Red Hat Enterprise Linux 7

SELinux 用户和管理员指南

Security-Enhanced Linux（SELinux）的基本和高级配置

Last Updated: 2022-10-23

Red Hat Enterprise Linux 7 SELinux 用户和管理员指南

Security-Enhanced Linux（SELinux）的基本和高级配置

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律通告

摘要

本书包括两个部分：SELinux 和管理受限服务.前者描述了 SELinux 功能的基本原理和原则，后者更侧重于设置和配置各种服务的实际任务。

目录

部分 I. SELINUX

本文档论述了 Security Enhanced Linux(SELinux)功能的基础知识和原则。

第 1 章 简介

SELinux（Security Enhanced Linux）提供了一个额外的系统安全层。SELinux 从根本上回答问题："May <subject> do <action> to <object>"，例如："Web 服务器访问用户主目录中的文件吗？"

基于用户、组和其他权限（称为 Discretionary Access Control(DAC)）的标准访问策略不允许系统管理员创建全面而精细的安全策略，例如限制特定应用程序仅查看日志文件，同时允许其他应用程序将新数据附加到日志文件

SELinux 使用强制访问控制（Mandatory Access Control，简称 MAC）。每个进程和系统资源都有一个特殊的安全标签，称为 *SELinux 上下文*。SELinux 上下文有时称为 *SELinux 标签*，它是一个提取系统级别细节并专注于实体的安全属性的标识符。这不仅提供了在 SELinux 策略中引用对象的一种一致方法，而且消除了在其他身份识别方法中可能存在的任何不确定性；例如，文件在利用绑定挂载的系统中可以有多个有效的路径名称。

SELinux 策略在一系列规则中使用这些上下文，它们定义进程如何相互交互以及与各种系统资源进行交互。默认情况下,策略不允许任何交互,除非规则明确授予了相应的权限。

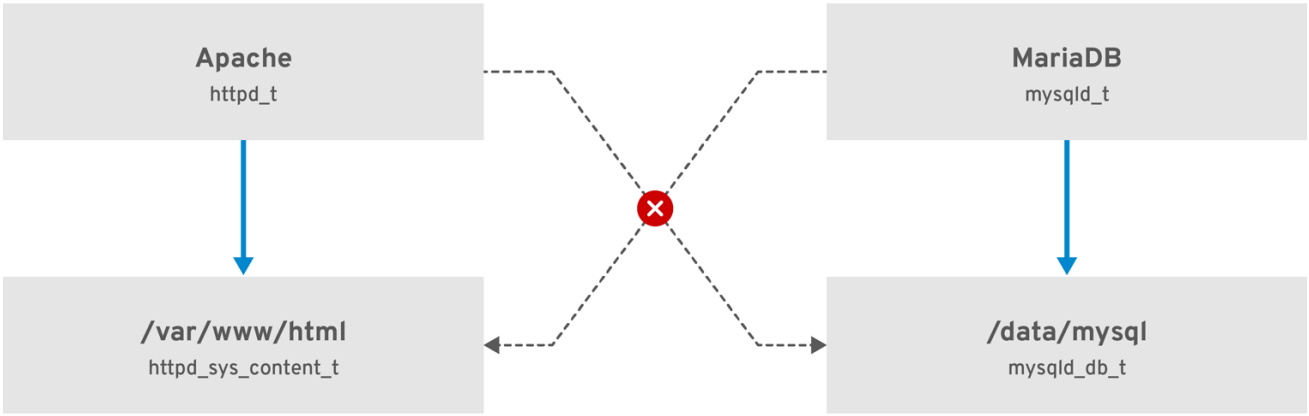
注意

务必要记住，在 DAC 规则后检查 SELinux 策略规则。如果 DAC 规则已拒绝了访问，则不会使用 SELinux 策略规则。这意味着，如果传统的 DAC 规则已阻止了访问，则不会在 SELinux 中记录拒绝信息。

SELinux 上下文包括以下字段： user（用户）、role（角色）、type（类型）和 security level（安全级别）。在 SELinux 策略中，SELinux 类型信息可能是最重要的。这是因为，最常用的、用于定义允许在进程和系统资源间进行的交互的策略规则会使用 SELinux 类型而不是 SELinux 的完整上下文。SELinux 类型通常以 **_t** 结尾。例如，Web 服务器的类型名称是 **httpd_t**。通常位于 **/var/www/html/** 中的文件和目录的类型上下文是 **httpd_sys_content_t**。通常位于 **/tmp 和 /var/tmp/ 中的文件和目录的类型上下文是 tmp_t**。Web 服务器端口的类型上下文是 **http_port_t**。

例如，有一个策略规则允许 Apache（作为 **httpd_t**运行的 Web 服务器进程）访问通常位于 **/var/www/html/** 及其他 Web 服务器目录(httpd**_sys_content_t**)中的文件和目录。策略中没有针对通常位于 **/tmp 和 /var/tmp** / 中的文件的允许规则，因此不允许访问。有了 SELinux，即使 Apache 被破坏并且恶意的脚本可以访问，它仍然无法访问 **/tmp** 目录。

图 1.1. SELinux 允许以 httpd_t 身份运行的 Apache 进程访问 /var/www/html/ 目录，并且它拒绝同一进程访问 /data/mysql/ 目录，因为 httpd_t 和 mysqld_db_t 类型上下文没有允许规则。另一方面，作为 mysqld_t 运行的 MariaDB 进程能够访问 /data/mysql/ 目录，SELinux 也正确拒绝类型为 mysqld_t 的进程，以访问标记为 httpd_sys_content_t 的 /var/www/html/ 目录。



[D]

其它资源

如需更多信息，请参阅以下文档：

- **apropos selinux 命令列出的 selinux (8)手册页** 和 man page。
- 安装 selinux-policy-doc 软件包时，**man -k _selinux 命令列出的 man** page。如需更多信息，请参阅 第 11.3.3 节 “为服务手动页面”。
- SELinux 颜色书
- SELinux Wiki FAQ

1.1. 运行 SELINUX 的好处

SELinux 提供以下优点：

- 所有进程和文件都被标记。SELinux 策略规则定义了进程如何与文件交互，以及进程如何相互交互。只有存在明确允许的 SELinux 策略规则时，才能允许访问。
- 精细访问控制。传统的 UNIX 通过用户的授权、基于 Linux 的用户和组进行控制。而 SELinux 的访问控制基于所有可用信息，如 SELinux 用户、角色、类型以及可选的安全级别。
- SELinux 策略由系统管理员进行定义，并在系统范围内强制执行。
- 改进了权限升级攻击的缓解方案。进程在域中运行，因此是相互分离的。SELinux 策略规则定义了如何处理访问文件和其它进程。如果某个进程被破坏，攻击者只能访问该进程的正常功能，而且只能访问已被配置为可以被该进程访问的文件。例如：如果 Apache HTTP 服务器被破坏，攻击者无法使用该进程读取用户主目录中的文件，除非添加或者配置了特定的 SELinux 策略规则允许这类访问。
- SELinux 可以用来强制实施数据机密性和完整性，同时保护进程不受不可信输入的影响。

但是，SELinux 本身并不是：

- 防病毒软件,
- 用来替换密码、防火墙和其它安全系统,
- 多合一的安全解决方案。

SELinux 旨在增强现有的安全解决方案，而不是替换它们。即使运行 SELinux，也务必要继续遵循良好的安全实践，例如保持软件更新、使用难以猜测的密码或防火墙。

1.2. 示例

以下示例演示了 SELinux 如何提高安全性：

- 默认操作为 deny（拒绝）。如果 SELinux 策略规则不存在允许访问（如允许进程打开一个文件），则拒绝访问。
- SELinux 可以限制 Linux 用户。SELinux 策略中包括很多受限制的 SELinux 用户。可将 Linux 用户映射到受限制的 SELinux 用户，以便利用其使用的安全规则和机制。例如，将 Linux 用户映射到 SELinux **user_u** 用户，会导致 Linux 用户无法运行（除非另外配置）设置用户 ID(setuid)应用程序，如 **sudo** 和 **su**。如需更多信息，请参阅 第 3.3 节 “受限制和未限制的用户”。
- 增加进程和数据的分离。进程在自己的域中运行，阻止进程访问其他进程使用的文件，并阻止进程访问其他进程。例如：在运行 SELinux 时，除非有其他配置，攻击者将无法侵入 Samba 服务器，然后使用 Samba 服务器作为攻击向量读取和写入其它进程使用的文件（如 MariaDB 数据库）。
- SELinux 可帮助缓解配置错误带来的破坏。不同的 DNS 服务器通常会在彼此间复制信息，这被称为区传输（zone transfer）.攻击者可以利用区传输来更新 DNS 服务器使其包括错误的信息。当在 Red Hat Enterprise Linux 中将 Berkeley Internet 名称域(BIND)作为 DNS 服务器运行时，即使管理员忘记限制哪些服务器可以执行区域传输，默认的 SELinux 策略也会阻止区域文件 [1] 通过 BIND **命名** 守护进程本身和其他进程，使用区域传送进行更新。
- 请参阅 NetworkWorld.com 文章 A Stbelt for server 软件：SELinux 阻止实际利用[2]有关 SELinux 的背景信息，以及 SELinux 已阻止的各种漏洞的信息。

1.3. SELINUX ARCHITECTURE

SELinux 是一个内置在 Linux 内核中的 Linux 安全模块（LSM）。内核中的 SELinux 子系统由安全策略驱动，该策略由管理员控制并在引导时载入。系统中所有与安全性相关的、内核级别的访问操作都会被 SELinux 截取，并在加载的安全策略上下文中检查。如果载入的策略允许操作，它将继续进行。否则,操作会被阻断，进程会收到一个错误。

SELinux 决策（如允许或禁止访问）会被缓存。这个缓存被称为 Access Vector Cache（AVC）。通过使用这些缓存的决定，可以较少对 SELinux 策略规则的检查，这会提高性能。请记住，如果 DAC 规则已首先拒绝了访问，则 SELinux 策略规则无效。

1.4. SELINUX 状态和模式

SELinux 可使用三种模式之一运行：disabled、permissive 或 enforcing 模式。

强烈建议不要使用禁用（disabled）模式。它不仅会使系统避免强制使用 SELinux 策略，还会避免为任何持久对象（如文件）添加标签，这使得在以后启用 SELinux 非常困难。

在 permissive 模式中，系统会象 enforcing 模式一样加载安全策略，包括标记对象并在日志中记录访问拒绝条目，但它并不会拒绝任何操作。虽然不建议在生产环境中使用 permissive 模式，但 permissive 模式对 SELinux 策略开发很有帮助。

Enforcing 模式是默认操作模式，在 enforcing 模式下 SELinux 可正常运行，并在整个系统中强制实施载入的安全策略。

使用 **setenforce** 实用程序在 enforcing 模式和 permissive 模式之间变化。**使用 setenforce** 进行的更改在重启后不会保留。要更改为强制模式，请以 Linux root 用户身份输入 **setenforce 1** 命令。要更改为 permissive 模式，请输入 **setenforce 0** 命令。使用 **getenforce** 工具来查看当前的 SELinux 模式：

```
~]# getenforce
Enforcing
```

```
~]# setenforce 0
~]# getenforce
Permissive
```

```
~]# setenforce 1
~]# getenforce
Enforcing
```

在 Red Hat Enterprise Linux 中，您可以在系统处于 enforcing 模式时将单个域设置为 permissive 模式。例如，使 **httpd_t** 域为 permissive 模式：

```
~]# semanage permissive -a httpd_t
```

如需更多信息，请参阅 第 11.3.4 节 “许可域”。

注意

第 4.4 节 “SELinux 状态和模式中的永久性更改” 涵盖了持久性状态和模式更改。

1.5. 其它资源

红帽身份管理(IdM)提供集中解决方案来定义 SELinux 用户映射。详情请参阅 Linux 域身份、身份验证和策略指南中的定义 SELinux 用户映射。

[1] 包括 DNS 服务器所使用的信息（如主机名到 IP 地址映射）的文本文件。

[2] 达利亚州马提提."服务器软件的空间：SELinux 阻止现实利用"。2008 年 2 月 24 日发布.访问日期：2009 年 8 月 27 日： http://www.networkworld.com/article/2283723/lan-wan/a-seatbelt-for-server-software--selinux-blocks-real-world-exploits.html

第 2 章 SELINUX CONTEXTS

进程和文件使用 SELinux 上下文标记，其中含有其他信息，如 SELinux 用户、角色、类型以及可选的级别。运行 SELinux 时，所有这些信息都用于做出访问控制决策。在红帽企业 Linux 中，SELinux 提供了基于角色的访问控制(RBAC)、类型强制(TE)和可选的多级别安全(MLS)的组合。

以下是显示 SELinux 上下文的示例。SELinux 上下文在运行 SELinux 的 Linux 操作系统上用于进程、Linux 用户和文件。使用以下命令查看文件和目录的 SELinux 上下文：

```
~]$ ls -Z file1
-rwxrw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0      file1
```

SELinux 上下文遵循 *SELinux user:role:type:level* 语法。这些字段如下：

SELinux user

SELinux 用户身份是策略已知的身份，它授权用于一组特定的角色，以及特定的 MLS/MCS 范围。每个 Linux 用户都使用 SELinux 策略映射到 SELinux 用户。这允许 Linux 用户继承对 SELinux 用户的限制。映射的 SELinux 用户身份在该会话中进程的 SELinux 上下文中使用，以定义它们可以进入哪些角色和级别。以 root 身份输入以下命令，查看 SELinux 和 Linux 用户帐户之间的映射列表（您需要安装 policycoreutils-python 软件包）：

```
~]# semanage login -l
Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

输出可能因系统稍有不同：

- **Login Name** 列中列出了 Linux 用户。
- **SELinux 用户** 列中列出了 Linux 用户映射至哪个 SELinux 用户。对于进程，SELinux 用户限制可以访问哪些角色和级别。
- **MLS/MCS Range** 列是多级别安全(MLS)和多类别安全(MCS)使用的级别。
- **Service** 列决定了正确的 SELinux 上下文，在该上下文中，Linux 用户应当登录该系统。默认情况下，使用星号(*)字符，它代表任何服务。

role

SELinux 的一部分是基于角色的访问控制(RBAC)安全模型。该角色是 RBAC 的一个属性。SELinux 用户获得角色的授权，并且域对角色进行了授权。该角色充当域和 SELinux 用户之间的中介。可以输入的角色决定了可以输入哪些域；最后，这控制可以访问哪些对象类型。这有助于降低对特权升级攻击的漏洞。

type

type 是 Type Enforcement 的属性。类型定义进程的域，以及文件的类型。SELinux 策略规则定义类型如何互相访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

level

级别是 MLS 和 MCS 的属性。MLS 范围是一对级别，如果级别不同，则写为 *低级别-高级别*，*如果级别相同（**s0-s0** 与 **s0** 相同）。每个级别都是敏感度类别对，类别是可选的。如果有类别，则该级别被写为 敏感度：category-set。如果没有类别，它将被写为 敏感度。*

*如果设置的类别是连续的序列，则可缩写它。例如，**c0.c3** 与 **c0,c1,c2,c3** 相同。**/etc/selinux/targeted/setrans.conf** 文件将级别 **(s0:c0)** 映射到人类可读形式（即 **CompanyConfidential**）。在红帽企业 Linux 中，目标策略强制 MCS，在 MCS 中，只有一个敏感度 **s0**。红帽企业 Linux 中的 MCS 支持 1024 个不同的类别：c **0** 到 **c1023**。s**0-s0:c0.c1023** 具有敏感度 **s0**，并授权所有类别。*

MLS 强制执行 Bell-La Padula Mandatory 访问模型，用于标记安全保护配置文件(LSPP)环境。要使用 MLS 限制，请安装 selinux-policy-mls 软件包，并将 MLS 配置为默认 SELinux 策略。Red Hat Enterprise Linux 附带的 MLS 策略省略了不属于评估配置的许多程序域，因此桌面工作站上的 MLS 不可用（不支持 X Window 系统）；但是，可以构建包含所有程序域的 MLS 策略。有关 MLS 配置的详情请参考 第 4.13 节 "多级别安全（MLS）"。

2.1. 域转换

*个域中的进程通过执行具有新域 **入口点** 类型的应用，过渡到另一个域。**入口点** 权限用于 SELinux 策略，并控制哪些应用程序可用于进入域。以下示例演示了一个域转换：*

过程 2.1. 域转换示例

1. *用户想要更改其密码。为此，他们将运行 **passwd** 实用程序。**/usr/bin/passwd** 可执行文件使用 **passwd_exec_t** 类型标记：*

   ```
   ~]$ ls -Z /usr/bin/passwd
   -rwsr-xr-x  root root system_u:object_r:passwd_exec_t:s0 /usr/bin/passwd
   ```

   ***passwd** 实用程序访问 **/etc/shadow**，其标有 **shadow_t** 类型：*

   ```
   ~]$ ls -Z /etc/shadow
   -r--------. root root system_u:object_r:shadow_t:s0    /etc/shadow
   ```

2. *SELinux 策略规则规定，在 **passwd_t** 域中运行的进程被允许读取和写入使用 **shadow_t** 类型标记的文件。**shadow_t** 类型仅应用于密码更改所需的文件。这包括 **/etc/gshadow**、**/etc/shadow** 及其备份文件。*

3. *SELinux 策略规则指出，**passwd_t 域 的入口点** 权限设置为 **passwd_exec_t** 类型。*

4. *用户运行 **passwd** 实用程序时，用户的 shell 进程将过渡到 **passwd_t** 域。使用 SELinux 时，由于默认操作是拒绝，并且存在允许 **passwd_t** 域中运行的（其他事情）应用访问 **shadow_t** 类型的文件的规则，passwd **应用** 被允许访问 **/etc/shadow**，并更新用户的密码。*

*这个示例并不详尽，用作解释域转换的基本示例。尽管有一个实际规则允许 **passwd_t** 域中运行的主题访问标有 **shadow_t** 文件类型的对象，但必须满足其他 SELinux 策略规则，才能让使用者转换到新域。在本例中，Type Enforcement 可确保：*

- ***passwd_t** 域只能通过执行标有 **passwd_exec_t** 类型的应用来输入；只能从授权的共享库（如 **lib_t** 类型）执行；并且无法执行任何其他应用。*
- *只有授权的域（如 **passwd_t**）可以写入到带有 **shadow_t** 类型标记的文件。即使其他进程使用超级用户特权运行，这些进程也无法写入带有 **shadow_t** 类型标记的文件，因为它们不在 **passwd_t** 域中运行。*

- 只有授权的域可以转换到 **passwd_t** 域。例如，在 **sendmail _t 域中运行的 sendmail** 进程没有执行 **passwd** 的合法原因；因此，它永远不会转换到 **passwd_t** 域。
- 在 **passwd_t** 域中运行的进程只能读取和写入授权类型，例如使用 **etc_t 或 shadow_t** 类型标记的文件。这可防止 **passwd** 应用被欺骗为读取或编写任意文件。

2.2. 进程的 SELINUX 上下文

使用 **ps -eZ** 命令查看进程的 SELinux 上下文。例如：

过程 2.2. 查看 passwd 实用程序的 SELinux 上下文

1. 打开终端，如 **Applications → System Tools → Terminal**。
2. 运行 **passwd** 实用程序。不要输入新密码：

```
~]$ passwd
Changing password for user user_name.
Changing password for user_name.
(current) UNIX password:
```

3. 打开一个新标签页或其他终端，再输入以下命令。输出结果类似以下：

