

```
[student@serverb ~]$ sudo tuned-adm list
[sudo] password for student: student
Available profiles:
- balanced           - General non-specialized tuned profile
- desktop            - Optimize for the desktop use-case
- latency-performance - Optimize for deterministic performance at the cost of
                        increased power consumption
- network-latency    - Optimize for deterministic performance at the cost of
                        increased power consumption, focused on low latency
                        network performance
- network-throughput - Optimize for streaming network throughput, generally
                        only necessary on older CPUs or 40G+ networks
- powersave          - Optimize for low power consumption
- throughput-performance - Broadly applicable tuning that provides excellent
                           performance across a variety of common server workloads
- virtual-guest       - Optimize for running inside a virtual guest
- virtual-host        - Optimize for running KVM guests
Current active profile: virtual-guest
```

1.5. 将当前活动的调优配置文件更改为 balanced 配置文件。

```
[student@serverb ~]$ sudo tuned-adm profile balanced
```

1.6. 列出当前活动的调优配置文件的摘要信息。

使用 **tuned-adm profile_info** 命令确认当前活动的配置文件是否为 balanced 配置文件。

```
[student@serverb ~]$ sudo tuned-adm profile_info
Profile name:
balanced

Profile summary:
General non-specialized tuned profile
...output omitted...
```

2. serverb 上两个进程的 CPU 使用百分比较高。请将每个进程的 nice 级别调整为 10，以便为其他进程腾出更多 CPU 时间。

2.1. 确定 serverb 上占用 CPU 资源最多的两个进程。占用 CPU 资源最多的进程列在命令输出的最后面。CPU 百分比值会有所不同。

```
[student@serverb ~]$ ps aux --sort=pcpu
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
...output omitted...
root      2983  100  0.0 228360  1744 ?        R<   21:08   0:23 md5sum /dev/zero
root      2967   101  0.0 228360  1732 ?        RN    21:08   0:23 sha1sum /dev/zero
[student@serverb ~]$
```

2.2. 确定占用 CPU 资源最多的两个进程各自的当前 nice 级别。

```
[student@serverb ~]$ ps -o pid,pcpu,nice,comm $(pgrep sha1sum;pgrep md5sum)
 PID %CPU NI COMMAND
 2967 99.6  2 sha1sum
 2983 99.7 -2 md5sum
```

2.3. 使用 **sudo renice -n 10 2967 2983** 命令将每个进程的 nice 级别调整为 **10**。使用在上一个命令输出中确定的 PID 值。

```
[student@serverb ~]$ sudo renice -n 10 2967 2983
[sudo] password for student:
2967 (process ID) old priority 2, new priority 10
2983 (process ID) old priority -2, new priority 10
```

2.4. 验证每个进程的当前 nice 级别是否为 10。

```
[student@serverb ~]$ ps -o pid,pcpu,nice,comm $(pgrep sha1sum;pgrep md5sum)
 PID %CPU NI COMMAND
 2967 99.6 10 sha1sum
 2983 99.7 10 md5sum
```

2.5. 从 **serverb** 退出。

```
[student@serverb ~]$ exit
logout
Connection to serverb closed.
[student@workstation ~]$
```

评估

在 **workstation** 上，运行 **lab tuning-review grade** 命令来确认是否成功完成本实验练习。

```
[student@workstation ~]$ lab tuning-review grade
```

完成

在 **workstation** 上，运行 **lab tuning-review finish** 脚本来完成本练习。

```
[student@workstation ~]$ lab tuning-review finish
```

本实验到此结束。

总结

在本章中，您学到了：

- **tuned** 服务可基于预定义的所选调优配置文件自动修改设备设置，以满足特定的系统需求。
- 要恢复所选的配置文件对系统设置所做的所有更改，可切换到另一个配置文件或停用 **tuned** 服务。
- 系统会为进程分配一个相对优先级，以确定其 CPU 访问权限。该优先级被称为进程的 **nice** 值。
- **nice** 命令将在进程启动时为其分配优先级。**renice** 命令可修改正在运行的进程的优先级。

海量视频题库 myitpub.com QQ:5565462
www.52myit.com

章 4

使用 ACL 控制对文件的访问

目标

解释并设置文件的访问控制列表 (ACL)，以处理需要复杂用户和组访问权限的情况。

培训目标

- 描述 ACL 的用例，识别设置了 ACL 的文件，并解释这些 ACL 的影响。
- 设置和删除文件上的 ACL，定义新创建文件上由目录自动设置的默认 ACL。

章节

- 解释文件 ACL (及引导式练习)
- 使用 ACL 保护文件安全 (及引导式练习)

实验

使用 ACL 控制对文件的访问

解释文件 ACL

培训目标

学完本节后，您应能够：

- 描述 ACL 和文件系统挂载选项。
- 使用 **ls** 和 **getfac1** 查看并解释 ACL。
- 描述 ACL 掩码和 ACL 权限优先级。
- 确定默认情况下红帽企业 Linux 使用 ACL 的位置。

访问控制列表概念

当文件仅由单个所有者和指定的一组人使用时，标准 Linux 文件权限即可满足要求。但是，有些用例要求多个指定的用户和组以不同的文件权限集来访问文件。访问控制列表 (ACL) 便提供了这一功能。

