalAddress** 和 **cn** 等属性，但可能只有对 **manager** 和 **salary** 属性的读取访问权限。

getEffectiveRights 搜索使用以下实体：

- *请求者*.当 **getEffectiveRights** 搜索出现问题时，它是经过身份验证的条目。
- 您将评估其权限 *的主题*。它在 GER 控制中被定义为授权 **DN**。
- *目标*。您可以通过搜索基础、搜索过滤器和请求的属性列表来定义它。

8.1. GET EFFECTIVES SEARCH PERMISSIONS

任何 *Get Effective Rights* (GER)搜索显示任何条目都可以具有以下的访问权限：

- *低级别权限*，即条目的权限。该访问权限显示用户 *A 可以在用户 B 条目上执行哪些操作*。
- *第二级权限* 显示用户 *A* 具有的给定属性的权限。*用户 A* 可能会对同一条目的不同属性具有不同的访问权限。用户拥有的任何访问控制都是对该条目的有效权限。

例如：

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

GER 搜索对条目和属性有以下访问权限：

表 8.1. 条目权利

权限	描述
a	添加一个条目。
d	删除此条目。
n	重命名 DN。
v	查看条目。

表 8.2. 属性权利

权限	描述
r	读.
s	搜索.
w	写入(**mod-add**)。
o	Obliterate (**mod-del**)。与删除类似。
c	比较.
W	自我写入.
O	自我删除.

8.2. GET EFFECTIVE RIGHTS 搜索格式

Get valid rights (GER)是一个扩展目录搜索。要使用它，您必须使用 **ldapsearch** 命令将 **-E** 选项传递给轻量级目录访问协议(LDAP)控制。例如：

```
# 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** 是子树或条目的基本 DN，您可以搜索 GER 主题。
  如果搜索基础是一个特定条目 DN，或者结果只返回一个条目，则结果会显示请求者在该特定条目上具有的权限。如果多个条目与过滤器匹配，则搜索会返回每个匹配条目，且每个条目具有请求者的权限。

- **1.3.6.1.4.1.42.2.27.9.5.2** 选项是 GER 控制的对象标识符。
  感叹号(**!**)定义搜索操作在服务器不支持此控制时返回错误(**!**)还是返回任何内容。

- *GER_subject* 是您检查权限的用户。您可以将 *GER_subject* 留空(**dn:**)，以获取匿名用户权利的结果。

- 可选属性 *List* 将 GER 结果限制为指定的属性或对象类，例如 ***mail*** 属性。

- 使用星号(*)符号返回所有属性。
- 使用加号(+)符号返回操作属性。

GER 选项向 **ldapsearch** 结果中添加额外信息，显示特定用户具有的权限。该 GER 主题用户可以通过附加选项 **-D** 对他们自己的条目请求权限。

如果请求者不是目录管理器用户，则请求者只能看到 GER 主题在请求者条目上具有的权限。所有其他条目会返回对有效权限不足的访问错误。

普通用户运行 GER 搜索的情况是常见的：

- 用户 A 检查其具有其他目录条目的权限。
- 用户 A 检查他对个人条目具有的权利。
- 用户 A 检查用户 B 对用户 A 条目具有的权利。

8.3. GET EFFECTIVE RIGHTS 搜索的常见场景

以下示例显示了使用 Get Effective Rights 搜索的常见情况。

8.3.1. Get Effective Rights 搜索的一般示例

当您需要使用 Get Effective Rights (GER)搜索时，最常见的情况是：

1. 检查个人权限.当用户 A 正在检查个人条目的权限时。例如，Ted Morris 希望检查他条目的权限：

例 8.1. 检查个人权限（用户 A 用户到用户 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
```

在本例中，**-b** 选项也具有请求者的 DN。

2. 检查其他用户的权限.例如，Ted Morris 是一个经理，需要检查其下级的 Dave Miller 条目：

例 8.2. 检查其他用户的权限（用户 A 到用户 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
```

在本例中，Ted Morris 具有对 Dave Miller 条目的所有属性的读取、搜索、比较、修改和删除权限。

3. 作为目录管理器，检查一个用户对另一个用户条目的权限。例如，Directory Manager 正在检查 Jane Smith 作为经理对她从属 Ted Morris 的条目具有哪些权利：

例 8.3. 目录管理器检查一个用户对另一个用户的权限