```
~]$ ps -eZ | grep passwd
unconfined_u:unconfined_r:passwd_t:s0-s0:c0.c1023 13212 pts/1 00:00:00 passwd
```

4. 在第一个标签页/终端中，按 **Ctrl+C** 以取消 **passwd** 工具。

在本例中，当执行 **passwd** 实用程序（使用 **passwd_exec_t** 类型标记）时，用户的 shell 进程将过渡到 **passwd_t** 域。请记住，类型定义了进程的域，以及文件的类型。

要查看所有正在运行的进程的 SELinux 上下文，请再次运行 **ps** 实用程序。请注意，以下是输出结果的截断示例，在您的系统上可能有所不同：

```
]$ ps -eZ
system_u:system_r:dhcpc_t:s0            1869 ?  00:00:00 dhclient
system_u:system_r:sshd_t:s0-s0:c0.c1023  1882 ?  00:00:00 sshd
system_u:system_r:gpm_t:s0              1964 ?  00:00:00 gpm
system_u:system_r:crond_t:s0-s0:c0.c1023 1973 ?  00:00:00 crond
system_u:system_r:kerneloops_t:s0       1983 ?  00:00:05 kerneloops
system_u:system_r:crond_t:s0-s0:c0.c1023 1991 ?  00:00:00 atd
```

system_r 角色用于系统进程，如守护进程。类型强制随后分隔每个域。

2.3. 用户的 SELINUX 上下文

使用以下命令查看与您的 Linux 用户关联的 SELinux 上下文：

```
~]$ id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

在 Red Hat Enterprise Linux 中，Linux 用户默认运行无限。此 SELinux 上下文显示，Linux 用户映射到 SELinux **unconfined_u** 用户，以 **unconfined_r** 角色运行，并在 **unconfined_t** 域中运行。s **0-s0** 是

*MLS 范围，本例中仅与 **s0** 相同。用户有权访问的类别由 **c0.c1023** 定义，这是所有类别（c**0** 到 **c1023**）。*

第 3 章 目标策略

targeted 策略是 Red Hat Enterprise Linux 中使用的默认 SELinux 策略。使用 targeted 策略时，目标在受限域中运行的进程以及不是目标的进程在未限制的域中运行。例如，默认情况下，登录用户在 **unconfined_t** 域中运行，由 init 启动的系统进程在 **unconfined_service_t** 域中运行；这两个域都是未限制的。

可执行和可写入的内存检查可能同时适用于受限制和不受限制的域。但是，默认情况下，在不受限制的域中运行的进程可以分配可写入内存并执行它。这些内存检查可以通过设置布尔值来启用，这将允许在运行时修改 SELinux 策略。稍后将探讨布尔值配置。

3.1. 受限制的进程

侦听网络的所有服务（如 **sshd** 或 **httpd**）都受到 Red Hat Enterprise Linux 的限制。此外，大多数以 root 用户身份运行并执行用户任务的进程（如 **passwd** 实用程序）都会受限制。当进程受限制时，它会在自己的域中运行，如 **httpd_t 域中运行的 httpd** 进程。如果一个受攻击者限制的进程受到 SELinux 策略配置的影响，攻击者对资源的访问权限和可能受到的破坏会受到限制。

完成这个步骤以确保启用 SELinux，并准备执行以下示例：

过程 3.1. 如何验证 SELinux 状态

1. 确认 SELinux 已启用，正在以强制模式运行，并且正在使用 targeted 策略。正确的输出应类似如下：

```
~]$ sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Max kernel policy version:      30
```

   有关更改 SELinux 模式的详情，请查看 第 4.4 节 "SELinux 状态和模式中的永久性更改"。

2. 以 root 用户身份，在 **/var/www/html/** 目录中创建一个文件：

```
~]# touch /var/www/html/testfile
```

3. 输入以下命令查看新创建的文件的 SELinux 上下文：

```
~]$ ls -Z /var/www/html/testfile
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/testfile
```

   默认情况下，Linux 用户在 Red Hat Enterprise Linux 中运行没有限制，这就是为何 **testfile** 文件带有 SELinux **unconfined_u** 用户标记的原因。RBAC 用于进程，而非文件。角色对文件没有意义；object **_r** 角色是用于文件的通用角色（在持久存储和网络文件系统上）。在 **/proc** 目录下，与进程相关的文件可以使用 **system_r** 角色。**httpd_sys_content_t** 类型允许 **httpd** 进程访问此文件。

以下示例演示了 SELinux 如何阻止 Apache HTTP 服务器(**httpd**)读取未正确标记的文件，例如供 Samba 使用的文件。这是个示例，不应在生产环境中使用。它假定已安装了 httpd 和 wget 软件包，使用了 SELinux 目标策略，并且 SELinux 处于强制模式。

过程 3.2. 受限进程示例

1. 以 root 用户身份，启动 **httpd** 守护进程：

```
~]# systemctl start httpd.service
```

确认 服务正在运行。输出中应包括以下信息（只有时间戳有所不同）：

```
~]$ systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since Mon 2013-08-05 14:00:55 CEST; 8s ago
```

2. 更改到 Linux 用户对其具有写入权限的目录，并输入以下命令。除非对默认配置进行了更改，否则这个命令会成功：

```
~]$ wget http://localhost/testfile
--2009-11-06 17:43:01--  http://localhost/testfile
Resolving localhost... 127.0.0.1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 0 [text/plain]
Saving to: `testfile'

[ <=>                                 ] 0      --.-K/s   in 0s

2009-11-06 17:43:01 (0.00 B/s) - `testfile' saved [0/0]
```

3. **chcon** 命令重新标记文件；但是，当文件系统重新标记时，此类标签更改不会保留。若要在文件系统重新标记后保留永久更改，请使用 **semanage** 实用程序，稍后将对此进行讨论。以 root 用户身份，输入以下命令将类型更改为 Samba 使用的类型：

```
~]# chcon -t samba_share_t /var/www/html/testfile
```

输入以下命令查看更改：

```
~]$ ls -Z /var/www/html/testfile
-rw-r--r--  root root unconfined_u:object_r:samba_share_t:s0 /var/www/html/testfile
```

4. 请注意，当前的 DAC 权限允许 **httpd** 进程访问 **testfile**。更改到您的用户具有写入访问权限的目录，并输入以下命令。除非对默认配置进行了更改，否则这个命令会失败：

```
~]$ wget http://localhost/testfile
--2009-11-06 14:11:23--  http://localhost/testfile
Resolving localhost... 127.0.0.1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 403 Forbidden
2009-11-06 14:11:23 ERROR 403: Forbidden.
```

5. 以 root 用户身份，删除 **testfile** ：

```
~]# rm -i /var/www/html/testfile
```

6. 如果您不需要 **httpd** 以 root 用户身份运行，请输入以下命令停止它：

```
~]# systemctl stop httpd.service
```

这个示例演示了 SELinux 增加的额外安全性。虽然 DAC 规则允许 **httpd** 进程在第 2 步中访问 **testfile**，因为该文件使用 **httpd** 进程无法访问的类型进行标记，但 SELinux 会拒绝访问。

如果 **auditd** 守护进程正在运行，类似如下的错误会记录到 **/var/log/audit/audit.log** 中：

```
type=AVC msg=audit(1220706212.937:70): avc:  denied  { getattr } for  pid=1904 comm="httpd"
path="/var/www/html/testfile" dev=sda5 ino=247576 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0  tclass=file

type=SYSCALL msg=audit(1220706212.937:70): arch=40000003 syscall=196 success=no exit=-13
a0=b9e21da0 a1=bf9581dc a2=555ff4 a3=2008171 items=0 ppid=1902 pid=1904 auid=500 uid=48
gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=1 comm="httpd"
exe="/usr/sbin/httpd" subj=unconfined_u:system_r:httpd_t:s0 key=(null)
```

另外，类似如下的错误会记录到 **/var/log/httpd/error_log** 中：

```
[Wed May 06 23:00:54 2009] [error] [client 127.0.0.1] (13)Permission denied: access to /testfile
denied
```

3.2. 未限制的进程

未限制的进程在不受限制的域中运行，例如 **init** 执行的非限制服务最终在 **unconfined_service_t** 域中运行，内核执行的无限制服务最终在 **kernel_t** 域中运行，未限制的 Linux 用户执行的无限制服务最终在 **unconfined_t** 域中运行。对于不受限制的进程，会应用 SELinux 策略规则，但存在允许无限制域中运行的进程的策略规则几乎所有访问权限。在不受限制的域中运行的进程只能回退到使用 DAC 规则。如果不受限制的进程受到破坏，SELinux 不会阻止攻击者获得系统资源和数据的访问权限，当然，仍会使用 DAC 规则。SELinux 是 DAC 规则基础上的安全增强 - 它不会取代它们。

要确定启用了 SELinux 且系统已准备好执行以下示例，请完成 第 3.1 节 "受限制的进程" 所述的 过程 3.1, "如何验证 SELinux 状态"。

以下示例演示了在运行 unconfined 时，Apache HTTP 服务器(httpd)如何访问供 Samba 使用的数据。请注意，在 Red Hat Enterprise Linux 中，默认情况下 **httpd** 进程在受限的 **httpd_t** 域中运行。这是个示例，不应在生产环境中使用。它假定已安装了 httpd、wget、dbus 和 audit 软件包，使用了 SELinux targeted 策略，并且 SELinux 处于强制模式。

过程 3.3. Unconfined 进程示例

1. **chcon** 命令重新标记文件；但是，当文件系统重新标记时，此类标签更改不会保留。若要在文件系统重新标记后保留永久更改，请使用 **semanage** 实用程序，稍后将对此进行讨论。以 root 用户身份，输入以下命令将类型更改为 Samba 使用的类型：

```
~]# chcon -t samba_share_t /var/www/html/testfile
```

查看更改：

```
~]$ ls -Z /var/www/html/testfile
-rw-r--r--  root root unconfined_u:object_r:samba_share_t:s0 /var/www/html/testfile
```

2. 输入以下命令确认 **httpd** 进程没有在运行：

```
~]$ systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: inactive (dead)
```

如果输出不同，以 root 用户身份输入以下命令停止 **httpd** 进程：

```
~]# systemctl stop httpd.service
```

3. 要使 **httpd** 进程以 root 用户身份运行不受限制，请输入以下命令将 **/usr/sbin/httpd** 文件的类型更改为没有转换到受限域的类型：

```
~]# chcon -t bin_t /usr/sbin/httpd
```

4. 确认 **/usr/sbin/httpd** 使用 **bin_t** 类型标记：

```
~]$ ls -Z /usr/sbin/httpd
-rwxr-xr-x. root root system_u:object_r:bin_t:s0       /usr/sbin/httpd
```

5. 以 root 用户身份，启动 **httpd** 进程并确认它已成功启动：

```
~]# systemctl start httpd.service
```

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since Thu 2013-08-15 11:17:01 CEST; 5s ago
```

6. 输入以下命令查看在 **unconfined_service_t** 域中运行的 **httpd** ：

```
~]$ ps -eZ | grep httpd
system_u:system_r:unconfined_service_t:s0 11884 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11885 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11886 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11887 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11888 ? 00:00:00 httpd
system_u:system_r:unconfined_service_t:s0 11889 ? 00:00:00 httpd
```

7. 更改到 Linux 用户对其具有写入权限的目录，并输入以下命令。除非对默认配置进行了更改，否则这个命令会成功：

```
~]$ wget http://localhost/testfile
--2009-05-07 01:41:10--  http://localhost/testfile
Resolving localhost... 127.0.0.1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
```

```
Length: 0 [text/plain]
Saving to: `testfile'

[ <=>                   ]--.-K/s   in 0s

2009-05-07 01:41:10 (0.00 B/s) - `testfile' saved [0/0]
```

虽然 **httpd** 进程无法访问使用 **samba_share_t** 类型标记的文件，但 **httpd** 在未限制的 **unconfined_service_t** 域中运行，并返回使用 DAC 规则，因此 **wget 命令可以** 成功运行。如果 **httpd** 在受限的 **httpd_t** 域中运行，**wget** 命令会失败。

8. **restorecon** 实用程序恢复文件的默认 SELinux 上下文。以 root 用户身份，输入以下命令恢复 **/usr/sbin/httpd** 的默认 SELinux 上下文：

```
~]# restorecon -v /usr/sbin/httpd
restorecon reset /usr/sbin/httpd context system_u:object_r:unconfined_exec_t:s0->system_u:object_r:httpd_exec_t:s0
```

确认 **/usr/sbin/httpd** 使用 **httpd_exec_t** 类型标记：

```
~]$ ls -Z /usr/sbin/httpd
-rwxr-xr-x  root root system_u:object_r:httpd_exec_t:s0 /usr/sbin/httpd
```

9. 以 root 身份，输入以下命令来重启 **httpd**。重启后，确认 **httpd** 是否在受限的 **httpd_t** 域中运行：

```
~]# systemctl restart httpd.service
```

```
~]$ ps -eZ | grep httpd
system_u:system_r:httpd_t:s0    8883 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8884 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8885 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8886 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8887 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8888 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    8889 ?        00:00:00 httpd
```

10. 以 root 用户身份，删除 **testfile** ：

```
~]# rm -i /var/www/html/testfile
rm: remove regular empty file `/var/www/html/testfile'? y
```

11. 如果您不需要 **httpd** 运行，请以 root 用户身份输入以下命令来停止 **httpd** ：

```
~]# systemctl stop httpd.service
```

这些部分中的示例演示了如何保护数据不受被入侵限制进程的影响（由 SELinux 保护），以及攻击者如何更加轻松地访问数据免受损坏的非限制进程（不受 SELinux 保护）。

3.3. 受限制和未限制的用户

每个 Linux 用户都使用 SELinux 策略映射到 SELinux 用户。这可允许 Linux 用户继承对 SELinux 用户的限制。以 root 用户身份运行 **semanage login -l** 命令来查看这个 Linux 用户映射：

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

在 Red Hat Enterprise Linux 中，Linux 用户默认映射到 SELinux **__default__** 登录，这映射到 SELinux **unconfined_u** 用户。下面一行定义了默认映射：

```
__default__             unconfined_u            s0-s0:c0.c1023
```

以下步骤演示了如何向系统添加新的 Linux 用户以及如何将该用户映射到 SELinux **unconfined_u** 用户。它假设 root 用户运行没有限制，这与 Red Hat Enterprise Linux 中的默认设置相同：

过程 3.4. 将新 Linux 用户映射到 SELinux unconfined_u 用户

1. 以 root 用户身份，输入以下命令创建一个名为 **newuser** 的新 Linux 用户：

   ```
   ~]# useradd newuser
   ```

2. 要将密码分配给 Linux **newuser** 用户：以 root 用户身份输入以下命令：

   ```
   ~]# passwd newuser
   Changing password for user newuser.
   New UNIX password: Enter a password
   Retype new UNIX password: Enter the same password again
   passwd: all authentication tokens updated successfully.
   ```

3. 从当前会话注销，然后以 Linux **newuser** 用户身份登录。登录时，**pam_selinux** PAM 模块会自动将 Linux 用户映射到 SELinux 用户（本例中为 **unconfined_u**），并设置生成的 SELinux 上下文。然后，将使用此上下文启动 Linux 用户的 shell。输入以下命令查看 Linux 用户的上下文：

   ```
   [newuser@localhost ~]$ id -Z
   unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
   ```

   注意

   如果您不再需要系统上的 **newuser** 用户，请注销 Linux **newuser** 的会话，使用您的帐户登录，然后以 root 用户身份运行 **userdel -r newuser** 命令。它将删除 **新用户** 及其主目录。

受限制和不受限制的 Linux 用户会受到可执行和可写入的内存检查，也受到 MCS 或 MLS 的限制。

要列出可用的 SELinux 用户，请输入以下命令：

```
~]$seinfo -u
Users: 8
   sysadm_u
```

```
system_u
xguest_u
root
guest_u
staff_u
user_u
unconfined_u
```

请注意，**seinfo** 命令由 setools-console 软件包提供，该软件包默认不安装。

如果一个未限制的 Linux 用户执行一个应用程序，这个应用程序被 SELinux 策略定义为可以从 **unconfined_t** 域转换到其自身限制域的应用程序，则未限制的 Linux 用户仍会受到那个受限制域的限制。这样做的安全优点是，即使 Linux 用户的运行没有限制，但应用程序仍受限制。因此，对应用程序中漏洞的利用会被策略限制。

同样，我们可以将这些检查应用到受限制的用户。每个受限制的 Linux 用户都受到一个受限的用户域的限制。SELinux 策略还可定义从受限制的用户域转换到自己受限制的目标域转换。在这种情况下，受限制的 Linux 用户会受到那个目标受限域的限制。重点是，根据用户的角色，把特定的权限与受限制的用户相关联。在下表中，您可以看到 Red Hat Enterprise Linux 中 Linux 用户的基本受限制域示例：

表 3.1. SELinux 用户功能

用户	角色	域	X 窗口系统	su 或 sudo	在主目录和 /tmp 中执行（默认）	网络
sysadm_u	sysadm_r	sysadm_t	是	**su** 和 **sudo**	是	是
staff_u	staff_r	staff_t	是	仅 **sudo**	是	是
user_u	user_r	user_t	是	否	是	是
guest_u	guest_r	guest_t	否	否	是	否
xguest_u	xguest_r	xguest_t	是	否	是	仅 Firefox

- **user_t**、**guest_t** 和 **xguest_t** 域中的 Linux 用户只能在 SELinux 策略允许的情况下运行 set 用户 ID(setuid)应用（如 **passwd**）。这些用户无法运行 **su** 和 **sudo** setuid 应用程序，因此无法使用这些应用程序成为 root 用户。
- **sysadm_t**、**staff_t**、**user _t** 和 **xguest_t** 域中的 Linux 用户可以使用 X Window 系统和终端登录。
- 默认情况下，**staff_t** 中的 Linux 用户、**user_t**、**guest_t** 和 **xguest_t** 域中的 Linux 用户可以在其主目录和 **/tmp** 中执行应用程序。要防止他们在他们有写入访问权限的目录中执行应用程序（继承用户的权限），将 **guest_exec_content** 和 **xguest_exec_content** 布尔值设置为 **off**。这有助于防止有缺陷或恶意的应用程序修改用户的文件。

  有关允许和阻止用户在主目录和 **/tmp** 中执行应用程序的详情，请查看 第 6.6 节 “用户执行应用程序的布尔值”。
- **xguest_t** 域中的唯一网络访问 Linux 用户是 **Firefox** 连接到网页。

请注意，**system_u** 是系统进程和对象的特殊用户身份。它绝对不能和 Linux 用户关联。此外，**unconfined_u** 和 **root** 是未限制的用户。因此，它们没有包括在上述 SELinux 用户功能表中。

除了已提到的 SELinux 用户外，还有特殊的角色可以映射到这些用户。这些角色决定了 SELinux 允许这些用户可以做什么：

- **webadm_r** 只能管理与 Apache HTTP 服务器相关的 SELinux 类型。如需更多信息，请参阅 第 13.2 节 "类型"。
- **dbadm_r** 只能管理与 MariaDB 数据库和 PostgreSQL 数据库管理系统相关的 SELinux 类型。如需更多信息，请参阅 第 20.2 节 "类型" 和 第 21.2 节 "类型"。
- **logadm_r** 只能管理与 **syslog** 和 **auditlog** 进程相关的 SELinux 类型。
- **secadm_r** 只能管理 SELinux。
- **auditadm_r** 只能管理与 **audit** 子系统相关的进程。

要列出所有可用的角色，请输入以下命令：