借助 ACL，您可以使用与常规文件权限相同的权限标志（读取、写入和执行）向由用户名、组名、UID 或 GID 标识的多个用户和组授予权限。除了文件所有者和文件的组从属关系之外，这些额外的用户和组分别被称为指定用户和指定组，因为它们不是在长列表中指定的，而是在 ACL 中指定的。

用户可以对属于自己的文件和目录设置 ACL。被分配了 **CAP_FOWNER** Linux 功能的特权用户可以对任何文件或目录设置 ACL。新文件和子目录会自动从父目录的默认 ACL（若已设置）中继承 ACL 设置。与常规文件的访问规则相似，父目录层次结构至少需要其他搜索（执行）权限集，以便启用指定用户和指定组的访问权限。

文件系统 ACL 支持

文件系统需挂载已启用的 ACL 支持。XFS 文件系统内置有 ACL 支持。其他文件系统（例如在红帽企业 Linux 8 上创建的 ext3 或 ext4）默认情况下会启用 **acl** 选项，但在早期版本中，应确认是否启用了 ACL 支持。要启用文件系统 ACL 支持，请对 **mount** 命令或在 **/etc/fstab** 配置文件的文件系统条目中使用 **ACL** 选项。

查看和解释 ACL 权限

ls -l 命令仅输出最少的 ACL 设置详细信息：

```
[user@host content]$ ls -l reports.txt
-rwxrwx---+ 1 user operators 130 Mar 19 23:56 reports.txt
```

10 字符权限字符串末尾的加号 (+) 表示该文件上存在带有若干条目的扩展 ACL 结构。

用户：

显示用户 ACL 设置，其与标准的用户文件设置相同；**rwx**。

组：

显示当前的 ACL 掩码设置，而不是组所有者设置；**rwx**。

其他：

显示其他 ACL 设置，其与标准的其他文件设置相同；无访问权限。



重要

如果使用 **chmod** 更改具有 ACL 的文件的组权限，则不会更改组所有者权限，而是更改 ACL 掩码。如果目的是更新文件的组所有者权限，需使用 **setfacl -m g::perms file**。

查看文件 ACL

要显示文件上的 ACL 设置，使用 **getfacl file**：

```
[user@host content]$ getfacl reports.txt
# file: reports.txt
# owner: user
# group: operators
user::rwx
user:consultant3:---
user:1005:rwx      #effective:rw-
group::rwx         #effective:rw-
group:consultant1:r--
group:2210:rwx     #effective:rw-
mask::rw-
other::---
```

查看上面示例中的每个部分：

注释条目：

```
# file: reports.txt
# owner: user
# group: operators
```

前三行是注释，用于识别文件名、所有者 (**user**) 和组所有者 (**operators**)。如果存在任何其他文件标志（如 **setuid** 或 **setgid**），则会出现第四行注释来显示所设置的标志。

用户条目：

```
user::rwx
user:consultant3:---
user:1005:rwx      #effective:rw-
```

- ① 文件所有者权限。**user** 的权限为 **rwx**。
- ② 指定用户权限。与此文件相关联的每位指定用户均有一个条目。**consultant3** 没有任何权限。
- ③ 指定用户权限。UID 1005 具有 **rwx**，但是掩码将有效权限仅限制为 **rw**。

组条目：

```
group::rwx      #effective:rwx- ①  
group:consultant1:r-- ②  
group:2210:rwx      #effective:rwx- ③
```

- ① 组所有者权限。operators 具有 rwx，但是掩码将有效权限仅限制为 rw。
- ② 指定组权限。与此文件相关联的每个指定组均有一个条目。consultant1 仅有 r 权限。
- ③ 指定组权限。GID 2210 具有 rwx，但是掩码将有效权限仅限制为 rw。

掩码条目：

```
mask::rw-
```

掩码设置显示可能为所有指定用户、组所有者和指定组提供的最大权限。UID 1005、operators 和 GID 2210 无法执行此文件，即便每个条目均已设置执行权限。

其他条目：

```
other::---
```

其他或“全局”权限。所有其他的 UID 和 GID 均无任何权限。

查看目录 ACL

要显示目录上的 ACL 设置，使用 **getfacl directory** 命令：

```
[user@host content]$ getfacl .  
# file: .  
# owner: user  
# group: operators  
# flags: -s-  
user::rwx  
user:consultant3:---  
user:1005:rwx  
group::rwx  
group:consultant1:r-x  
group:2210:rwx  
mask::rwx  
other::---  
default:user::rwx  
default:user:consultant3:---  
default:group::rwx  
default:group:consultant1:r-x  
default:mask::rwx  
default:other::---
```

查看上面示例中的每个部分：

打开注释条目：

```
# file: .
# owner: user
# group: operators
# flags: -s-
```

前三行是注释，用于识别目录名、所有者 (**user**) 和组所有者 (**operators**)。如果存在任何其他目录标志（如 **setuid**、**setgid**、**sticky**），则会出现第四行注释来显示所设置的标志；本例为 **setgid**。