```
# 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
```

如果用户没有权限，则结果会显示无法访问错误不足：

例 8.4. 对条目没有权限

```
# 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. 检查另一个用户在您的条目上具有哪些权限。例如，Ted Morris 检查 Dave Miller 在 Ted Morris 的条目上有哪些权利：

例 8.5. 检查另一个用户在您的条目中的权限

```
# 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
```

在本例中，Dave Miller 有权查看条目的 DN，并可读取、搜索和比较 ***ou***、***givenName***、***l*** 和其他属性。他对 ***userPassword*** 属性没有任何权限。

8.3.2. Get Effective Rights 搜索不存在的属性示例

默认情况下，条目中的属性没有值。使用带有 Get Effective Rights (GER)搜索的星号(*)返回条目的每个属性，包括未在条目上设置的属性。

例 8.6. 检查条目的每个属性的权限

```
# 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
```

在本例中，***secretary*** 属性没有被设置，但您仍可在 GER 搜索结果中看到它。

8.3.3. Get Effective Rights 搜索特定属性或对象类的示例

本节中的示例演示了如何搜索对特定属性、属性集合以及属于条目的对象类的所有属性的权限。

1. 通过列出条目的具体属性来获取权利(GER)搜索结果。例如：

   例 8.7. 获取特定属性的搜索结果

   ```
   # 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. GER 以 *属性@objectClass* 格式搜索条目的对象类的特定属性。请求者必须是目录管理器。

   例 8.8. 获取对象类的特定属性的搜索结果

   ```
   # 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
   ```

   您可以使用星号(*)返回对象类的所有属性，格式为 **@objectClass*。搜索结果还包括不存在的属性。

8.3.4. Get Effective Rights 搜索不存在的条目的示例

本例演示了如何在用户条目（尚不存在）上检查特定用户的权限。在这种情况下，服务器会在子树中生成模板条目，您可以使用 Get Effective Rights (GER)搜索。要检查不存在的条目，Get Effective Rights (GER)搜索可以使用指定的对象类生成带有此条目的所有潜在属性的模板条目。

当服务器创建模板条目时，它使用对象类定义中的第一个 MUST 属性来创建 RDN 属性。如果 MUST 属性不存在，服务器将使用 MAY 属性。通过将 RDN 值传递给对象类，格式为 *@objectclass:rdn_attribute*。

例如，要检查 **scarter** 对带有 ***uidNumber*** 作为其 RDN 的不存在的 POSIX 条目的权限：

例 8.9. 检查不存在的条目的权限

```
# 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. Get Effective Rights 搜索操作属性的示例

ldapsearch 命令不会返回操作属性。使用加号符号(**+**)搜索它们。使用 **+** 仅返回您可以在条目上使用的操作属性。

例 8.10. 搜索操作属性

```
# 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 Effectives 结果和访问控制规则示例

有效的访问控制列表(ACL)定义用户的 Get Access Rights (GER)。

例 8.11. 访问控制列表

```
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
```

在本例中，ACL 不包含 **dc=example,dc=com** 子树。这会导致 GER 搜索结果显示用户没有 **dc=example,dc=com** 条目：

例 8.12. 带有未设置 ACL 的 GER 搜索结果

```
# 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
```

要查看结果，您必须是目录管理器，否则结果为空白。

8.4. GET EFFECTIVE RIGHT 返回代码

如果发生错误，则 Get Effective Rights (GER)搜索结果会返回一个错误代码。下表描述了错误代码：

表 8.3. 错误代码

代码	描述
0	成功完成。

代码	描述
1	操作错误。
12	Critical 扩展不可用。如果关键表达式设为 **true**，则条目上不存在有效的权限。
16	没有这样的属性。
17	未定义属性类型。
21	无效的属性语法。
50	权限不足。
52	不可用。
53	取消停止执行。
80	其他.