```
~]$ seinfo -r
```

如前文所述，se **info** 命令由 setools-console 软件包提供，该软件包默认不安装。

3.3.1. sudo 转换和 SELinux 角色

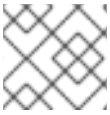
在某些情况下，受限制的用户需要执行需要 root 特权的管理任务。为此，这种受限用户必须使用 **sudo** 命令获得 受限的管理员 SELinux 角色。**sudo** 命令用于为可信用户授予管理访问权限。当用户在管理命令之前加上 **sudo** 时，系统会提示他们输入 自己的 密码。然后，当它们经过身份验证并假定允许 命令时，将像 root 用户一样执行管理命令。

如 表 3.1 "SELinux 用户功能" 所示，默认情况下，只有 **staff_u 和 sysadm_u** SELinux 受限用户可以使用 **sudo**。当此类用户通过 **sudo** 执行命令时，可以根据 **/etc/sudoers 配置文件中指定的规则或在 /etc/sudoers.d/** 目录中相应的规则更改其角色（如果存在此类文件）。

有关 **sudo** 的更多信息，请参阅《红帽企业 Linux 7 系统管理员指南》中的 获取 特权 章节。

过程 3.5. 配置 sudo 转换

此流程演示了如何设置 **sudo** 将新创建的 SELinux_user_u 受限用户从 default_role_r 转换为 administrator_r 管理员角色。



注意

要为已存在的 SELinux 用户配置受限管理员角色，请跳过前两个步骤。

1. 创建一个新的 SELinux 用户，并为这个用户指定默认 SELinux 角色和一个补充的受限管理员角色：

```
~]# semanage user -a -r s0-s0:c0.c1023 -R "default_role_r administrator_r" SELinux_user_u
```

2. 设置默认的 SElinux 策略上下文文件。例如，若要与 **staff_u SELinux 用户使用相同的 SELinux 规则**，请复制 **staff_u** 上下文文件：

```
~]# cp /etc/selinux/targeted/contexts/users/staff_u
/etc/selinux/targeted/contexts/users/SELinux_user_u
```

3. 将新创建的 SELinux 用户映射到现有 Linux 用户：

```
semanage login -a -s SELinux_user_u -rs0:c0.c1023 linux_user
```

4. 在 **/etc/sudoers.d/** 目录中，使用与您的 Linux 用户相同的名称创建新配置文件，并将以下字符串添加到其中：

```
~]# echo "linux_user ALL=(ALL) TYPE=administrator_t ROLE=administrator_r /bin/bash " >
/etc/sudoers.d/linux_user
```

5. 使用 **restorecon** 工具重新标记 linux_user 主目录：

```
~]# restorecon -FR -v /home/linux_user
```

6. 以新创建的 Linux 用户身份登录到该系统，检查用户是否标记了默认的 SELinux 角色：

```
~]$ id -Z
SELinux_user_u:default_role_r:SELinux_user_t:s0:c0.c1023
```

7. 运行 **sudo** 将用户的 SELinux 上下文更改为 **/etc/sudoers.d/linux_user** 中指定的辅助 SELinux 角色。与 **sudo** 一起使用的 **-i** 选项会导致执行交互式 shell：

```
~]$ sudo -i
~]# id -Z
SELinux_user_u:administrator_r:administrator_t:s0:c0.c1023
```

要更好地了解占位符，如 default_role_r 或 administrator_r，请参阅以下示例。

例 3.1. 配置 sudo 转换

这个示例创建一个新的 SELinux 用户 **confined_u**，默认分配的 **staff_r**，使用 **sudo** 将 **confined_u** 从 **staff_r** 的角色更改为 **webadm_r**。

- 以 root 用户身份在 **sysadm_r** 或 **unconfined_r** 角色中输入所有命令。

```
~]# semanage user -a -r s0-s0:c0.c1023 -R "staff_r webadm_r" confined_u
~]# cp /etc/selinux/targeted/contexts/users/staff_u
/etc/selinux/targeted/contexts/users/confined_u
~]# semanage login -a -s confined_u -rs0:c0.c1023 linux_user
~]# restorecon -FR -v /home/linux_user
~]# echo "linux_user ALL=(ALL) ROLE=webadm_r TYPE=webadm_t /bin/bash " >
/etc/sudoers.d/linux_user
```

- 以新创建的 Linux 用户身份登录到该系统，检查用户是否标记了默认的 SELinux 角色：

```
~]$ id -Z
confined_u:staff_r:staff_t:s0:c0.c1023
~]$ sudo -i
```

```
~]# id -Z
confined_u:webadm_r:webadm_t:s0:c0.c1023
```

第 4 章 使用 SELINUX

*以下小节简要介绍 Red Hat Enterprise Linux 中的主要 SELinux 软件包；安装和更新软件包；使用哪些日志文件；主要的 SELinux 配置文件；启用和禁用 SELinux；SELinux 模式；临时和永久更改文件和目录标签；使用 **mount** 命令覆盖文件系统标签；挂载 NFS 卷，以及在复制和存档文件和目录时保留 SELinux 上下文。*

4.1. SELINUX 软件包

在 Red Hat Enterprise Linux 完整安装中，SELinux 软件包会被默认安装，除非在安装过程中手动排除它们。如果以文本模式执行最小安装，则默认情况下不会安装 policycoreutils-python 和 policycoreutils-gui 软件包。此外，SELinux 默认以强制模式运行，并使用 SELinux 目标策略。默认情况下，您的系统中会安装以下 SELinux 软件包：

- *policycoreutils 提供 **restorecon**、**secon**、**setfiles**、se**module**、load_**policy** 和 **setsebool** 等实用程序，用于操作和管理 SELinux。*
- *selinux-policy 提供 基本目录结构、**selinux-policy.conf** 文件和 RPM 宏。*
- *selinux-policy-targeted 提供了 SELinux targeted 策略。*
- *libselinux - 为 SELinux 应用程序提供 API.*
- *libselinux-utils 提供 **avcstat**、**getenforce**、**getsebool**、**matchpathcon**、selinux**conlist**、selinux**defcon**、selinux**enabled** 和 **setenforce** 实用程序。*
- *libselinux-python 为开发 SELinux 应用程序提供 Python 绑定。*

*默认情况下不会安装以下软件包，但可通过运行 **yum install <package-name>** 命令选择性地安装：*

- *selinux-policy-devel 提供用于创建自定义 SELinux 策略和策略模块的实用程序。*
- *selinux-policy-doc 提供说明如何使用各种服务配置 SELinux 的 man page。*
- *selinux-policy-mls 提供了 MLS（多级安全性）SELinux 策略。*
- *setroubleshoot-server 会在 SELinux 拒绝访问时将拒绝消息转换为此软件包中也提供的 **sealert** 实用程序可以查看的详细信息。*
- *setools-console 提供 Tresys Technology SETools 分发 、用于分析和查询策略、审计日志监控和报告以及文件上下文管理的实用程序和库。 setools 软件包是 SETools 的 meta-package。setools-gui 软件包提供了 apol **和 seaudit** 实用程序。setools-console 软件包提供了 **sechecker**、se**diff**、**seinfo**、**sesearch** 和 **findcon** 命令行实用程序。有关这些工具的信息，请参阅 Tresys Technology SETools 页面。请注意，只有启用 Red Hat Network Optional 频道时才会提供 setools 和 setools-gui 软件包。如需更多信息 ， 请参阅覆盖范围详情。*
- *mcstrans 级别（如 **s0-s0:c0.c1023** ）转换为更易于阅读的形式，如 **SystemLow-SystemHigh**。*
- *policycoreutils-python 提供 **semanage**、**audit 2allow**、**audit 2why** 和 **chcat** 等实用程序，用于操作和管理 SELinux。*
- *policycoreutils-gui 提供 **system-config-selinux**，这是一个用于管理 SELinux 的图形实用程序。*

4.2. 使用哪个日志文件

在 Red Hat Enterprise Linux 中，默认情况下会安装 dbus 和 audit 软件包，除非它们已从默认软件包选择中删除。setroubleshoot-server 必须使用 Yum 安装（使用 **yum install setroubleshoot-server** 命令）。

如果 **auditd** 守护进程正在运行，则 SELinux 拒绝信息会默认写入 **/var/log/audit/audit.log** ：

```
type=AVC msg=audit(1223024155.684:49): avc:  denied  { getattr } for  pid=2000 comm="httpd"
path="/var/www/html/file1" dev=dm-0 ino=399185 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=system_u:object_r:samba_share_t:s0 tclass=file
```

另外，类似以下的消息被写入 **/var/log/message** 文件中：

```
May 7 18:55:56 localhost setroubleshoot: SELinux is preventing httpd (httpd_t) "getattr" to
/var/www/html/file1 (samba_share_t). For complete SELinux messages. run sealert -l de7e30d6-
5488-466d-a606-92c9f40d316d
```

在红帽企业 Linux 7 中，**setroubleshootd** 不再作为服务持续运行。但是，它仍用于分析 AVC 信息。两个新程序可在需要时启动 **setroubleshoot** ：

- **sedispatch** 实用程序作为 **audit** 子系统的一部分运行。当返回 AVC 拒绝消息时，**sedispatch** 使用 **dbus** 发送消息。如果这些消息已在 **运行**，这些消息会直接设置。如果没有运行，则 **sedispatch 会自动** 启动它。
- **seapplet** 实用程序在系统工具栏中运行，等待 **setroubleshootd** 中的 dbus 消息。它启动通知 bubble，允许用户查看 AVC 消息。

过程 4.1. 自动启动守护进程

1. 要将 **auditd** 和 **rsyslog** 守护进程配置为在引导时自动启动，以 root 用户身份输入以下命令：

   ```
   ~]# systemctl enable auditd.service
   ```

   ```
   ~]# systemctl enable rsyslog.service
   ```

2. 要确保启用了守护进程，在 shell 提示符后输入以下命令：

   ```
   ~]$ systemctl is-enabled auditd
   enabled
   ```

   ```
   ~]$ systemctl is-enabled rsyslog
   enabled
   ```

   或者，使用 **systemctl status service-name.service** 命令并搜索命令输出 **中启用** 的关键字，例如：

   ```
   ~]$ systemctl status auditd.service | grep enabled
   auditd.service - Security Auditing Service
     Loaded: loaded (/usr/lib/systemd/system/auditd.service; enabled)
   ```

要了解更多有关 **systemd** 守护进程如何管理系统服务的信息，请参阅《系统管理员指南》中的管理系统服务 章节。

4.3. 主配置文件

/etc/selinux/config 文件是主要的 SELinux 配置文件。它控制 SELinux 是否已启用或禁用，以及使用了哪个 SELinux 模式和 SELinux 策略：

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

SELINUX=

***SELINUX** 选项设置 SELinux 是否已禁用或启用，以及在哪个模式下（enforcing 或 permissive）正在运行：*

- *使用 **SELINUX=enforcing** 时，SELinux 策略强制实施，SELinux 根据 SELinux 策略规则拒绝访问。记录拒绝消息。*
- *使用 **SELINUX=permissive** 时，不会强制执行 SELinux 策略。SELinux 不会拒绝访问，但会记录对于在强制模式下运行 SELinux 时拒绝的操作。*
- *当使用 **SELINUX=disabled** 时，SELinux 被禁用，SELinux 模块没有在 Linux 内核中注册，仅使用 DAC 规则。*

SELINUXTYPE=

***SELINUXTYPE** 选项设置要使用的 SELinux 策略。targeted 策略是默认策略。只有在您想要使用 MLS 策略时，才更改这个选项。有关如何启用 MLS 策略的详情请参考 第 4.13.2 节 "在 SELinux 中启用 MLS"。*

4.4. SELINUX 状态和模式中的永久性更改

如 第 1.4 节 "SELinux 状态和模式" 所述，SELinux 可以启用或禁用。启用后，SELinux 有两个模式：enforcing 和 permissive。

*使用 **getenforce** 或 **sestatus** 命令检查 SELinux 在哪个模式下运行。**getenforce** 命令返回 **Enforcing**、**Pmissive** 或 **Disabled**。*

***sestatus** 命令返回 SELinux 状态以及正在使用的 SELinux 策略：*

```
~]$ sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Max kernel policy version:      30
```

注意

当系统以 permissive 模式运行 SELinux 时，用户可以错误地标记文件。在禁用 SELinux 时创建的文件根本不标记。这个行为会在更改为 enforcing 模式时导致问题，因为文件被错误地标记或者根本不标记文件。为了防止不正确标记和未标记的文件造成问题，当从 disabled 状态改为 permissive 或 enforcing 模式时会自动重新标记文件系统。

4.4.1. 启用 SELinux

启用后，SELinux 可使用两种模式之一运行：enforcing 或 permissive。以下小节介绍了如何永久更改这些模式。

在之前禁用的系统中启用 SELinux 时，为了避免问题（如系统无法引导或进程失败），红帽建议按照以下步骤执行：

1. 以 permissive 模式启用 SELinux。如需更多信息，请参阅 第 4.4.1.1 节 “Permissive 模式”。
2. 重启您的系统。
3. 检查 SELinux 拒绝信息。如需更多信息，请参阅 第 11.3.5 节 “搜索和查看地址”。
4. 如果没有拒绝的操作，切换到 enforcing 模式。如需更多信息，请参阅 第 4.4.1.2 节 “强制模式”。

要在 enforcing 模式下使用 SELinux 运行自定义应用程序，请选择以下之一：

- 在 **unconfined_service_t** 域中运行您的应用程序。如需更多信息，请参阅 第 3.2 节 “未限制的进程”。
- 为应用程序编写新策略。如需更多信息，请参阅编写自定义 SELinux 策略知识库文章。

4.4.1.1. Permissive 模式

当 SELinux 是以 permissive 模式运行时，不会强制 SELinux 策略。系统可保持正常操作，SELinux 不会拒绝任何操作，而只是记录 AVC 信息，它们可用于故障排除、调试和 SELinux 策略改进。每个 AVC 在这个示例中仅记录一次。

要将模式永久改为 permissive，请按照以下步骤操作：

过程 4.2. 进入许可模式

1. 编辑 **/etc/selinux/config** 文件，如下所示：

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=permissive
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

2. 重启系统：

```
~]# reboot
```

4.4.1.2. 强制模式

当 SELinux 处于 enforcing 模式时，它会强制 SELinux 策略并根据 SELinux 策略规则拒绝访问。在 Red Hat Enterprise Linux 中，当系统最初使用 SELinux 安装时，默认启用 enforcing 模式。

如果禁用了 SELinux，请按照以下步骤将模式再次改为 enforcing：

过程 4.3. 进入强制模式

这个过程假设安装了 selinux-policy-targeted、selinux-policy、libselinux、libselinux-python、libselinux-utils、policycoreutils 和 policycoreutils-python 软件包。要验证软件包是否已安装，请使用以下命令：

```
rpm -q package_name
```

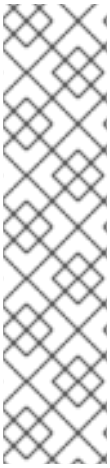
1. 编辑 **/etc/selinux/config** 文件，如下所示：

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

2. 重启系统：

```
~]# reboot
```

在下一次引导中，SELinux 会重新标记系统中的所有文件和目录，并为禁用 SELinux 时创建的文件和目录添加 SELinux 上下文。



注意

切换到 enforcing 模式后，SELinux 可能会因为不正确或缺少 SELinux 策略规则而拒绝某些操作。要查看 SELinux 拒绝的操作，以 root 用户身份输入以下命令：

```
~]# ausearch -m AVC,USER_AVC,SELINUX_ERR -ts today
```

另外，安装 setroubleshoot-server 软件包后，以 root 用户身份输入以下命令：

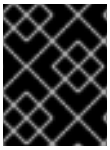
```
~]# grep "SELinux is preventing" /var/log/messages
```

如果 SELinux 拒绝某些操作，请参阅 第 11 章 故障排除 以获得有关故障排除的信息。

第 1.4 节 "SELinux 状态和模式" 涵盖了对模式的临时更改。

4.4.2. 禁用 SELinux

禁用 SELinux 时，SELinux 策略不被加载；它不会被强制执行，也不会记录 AVC 信息。因此，运行 第 1.1 节 "运行 SELinux 的好处" 中列出的 SELinux 的所有益处都丢失。



重要

红帽强烈建议您使用 permissive 模式，而不是永久禁用 SELinux。有关 permissive 模式的更多信息，请参阅 第 4.4.1.1 节 "Permissive 模式"。

要永久禁用 SELinux，请按照以下步骤执行：

过程 4.4. 禁用 SELinux

1. 在 **/etc/selinux/config** 文件中配置 **SELINUX=disabled** ：

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

2. 重启您的系统。重启后，确认 **getenforce** 命令返回 **Disabled**:

```
~]$ getenforce
Disabled
```

4.5. 在引导时更改 SELINUX 模式

在引导时，您可以设置几个内核参数来更改 SELinux 的运行方式：

enforcing=0

设置此参数可让系统以 permissive 模式启动，这在进行故障排除时非常有用。如果您的文件系统被破坏，使用 permissive 模式可能是唯一的选择。在 permissive 模式中，系统将继续正确创建标签。在这个模式中产生的 AVC 信息可能与 enforcing 模式不同。

在 permissive 模式中，只报告来自于同一拒绝的一系列操作的第一个拒绝信息。然而，在 enforcing 模式中，您可能会得到一个与读取目录相关的拒绝信息，应用程序将停止。在 permissive 模式中，您会得到相同的 AVC 信息，但应用程序将继续读取目录中的文件，并为因为每个拒绝额外获得一个 AVC。

selinux=0

这个参数会导致内核不载入 SELinux 构架的任意部分。初始化脚本会注意到系统使用 **selinux=0** 参数引导，并更改 **/.autorelabel** 文件。这会导致系统在下次使用 SELinux enabled 模式引导时自动重新标记。

重要

红帽不推荐使用 **selinux=0** 参数。要调试您的系统，首选使用 permissive 模式。

autorelabel=1

这个参数强制系统使用类似以下命令的重新标记：

```
~]# touch /.autorelabel
~]# reboot
```

如果系统标签包含大量错误，您可能需要以 permissive 模式引导，以便自动标签成功。

有关 SELinux 的其他内核引导参数，如 **checkreqprot**，请查看 **/usr/share/doc/kernel-doc-<KERNEL_VER>/Documentation/kernel-parameters.txt** 文件。本文档随 kernel-doc 软件包一起安装。使用安装的内核的版本号替换 <KERNEL_VER> 字符串，例如：

```
~]# yum install kernel-doc
~]$ less /usr/share/doc/kernel-doc-3.10.0/Documentation/kernel-parameters.txt
```

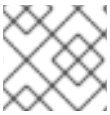
4.6. 布尔值

布尔值允许在运行时更改 SELinux 策略的部分，而不必知晓 SELinux 策略的编写。这允许更改，例如允许服务访问 NFS 卷，而无需重新加载或重新编译 SELinux 策略。

4.6.1. 列出布尔值

有关布尔值列表，说明每个布尔值是什么，以及它们是开启还是关闭，以 Linux root **用户身份运行 semanage boolean -l** 命令。以下示例没有列出所有布尔值，输出会为了简洁而缩短：

```
~]# semanage boolean -l
SELinux boolean                 State  Default Description


smartmon_3ware                  (off  ,  off)  Determine whether smartmon can...
mpd_enable_homedirs             (off  ,  off)  Determine whether mpd can traverse...
```



注意

要获得更详细的描述，请安装 selinux-policy-devel 软件包。

SELinux 布尔值 列中列出了布尔值名称。**Description** 列中列出了布尔值是 on 还是 off，以及它们的作用。

getsebool -a 命令列出布尔值，无论布尔值是开启还是关闭，但不给出每个布尔值的说明。以下示例没有列出所有布尔值：

```
~]$ getsebool -a
cvs_read_shadow --> off
daemons_dump_core --> on
```

运行 **getsebool boolean-name** 命令以仅列出 boolean-name 布尔值的状态：