标准 ACL 条目：

```
user::rwx
user:consultant3:---
user:1005:rwx
group::rwx
group:consultant1:r-x
group:2210:rwx
mask::rwx
other::---
```

此目录上的 ACL 权限与前述文件示例中的相同，但适用于目录。其关键区别在于，这些条目包含了执行权限（适用时），以允许目录搜索权限。

默认用户条目：

```
default:user::rwx
default:user:consultant3:---
```

- ① 默认文件所有者 ACL 权限。文件所有者将获得 **rwx**，即可以读取/写入新文件，并在新子目录上执行操作。
- ② 默认指定用户 ACL 权限。每位指定用户均有一个条目，他们将自动获得应用到新文件或子目录的默认 ACL。**consultant3** 始终默认设置为没有任何权限。

默认组条目：

```
default:group::rwx
default:group:consultant1:r-x
```

- ① 默认组所有者 ACL 权限。文件组所有者将获得 **rwx**，即可以读取/写入新文件，并在新子目录上执行操作。
- ② 默认指定组 ACL 权限。每个指定组均有一个条目，他们将自动获得默认 ACL。**consultant1** 将获得 **rx**，即只能够读取新文件，并在新子目录上执行操作。

默认 ACL 掩码条目：

```
default:mask::rwx
```

默认掩码设置显示可能为所有新建文件或目录（其具有指定用户 ACL、组所有者 ACL 或指定组 ACL）提供的初始最大权限：读取和写入新文件，对新子目录的执行权限。新文件永远不可获得执行权限。

默认其他条目：

```
default:other::---
```

默认其他或“全局”权限。所有其他 UID 和 GID 对新文件或新子目录均没有任何权限。

上述示例中的 **默认** 条目不包括指定用户 (UID **1005**) 和指定组 (GID **2210**)；因此，它们不会自动获得添加至任何新文件或新子目录中的初始 ACL 条目。这会有效地将它们限制在其已具有 ACL 的文件和子目录中，或者限制在相关文件所有者后来使用 **setfacl** 添加了 ACL 的文件和子目录中。这些条目仍可创建自己的文件和子目录。



注意

getfacl 命令的输出可用作 **setfacl** 的输入，以恢复 ACL 或者从源文件或目录复制 ACL 并将其保存到新文件中。例如，若要从备份恢复 ACL，可使用 **getfacl -R /dir1 > file1** 为目录及其内容生成递归 ACL 输出转储文件。而后，这一输出可用于恢复原始的 ACL，从而将保存的输出作为输入传给 **setfacl** 命令。例如，要对当前路径中的同一目录执行批量更新，请使用以下命令：**setfacl --set-file=file1**

ACL 掩码

ACL 掩码定义可授予指定用户、组所有者和指定组的最大权限。它不限制文件所有者或其他用户的权限。所有实施 ACL 的文件和目录都将具有 ACL 掩码。

可使用 **getfacl** 查看掩码，并通过 **setfacl** 显式设置掩码。如果未显式设置，系统会自动计算并添加掩码；但也可从父目录默认掩码设置中继承掩码。默认情况下，每当添加、修改或删除任何受影响的 ACL 时，均会重新计算掩码。

ACL 权限优先级

在决定一个进程（正在运行的程序）能否访问文件时，将按如下所示应用文件权限和 ACL：

- 如果正在以文件所有者身份运行进程，则应用文件的用户 ACL 权限。
- 如果正在以指定用户 ACL 条目中列出的用户身份运行进程，则应用指定用户 ACL 权限（只要掩码允许）。
- 如果正在以与文件的组所有者相匹配的组身份运行进程，或者以具有显式指定组 ACL 条目的组身份运行进程，则应用相匹配的 ACL 权限（只要掩码允许）。
- 否则，将应用文件的其他 ACL 权限。

操作系统使用 ACL 的示例

红帽企业 Linux 上有展示 ACL 用于扩展权限需求的典型示例。

Systemd 日志文件上的 ACL

systemd-journald 使用 ACL 条目向 **adm** 和 **wheel** 组授予对 **/run/log/journal/cb44...8ae2/system.journal** 文件的读取访问权限。此 ACL 允许 **adm** 和 **wheel** 组的成员读取由 **journalctl** 管理的日志，而无需赋予对 **/var/log/** 中具有特权的内容（如 **messages**、**secure** 或 **audit**）的特殊权限。

由于 **systemd-journald** 配置的缘故，**system.journal** 文档的父文件夹可以改变，但 **systemd-journald** 会自动将 ACL 应用于新文件夹和文件。



注意

当 `systemd-journald` 配置为使用持久存储时，系统管理员应在 `/var/log/journal/` 文件夹上设置 ACL。

```
[user@host ]$ getfacl /run/log/journal/cb44...8ae2/system.journal
getfacl: Removing leading '/' from absolute path names
# file: run/log/journal/cb44...8ae2/system.journal
# owner: root
# group: systemd-journal
user::rw-
group::r--
group:adm:r--
group:wheel:r--
mask::r--
other::---
```

受 `systemd` 管理的设备上的 ACL

`systemd-udev` 使用一组 `udev` 规则，它们将启用某些设备的 `uaccess` 标记，如 CD/DVD 播放器或刻录机、USB 存储设备、声卡等等。上述 `udev` 规则可在这些设备上设置 ACL，从而允许登录到图形用户界面（例如 `gdm`）的用户完全控制这些设备。

在用户从 GUI 注销之前，ACL 将保持活动状态。下一个登录 GUI 的用户会为所需的设备应用新 ACL。

在下例中，您可以看到 `user` 有一个 ACL 条目，它将 `rw` 权限应用于 `/dev/sr0` 设备（一个 CD/DVD 驱动器）。

```
[user@host ]$ getfacl /dev/sr0
getfacl: Removing leading '/' from absolute path names
# file: dev/sr0
# owner: root
# group: cdrom
user::rw-
user:group:rw-
group::rw-
mask::rw-
other::---
```



参考文献

`acl(5)`、`getfacl(1)`、`journald.conf(5)`、`ls(1)`、`systemd-journald(8)` 和 `systemd-udevd(8)` man page

► 小测验

解释文件 ACL

将下列项目与表格中对应的项匹配。

getfacl /directory

default:m::rx /directory

default:user:mary:rx /directory

g::rw /directory

g::rw file

group:hug:rwx /directory

user::rx file

user:mary:rx file

描述	ACL 操作
显示目录上的 ACL。	
具有文件读取和执行权限的指定用户。	
具有文件读取和执行权限的文件所有者。	
授予目录组所有者的目录读取和写入权限。	
授予文件组所有者的文件读取和写入权限。	
授予指定组的目录读取、写入和执行权限。	
设置为默认掩码的读取和执行权限。	
获得新文件初始读取权限、新子目录读取和执行权限的指定用户。	

► 解决方案

解释文件 ACL

将下列项目与表格中对应的项匹配。

描述	ACL 操作
显示目录上的 ACL。	getfacl /directory
具有文件读取和执行权限的指定用户。	user:mary:rx file
具有文件读取和执行权限的文件所有者。	user::rx file
授予目录组所有者的目录读取和写入权限。	g::rw /directory
授予文件组所有者的文件读取和写入权限。	g::rw file
授予指定组的目录读取、写入和执行权限。	group:hug:rwx /directory
设置为默认掩码的读取和执行权限。	default:m::rx /directory
获得新文件初始读取权限、新子目录读取和执行权限的指定用户。	default:user:mary:rx /directory

使用 ACL 保护文件安全

培训目标

学完本节后，您应能够：

- 使用 **setfacl** 更改常规 ACL 文件权限。
- 控制新文件和目录的默认 ACL 文件权限。

更改 ACL 文件权限

使用 **setfacl** 添加、修改或删除文件和目录的标准 ACL。

ACL 采用普通的权限文件系统表示法：“**r**”表示读取权限，“**w**”表示写入权限，“**x**”表示执行权限。“**-**”（短划线）表示缺少相关权限。在（以递归方式）设置 ACL 时，大写字母“**X**”可用于表示：如果文件还没有相关的执行权限，则只应设置目录（而非常规文件）的执行权限。这一行为与 **chmod** 相同。

添加或修改 ACL

可以使用 **-m** 选项通过命令行设置 ACL，或使用 **-M** 选项（使用 “**-**”（短划线），而不使用 **stdin** 的文件名）通过文件传递 ACL。这两个选项是“修改”选项；它们会为文件或目录添加新的 ACL 条目，或替换特定的现有 ACL 条目。文件或目录的任何其他现有 ACL 条目均保持不变。



注意

使用 **--set** 或 **--set-file** 选项来完全替换文件的 ACL 设置。

首次定义文件的 ACL 时，如果添加操作不包含文件所有者、组所有者或其他权限的设置，则系统会基于当前标准文件权限来设置以上权限（这些设置也称为基础 ACL 条目，且无法删除），系统也会计算并添加新掩码值。

要添加或修改用户 ACL 或指定用户 ACL，请执行以下操作：

```
[user@host ~]$ setfacl -m u:name:rX file
```

如果 name 留空，则它适用于文件所有者，否则，name 可以是用户名或 UID 值。在本例中，授予的权限为只读和执行（如果已设置）（除非 file 为目录，此时目录会设置执行权限，以允许进行目录搜索）。

ACL 文件所有者和标准文件所有者权限同等；因此，使用文件所有者权限中的 **chmod** 等同于使用文件所有者权限中的 **setfacl**。**chmod** 对指定用户没有影响。

要添加或修改组 ACL 或指定组 ACL，请执行以下操作：

```
[user@host ~]$ setfacl -m g:name:rw file
```

这与添加或修改用户 ACL 条目的模式相同。如果 name 留空，则它适用于组所有者。否则，请指定指定组的组名或 GID 值。本例中的权限为读取和写入权限。

chmod对于具有 ACL 设置的文件，对任何组权限都没有影响，但会更新 ACL 掩码。

要添加或修改其他 ACL，请执行以下操作：

```
[user@host ~]$ setfacl -m o::-- file
```

其他 ACL 仅接受权限设置。其他 ACL 的典型权限设置为：不具有任何权限，使用短划线 (-) 进行设置；只读权限，像往常一样使用 r 进行设置。当然，您可以设置任何标准权限。