```
~]$ getsebool cvs_read_shadow
cvs_read_shadow --> off
```

使用空格分隔列表列出多个布尔值：

```
~]$ getsebool cvs_read_shadow daemons_dump_core
cvs_read_shadow --> off
daemons_dump_core --> on
```

4.6.2. 配置布尔值

在 **setsebool boolean_name on/off** 表单中运行 setsebool 实用程序来启用或禁用布尔值。

以下示例演示了配置 **httpd_can_network_connect_db** 布尔值：

过程 4.5. 配置布尔值

1. 默认情况下，httpd **_can_network_connect_db** 布尔值是 off，阻止 Apache HTTP 服务器脚本和模块连接到数据库服务器：

   ```
   ~]$ getsebool httpd_can_network_connect_db
   httpd_can_network_connect_db --> off
   ```

2. 要临时启用 Apache HTTP 服务器脚本和模块来连接到数据库服务器，以 root 用户身份输入以下命令：

   ```
   ~]# setsebool httpd_can_network_connect_db on
   ```

3. 使用 **getsebool** 工具验证该布尔值是否已启用：

   ```
   ~]$ getsebool httpd_can_network_connect_db
   httpd_can_network_connect_db --> on
   ```

   这允许 Apache HTTP 服务器脚本和模块连接到数据库服务器。

4. 此更改在重新启动后不会保留。要在重启后保留更改，以 root 用户身份在命令 **中运行 setsebool -P boolean-name** .[3]

   ```
   ~]# setsebool -P httpd_can_network_connect_db on
   ```

4.6.3. Shell 自动完成

可以通过 **getsebool、setsebool 和 semanage** 实用程序使用 **shell 自动完成。将自动完成与 getsebool 和 setsebool 一起使用以完成命令行参数和布尔值。要只列出命令行参数，请在命令名称后面添加连字符("-")，然后按 Tab 键 ：**

```
~]# setsebool -[Tab]
-P
```

要完成布尔值，请开始写入布尔值名称，然后按 Tab：

```
~]$ getsebool samba_[Tab]
samba_create_home_dirs   samba_export_all_ro      samba_run_unconfined
samba_domain_controller  samba_export_all_rw      samba_share_fusefs
samba_enable_home_dirs   samba_portmapper         samba_share_nfs
```

```
~]# setsebool -P virt_use_[Tab]
virt_use_comm     virt_use_nfs      virt_use_sanlock
virt_use_execmem  virt_use_rawip    virt_use_usb
virt_use_fusefs   virt_use_samba    virt_use_xserver
```

semanage 实用程序与多个命令行参数一同使用，这些参数逐个完成。semanage 命令的第一个参数是一个 选项，用于指定管理 SELinux 策略的哪个部分：

```
~]# semanage [Tab]
boolean     export      import      login       node        port
dontaudit   fcontext    interface   module      permissive  user
```

然后，一个或多个命令行参数如下：

```
~]# semanage fcontext -[Tab]
-a          -D          --equal      --help       -m          -o
--add       --delete    -f           -l           --modify    -S
-C          --deleteall  --ftype     --list       -n          -t
-d          -e          -h           --locallist  --noheading  --type
```

最后，完成特定 SELinux 条目的名称，如布尔值、SELinux 用户、域或其他条目。开始输入条目并按 Tab：

```
~]# semanage fcontext -a -t samba<tab>
samba_etc_t                  samba_secrets_t
sambagui_exec_t              samba_share_t
samba_initrc_exec_t          samba_unconfined_script_exec_t
samba_log_t                  samba_unit_file_t
samba_net_exec_t
```

命令行参数可以链接到命令中：

```
~]# semanage port -a -t http_port_t -p tcp 81
```

4.7. SELINUX 上下文 - 标记文件

在运行 SELinux 的系统上，所有进程和文件都采用代表安全相关信息的标记方式。此信息称为 SELinux 上下文。对于文件，可使用 ls -Z 命令查看：

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

在本例中，SELinux 提供了用户(unconfined_u)、角色(object_r)、类型(user_home_t)和级别(s0)。此信息用于制定访问控制决策。在 DAC 系统上，访问权限根据 Linux 用户和组群 ID 进行控制。在 DAC 规则后检查 SELinux 策略规则。如果 DAC 规则首先拒绝访问，则不会使用 SELinux 策略规则。

注意

默认情况下，新建的文件和目录继承其父目录的 SELinux 类型。例如，当在 /etc 目录中使用 etc_t 类型创建新文件时，新文件将继承同一类型：

```
~]$ ls -dZ - /etc
drwxr-xr-x. root root system_u:object_r:etc_t:s0       /etc
```

```
~]# touch /etc/file1
```

```
~]# ls -lZ /etc/file1
-rw-r--r--. root root unconfined_u:object_r:etc_t:s0   /etc/file1
```

SELinux 提供多个命令来管理文件系统标签，如 chcon、semanage fcontext、restorecon 和 matchpathcon。

4.7.1. 临时更改：chcon

chcon 命令更改文件的 SELinux 上下文。但是，通过 chcon 命令所做的更改不会在文件系统重新标记或执行 restorecon 命令之间保留。SELinux 策略控制用户是否能够修改任何给定文件的 SELinux 上下文。使用 chcon 时，用户提供要更改的所有或部分 SELinux 上下文。文件类型不正确是 SELinux 拒绝访问的常见原因。

快速参考

- 运行 chcon -t 类型 file-name 命令更改文件类型，其中 type 是 SELinux 类型，如 httpd_sys_content_t，file-name 是文件或目录名称：

```
~]$ chcon -t httpd_sys_content_t file-name
```

- 运行 **chcon -R -t** 类型 **directory-name** 命令以更改目录及其内容的类型，其中 **type** 是 **SELinux** 类型，如 **httpd_sys_content_t**，**directory-name** 是目录名称：

```
~]$ chcon -R -t httpd_sys_content_t directory-name
```

过程 **4.6.** 更改文件或目录的类型

以下步骤演示了更改类型，而不是 **SELinux** 上下文的其他属性。对于目录，本节中的示例的作用相同，例如，如果 **file1** 是目录。

1. 更改到您的主目录。

2. 创建新文件并查看其 **SELinux** 上下文：

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

在本例中，**file 1** 的 **SELinux** 上下文包括 **SELinux unconfined_u** 用户、**object _r** 角色、**user_home_t** 类型和 **s0** 级别。有关 **SELinux** 上下文的每个部分的描述，请参阅 **第 2 章 SELinux Contexts**。

3. 输入以下命令将类型更改为 **samba_share_t**。**t** 选项仅更改类型。然后查看更改：

```
~]$ chcon -t samba_share_t file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:samba_share_t:s0 file1
```

4. 使用以下命令恢复 **file1** 文件的 **SELinux** 上下文。使用 **-v** 选项查看哪些更改：

```
~]$ restorecon -v file1
restorecon reset file1 context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:user_home_t:s0
```

在本例中，上一类型 **samba_share_t** 已恢复到正确的 **user_home_t** 类型。使用目标策略（**Red Hat Enterprise Linux** 中的默认 **SELinux** 策略）时，**restorecon** 命令读取 **/etc/selinux/targeted/contexts/files/** 目录中的文件，以查看应具有哪些 **SELinux** 上下文文件。

过程 **4.7.** 更改目录及其内容类型

以下示例演示了创建新目录，并将目录的文件类型及其内容更改为 **Apache HTTP** 服务器使用的类型。如果您希望 **Apache HTTP** 服务器使用不同的文档根目录（而不是 **/var/www/html/**），则使用本示例中的配置：

1. 以 **root** 用户身份，在此目录中创建一个新的 **Web/** 目录，再创建 **3** 个空文件（**file1**、**file2** 和 **file3**）。其中包含的 **web/** 目录和文件使用 **default_t** 类型标记：

```
~]# mkdir /web
```

```
~]# touch /web/file{1,2,3}
```

```
~]# ls -dZ /web
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /web
```

```
~]# ls -lZ /web
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file3
```

2. 以 **root** 用户身份，输入以下命令将 **web/** 目录（及其内容）的类型改为 **httpd_sys_content_t** ：

```
~]# chcon -R -t httpd_sys_content_t /web/
```

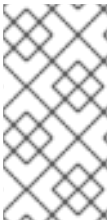
```
~]# ls -dZ /web/
drwxr-xr-x  root root unconfined_u:object_r:httpd_sys_content_t:s0 /web/
```

```
~]# ls -lZ /web/
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file3
```

3. 要恢复默认 SELinux 上下文，以 root 用户身份使用 restorecon 工具程序：

```
~]# restorecon -R -v /web/
restorecon reset /web context unconfined_u:object_r:httpd_sys_content_t:s0->system_u:object_r:default_t:s0
restorecon reset /web/file2 context unconfined_u:object_r:httpd_sys_content_t:s0->system_u:object_r:default_t:s0
restorecon reset /web/file3 context unconfined_u:object_r:httpd_sys_content_t:s0->system_u:object_r:default_t:s0
restorecon reset /web/file1 context unconfined_u:object_r:httpd_sys_content_t:s0->system_u:object_r:default_t:s0
```

有关 chcon 的详情，请查看 chcon(1) 手册页。



注意

类型强制是 SELinux targeted 策略中使用的主要权限控制。在大多数情况下，可以忽略 SELinux 用户和角色。

4.7.2. 持久性更改：semanage fcontext

semanage fcontext 命令用于更改文件的 SELinux 上下文。要显示新创建的文件和目录的上下文，以 root 用户身份输入以下命令：

```
~]# semanage fcontext -C -l
```

以下实用程序将使用 semanage fcontext 进行的更改。当重新标记文件系统时，会使用 setfiles 实用程序，并且 restorecon 实用程序恢复默认 SELinux 上下文。这意味着 semanage fcontext 所做的更改是持久的，即使文件系统被重新标记。SELinux 策略控制用户是否能够修改任何给定文件的 SELinux 上下文。

快速参考

在文件系统重新标记后进行 SELinux 上下文更改：

1. 输入以下命令，记得使用文件或目录的完整路径：

```
~]# semanage fcontext -a options file-name|directory-name
```

2. 使用 restorecon 程序应用上下文更改：

```
~]# restorecon -v file-name|directory-name
```

将正则表达式与 semanage fcontext 搭配使用

要使 semanage fcontext 命令正常工作，您可以使用完全限定路径或 兼容 Perl 的正则表达式 (PCRE)。使用的唯一 PCRE 标志是 PCRE2_DOTALL，这使得 . 通配符匹配任何内容，包括新行。代表路径的字符串被处理为字节，这意味着非 ASCII 字符不会被单个通配符匹配。

请注意，使用 semanage fcontext 指定的文件上下文定义是按照相反的顺序评估它们的定义：最新的条目首先评估，而不考虑时间长度。存储在 file_contexts.local 中的本地文件上下文修改的优先级高于策略模块。这意味着，每当 file_contexts.local 中找到与给定文件路径的匹配时，不会考虑其他文件上下文定义。

重要

使用 semanage fcontext 命令指定的文件上下文定义有效覆盖所有其他文件上下文定义。因此，所有正则表达式都应尽可能具体，以避免对文件系统中其他部分产生意外影响。

有关文件上下文定义和标志中使用的正则表达式类型的更多信息，请参阅 semanage-fcontext(8)手册页。

过程 4.8. 更改文件或目录的类型

以下示例演示了更改文件类型，而不是 SELinux 上下文的其他属性。此示例对目录起作用，例如 file1 是目录。

1. 以 root 用户身份，在 /etc 目录中创建一个新文件。默认情况下，/etc 中新创建的文件使用 etc_t 类型标记：

```
~]# touch /etc/file1
```

```
~]$ ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0       /etc/file1
```

要列出有关目录的信息，请使用以下命令：

```
~]$ ls -dZ directory_name
```

2. 以 root 身份，输入以下命令将 file1 类型更改为 samba_share_t。a 选项添加新记录，而 -t 选项则定义类型(samba_share_t)。请注意，运行这个命令不会直接更改类型；file 1 仍然使用 etc_t 类型进行标记：

```
~]# semanage fcontext -a -t samba_share_t /etc/file1
```

```
~]# ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0       /etc/file1
```

```
~]$ semanage fcontext -C -l
/etc/file1    unconfined_u:object_r:samba_share_t:s0
```

3. 以 root 用户身份，使用 restorecon 实用程序更改类型。因为 semanage 向 /etc/file1 的 file_contexts.local 中添加了一个条目，所以restorecon 会将类型改为 samba_share_t ：

```
~]# restorecon -v /etc/file1
restorecon reset /etc/file1 context unconfined_u:object_r:etc_t:s0-
>system_u:object_r:samba_share_t:s0
```

过程 4.9. 更改目录及其内容类型

以下示例演示了创建新目录，并将目录的文件类型及其内容更改为 Apache HTTP 服务器使用的类型。如果您希望 Apache HTTP 服务器使用不同的文档根目录而不是 /var/www/html/，则使用这个示例中的配置：

1. 以 root 用户身份，在此目录中创建一个新的 Web/ 目录，再创建 3 个空文件（file1、file2 和 file3）。其中包含的 web/ 目录和文件使用 default_t 类型标记：

```
~]# mkdir /web
```

```
~]# touch /web/file{1,2,3}
```

```
~]# ls -dZ /web
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /web
```

```
~]# ls -lZ /web
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file1
```

```
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file3
```

2. 以 root 身份，输入以下命令将 web/ 目录的类型及其文件更改为 httpd_sys_content_t。a 选项添加新记录，-t 选项定义类型(httpd_sys_content_t)。"/web(/.*)?" 正则表达式会导致 semanage 对 web/ 以及其中的文件应用更改。请注意，运行这个命令不会直接更改类型；其中的 web/ 和文件仍使用 default_t 类型进行标记：

```
~]# semanage fcontext -a -t httpd_sys_content_t "/web(/.*)?"
```

```
~]$ ls -dZ /web
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /web
```

```
~]$ ls -lZ /web
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:default_t:s0 file3
```

semanage fcontext -a -t httpd_sys_content_t "/web(/.*)?" 命令将以下条目添加到 /etc/selinux/targeted/contexts/files/file_contexts.local 中：

```
/web(/.*)?    system_u:object_r:httpd_sys_content_t:s0
```

3. 以 root 用户身份，使用 restorecon 实用程序更改 web/ 的类型，以及其中的所有文件。r 用于递归，这意味着 web/ 下的所有文件和目录都标有 httpd_sys_content_t 类型。因为 semanage 向 file.contexts.local 为 /web(/.*)? 添加了一个条目，所以 restorecon 将类型改为 httpd_sys_content_t ：

```
~]# restorecon -R -v /web
restorecon reset /web context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /web/file2 context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /web/file3 context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /web/file1 context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

请注意，默认情况下，新创建的文件和目录继承其父目录的 SELinux 类型。

过程 4.10. 删除添加的上下文

以下示例演示了添加和删除 SELinux 上下文。如果上下文是正则表达式的一部分，如 /web(/.*)?，请在正则表达式两处使用引号：

```
~]# semanage fcontext -d "/web(/.*)?"
```

1. 要删除上下文，请输入以下命令，其中 file-name |directory-name 是 file_contexts.local 中的第一个部分：

   ```
   ~]# semanage fcontext -d file-name|directory-name
   ```

   以下是 file_contexts.local 中的上下文示例：

   ```
   /test    system_u:object_r:httpd_sys_content_t:s0
   ```

   第一部分被测试。要在运行 restorecon 后，或文件系统重新标记后，要防止 test/ 目录被标记为 httpd_sys_content_t，请以 root 身份输入以下命令从 file_contexts.local 中删除上下文：

   ```
   ~]# semanage fcontext -d /test
   ```

2. 以 root 用户身份，使用 restorecon 实用程序恢复默认 SELinux 上下文。

有关 semanage 的详情请参考 semanage(8) 和 semanage-fcontext(8) 手册页。

重要

使用 semanage fcontext -a 更改 SELinux 上下文时，请使用文件或目录的完整路径以避免在文件系统重新标记或运行 restorecon 命令后文件被错误标记。

4.7.3. 如何确定文件上下文

确定文件上下文基于文件上下文定义，这些定义在系统安全策略中指定（.fc 文件）。根据系统策略，semanage 生成 file_contexts.homedirs 和 file_contexts 文件。

系统管理员可以使用 semanage fcontext 命令自定义文件上下文定义。此类自定义存储在 file_contexts.local 文件中。

当标签实用程序（如 matchpathcon 或 restorecon）确定给定路径的正确标签时，它会首先搜索本

地更改(file_contexts.local)。如果实用程序找不到匹配的模式，它将搜索 file_contexts.homedirs 文件，最后搜索 file_contexts 文件。但是，每当找到与给定文件路径的匹配项时，搜索结束时，实用程序会查找任何其他文件上下文定义。这意味着与主目录相关的文件上下文的优先级高于其余的优先级，本地自定义会覆盖系统策略。

在评估之前，由 系统策略 指定的 文件上下文定义（file_contexts.homedir s 和 file_contexts 文件的内容）按照句子长度（在任意通配符之前修复路径前缀）进行排序。这意味着选择最具体的路径。但是，使用 semanage fcontext 指定的文件上下文定义按照其定义方式顺序进行反序评估：最新的条目首先评估，而不考虑时间长度。

有关以下信息的更多信息：

- 使用 chcon 更改文件的上下文，请参阅 第 4.7.1 节 “临时更改：chcon”。
- 使用 semanage fcontext 更改并添加文件上下文定义，请参阅 第 4.7.2 节 “持久性更改：semanage fcontext”。
- 通过 system-policy 操作更改并添加文件上下文定义，请参阅 第 4.10 节 “维护 SELinux 标签” 或 第 4.12 节 “优先处理和禁用 SELinux 策略模块”。

4.8. FILE_T 和 DEFAULT_T 类型

使用支持扩展属性(EA) 的文件系统时，file_t 类型是尚未分配 EA 值的默认文件类型。这个类型仅用于此目的，且在正确标记的文件系统中不存在，因为运行 SELinux 的系统上的所有文件都应具有适当的 SELinux 上下文，并且 file_t 类型永远不会在文件上下文配置中使用[4].

default_t 类型用于与 file-context 配置中任何模式不匹配的文件，以便这些文件能够与磁盘上没有上下文的文件区分，并且通常对受限制的域无法访问。例如，如果您创建新的顶级目录，如 mydirectory/，则可以使用 default_t 类型标记此目录。如果服务需要访问此目录，您需要更新此位置的 file-contexts 配置。有关在 file-context 配置中添加上下文的详情，请查看 第 4.7.2 节 “持久性更改：semanage fcontext”。

4.9. 挂载文件系统

默认情况下，挂载支持扩展属性的文件系统时，每个文件的安全性上下文可以从 文件的 security.selinux 扩展属性获取。不支持扩展属性的文件系统中的文件将从策略配置中分配单一的默认安全上下文，具体取决于文件系统类型。

使用 mount -o context 命令覆盖现有的扩展属性，或者为不支持扩展属性的文件系统指定不同的默认上下文。如果您不信任文件系统来提供正确的属性，比如在多个系统中使用的可移动介质，这很有用。mount -o context 命令还可用于支持不支持扩展属性的文件系统的标记，如文件分配表(FAT)或 NFS 卷。使用 context 选项指定的上下文 不会写入磁盘：原始上下文将被保留，如果文件系统最初具有扩展属性，则在不带 上下文 挂载时会看到。

有关文件系统标记的更多信息，请参阅 James Morris 的"SELinux 中的文件系统标签"文章：
http://www.linuxjournal.com/article/7426

4.9.1. 上下文挂载

要挂载具有指定上下文的文件系统，覆盖现有的上下文（如果存在），或者为不支持扩展属性的文件系统指定不同的默认上下文，以 root 用户身份，在挂载所需的 文件系统时使用 mount -o context=SELinux_user:role:type:level 命令。上下文更改不会写入磁盘。默认情况下，客户端上的 NFS 挂载使用 NFS 卷策略定义的默认上下文标记。在常见策略中，此默认上下文使用 nfs_t 类型。如果不使用其他挂载选项，这可能会阻止使用其他服务（如 Apache HTTP 服务器）共享 NFS 卷。以下示例挂载了 NFS 卷，以便可以使用 Apache HTTP 服务器进行共享：

```
~]# mount server:/export /local/mount/point -o \ context="system_u:object_r:httpd_sys_content_t:s0"
```