ACL 其他权限及标准其他权限同等，因此使用其他权限中的 **chmod** 等同于使用其他权限中的 **setfacl**。

您可以使用同一命令添加多个条目；请使用以逗号分隔的条目列表：

```
[user@host ~]$ setfacl -m u::rwx,g:consultants:rX,o::-- file
```

该操作会将文件所有者设置为拥有读取、写入和执行权限，将指定组 **consultants** 设置为拥有只读和条件执行权限，并将所有其他用户限制为无权限。组所有者将维持其现有文件或 ACL 权限，并且其他“指定”条目保持不变。

使用 **getfacl** 作为输入

您可以将 **getfacl** 的输出用作 **setfacl** 的输入：

```
[user@host ~]$ getfacl file-A | setfacl --set-file=- file-B
```

--set-file 选项可接受来自文件或 stdin 的输入。短划线字符 (-) 指定使用 stdin。在此例中，file-B 与 file-A 的 ACL 设置相同。

设置明确的 ACL 掩码

您可以明确设置文件或目录的 ACL 掩码，以限制指定用户、组所有者和指定组的最大有效权限。这限制了任何超出掩码的现有权限，但不会影响比掩码的许可更少的权限。

```
[user@host ~]$ setfacl -m m::r file
```

该操作会添加掩码值，该掩码值会将任何指定用户、组所有者及任何指定组限制为拥有只读权限，而不考虑它们的现有设置。文件所有者和其他用户不受掩码设置的影响。

getfacl 会在受掩码设置限制的条目旁边显示 **有效** 注释。



重要

默认情况下，每当修改或删除受影响的 ACL 设置（指定用户、组所有者或指定组）时，系统都会重新计算 ACL 掩码，进而可能重新设置上一个明确的掩码设置。

为避免重新计算掩码，请使用 **-n** 选项，或者将掩码设置 (**-m m::perms**) 包含在会修改受掩码影响的 ACL 设置的任何 **setfacl** 操作中。

递归 ACL 修改

在目录上设置 ACL 时，请使用 **-R** 选项以递归方式应用 ACL。记住，您可能以递归方式使用“**X**”（大写 X）权限，这样，设置了执行权限的文件会保留该设置，而目录会设置执行权限，以允许进行目录搜索。以非递归方式设置 ACL 时，最好也使用大写“**X**”，因为这可防止管理员意外地向普通文件添加执行权限。

```
[user@host ~]$ setfacl -R -m u:name:rX directory
```

该操作会将用户 name 添加到 directory 目录及所有现有文件和子目录中，从而设置只读和条件执行权限。

删除 ACL

除了不指定“:perms”外，删除特定 ACL 条目与修改操作的基本形式相同。

```
[user@host ~]$ setfacl -x u:name,g:name file
```

该操作只会从文件或目录 ACL 中删除指定用户和指定组。任何其他现有 ACL 条目均保持活动。

可以在同一 **setfacl** 操作中包含删除 (-x) 和修改 (-m) 操作。

掩码只能在未设置其他 ACL（无法删除的基础 ACL 除外）的情况下删除，因此必须最后删除。文件不再有任何 ACL，并且 **ls -l** 不会在权限字符串旁边显示加号 (+)。或者，要删除文件或目录的所有 ACL 条目（包括目录的默认 ACL），请使用以下命令：

```
[user@host ~]$ setfacl -b file
```

控制默认 ACL 文件权限

为了确保在目录中创建的文件和目录继承特定的 ACL，请在目录上使用默认 ACL。您可以设置默认 ACL 并进行任何标准 ACL 设置，包括默认掩码。

目录本身仍然需要具备标准 ACL 才能进行访问权限控制，因为默认 ACL 不会对目录实施访问权限控制；它们仅提供 ACL 权限继承支持。例如：

```
[user@host ~]$ setfacl -m d:u:name:rx directory
```

该命令会添加一个默认的指定用户 (**d:u:name**)，该用户拥有对子目录的只读权限和执行权限。

用于为各个 ACL 类型添加默认 ACL 的 **setfacl** 命令与用于标准 ACL 的命令完全相同，但是以 **d:** 开头。或者，在命令行使用 **-d** 选项。



重要

在设置目录的默认 ACL 时，请通过在默认 ACL 中包含执行权限来确保用户可以访问其中创建的新子目录的内容。

用户不会自动设置对新创建的常规文件的执行权限，因为与新目录不同，新常规文件的 ACL 掩码为 **rw-**。

 **注意**

新文件和新子目录会继续从创建用户设置自己的 UID 和主要组 GID 值，不过，当父目录 **setgid** 标志启用时除外，在这种情况下，主要组 GID 与父目录 GID 相同。

删除默认 ACL 条目

删除默认 ACL 的方式与删除标准 ACL 一样，也是以 **d:** 开头，或使用 **-d** 选项。

```
[user@host ~]$ setfacl -x d:u:name directory
```