此文件系统上新创建的文件和目录似乎具有通过 -o 上下文指定的 SELinux 上下文。但是，由于这些更改不会写入磁盘，因此使用此选项指定的上下文不会在挂载之间保留。因此，这个选项必须与每个挂载中指定的相同上下文一起使用，以保持所需的上下文。有关使上下文挂载持久的更多信息，请参阅 第 4.9.5 节 “使上下文挂载持久”。

类型强制是 SELinux targeted 策略中使用的主要权限控制。在大多数情况下，可以忽略 SELinux 用户和角色，因此，在通过 -o 上下文覆盖 SELinux 上下文时，请使用 SELinux system_u 用户和 object_r 角色，并专注于类型。如果您不使用 MLS 策略或多类别安全性，请使用 s0 级别。

注意

使用 上下文 选项挂载文件系统时，禁止用户和进程进行上下文更改。例如，在通过 上下文 选项挂载的文件系统中运行 chcon 命令会导致 Operation not supported 错误。

4.9.2. 更改默认上下文

如 第 4.8 节 “file_t 和 default_t 类型” 所述，在支持扩展属性的文件系统中，当访问磁盘中缺少 SELinux 上下文的文件时，会将其视为 SELinux 策略定义的默认上下文。在常见策略中，此默认上下文使用 file_t 类型。如果需要使用其他默认上下文，请使用 defcontext 选项挂载文件系统。

以下示例将 /dev/sda2 上新创建的文件系统挂载到新创建的 test/ 目录。它假设 /etc/selinux/targeted/contexts/files/ 中没有为 test/ 目录定义上下文的规则：

```
~]# mount /dev/sda2 /test/ -o defcontext="system_u:object_r:samba_share_t:s0"
```

在本例中：

- defcontext 选项定义 system_u:object_r:samba_share_t:s0 是"未标记文件的默认安全上下文"。[5].
- 挂载后，文件系统的根目录(test/)将被视为使用 defcontext 指定的上下文标记（此标签不存储在磁盘上）。这会影响 test/ 下创建文件的标签：新文件继承 samba_share_t 类型，这些标签存储在磁盘上。
- 当使用 defcontext 选项挂载文件系统时，在 test/ 下创建的文件保留其标签。

4.9.3. 挂载 NFS 卷

默认情况下，客户端上的 NFS 挂载使用 NFS 卷策略定义的默认上下文标记。在常见策略中，此默认上下文使用 nfs_t 类型。根据策略配置，Apache HTTP 服务器和 MariaDB 等服务可能无法读取使用 nfs_t 类型标记的文件。这可能会导致标记为此类型的文件系统被挂载，然后被其他服务读取或导出。

如果要挂载 NFS 卷并使用其他服务读取或导出该文件系统，在挂载时使用 上下文 选项来覆盖 nfs_t 类型。使用以下上下文选项挂载 NFS 卷，以便可以使用 Apache HTTP 服务器共享它们：

```
~]# mount server:/export /local/mount/point -o context="system_u:object_r:httpd_sys_content_t:s0"
```

由于这些更改不会写入磁盘，因此使用此选项指定的上下文不会在挂载之间保留。因此，这个选项必须与每个挂载中指定的相同上下文一起使用，以保持所需的上下文。有关使上下文挂载持久的更多信息，请参阅 第 4.9.5 节 “使上下文挂载持久”。

除了使用 上下文 选项挂载文件系统外，也可启用布尔值，以允许服务访问使用 nfs_t 类型标记的文件系统。有关配置布尔值以允许服务访问 nfs_t 类型的步骤，请参阅 第 II 部分 “管理受限服务”。

4.9.4. 多个 NFS 挂载

当从同一 NFS 导出挂载多个挂载时，尝试使用不同的上下文覆盖每个挂载的 SELinux 上下文时，会导致后续挂载命令失败。在以下示例中，NFS 服务器具有一个导出 导出/，它有两个子目录 web/ 和 database/。以下命令尝试从一个 NFS 导出中尝试两个挂载，并尝试覆盖每个导出的上下文：

```
~]# mount server:/export/web /local/web -o context="system_u:object_r:httpd_sys_content_t:s0"
```

```
~]# mount server:/export/database /local/database -o context="system_u:object_r:mysqld_db_t:s0"
```

第二个挂载命令失败，以下内容记录到 /var/log/messages ：

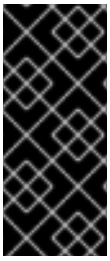
```
kernel: SELinux: mount invalid.  Same superblock, different security settings for (dev 0:15, type nfs)
```

要从单个 NFS 导出挂载多个挂载，每个挂载都有不同的上下文，请使用 -o nosharecache,context 选项。以下示例从单个 NFS 导出挂载多个挂载，每个挂载都有不同的上下文（允许单个服务访问每个导出）：

```
~]# mount server:/export/web /local/web -o
nosharecache,context="system_u:object_r:httpd_sys_content_t:s0"
```

```
~]# mount server:/export/database /local/database -o \
nosharecache,context="system_u:object_r:mysqld_db_t:s0"
```

在本例中，server :/export/web 在本地挂载到 /local/web/ 目录，所有文件都使用 httpd_sys_content_t 类型进行标记，允许 Apache HTTP 服务器访问。server:/export/database 在本地挂载到 /local/database/，所有文件都被标记为 mysqld_db_t 类型，允许 MariaDB 访问。这些类型更改不会写入磁盘。



重要

nosharecache 选项允许您多次挂载具有不同上下文的导出同一子目录，例如多次挂载 /export/web/。不要多次使用不同的上下文从导出中挂载同一子目录，因为这会产生重叠挂载，其中文件可在两个不同的上下文下访问。

4.9.5. 使上下文挂载持久

要使上下文挂载在重新挂载和重新启动后持久保留，请在 /etc/fstab 文件或自动挂载程序映射中添加文件系统条目，并使用所需的上下文作为挂载选项。以下示例为 NFS 上下文挂载在 /etc/fstab 中添加一个条目：

```
server:/export /local/mount/ nfs context="system_u:object_r:httpd_sys_content_t:s0" 0 0
```

4.10. 维护 SELINUX 标签

这些小节描述了复制、移动和存档文件和目录时 SELinux 上下文会发生什么情况。它还说明了在复制和存档时如何保留上下文。

4.10.1. 复制文件和目录

复制文件或目录时，如果文件或目录不存在，则会创建一个新文件或目录。该新文件或目录的上下文基于默认标记规则，而不是原始文件或目录的上下文，除非使用选项来保留原始上下文。例如，在用户主目录中创建的文件使用 user_home_t 类型进行标记：

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

如果此类文件复制到另一个目录，如 /etc，则根据 /etc 的默认标记规则创建新文件。在没有附加选项的情况下复制文件可能无法保留原始上下文：

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

```
~]# cp file1 /etc/
```

```
~]$ ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0   /etc/file1
```

将 file1 复制到 /etc 时，如果 /etc/file1 不存在，则创建 /etc/file1 作为新文件。如上例所示，根据 default-labeling 规则，/etc/file1 使用 etc_t 类型进行标记。

在现有文件上复制文件时，现有文件的上下文将被保留，除非用户指定了 cp 选项来保留原始文件的上下文，如 --preserve=context。SELinux 策略可能会阻止在复制期间保留上下文。

过程 4.11. 在不保留 SELinux 上下文的情况下复制

此流程显示，当使用 cp 命令复制文件时，如果没有给出任何选项，则类型将从目标父目录继承。

1. 在用户的主目录中创建文件。该文件使用 user_home_t 类型标记：

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

2. **/var/www/html/ 目录使用 httpd_sys_content_t 类型标记，如下所示：**

```
~]$ ls -dZ /var/www/html/
drwxr-xr-x  root root system_u:object_r:httpd_sys_content_t:s0 /var/www/html/
```

3. **将 file1 复制到 /var/www/html/ 时，它会继承 httpd_sys_content_t 类型：**

```
~]# cp file1 /var/www/html/
```

```
~]$ ls -Z /var/www/html/file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/file1
```

过程 4.12. 复制时保留 SELinux 上下文

这个过程演示了如何在复制时使用 --preserve=context 选项来保留上下文。

1. **在用户的主目录中创建文件。该文件使用 user_home_t 类型标记：**

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

2. **/var/www/html/ 目录使用 httpd_sys_content_t 类型标记，如下所示：**

```
~]$ ls -dZ /var/www/html/
drwxr-xr-x  root root system_u:object_r:httpd_sys_content_t:s0 /var/www/html/
```

3. **使用 --preserve=context 选项可在复制操作期间保留 SELinux 上下文。如下所示，文件复制到 /var/www/html/ 时保留 user_home_t 的文件1 类型：**

```
~]# cp --preserve=context file1 /var/www/html/
```

```
~]$ ls -Z /var/www/html/file1
-rw-r--r--  root root unconfined_u:object_r:user_home_t:s0 /var/www/html/file1
```

过程 4.13. 复制和更改上下文

这个过程演示了如何使用 --context 选项更改目标副本的上下文。以下示例在用户的主目录中执行：

1. **在用户的主目录中创建文件。该文件使用 user_home_t 类型标记：**

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

2. **使用 --context 选项来定义 SELinux 上下文：**

```
~]$ cp --context=system_u:object_r:samba_share_t:s0 file1 file2
```

3. **如果没有 --context，file2 将使用 unconfined_u:object_r:user_home_t 上下文标记：**

```
~]$ ls -Z file1 file2
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
-rw-rw-r--  user1 group1 system_u:object_r:samba_share_t:s0 file2
```

过程 4.14. 使用现有文件复制文件

此流程显示，当将文件复制到现有文件时，除非使用选项保留上下文，否则现有文件的上下文将被保留。

1. **以 root 用户身份，在 /etc 目录中创建一个新 文件 file1。如下所示，该文件使用 etc_t 类型进行标记：**

```
~]# touch /etc/file1
```

```
~]$ ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0   /etc/file1
```

2. **在 /tmp 目录中创建另一个 文件 file2。如下所示，该文件使用 user_tmp_t 类型进行标记：**

```
~]$ touch /tmp/file2
```

```
~$ ls -Z /tmp/file2
-rw-r--r--  root root unconfined_u:object_r:user_tmp_t:s0 /tmp/file2
```

3. 使用 file 2 覆盖 file 1 ：

```
~]# cp /tmp/file2 /etc/file1
```

4. 复制后，以下命令显示使用 etc_t 类型标记的 file1，而不是替换 / etc/file1 的 /tmp/file2 中的 user_tmp_t 类型：

```
~]$ ls -Z /etc/file1
-rw-r--r--  root root unconfined_u:object_r:etc_t:s0   /etc/file1
```

重要

复制文件和目录，而不是移动它们。这有助于确保它们标记为正确的 SELinux 上下文。不正确的 SELinux 上下文可能会阻止进程访问此类文件和目录。

4.10.2. 移动文件和目录

文件和目录在移动时保留其当前 SELinux 上下文。在很多情况下，这对他们移动到的位置来说不正确。以下示例演示了将文件从用户的主目录移动到 /var/www/html/ 目录，供 Apache HTTP 服务器使用。因为文件被移动，所以不会继承正确的 SELinux 上下文：

过程 4.15. 移动文件和目录

1. 更改到您的主目录，并在其中创建文件。该文件使用 user_home_t 类型标记：

```
~]$ touch file1
```

```
~]$ ls -Z file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 file1
```

2. 输入以下命令查看 /var/www/html/ 目录的 SELinux 上下文：

```
~]$ ls -dZ /var/www/html/
drwxr-xr-x  root root system_u:object_r:httpd_sys_content_t:s0 /var/www/html/
```

默认情况下，/var/www/html/ 被标记为 httpd_sys_content_t 类型。在 /var/www/html/ 下创建的文件和目录继承此类型，因此，它们使用此类型进行标记。

3. 以 root 身份，将 file1 移到 /var/www/html/。因为这个文件被移动，它会保留当前的 user_home_t 类型：

```
~]# mv file1 /var/www/html/
```

```
~]# ls -Z /var/www/html/file1
-rw-rw-r--  user1 group1 unconfined_u:object_r:user_home_t:s0 /var/www/html/file1
```

默认情况下，Apache HTTP 服务器无法读取标记为 user_home_t 类型的文件。如果组成 Web 页面的所有文件都标有 user_home_t 类型，或者 Apache HTTP 服务器无法读取的其他类型，则尝试使用 Web 浏览器（如 Mozilla Firefox ）访问它们时将拒绝权限。

重要

使用它们v 命令 移动文件和目录可能会导致 SELinux 上下文不正确，从而导致进程（如 Apache HTTP 服务器和 Samba）访问这些文件和目录。

4.10.3. 检查默认 SELinux 上下文

使用 matchpathcon 实用程序检查文件和目录是否有正确的 SELinux 上下文。此实用程序查询 系统策略，然后提供与文件路径关联的默认安全性上下文。[6] 以下示例演示了使用 matchpathcon 验证 /var/www/html/ 目录中的文件是否已正确标记：

过程 4.16. 使用 matchpathcon检查默认 SELinux Conxtext

1. 以 root 用户身份，在 /var/www/html/ 目录中创建三个文件（ file1、file2 和 file3 ）。这些文件继承 /var/www/html/ 中的 httpd_sys_content_t 类型：

```
~]# touch /var/www/html/file{1,2,3}
```

```
~]# ls -Z /var/www/html/
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file3
```

2. 以 root 身份，将 file1 类型更改为 samba_share_t。请注意，Apache HTTP 服务器无法读取标有 samba_share_t 类型的文件或目录。

```
~]# chcon -t samba_share_t /var/www/html/file1
```

3. **matchpathcon -V 选项可将当前 SELinux 上下文与 SELinux 策略中正确的默认上下文进行比较。输入以下命令检查 /var/www/html/ 目录中的所有文件：**

```
~]$ matchpathcon -V /var/www/html/*
/var/www/html/file1 has context unconfined_u:object_r:samba_share_t:s0, should be
system_u:object_r:httpd_sys_content_t:s0
/var/www/html/file2 verified.
/var/www/html/file3 verified.
```

matchpathcon 命令的以下输出解释 file1 被标记为 samba_share_t 类型，但应使用 httpd_sys_content_t 类型进行标记：

```
/var/www/html/file1 has context unconfined_u:object_r:samba_share_t:s0, should be
system_u:object_r:httpd_sys_content_t:s0
```

要解决标签问题并允许 Apache HTTP 服务器以 root 用户身份访问 file1，请使用 restorecon 实用程序：

```
~]# restorecon -v /var/www/html/file1
restorecon reset /var/www/html/file1 context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

4.10.4. 使用 tar归档文件

默认情况下，tar 实用程序不保留扩展属性。由于 SELinux 上下文存储在扩展属性中，因此存档文件时可能会丢失上下文。使用 tar --selinux 命令创建保留上下文的归档并从存档中恢复文件。如果 tar 归档包含没有扩展属性的文件，或者您希望扩展属性与系统默认值匹配，请使用 restorecon 工具：

```
~]$ tar -xvf archive.tar | restorecon -f -
```

请注意，根据 目录，您可能需要 root 用户来运行 restorecon。

以下示例演示了创建一个保留 SELinux 上下文的 tar 归档：

过程 4.17. 创建 tar 归档

1. 进入 **/var/www/html/** 目录并查看其 **SELinux** 上下文：

```
~]$ cd /var/www/html/
```

```
html]$ ls -dZ /var/www/html/
drwxr-xr-x. root root system_u:object_r:httpd_sys_content_t:s0 .
```

2. 以 **root** 用户身份，在 **/var/www/html/** 中创建三个文件（**file1**、**file2** 和 **file3**）。这些文件继承 **/var/www/html/** 中的 **httpd_sys_content_t** 类型：

```
html]# touch file{1,2,3}
```

```
html]$ ls -Z /var/www/html/
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file3
```

3. 以 **root** 身份，输入以下命令创建名为 **test.tar** 的 **tar** 存档。使用 **--selinux** 保留 **SELinux** 上下文：

```
html]# tar --selinux -cf test.tar file{1,2,3}
```

4. 以 **root** 用户身份，创建一个名为 **test/** 的新目录，然后允许所有用户对其进行完全访问：

```
~]# mkdir /test
```

```
~]# chmod 777 /test/
```

5. 将 **test.tar** 文件复制到 **test/** 中：

```
~]$ cp /var/www/html/test.tar /test/
```

6. 更改到 **test/** 目录。在该目录中后，输入以下命令提取 **tar** 存档。再次指定 **--selinux** 选项，否则 **SELinux** 上下文将更改为 **default_t** ：

```
~]$ cd /test/
```

```
test]$ tar --selinux -xvf test.tar
```

7. 查看 SELinux 上下文。httpd_sys_content_t 类型已被保留，而不是更改为 default_t，如果 --selinux 没有被使用，这会发生这样的情况：

```
test]$ ls -lZ /test/
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file3
-rw-r--r--  user1 group1 unconfined_u:object_r:default_t:s0 test.tar
```

8. 如果不再需要 test/ 目录，以 root 用户身份输入以下命令删除它，以及其中的所有文件：

```
~]# rm -ri /test/
```

有关 tar 的详情请参考 tar(1) 手册页，如保留所有扩展属性的 --xattrs 选项。

4.10.5. 使用 星级归档文件

默认情况下， star 实用程序 不保留扩展属性。由于 SELinux 上下文存储在扩展属性中，因此存档文件时可能会丢失上下文。使用 星 -xattr -H=exustar 命令创建保留上下文的存档。默认情况下不安装 星级软件包。要安装 星形，请以 root 用户身份运行 yum install star 命令。

以下示例演示了创建一个保留 SELinux 上下文 的星级 归档：

过程 4.18. 创建 星级 存档

1. 以 root 用户身份，在 /var/www/html/ 中创建三个文件（ file1、file2 和 file3 ）。这些文件继承 /var/www/html/ 中的 httpd_sys_content_t 类型：

```
~]# touch /var/www/html/file{1,2,3}
```

```
~]# ls -Z /var/www/html/
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 file3
```

2. 更改到 /var/www/html/ 目录。在这个目录中，以 root 身份输入以下命令来创建一个名为 test.star 的星 级存档：

```
~]$ cd /var/www/html
```

```
html]# star -xattr -H=exustar -c -f=test.star file{1,2,3}
star: 1 blocks + 0 bytes (total of 10240 bytes = 10.00k).
```

3. **以 root 用户身份，创建一个名为 test/ 的新目录，然后允许所有用户对其进行完全访问：**

```
~]# mkdir /test
```

```
~]# chmod 777 /test/
```

4. **输入以下命令将 test.star 文件复制到 test / 中：**

```
~]$ cp /var/www/html/test.star /test/
```

5. **更改为 test/。在这个目录中后，输入以下命令提取 星级 归档：**

```
~]$ cd /test/
```

```
test]$ star -x -f=test.star
star: 1 blocks + 0 bytes (total of 10240 bytes = 10.00k).
```

6. **查看 SELinux 上下文。httpd_sys_content_t 类型已被保留，而不是更改为 default_t，如果未使用 -xattr -H=exustar 选项，则会发生这种情况：**

```
~]$ ls -lZ /test/
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file2
-rw-r--r--  user1 group1 unconfined_u:object_r:httpd_sys_content_t:s0 file3
-rw-r--r--  user1 group1 unconfined_u:object_r:default_t:s0 test.star
```

7. **如果不再需要 test/ 目录，以 root 用户身份输入以下命令删除它，以及其中的所有文件：**