该命令会删除上一个示例中添加的默认 ACL 条目。

要删除目录的所有默认 ACL 条目，请使用 **setfacl -k directory**。



参考文献

acl(5)、**setfacl(1)** 和 **getfacl(1)** man page

► 指导练习

使用 ACL 保护文件安全

在本练习中，您将使用 ACL 条目为组授予目录访问权限并拒绝用户访问，设置目录的默认 ACL，以及确认在该目录中创建的新文件是否继承默认 ACL。

成果

您应能够：

- 使用 ACL 条目为组授予访问权限并拒绝其中一个成员访问。
- 验证现有的文件和目录是否反映了新的 ACL 权限。
- 设置目录的默认 ACL，并确认新文件和目录是否继承其配置。

在你开始之前

以 `student` 用户身份并使用 `student` 作为密码登录 `workstation`。

在 `workstation` 上，运行 `lab acl-secure start` 命令。该命令将运行一个起始脚本，它将确定 `servera` 计算机是否可从网络访问。此外，它还会创建本练习中使用的用户、组、目录和文件。

```
[student@workstation ~]$ lab acl-secure start
```

Operators 和 Consultants 是一家 IT 支持公司的成员。他们需要共享信息。`servera` 中包含有正确配置的共享目录，它位于托管文件的 `/shares/content`。

目前，仅 `operators` 组的成员有权访问该目录，但 `consultants` 组的成员需要对该目录具有完全访问权限。

`consultant1` 用户是 `consultants` 组的一个成员，但经常引起问题，因此该用户不应具有此目录的访问权限。

您的任务是向目录及其内容中添加相应 ACL 条目，这样，`consultants` 组的成员会拥有完全的访问权限，但会拒绝 `consultant1` 用户拥有任何访问权限。确保将来存储在 `/shares/content` 中的文件和目录应用相应的 ACL 条目。

重要信息：

- `sysadmin1` 和 `operator1` 用户是 `operators` 组的成员。
- `consultant1` 和 `consultant2` 用户是 `consultants` 组的成员。
- 目录 `/shares/content` 中包含一个名为 `server-info` 的子目录和大量文件，可以用来测试 ACL。此外，`/shares/content` 目录中还包含一个名为 `loadvg.sh` 的执行脚本，也可以用于测试。
- `sysadmin1`、`operator1`、`consultant1` 和 `consultant2` 用户的密码均为 `redhat`。
- 所有更改均应发生在 `/shares/content` 目录及其文件中。请勿调整 `/shares` 目录。

► 1. 登录 servera 并切换到 root 用户。

- 1.1. 使用 **ssh** 命令，以 **student** 用户身份登录 **servera**。系统已配置为使用 SSH 密钥来进行身份验证，因此不需要提供密码。

```
[student@workstation ~]$ ssh student@servera  
...output omitted...  
[student@servera ~]$
```

- 1.2. 使用 **sudo -i** 命令，切换为 **root** 用户。**student** 用户的密码为 **student**。

```
[student@servera ~]$ sudo -i  
[sudo] password for student: student  
[root@servera ~]#
```

► 2. 向 **/shares/content** 目录及其所有内容中添加指定的 ACL。

- 2.1. 使用 **setfacl** 以递归方式更新 **/shares/content** 目录，从而授予 **consultants** 组读取、写入和条件执行权限。

```
[root@servera ~]# setfacl -Rm g:consultants:rwx /shares/content
```

-R 选项表示递归，**-m** 选项表示修改/添加，**rwx** 表示应用读取、写入和条件执行权限。

- 2.2. 使用 **setfacl** 以递归方式更新 **/shares/content** 目录，从而拒绝 **consultant1** 用户拥有 **consultants** 组的任何访问权限。

```
[root@servera ~]# setfacl -Rm u:consultant1:- /shares/content
```

-R 选项表示递归，**-m** 选项表示修改/添加，**-** 表示不允许访问。

► 3. 添加指定 ACL 作为默认 ACL，以支持将来的文件和目录添加。

- 3.1. 使用 **setfacl** 添加 **consultants** 组的默认访问规则。授予 **content** 目录的读取、写入和执行权限。

```
[root@servera ~]# setfacl -m d:g:consultants:rwx /shares/content
```

-m 选项表示修改/添加，**d:g** 表示默认组，**rwx** 表示应用读取/写入/执行权限（正确创建和访问子目录所需的权限）

- 3.2. 使用 **setfacl** 添加 **consultant1** 用户的默认访问规则。拒绝对 **content** 目录的所有访问。

```
[root@servera ~]# setfacl -m d:u:consultant1:- /shares/content
```

-m 选项表示修改/添加，**d:u** 表示默认用户，**-** 表示没有权限

► 4. 验证 ACL 更改。

consultant2 应当能够读取任何文件并创建带有新文件的新目录。

consultant1 应当无法读取、写入或执行任何文件；这包括无法列出目录内容。

使用 **su - user** 切换到您的测试用户。使用 **exit** 或 **Ctrl+D** 退出测试用户 shell。

```
[root@servera ~]# exit  
[student@servera ~]$ su - consultant2  
Password: redhat  
[consultant2@servera ~]$ cd /shares/content/
```

4.1. 使用 **cat** 检查 **consultant2** 是否可以读取文件。

```
[consultant2@servera content]$ cat serverb-loadavg.txt  
#####  
serverb.lab.example.com  
#####  
Wed Mar 25 15:25:19 EDT 2019  
#####  
ldavg 0.18, 0.06, 0.05  
#####
```

4.2. 使用 **loadavg.sh** 脚本检查 **consultant2** 是否可以执行文件。

```
[consultant2@servera content]$ ./loadavg.sh  
ldavg 0.00, 0.00, 0.04
```

4.3. 创建名为 **reports** 的目录。

使用 **echo** 创建包含某些内容的文件，命名文件 **test.txt** 并将其放入新目录。
完成后，切换回 **student** 用户。

```
[consultant2@servera content]$ mkdir reports  
[consultant2@servera content]$ echo "TEST REPORT" > reports/test.txt  
[consultant2@servera content]$ exit  
logout  
[student@servera ~]$
```

4.4. 以 **consultant1** 用户身份登录。以 **consultant1** 用户身份，使用 **cd** 尝试更改目录，并且尝试用 **ls** 列出目录。这两个命令将失败，并显示 **Permission denied** 的消息。

尝试由 **consultant2** 使用的一个或多个命令，但以 **consultant1** 身份，进一步验证其缺少访问权限。请使用完整路径 **/shares/content**，因为无法使用 **cd** 来更改到该目录。

测试完 **consultant1** 后，切换回 **student**。

```
[student@servera ~]$ su - consultant1  
Password: redhat  
[consultant1@servera ~]$ cd /shares/content/  
-bash: cd: /shares/content/: Permission denied  
[consultant1@servera ~]$ ls /shares/content/  
ls: cannot open directory '/shares/content/': Permission denied  
[consultant1@servera ~]$ cat /shares/content/serverb-loadavg.txt  
cat: /shares/content/serverb-loadavg.txt: Permission denied
```

```
[consultant1@servera ~]$ exit  
logout  
[student@servera ~]$
```

- 4.5. 以 sysadmin1 用户身份登录。使用 **getfacl** 查看 **/shares/content** 上的所有 ACL 条目和 **/shares/content/reports** 上的 ACL 条目。

测试完 consultant1 后，切换回 student。

```
[student@servera ~]$ su - sysadmin1  
Password: redhat  
[sysadmin1@servera ~]$ getfacl /shares/content  
getfacl: Removing leading '/' from absolute path names  
# file: shares/content/  
# owner: root  
# group: operators  
# flags: -s-  
user::rwx  
user:consultant1:---  
group::rwx  
group:consultants:rwx  
mask::rwx  
other::---  
default:user::rwx  
default:user:consultant1:---  
default:group::rwx  
default:group:consultants:rwx  
default:mask::rwx  
default:other:---  
  
[sysadmin1@servera ~]$ getfacl /shares/content/reports  
getfacl: Removing leading '/' from absolute path names  
# file: shares/content/reports  
# owner: consultant2  
# group: operators  
# flags: -s-  
user::rwx  
user:consultant1:---  
group::rwx  
group:consultants:rwx  
mask::rwx  
other::---  
default:user::rwx  
default:user:consultant1:---  
default:group::rwx  
default:group:consultants:rwx  
default:mask::rwx  
default:other:---  
  
[sysadmin1@servera ~]$ exit  
logout  
[student@servera ~]$
```

- 4.6. 从 servera 注销。

```
[student@servera ~]$ exit  
logout  
Connection to servera closed.  
[student@workstation ~]$
```

完成

在 workstation 上，运行 **lab acl-secure finish** 脚本来完成本练习。

```
[student@workstation ~]$ lab acl-secure finish
```

本引导式练习到此结束。

► 开放研究实验

使用 ACL 控制对文件的访问

任务执行清单

在本实验中，您将为两个组中的用户设置一个协作目录，从而组合 set-GID 权限和默认 ACL 条目以提供正确的访问权限。

成果

您应能够：

- 在文件夹上配置 set-GID 权限，以继承内部文件和文件夹的组所有权。
- 配置 ACL 条目以允许或拒绝用户和组对文件和目录的读取/写入/执行权限。
- 配置默认 ACL 以对新文件和目录自动获取正确的 ACL 和文件权限。

在你开始之前

以 `student` 用户身份并使用 `student` 作为密码登录 `workstation`。

在 `workstation` 上，运行 `lab acl-review start` 命令。该命令将运行一个起始脚本，它将确定 `serverb` 计算机是否可从网络访问。此外，它还会创建本练习中使用的用户、组、目录和文件。

```
[student@workstation ~]$ lab acl-review start
```

某个证券金融机构正在设置可存放案例文件的协作共享目录，`managers` 组的成员将对这些文件具有读取和写入权限。

该机构的联合创始人 `manager1` 已决定，`contractors` 组成员也应该能够读取和写入此共享目录。不过，`manager1` 并不信任 `contractor3` 用户（`contractors` 组的成员），因此 `contractor3` 对该目录的访问权限应限定为只读。