```
~]# rm -ri /test/
```

8. **如果不再需要 星号，请以 root 用户身份删除软件包：**

```
~]# yum remove star
```

有关 星号 的详情，请查看 **star(1)** 手册页。

4.11. 收集工具的信息

下面列出的工具是提供格式良好的信息的命令行工具，如访问向量缓存统计信息或类、类型或布尔值的数量。

avcstat

这个命令自启动以来提供访问向量缓存统计的简短输出。您可以通过指定以秒为单位的时间间隔来实时观察统计信息。这自初始输出后提供更新的统计信息。使用的统计信息文件是 **/sys/fs/selinux/avc/cache_stats**，您可以使用 **-f /path/to/file** 选项指定不同的缓存文件。

```
~]# avcstat
   lookups       hits     misses     allocs   reclaims      frees
  47517410   47504630      12780      12780      12176      12275
```

seinfo

此实用程序可用于描述策略的细分，如类、类型、布尔值、允许规则等。**se info** 是使用 **policy.conf** 文件、二进制策略文件、模块列表或策略列表作为输入的命令行实用程序。您必须安装 **setools-console** 软件包才能使用 **seinfo** 工具程序。

seinfo 的输出结果会因二进制和源文件而异。例如，策略源文件使用 **{ }** 方括号将多个规则元素分组到一行中。个属性也会产生类似的效果，其中单个属性可扩展到一个或多个类型。由于这些已扩展且在二进制策略文件中不再相关，因此搜索结果中的返回值为零。但是，规则数量会显著增加，因为每个之前使用方括号的行规则现在是多个单独的行。

某些项目不存在于二进制策略中。例如，不会在编译策略时检查允许规则，而不是在运行时检查，初始安全标识符(SID)也不是二进制策略的一部分，因为它们是在内核在启动期间加载策略前必需的。

```
~]# seinfo

Statistics for policy file: /sys/fs/selinux/policy
Policy Version & Type: v.28 (binary, mls)

   Classes:            77    Permissions:       229
   Sensitivities:       1    Categories:       1024
   Types:            3001    Attributes:        244
   Users:               9    Roles:              13
   Booleans:          158    Cond. Expr.:       193
   Allow:          262796    Neverallow:          0
   Auditallow:         44    Dontaudit:      156710
   Type_trans:      10760    Type_change:        38
```

```
  Type_member:        44    Role allow:         20
  Role_trans:        237    Range_trans:      2546
  Constraints:        62    Validatetrans:       0
  Initial SIDs:       27    Fs_use:             22
  Genfscon:           82    Portcon:           373
  Netifcon:            0    Nodecon:             0
  Permissives:        22    Polcap:              2
```

seinfo 工具还可使用 domain 属性列出类型数量，从而估算不同受限制的进程数量：

```
~]# seinfo -adomain -x | wc -l
550
```

并非所有域类型都被限制。要查看未限制域的数量，请使用 unconfined_domain 属性：

```
~]# seinfo -aunconfined_domain_type -x | wc -l
52
```

可以使用 --permissive 选项计算 permissive 域：

```
~]# seinfo --permissive -x | wc -l
31
```

删除上述命令中的其他 | wc -l 命令以查看完整列表。

sesearch

您可以使用 sesearch 实用程序搜索策略中的特定规则。可以搜索策略源文件或二进制文件。例如：

```
~]$ sesearch --role_allow -t httpd_sys_content_t
Found 20 role allow rules:
   allow system_r sysadm_r;
   allow sysadm_r system_r;
   allow sysadm_r staff_r;
   allow sysadm_r user_r;
   allow system_r git_shell_r;
   allow system_r guest_r;
   allow logadm_r system_r;
   allow system_r logadm_r;
   allow system_r nx_server_r;
   allow system_r staff_r;
   allow staff_r logadm_r;
   allow staff_r sysadm_r;
   allow staff_r unconfined_r;
   allow staff_r webadm_r;
```

```
allow unconfined_r system_r;
allow system_r unconfined_r;
allow system_r user_r;
allow webadm_r system_r;
allow system_r webadm_r;
allow system_r xguest_r;
```

sesearch 工具可以提供 允许 规则的数量：

```
~]# sesearch --allow | wc -l
262798
```

和 dontaudit 规则 的数量：

```
~]# sesearch --dontaudit | wc -l
156712
```

4.12. 优先处理和禁用 SELINUX 策略模块

/etc/selinux/ 中的 SELinux 模块存储允许对 SELinux 模块使用优先级。以 root 用户身份输入以下命令显示两个具有不同优先级的模块目录：

```
~]# ls /etc/selinux/targeted/active/modules
100  400  disabled
```

semodule 实用程序使用的默认优先级为 400，SELinux -policy 软件包中使用的优先级为 100，因此您可以找到安装优先级 100 的大部分 SELinux 模块。

您可以使用具有相同名称且具有更高优先级的修改模块覆盖现有模块。如果有更多模块具有相同名称和不同优先级，则构建策略时仅使用优先级最高的模块。

例 4.1. 使用 SELinux 策略模块优先级

使用修改的文件上下文准备新模块。使用 semodule -i 命令安装模块，并将模块的优先级设置为 400。以下示例中使用 sandbox.pp。

```
~]# semodule -X 400 -i sandbox.pp
~]# semodule --list-modules=full | grep sandbox
400 sandbox          pp
100 sandbox          pp
```

要返回默认模块，以 root 用户身份输入 semodule -r 命令：

```
~]# semodule -X 400 -r sandbox
libsemanage.semanage_direct_remove_key: sandbox module at priority 100 is now active.
```

禁用系统策略模块

要禁用 系统策略 模块，以 root 用户身份输入以下命令：

```
semodule -d MODULE_NAME
```

警告

如果您使用 semodule -r 命令删除 系统策略 模块，它会在您的系统存储上删除，您无法再次载入它。为避免不必要的重新安装 selinux-policy-targeted 软件包来恢复 所有系统策略 模块，请使用 semodule -d 命令。

4.13. 多级别安全（MLS）

多级安全技术指的是一种安全策略，它强制执行Bell-La Padula Mandatory Access Model。在 MLS 下，用户和进程称为 主体，而系统的文件、设备和其他被动组件称为 对象。主体和对象都标记为安全级别，该级别要求主体的授权或对象分类。每个安全级别都由一个 敏感度 组成， 例如，内部发行时间表按照具有机密敏感度的内部文件类别提交。

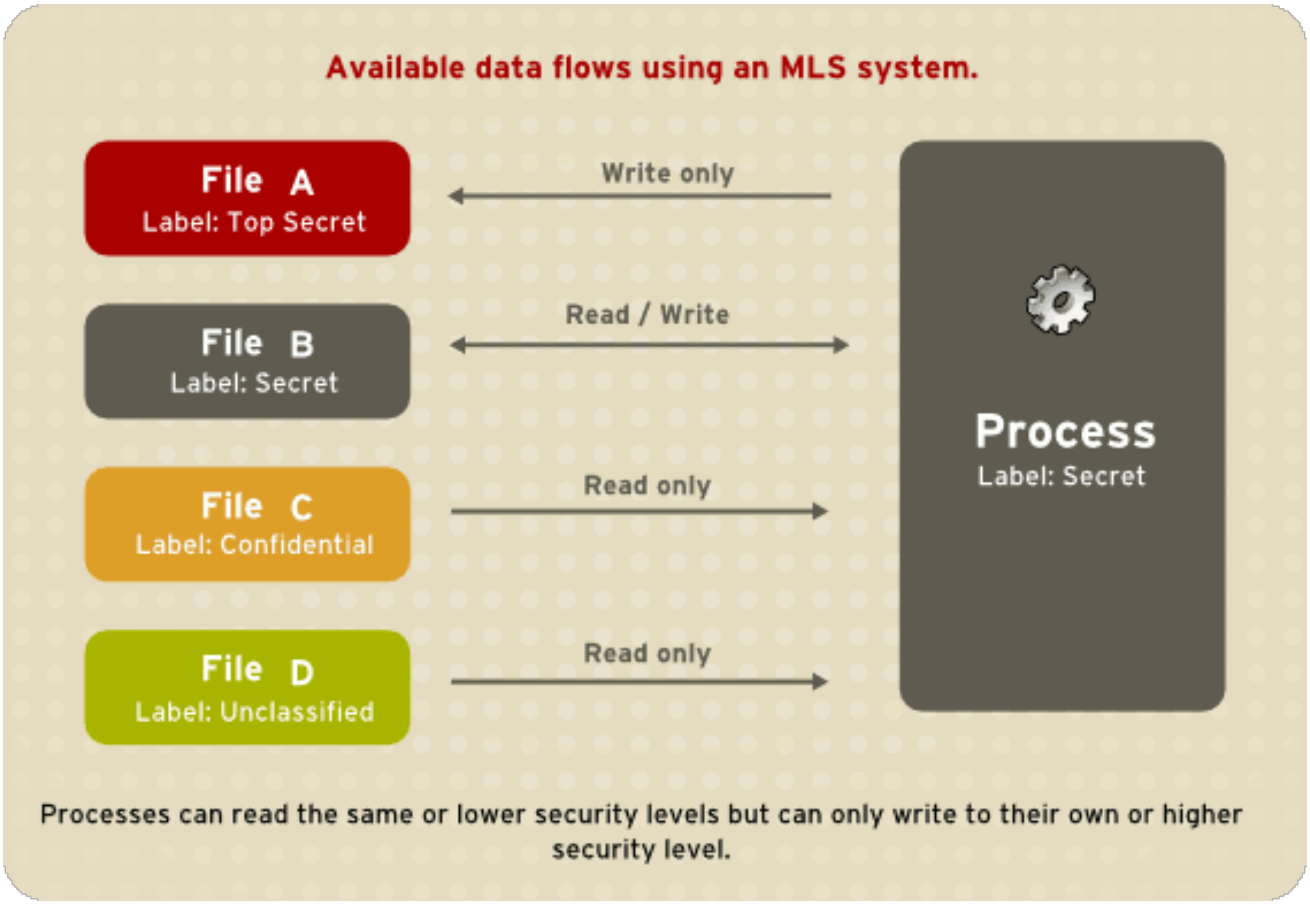
图 4.1 “国家级别” 显示美国国防社区最初设计的许可级别。关于上述内部时间表示例，只有获得机密许可的用户才能查看机密类别中的文件。但是，只有保密许可的用户不得查看需要更高等级或资格的文件；只允许他们读取较低级别的文件，并且能够访问更高等级的文件。

图 4.1. 国家级别

[D]

图 4.2 “使用 MLS 允许的数据流” 显示在"Secret"安全级别下运行的对象和具有不同安全级别的各种对象之间的所有允许的数据流。简而言之，Bell-LaPadula 模型强制执行两种属性： 无读写 。

图 4.2. 使用 MLS 允许的数据流



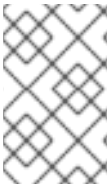
[D]

4.13.1. MLS 和系统特权

MLS 访问规则始终与传统的访问权限（文件权限）结合使用。例如，如果具有安全等级为 "Secret" 的用户使用 Discretionary Access Control（DAC）来阻止其他用户对文件的访问，这也会阻止具有安全等级 "Top Secret" 的用户访问。务必要记住，在 DAC 规则后检查 SELinux MLS 策略规则。较高的安全许可不会自动授予任意浏览文件系统的权限。

拥有顶级别的用户不会自动获得多级系统的管理权限。虽然这些用户可能可以访问计算机上的所有信息，但是这与具有管理权是不同的。

4.13.2. 在 SELinux 中启用 MLS



注意

不建议在运行 X Window 系统的系统上使用 MLS 策略。

按照以下步骤在您的系统上启用 SELinux MLS 策略。

过程 4.19. 启用 SELinux MLS 策略

1. 安装 selinux-policy-mls 软件包：

```
~]# yum install selinux-policy-mls
```

2. 在启用 MLS 策略之前，必须使用 MLS 标签重新标记文件系统中的每个文件。重新标记文件系统时，可能会拒绝访问受限域，这可能会妨碍系统正确启动。要防止发生这种情况，请在 /etc/selinux/config 文件中配置 SELINUX=permissive。另外，通过配置 SELINUXTYPE=mls 来启用 MLS 策略。您的配置文件应如下所示：

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=permissive
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=mls
```

3. 确保 SELinux 以 permissive 模式运行：

```
~]# setenforce 0
```

```
~]$ getenforce
Permissive
```

4. 使用 fixfiles 脚本创建包含 -F 选项的 /.autorelabel 文件，以确保在下次重启时重新标记文件：

```
~]# fixfiles -F onboot
```

5. 重启您的系统。在下一次启动期间，将根据 MLS 策略重新标记所有文件系统。标签进程使用适当的 SELinux 上下文标记所有文件：

```
*** Warning -- SELinux mls policy relabel is required.
*** Relabeling could take a very long time, depending on file
```

```
*** system size and speed of hard drives.
***********
```

底部行中的每个 *（星 号）字符表示 1000 个已标记的文件。在上面的示例中，十一 * 字符代表 11000 个已标记的文件。标记所有文件所需的时间取决于系统上的文件数量以及硬盘驱动器的速度。在现代系统中，此过程只需 10 分钟。标签过程完成后，系统将自动重新引导。

6. 在 permissive 模式中，SELinux 策略不会被强制实施，但是仍会记录在 enforcing 模式运行时会被拒绝的操作。在更改为 enforcing 模式之前，以 root 身份输入以下命令，以确认 SELinux 在上一次启动期间没有拒绝操作。如果在上一次启动期间 SELinux 没有拒绝操作，这个命令不会返回任何输出。如果在引导过程中 SELinux 拒绝访问，请参阅 第 11 章 故障排除。

```
~]# grep "SELinux is preventing" /var/log/messages
```

7. 如果 /var/log/messages 文件中没有拒绝信息，或者您已解决所有现有的拒绝，请在 /etc/selinux/config 文件中配置 SELINUX=enforcing ：

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=mls
```

8. 重启您的系统并确保 SELinux 以 enforcing 模式运行：

```
~]$ getenforce
Enforcing
```

启用了 MLS 策略：

```
~]# sestatus |grep mls
Policy from config file:       mls
```

4.13.3. 使用特定 MLS 范围创建用户

按照以下步骤创建具有特定 MLS 范围的新 Linux 用户：

过程 4.20. 使用特定 MLS 范围创建用户

1. 使用 useradd 命令添加新的 Linux 用户，并将新的 Linux 用户映射到现有 SELinux 用户（本例中为 staff_u）：

```
~]# useradd -Z staff_u john
```

2. 为新创建的 Linux 用户分配密码：

```
prompt~]# passwd john
```

3. 以 root 身份输入以下命令，以查看 SELinux 和 Linux 用户之间的映射。输出应如下：

```
~]# semanage login -l
Login Name           SELinux User         MLS/MCS Range        Service

__default__          user_u               s0-s0                *
john                 staff_u              s0-s15:c0.c1023      *
root                 root                 s0-s15:c0.c1023      *
staff                staff_u              s0-s15:c0.c1023      *
sysadm               staff_u              s0-s15:c0.c1023      *
system_u             system_u             s0-s15:c0.c1023      *
```

4. 为用户 john 定义一个特定范围：

```
~]# semanage login --modify --range s2:c100 john
```

5. 再次查看 SELinux 和 Linux 用户之间的映射。请注意，用户 john 现在定义了特定的 MLS 范围：

```
~]# semanage login -l
Login Name           SELinux User         MLS/MCS Range        Service

__default__          user_u               s0-s0                *
john                 staff_u              s2:c100              *
root                 root                 s0-s15:c0.c1023      *
staff                staff_u              s0-s15:c0.c1023      *
sysadm               staff_u              s0-s15:c0.c1023      *
system_u             system_u             s0-s15:c0.c1023      *
```

6. 如果需要，请输入以下命令更正 john 主目录中的标签：

-

```
~]# chcon -R -l s2:c100 /home/john
```

4.13.4. 设置 Polyinstantiated 目录

/tmp 和 /var/tmp/ 目录通常由所有程序、服务和用户用于临时存储。但是，此类设置使得这些目录容易遭受竞争条件攻击，或者存在基于文件名的信息泄漏。SELinux 以 多形目录 的形式提供解决方案。这实际上意味着 /tmp 和 / var/tmp/ 都已实例化，使其对于每个用户都是私有的。启用目录实例化时，每个用户的 /tmp 和 /var/tmp / 目录都会自动挂载到 /tmp-inst 和 /var/tmp/tmp-inst 下。

按照以下步骤启用目录的多形化：

过程 4.21. 启用 Polyinstantiation 目录

1. 取消注释 /etc/security/namespace.conf 文件中的最后三行，以启用 /tmp、/var/tmp/ 和用户主目录的实例化：

```
~]$ tail -n 3 /etc/security/namespace.conf
/tmp     /tmp-inst/           level      root,adm
/var/tmp /var/tmp/tmp-inst/   level      root,adm
$HOME    $HOME/$USER.inst/    level
```

2. 在 /etc/pam.d/login 文件中确保为会话配置了 pam_namespace.so 模块：

```
~]$ grep namespace /etc/pam.d/login
session    required     pam_namespace.so
```

3. 重启您的系统。

4.14. 文件名称转换

文件名转换 功能允许策略作者在编写策略转换规则时指定文件名。可以编写一条声明为以下条件的规则：如果标有 A_t 的进程在标有 B_t 的目录中创建指定对象类，并且指定的对象类名为 objectname，它会获取标签 C_t。这种机制可以更加精细地控制系统上的进程。

如果不进行文件名转换，可以通过三种方式来标记对象：

- 默认情况下，对象从父目录继承标签。例如，如果用户在标有 etc_t 的目录中创建文件，则该文件也会被标记 etc_t。但是，当需要在具有不同标签的目录中有多个文件时，此方法没有用

处。

- 策略写入器可以在策略中编写一个规则，声明如下：如果类型为 **A_t** 的进程在标有 **B_t** 的目录中创建指定对象类，则对象会获得新的 **C_t** 标签。如果单个程序在同一目录中创建多个对象，每个对象需要单独的标签，则这种方法存在问题。此外，这些规则仅提供部分控制，因为未指定创建的对象的名称。

- 某些应用程序具有 **SELinux** 感知，允许此类应用程序询问系统特定路径的标签是什么。这些应用程序随后请求内核使用所需标签创建对象。识别 **SELinux** 的应用程序示例有 **rpm** 软件包管理器、**restorecon** 实用程序或 **udev** 设备管理器。但是，不能指示创建 **SELinux** 感知文件或目录的每个应用程序。通常需要在创建后使用正确的标签重新标记对象。否则，当受限域尝试使用该对象时，将返回 **AVC** 消息。

文件名转换功能减少了与不正确的标记相关的问题，提高了系统的安全性。策略作者可以正确声明，特定应用程序只能在指定目录中创建具有指定名称的文件。规则会考虑文件名，而不是文件路径。这是文件路径的基名。请注意，文件名转换使用 **strcmp()** 功能完成的完全匹配。不考虑使用正则表达式或通配符字符。

注意

文件路径在内核中可能有所不同，文件名转换不使用路径来确定标签。因此，此功能只影响初始文件创建，且不会修复已创建对象的不正确的标签。

例 4.2. 使用文件名转换的策略规则编写示例

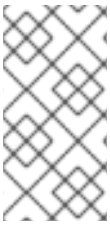
以下示例显示了带有文件名转换的策略规则：

```
filetrans_pattern(unconfined_t, admin_home_t, ssh_home_t, dir, ".ssh")
```

此规则规定，如果具有 **unconfined_t** 类型的进程在标有 **admin_home_t** 的目录中创建 **~/.ssh/** 目录，则 **~/.ssh/** 目录会获得标签 **ssh_home_t**。

以下是使用文件名转换编写策略规则的类似示例：