`manager1` 已创建用户和组，而且开始设置共享目录并复制一些模板文件。由于 `manager1` 太忙，这项工作就落到了您的身上。

您的任务是完成共享目录的设置。此目录及其所有内容均应属于 `managers` 组，并且应针对所有者和组（`managers`）将文件更新为可读取和可写入。其他用户不应具有任何权限。您还需要为 `contractors` 组提供读取和写入权限，但 `contractor3` 除外，因为他只能获得读取权限。确保您的设置适用于现有的和将来的文件。

重要信息：

- 共享目录：`serverb` 上的 `/shares/cases`。
- `manager1` 和 `manager2` 用户是 `managers` 组的成员。
- `contractor1`、`contractor2` 和 `contractor3` 用户是 `contractors` 组的成员。
- 目录中存在两个文件：`shortlist.txt` 和 `backlog.txt`。

- 五个用户的密码都是 redhat。
- 所有更改均应发生在 **/shares/cases** 目录及其文件中；请勿调整 **/shares** 目录。

1. **cases** 目录及其内容应属于 **managers** 组。新增至 **cases** 目录中的文件应自动属于 **managers** 组。现有文件的用户和组所有者应具有读取和写入权限，其他用户应没有任何权限。



注意

提示：切勿使用 **setfacl**。

2. 将 ACL 条目添加至 **cases** 目录（及其内容），以允许 **contractors** 组成员具有文件的读取/写入权限以及目录的执行权限。限制 **contractor3** 用户，使其只具有文件的读取权限和目录的执行权限。
3. 添加 ACL 条目，以确保 **cases** 目录中的任何新文件或目录均针对所有授权用户和组应用了正确的权限。
4. 验证您已正确更改了 ACL 和文件系统。

使用 **ls** 和 **getfacl** 检查 **/shares/cases** 上的设置。

以 **student** 用户身份，使用 **su - user** 先切换到 **manager1**，然后切换到 **contractor1**。验证您能够写入文件、读取文件、创建目录以及对新目录中的文件执行写入操作。使用 **ls** 检查新目录权限，并使用 **getfacl** 查看新目录的 ACL。

以 **student** 用户身份，使用 **su - contractor3** 切换用户。尝试写入一个文件（应会失败）并尝试创建新目录（应会失败）。作为 **contractor3** 用户，您应该能够读取 **cases** 目录中的 **shortlist.txt** 文件，并能够读取在由 **manager1** 或 **contractor1** 用户新建的任一目录中写入的“测试”文件。



注意

上述这组测试是您可用来执行以检查访问权限是否正确的其中一些测试。您应根据自己的环境来设计合适的访问权限验证测试。

评估

在 **workstation** 上，运行 **lab acl-review grade** 命令来确认是否成功完成本练习。

```
[student@workstation ~]$ lab acl-review grade
```

完成

在 **workstation** 上，运行 **lab acl-review finish** 命令来完成本练习。

```
[student@workstation ~]$ lab acl-review finish
```

本实验到此结束。

► 解决方案

使用 ACL 控制对文件的访问

任务执行清单

在本实验中，您将为两个组中的用户设置一个协作目录，从而组合 set-GID 权限和默认 ACL 条目以提供正确的访问权限。

成果

您应能够：

- 在文件夹上配置 set-GID 权限，以继承内部文件和文件夹的组所有权。
- 配置 ACL 条目以允许或拒绝用户和组对文件和目录的读取/写入/执行权限。
- 配置默认 ACL 以对新文件和目录自动获取正确的 ACL 和文件权限。

在你开始之前

以 `student` 用户身份并使用 `student` 作为密码登录 `workstation`。

在 `workstation` 上，运行 `lab acl-review start` 命令。该命令将运行一个起始脚本，它将确定 `serverb` 计算机是否可从网络访问。此外，它还会创建本练习中使用的用户、组、目录和文件。

```
[student@workstation ~]$ lab acl-review start
```

某个证券金融机构正在设置可存放案例文件的协作共享目录，`managers` 组的成员将对这些文件具有读取和写入权限。

该机构的联合创始人 `manager1` 已决定，`contractors` 组成员也应该能够读取和写入此共享目录。不过，`manager1` 并不信任 `contractor3` 用户（`contractors` 组的成员），因此 `contractor3` 对该目录的访问权限应限定为只读。

`manager1` 已创建用户和组，而且开始设置共享目录并复制一些模板文件。由于 `manager1` 太忙，这项工作就落到了您的身上。

您的任务是完成共享目录的设置。此目录及其所有内容均应属于 `managers` 组，并且应针对所有者和组（`managers`）将文件更新为可读取和可写入。其他用户不应具有任何权限。您还需要为 `contractors` 组提供读取和写入权限，但 `contractor3` 除外，因为他只能获得读取权限。确保您的设置适用于现有的和将来的文件。

重要信息：

- 共享目录：`serverb` 上的 `/shares/cases`。
- `manager1` 和 `manager2` 用户是 `managers` 组的成员。
- `contractor1`、`contractor2` 和 `contractor3` 用户是 `contractors` 组的成员。
- 目录中存在两个文件：`shortlist.txt` 和 `backlog.txt`。