```
filetrans_pattern(staff_t, user_home_dir_t, httpd_user_content_t, dir, "public_html")
filetrans_pattern(thumb_t, user_home_dir_t, thumb_home_t, file, "missfont.log")
filetrans_pattern(kernel_t, device_t, xserver_misc_device_t, chr_file, "nvidia0")
filetrans_pattern(puppet_t, etc_t, krb5_conf_t, file, "krb5.conf")
```



注意

文件名转换功能主要影响策略编写器，但用户可以注意到，几乎始终使用包含目录的默认标签创建文件对象，某些文件对象具有与策略中指定的不同标签。

4.15. 禁用 PTRACE（）

ptrace（） 系统调用允许一个进程观察和控制另一个进程的执行并更改其内存和寄存器。此调用主要供开发人员在调试期间使用，例如在使用 strace 实用程序时。不需要 ptrace（） 时，可以禁用它来提高系统安全性。这可以通过启用 deny_ptrace 布尔值（拒绝在 unconfined_t 域中运行的进程）对其他进程使用 ptrace（） 来完成。

deny_ptrace 布尔值默认为禁用。要启用它，以 root 用户身份在命令 中运行 setsebool -P deny_ptrace:

```
~]# setsebool -P deny_ptrace on
```

要验证是否启用了这个布尔值，请使用以下命令：

```
~]$ getsebool deny_ptrace
deny_ptrace --> on
```

要禁用这个布尔值，以 root 用户身份运行 setsebool -P deny_ptrace off 命令：

```
~]# setsebool -P deny_ptrace off
```

注意

setsebool -P 命令进行永久性更改。如果您不想在重新启动后保留更改，则不要使用 -P 选项。

这个布尔值只会影响作为 Red Hat Enterprise Linux 一部分的软件包。因此，第三方软件包仍然可以

使用 ptrace（） 系统调用。要列出允许使用 ptrace（）的 域，请输入以下命令。请注意，setools-console 软件包提供了 sesearch 实用程序，并且默认情况下未安装该软件包。

```
~]# sesearch -A -p ptrace,sys_ptrace -C | grep -v deny_ptrace | cut -d ' ' -f 5
```

4.16. 缩略图保护

缩略图图标可能会允许攻击者使用可移动介质（如 USB 设备或 CD）破坏锁定的计算机。当系统检测到可移动介质时，Nautilus 文件管理器会执行缩略图驱动程序代码，以便在适当的文件浏览器中显示缩略图图标，即使计算机已被锁定。这个行为不安全，因为如果缩略图可执行文件存在安全漏洞，攻击者可以使用缩略图驱动程序代码绕过锁屏幕，而不必输入密码。

因此，新的 SELinux 策略用于防止此类攻击。此策略确保所有缩略图驱动程序在屏幕锁定时被锁定。缩略图保护为受限制的用户和未限制的用户都启用。该策略会影响以下应用程序：

- /usr/bin/evince-thumbnailer
- /usr/bin/ffmpegthumbnailer
- /usr/bin/gnome-exe-thumbnailer.sh
- /usr/bin/gnome-nds-thumbnailer
- /usr/bin/gnome-xcf-thumbnailer
- /usr/bin/gsf-office-thumbnailer
- /usr/bin/raw-thumbnailer
- /usr/bin/shotwell-video-thumbnailer
- /usr/bin/totem-video-thumbnailer

- /usr/bin/whaaw-thumbnailer
- /usr/lib/tumbler-1/tumblerd
- /usr/lib64/tumbler-1/tumblerd

---

[3] 若要临时恢复到默认行为，请以 Linux root 用户身份运行 setsebool httpd_can_network_connect_db off 命令。对于重启后保留的更改，请运行 setsebool -P httpd_can_network_connect_db off 命令。

[4] /etc/selinux/targeted/contexts/files/ 目录中的文件定义了文件和目录的上下文。restorecon 读取此目录中的文件，并将 文件实用程序设置为将 文件和目录恢复到其默认上下文。

[5] 贾姆斯州米尔里斯."SELinux 中的文件系统标记"。2004 年 10 月 1 日发布.2008 年 10 月 14 日访问： http://www.linuxjournal.com/article/7426

[6] 有关 matchpathcon 的详情，请查看 matchpathcon(8) 手册页。

第 5 章 SEPOLICY SUITE

sepolicy 实用程序提供了一组查询已安装的 **SELinux** 策略的功能。这些功能是新的，或者之前由单独的实用程序（如 **sepolgen** 或 **se trans** ）提供的。该套件允许您生成转换报告、**man page** 甚至新的策略模块，从而使用户能够更轻松地访问并更好地了解 **SELinux** 策略。

policycoreutils-devel 软件包提供 **sepolicy**。以 **root** 用户身份输入以下命令安装 **sepolicy** ：

```
~]# yum install policycoreutils-devel
```

sepolicy 套件提供以下作为命令行参数调用的功能：

表 5.1. sepolicy 功能

功能	描述
布尔值	查询 SELinux 策略以查看布尔值的描述
通信	查询 SELinux 策略以查看域是否可以相互通信
generate	生成 SELinux 策略模块模板
gui	SELinux 策略的图形用户界面
interface	列出 SELinux 策略接口
manpage	生成 SELinux man page
network	查询 SELinux 策略网络信息
transition	查询 SELinux 策略并生成进程转换报告

5.1. SEPOLICY PYTHON BINDINGS

在以前的 **Red Hat Enterprise Linux** 版本中，**se tools** 软件包包括 **sesearch** 和 **seinfo** 实用程序。**sesearch** 实用程序用于在 **SELinux** 策略中搜索规则，而 **seinfo** 实用程序允许您查询策略中的各种其他组件。

在 **Red Hat Enterprise Linux 7** 中，添加了用于 **sesearch** 和 **seinfo** 的 **Python** 绑定，以便您可以通过 **sepolicy** 套件使用这些实用程序的功能。请参见以下示例：

```
> python
>>> import sepolicy
>>> sepolicy.info(sepolicy.ATTRIBUTE)
Returns a dictionary of all information about SELinux Attributes
>>>sepolicy.search([sepolicy.ALLOW])
Returns a dictionary of all allow rules in the policy.
```

5.2. 生成 SELINUX 策略模块： SEPOLICY GENERATE

在之前的 Red Hat Enterprise Linux 版本中， sepolgen 或 selinux-polgengui 实用程序用于生成 SELinux 策略。这些工具已合并到 sepolicy 套件。在 Red Hat Enterprise Linux 7 中， sepolicy generate 命令用于生成初始 SELinux 策略模块模板。

与 sepolgen 不同，不需要以 root 用户身份运行 sepolicy generate。此实用程序还会创建一个 RPM spec 文件，该文件可用于构建 RPM 软件包，该软件包可将策略软件包文件(NAME.pp)和接口文件 (NAME. if)安装到正确的位置，提供 SELinux 策略安装到内核并修复标记。设置脚本将继续安装 SELinux 策略并设置标签。此外，使用 sepolicy manpage 命令生成基于已安装策略的 man page。[7] 最后，sepolicy 生成 构建并将 SELinux 策略和 man page 编译到 RPM 软件包中，并可在其他系统上安装。

执行 sepolicy 生成 时，会生成以下文件：

NAME.te - 键入 enforcing 文件

此文件定义特定域的所有类型和规则。

NAME.if - 接口文件

此文件定义系统的默认文件上下文。它取 NAME.te 文件中创建的文件类型，并将文件路径关联到这些类型。工具（如 restorecon 和 rpm ）使用这些路径编写标签。

NAME_selinux.spec - RPM spec 文件

此文件是一个 RPM spec 文件，可安装 SELinux 策略并设置标签。此文件还会安装 interface 文件和描述策略的 man page。您可以使用 sepolicy manpage -d NAME 命令来生成 man page。

NAME.sh - helper shell 脚本

此脚本有助于在系统上编译、安装和修复标记。它还会根据安装的策略生成 man page，编译和构建适合在其他系统上安装的 RPM 软件包。

如果可以生成 SELinux 策略模块，se policy 会生成 从源域到目标域的所有生成的路径。有关 sepolicy 生成 的详情，请查看 sepolicy-generate(8) 手册页。

5.3. 了解域 转换：SEPOLICY 转换

在以前的版本中，setrans 实用程序用于检查两个域或进程类型之间是否可以转换，并打印出用于在这些域或进程间转换的所有中间类型。在 Red Hat Enterprise Linux 7 中，se trans 作为 sepolicy 套件的一部分提供，现在使用 sepolicy transition 命令。

sepolicy transition 命令查询 SELinux 策略并创建进程转换报告。sepolicy transition 命令需要两个命令行参数，即源域（通过 -s 选项指定）和目标域（通过 -t 选项指定）。如果只输入源域，se policy transition 将 列出源域可以转换到的所有可能域。以下输出不包含所有条目。“@” 字符表示 “执行”：

```
~]$ sepolicy transition -s httpd_t
httpd_t @ httpd_suexec_exec_t --> httpd_suexec_t
httpd_t @ mailman_cgi_exec_t --> mailman_cgi_t
httpd_t @ abrt_retrace_worker_exec_t --> abrt_retrace_worker_t
httpd_t @ dirsrvadmin_unconfined_script_exec_t --> dirsrvadmin_unconfined_script_t
httpd_t @ httpd_unconfined_script_exec_t --> httpd_unconfined_script_t
```

如果指定了目标域，se policy 转换将检查 SELinux 策略，了解从源域到目标域的所有转换路径并列出这些路径。以下输出不完整：

```
~]$ sepolicy transition -s httpd_t -t system_mail_t
httpd_t @ exim_exec_t --> system_mail_t
httpd_t @ courier_exec_t --> system_mail_t
httpd_t @ sendmail_exec_t --> system_mail_t
httpd_t ... httpd_suexec_t @ sendmail_exec_t --> system_mail_t
httpd_t ... httpd_suexec_t @ exim_exec_t --> system_mail_t
httpd_t ... httpd_suexec_t @ courier_exec_t --> system_mail_t
httpd_t ... httpd_suexec_t ... httpd_mojomojo_script_t @ sendmail_exec_t --> system_mail_t
```

有关 sepolicy 迁移的详情，请查看 sepolicy-transition(8) 手册页。

5.4. 生成 MAN PAGE: SEPOLICY MANPAGE

sepolicy manpage 命令根据记录进程域的 SELinux 策略生成 man page。因此，此类文档始终是最新的。每个自动生成的 man page 名称都由进程域名和 _selinux 后缀组成，如 httpd_selinux。

man page 包括多个部分，它们提供有关受限制域 SELinux 策略的各个部分的信息：

- *Entrypoints 部分包含域转换过程中需要执行的所有可执行文件。*
- *Process Types 部分列出了所有以与目标域相同的前缀开头的进程类型。*
- *Booleans 部分列出了与 域关联的布尔值。*
- *Port Types 部分中包含与域匹配的端口类型，并描述了分配给这些端口类型的默认端口号。*
- *Managed Files 部分描述了允许域写入的类型以及这些类型关联的默认路径。*
- *文件上下文 部分包含与域关联的所有文件类型，并描述了如何在系统上使用这些文件类型以及默认路径标记。*
- *共享文件 部分介绍如何使用域共享类型，如 public_content_t。*

有关 sepolicy manpage 的详情，请查看 sepolicy-manpage(8) 手册页。

---

[7] *有关 sepolicy manpage 的详情请参考 第 5.4 节 “生成 man page: sepolicy manpage”。*

第 6 章 限制用户

在 Red Hat Enterprise Linux 中，用户默认映射到 SELinux unconfined_u 用户。unconfined_u 运行的所有进程都在 unconfined_t 域中。这意味着用户可以在标准 Linux DAC 策略限制范围内访问系统。然而，很多受限制的 SELinux 用户在 Red Hat Enterprise Linux 中可用。这意味着可以将用户限制为有限的能力集。每个 Linux 用户都使用 SELinux 策略映射到 SELinux 用户，允许 Linux 用户继承对 SELinux 用户的限制，例如（取决于用户），无法：

- 运行 X Window 系统
- 使用网络
- 运行 setuid 应用程序（除非 SELinux 策略允许）
- 或运行 su 和 sudo 命令。

例如，SELinux user_u 用户运行的进程位于 user_t 域中。此类进程可以连接到网络，但无法运行 su 或 sudo 命令。这有助于防止用户访问系统。有关受限制用户及其功能的详情，请查看 第 3.3 节 “受限制和未限制的用户” 表 3.1 “SELinux 用户功能”。

6.1. LINUX 和 SELINUX 用户映射

以 root 用户身份输入以下命令查看 Linux 用户与 SELinux 用户之间的映射：

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

在 Red Hat Enterprise Linux 中，Linux 用户默认映射到 SELinux __default__ 登录（进而映射到 SELinux unconfined_u 用户）。使用 useradd 命令创建 Linux 用户时，如果没有指定任何选项，则会将其映射到 SELinux unconfined_u 用户。下面定义了 default-mapping：

```
__default__          unconfined_u         s0-s0:c0.c1023       *
```

6.2. 限制新 LINUX 用户：USERADD

映射到 SELinux unconfined_u 用户的 Linux 用户在 unconfined_t 域中运行。通过运行 id -Z 命令，以映射到 unconfined_u 的 Linux 用户登录来查看：

```
~]$ id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

当 Linux 用户在 unconfined_t 域中运行时，会应用 SELinux 策略规则，但存在允许 Linux 用户在 unconfined_t 域中运行的策略规则几乎所有访问权限。如果不受限制的 Linux 用户执行 SELinux 策略定义的应用程序可以从 unconfined_t 域转换到其自身限制的域，则未限制的 Linux 用户仍会受到那个受限制域的限制。这样做的安全优势在于，即使 Linux 用户运行没有限制，应用程序仍会受到限制，因此对应用程序中漏洞的利用也会受到策略的限制。

注意

这不会防止系统用户阻止系统。相反，用户和系统会受到保护，使其免受应用漏洞造成的破坏。

使用 useradd 命令创建 Linux 用户时，请使用 -Z 选项指定他们映射到哪个 SELinux 用户。以下示例创建了一个新的 Linux 用户 useruuser，并将该用户映射到 SELinux user_u 用户。映射到 SELinux user_u 用户的 Linux 用户在 user_t 域中运行。在此域中，Linux 用户无法运行 setuid 应用程序，除非 SELinux 策略允许它（如 passwd），且无法运行 su 或 sudo 命令，阻止他们使用这些命令成为 root 用户。

过程 6.1. 将新 Linux 用户限制为 user_u SELinux 用户

1. 以 root 用户身份，创建映射到 SELinux user_uuser 的新 Linux 用户(useru user)。

```
~]# useradd -Z user_u useruuser
```

2. 要查看 useruuser 和 user_u 之间的映射，以 root 用户身份输入以下命令：

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
useruuser            user_u               s0                   *
```

3. 以 root 用户身份，为 Linux useruuser 用户分配密码：

```
~]# passwd useruuser
Changing password for user useruuser.
New password: Enter a password
Retype new password: Enter the same password again
passwd: all authentication tokens updated successfully.
```

4. 从当前会话注销，然后以 Linux useruuser 用户身份登录。登录时，pam_selinux 模块会将 Linux 用户映射到 SELinux 用户（本例中为 user_u），并设置生成的 SELinux 上下文。然后，将使用此上下文启动 Linux 用户的 shell。输入以下命令查看 Linux 用户的上下文：

```
~]$ id -Z
user_u:user_r:user_t:s0
```

5. 从 Linux useruuser 的会话注销，然后使用您的帐户重新登录。如果您不希望 Linux useruuser 用户，以 root 用户身份输入以下命令删除它，及其主目录：

```
~]# userdel -Z -r useruuser
```

6.3. 限制现有 LINUX 用户：SEMANAGE LOGIN

如果 Linux 用户映射到 SELinux unconfined_u 用户（默认行为），并且您想要更改将其映射到哪个 SELinux 用户，请使用 semanage login 命令。以下示例创建一个名为 newuser 的新 Linux 用户，然后将该 Linux 用户映射到 SELinux user_u 用户：

过程 6.2. 将 Linux 用户映射到 SELinux 用户

1. 以 root 用户身份，创建新的 Linux 用户（新用户）。因为这个用户使用默认映射，所以它不会出现在 semanage login -l 输出中：

```
~]# useradd newuser
```

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

2. 要将 Linux newuser 用户映射到 SELinux user_u 用户，以 root 用户身份输入以下命令：

```
~]# semanage login -a -s user_u newuser
```

a 选项添加新记录，而 -s 选项指定要将 Linux 用户映射到的 SELinux 用户。最后一个参数 newuser 是您要映射到指定的 SELinux 用户的 Linux 用户。

3. 要查看 Linux newuser 用户与 user_u 之间的映射，请再次使用 semanage 工具程序：

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
newuser              user_u               s0                   *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

4. 以 root 用户身份，为 Linux newuser 用户分配密码：

```
~]# passwd newuser
Changing password for user newuser.
New password: Enter a password
Retype new password: Enter the same password again
passwd: all authentication tokens updated successfully.
```

5. 从当前会话注销，然后以 Linux newuser 用户身份登录。输入以下命令查看 新用户的 SELinux 上下文：

```
~]$ id -Z
user_u:user_r:user_t:s0
```

6. 从 Linux newuser 的会话注销，然后使用您的帐户重新登录。如果您不希望 Linux newuser 用户，以 root 用户身份输入以下命令删除它，及其主目录：

```
~]# userdel -r newuser
```

以 root 用户身份，删除 Linux newuser 用户与 user_u 之间的映射：

```
~]# semanage login -d newuser
```

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

6.4. 更改默认映射

在 Red Hat Enterprise Linux 中，Linux 用户默认映射到 SELinux __default__ 登录（进而映射到 SELinux unconfined_u 用户）。如果您希望新 Linux 用户和 Linux 用户未专门映射到 SELinux 用户，请使用 semanage login 命令更改默认映射。

例如，以 root 用户身份输入以下命令将默认映射从 unconfined_u 改为 user_u ：

```
~]# semanage login -m -S targeted -s "user_u" -r s0 __default__
```

验证 __default__ 登录是否已映射到 user_u ：

```
~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          user_u               s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

如果创建了新的 Linux 用户并且未指定 SELinux 用户，或者现有 Linux 用户登录且不匹配 semanage login -l 输出中的特定条目，则这些用户将映射到 user_u，如 _ default__ 登录所示。

要改回到默认行为，以 root 用户身份输入以下命令将 __default__ 登录映射到 SELinux unconfined_u 用户：

```
~]# semanage login -m -S targeted -s "unconfined_u" -r s0-s0:c0.c1023 __default__
```

6.5. XGUEST：KIOSK 模式

xguest 软件包提供一个 kiosk 用户帐户。该帐户用于保护人们传出并使用的计算机，例如库、银行、机场、信息台和咖啡店等机器。kiosk 用户帐户非常有限：基本上，它仅允许用户登录并使用 Firefox 浏

览 Internet 网站。客户机用户已分配给 xguest_u，请参阅 表 3.1 “SELinux 用户功能”当您注销时，使用此帐户登录时将丢失所有更改，如创建文件或更改设置。

设置 kiosk 帐户：

1. 以 root 用户身份，安装 xguest 软件包。根据需要安装依赖项：

```
~]# yum install xguest
```

2. 为了让各种用户使用 kiosk 帐户，该帐户不受密码保护，因此，只能在 SELinux 在强制模式下运行时，才能保护该帐户。在使用此帐户登录前，使用 getenforce 工具确认 SELinux 是否在 enforcing 模式下运行：

```
~]$ getenforce
Enforcing
```

否则，请参阅 第 4.4 节 “SELinux 状态和模式中的永久性更改” 以了解有关切换到 enforcing 模式的信息。如果 SELinux 处于 permissive 模式或禁用，则无法使用此帐户登录。

3. 您只能使用 GNOME 显示管理器(GDM)登录此帐户。安装 xguest 软件包后，将一个 guest 帐户添加到 GDM 登录屏幕。

6.6. 用户执行应用程序的布尔值

不允许 Linux 用户在其主目录中执行应用程序（继承用户的权限）和 /tmp 目录中，它们有写入访问权限可以帮助防止缺陷或恶意应用程序修改用户拥有的文件。

布尔值可用于更改此行为，并使用 setsebool 实用程序进行配置，后者必须以 root 身份运行。setsebool -P 命令进行永久性更改。如果您不希望在重启后保留更改，则不要使用 -P 选项：

guest_t

要防止 guest_t 域中的 Linux 用户执行其主目录和 /tmp 中的应用程序：

```
~]# setsebool -P guest_exec_content off
```

xguest_t

*防止 **xguest_t** 域中的 **Linux** 用户执行其主目录和 **/tmp** 中的应用程序：*

```
~]# setsebool -P xguest_exec_content off
```

user_t

*要防止 **user_t** 域中的 **Linux** 用户执行其主目录和 **/tmp** 中的应用程序：*

```
~]# setsebool -P user_exec_content off
```

staff_t

*要防止 **staff_t** 域中的 **Linux** 用户执行其主目录和 **/tmp** 中的应用程序：*

```
~]# setsebool -P staff_exec_content off
```

*将 **staff_exec_content** 布尔值打开 并允许 **staff _t** 域中的 **Linux** 用户在其主目录和 **/tmp** 中执行应用程序：*

```
~]# setsebool -P staff_exec_content on
```

第 7 章 使用 SANDBOX 保护程序

沙盒安全实用程序添加了一组 SELinux 策略，允许系统管理员在严格限制的 SELinux 域内运行应用程序。可以定义对打开新文件或访问网络的权限的限制。这样便可以安全地测试不受信任的软件的处理特征，而不会有损坏系统的风险。

7.1. 使用 SANDBOX 运行应用程序

在使用 sandbox 工具前，必须安装 policycoreutils-sandbox 软件包：

```
~]# yum install policycoreutils-sandbox
```

限制应用程序的基本语法为：

```
~]$ sandbox [options] application_under_test
```

要在 沙盒 中运行图形应用程序，请使用 -X 选项。例如：

```
~]$ sandbox -X evince
```

X 告知 沙盒 为应用程序设置受限的次要 X 服务器（在本例中为 evince），然后复制所需资源并在 用户的主目录 或 /tmp 目录中创建封闭的虚拟环境。

要保留从一个会话到下一个会话的数据：

```
~]$ sandbox -H sandbox/home -T sandbox/tmp -X firefox
```

请注意，sandbox/home 用于 /home，sandbox/tmp 用于 /tmp。不同的应用会放置在不同的受限环境中。应用以全屏模式运行，这会阻止访问其他功能。如前文所述，您无法打开或创建标记为 sandbox_x_file_t 的文件。

网络的访问最初也不可能在 沙盒 内。若要允许访问，可使用 sandbox_web_t 标签。例如，要启动 Firefox ：

```
~]$ sandbox -X -t sandbox_web_t firefox
```

警告

sandbox_net_t 标签允许不受限制的双向网络访问所有网络端口。sandbox_web_t 允许连接到仅 Web 浏览所需的端口。

只有在需要时才应谨慎使用 sandbox_net_t。

请参见 沙盒(8) 手册页，以及可用选项的完整列表。

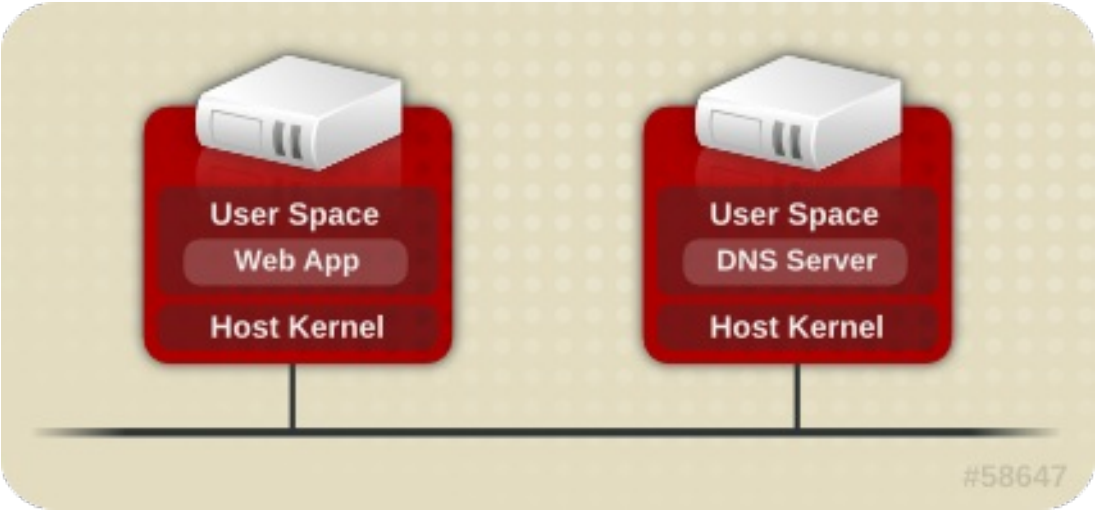
第 8 章 SVIRT

sVirt 是红帽企业 Linux 中包含的技术，其集成了 SELinux 和虚拟化。sVirt 应用强制访问控制(MAC)来在使用虚拟机时提高安全性。集成这些技术的主要原因是提高安全性并强化系统免受管理程序中可能用作面向主机的攻击向量或另一台虚拟机的攻击向量。

本章论述了 sVirt 如何与 Red Hat Enterprise Linux 中的虚拟化技术集成。

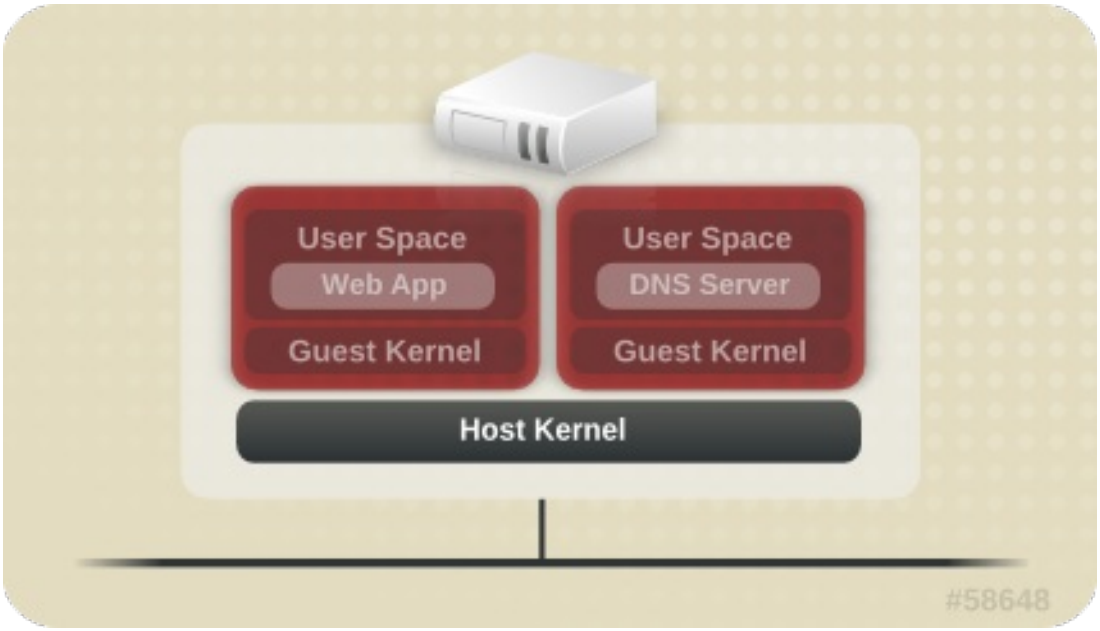
非虚拟化环境

在非虚拟化环境中，主机在物理上相互隔开，每个主机都具有自包含的环境，由 Web 服务器或 DNS 服务器等服务组成。这些服务直接与自己的用户空间（主机内核和物理主机）通信，为它们的服务直接提供网络。下图代表了一个非虚拟化环境：



虚拟化环境

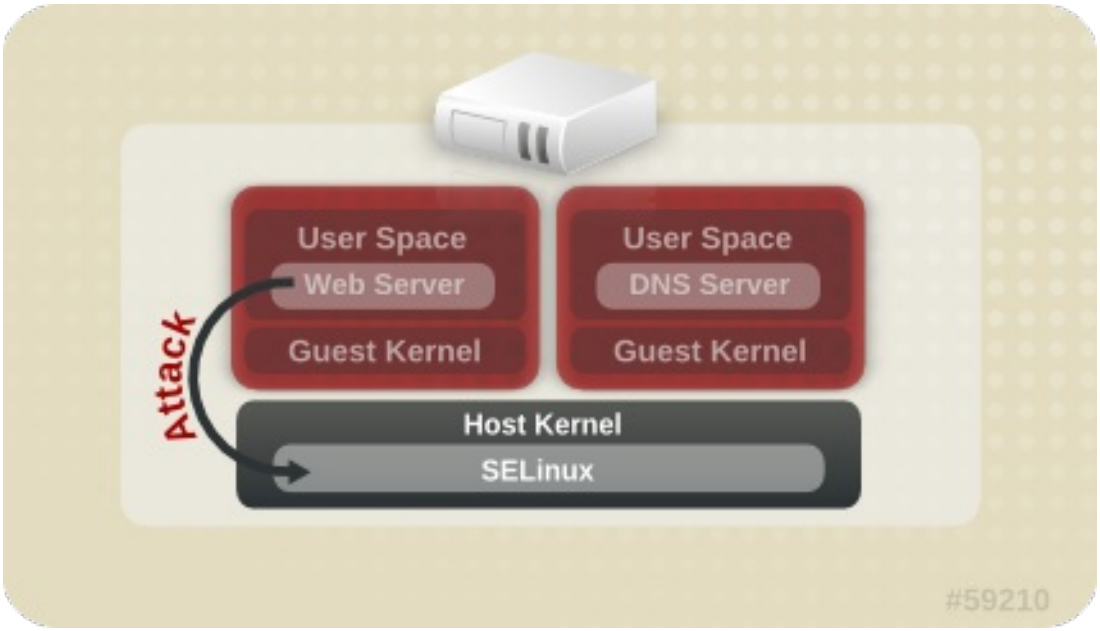
在虚拟化环境中，单个主机内核和物理主机中可以存放多个操作系统（作为"guest"）。下图代表虚拟化环境：



8.1. 安全性和虚拟化

如果未虚拟化服务，则物理分隔计算机。除网络攻击外，任何漏洞通常包含在受影响的计算机中。当服务分组在虚拟化环境中时，系统中会出现额外的漏洞。如果系统管理程序中存在可能被客户机实例利用的安全漏洞，此 guest 可能不仅能够攻击主机，而且能够攻击该主机上运行的其他虚拟客户机。这不是理论上的，虚拟机监控程序上已存在攻击。这些攻击可能扩展到客户机实例之外，并可公开其他客户端受到攻击。

sVirt 旨在隔离客户机并限制其启动后续攻击的能力（如果被利用）。在以下镜像中演示了这一点，其中攻击无法中断虚拟机并扩展到另一个主机实例：



SELinux 在实施强制访问控制(MAC)中引入了虚拟化实例的可插拔安全框架。sVirt 框架允许对客户机及其资源进行唯一标记。标记之后，可以应用可拒绝不同 guest 之间的访问的规则。

8.2. SVIRT 标记

与受 SELinux 保护的其他服务一样，sVirt 使用基于进程的机制和限制为客户机实例提供额外的安全层。在典型的使用中，您甚至不应注意到 sVirt 在后台工作。这部分论述了 sVirt 的标记功能。

如以下输出所示，使用 sVirt 时，每个虚拟机(VM)进程都会使用动态生成的级别进行标记并运行。每个进程都与具有不同级别的其他虚拟机隔离：

```
~]# ps -eZ | grep qemu

system_u:system_r:svirt_t:s0:c87,c520 27950 ?  00:00:17 qemu-kvm
system_u:system_r:svirt_t:s0:c639,c757 27989 ? 00:00:06 qemu-system-x86
```

实际磁盘镜像会自动标记为与进程匹配，如下输出所示：

```
~]# ls -lZ /var/lib/libvirt/images/*

system_u:object_r:svirt_image_t:s0:c87,c520   image1
```

下表概述了使用 sVirt 时可以分配的不同标签：

表 8.1. sVirt Labels

类型	SELinux Context	描述
虚拟机进程	system_u:system_r:svirt_t:MCS1	MCS1 是一个随机选择的 MCS 字段。目前支持约 500,000 个标签。
虚拟机镜像	system_u:object_r:svirt_image_t:MCS1	只有标有相同 MCS 字段的 *svirt_t* 进程才能读取/写入这些镜像文件和设备。
虚拟机共享读取/写入内容	system_u:object_r:svirt_image_t:s0	标有 *svirt_t 的所有* 进程都可写入 svirt_image_t:s0 文件和设备。
虚拟机镜像	system_u:object_r:virt_content_t:s0	镜像退出时使用的系统默认标签.不允许 *svirt_t* 虚拟进程读取使用该标签的文件/设备。

使用 sVirt 时也可以执行静态标记。静态标签允许管理员为虚拟机选择特定的标签，包括 MCS/MLS 字段。运行静态标记虚拟机的管理员负责在镜像文件上设置正确的标签。虚拟机将始终使用该标签启

动，sVirt 系统永远不会修改静态标记的虚拟机内容的标签。这允许 sVirt 组件在 MLS 环境中运行。您还可以根据您的要求，在系统上运行具有不同敏感度级别的多个虚拟机。

第 9 章 安全 LINUX 容器

Linux 容器(LXC)是一种低级虚拟化功能，允许您在一个系统上同时运行同一服务的多个副本。与完全虚拟化相比，容器不需要整个新系统进行引导，可以使用较少的内存，并且能够以只读方式使用基础操作系统。例如，LXC 允许您同时运行多个 Web 服务器，各自拥有自己的数据，同时共享系统数据，甚至以 root 用户身份运行。但是，在容器内运行特权进程可能会影响容器外运行的其他进程，或者在其他容器中运行的进程。保护 Linux 容器使用 SELinux 上下文，因此可防止在其中运行的进程相互交互或与主机交互。

Docker 应用程序是在红帽企业 Linux 中管理 Linux 容器的主要实用程序。作为替代方案，您也可以使用 libvirt 软件包提供的 virsh 命令行实用程序。

有关 Linux 容器的详情，请参阅开始使用容器。

第 10 章 SELINUX SYSTEMD 访问控制

在 Red Hat Enterprise Linux 7 中，系统服务由 systemd 守护进程控制。在之前的 Red Hat Enterprise Linux 版本中，可以通过两种方式启动守护进程：

- 在引导时，System V init 守护进程启动了 init.rc 脚本，然后此脚本启动了所需的守护进程。例如，在启动时启动的 Apache 服务器具有以下 SELinux 标签：

  ```
  system_u:system_r:httpd_t:s0
  ```

- 管理员手动启动 init.rc 脚本，导致 守护进程运行。例如，在 Apache 服务器上调用 service httpd restart 命令时，生成的 SELinux 标签如下所示：

  ```
  unconfined_u:system_r:httpd_t:s0
  ```

手动启动时，进程采用启动它的 SELinux 标签的用户部分，使得以上两个情况下的标记不一致。对于 systemd 守护进程，转换过程会非常不同。由于 systemd 使用 init_t 类型来处理启动和停止系统上守护进程的所有调用，因此在手动重启守护进程时，它可以覆盖标签的用户部分。因此，以上这两种情况下的标签都是 system_u:system_r:httpd_t:s0，SELinux 策略可能会被改进来控制哪些域可以控制哪些单元。

10.1. 服务的 SELINUX 访问权限

在以前的 Red Hat Enterprise Linux 版本中，管理员可以控制哪些用户或应用程序可以根据 System V Init 脚本标签启动或停止服务。现在，systemd 启动 和停止所有服务，用户和进程使用 systemctl 实用程序与 systemd 通信。systemd 守护进程能够参考 SELinux 策略，检查调用进程的标签以及调用者尝试管理的单元文件标签，然后询问 SELinux 是否允许调用者访问。这个方法可控制对关键系统功能的访问控制，其中包括启动和停止系统服务。

例如，之前，管理员必须允许 NetworkManager 执行 systemctl 将 D-Bus 消息发送到 systemd，后者反过来会启动或停止任何请求的服务 NetworkManager。事实上，NetworkManager 被允许执行 systemctl 可执行的 所有操作。也无法设置受限管理员，以便他们仅启动或停止特定的服务。

为了解决这些问题，systemd 也充当 SELinux 访问管理器。它可以检索运行 systemctl 的进程的标签，也可以检索向 systemd 发送 D-Bus 消息的进程的标签。然后守护进程会查找进程要配置的单元文件标签。最后，如果 SELinux 策略允许进程标签和单元文件标签之间的特定访问，systemd 可以从内核检索信息。这意味着，SELinux 现在可以限制需要与 systemd 交互特定服务的被破坏的应用。策略作者也可以使用这些精细的控制来限制管理员。策略更改涉及一个名为 service 的新类，其权限如下：

```
class service
```

```
{
        start
        stop
        status
        reload
        kill
        load
        enable
        disable
}
```

例如，策略作者现在可以允许域获取服务状态或启动和停止服务，但不能启用或禁用服务。在所有情况下，SELinux 和 systemd 中的访问控制操作都不匹配。一个映射被定义为带有 SELinux 访问检查的 systemd 方法调用。表 10.1 “在 SELinux 访问检查中映射 systemd 单元文件方法调用” 映射单元文件的访问检查，表 10.2 “在 SELinux 访问检查中映射 systemd 常规系统调用” 涵盖了系统访问检查。如果任一表中都没有找到匹配项，则调用 未定义的 系统检查。

表 10.1. 在 SELinux 访问检查中映射 systemd 单元文件方法调用

systemd 单元文件方法	SELinux 访问检查
DisableUnitFiles	disable
EnableUnitFiles	启用
GetUnit	status
GetUnitByPID	status
GetUnitFileState	status
kill	stop
KillUnit	stop
LinkUnitFiles	启用
ListUnits	status
LoadUnit	status
MaskUnitFiles	disable
PresetUnitFiles	启用
ReenableUnitFiles	启用
Reexecute	start

systemd 单元文件方法	SELinux 访问检查
reload	reload
ReloadOrRestart	start
ReloadOrRestartUnit	start
ReloadOrTryRestart	start
ReloadOrTryRestartUnit	start
ReloadUnit	reload
ResetFailed	stop
ResetFailedUnit	stop
Restart	start
RestartUnit	start
Start	start
StartUnit	start
StartUnitReplace	start
stop	stop
StopUnit	stop
TryRestart	start
TryRestartUnit	start
UnmaskUnitFiles	启用

表 10.2. 在 SELinux 访问检查中映射 systemd 常规系统调用

systemd 常规系统调用	SELinux 访问检查
ClearJobs	reboot
FlushDevices	halt

systemd 常规系统调用	SELinux 访问检查
Get	status
GetAll	status
GetJob	status
GetSeat	status
GetSession	status
GetSessionByPID	status
GetUser	status
halt	halt
introspect	status
KExec	reboot
KillSession	halt
KillUser	halt
ListJobs	status
ListSeats	status
ListSessions	status
ListUsers	status
LockSession	halt
PowerOff	halt
重启	reboot
SetUserLinger	halt
TerminateSeat	halt
TerminateSession	halt
TerminateUser	halt

例 10.1. 系统服务的 SELinux 策略

使用 sesearch 工具，您可以列出系统服务的策略规则。例如，调用 sesearch -A -s NetworkManager_t -c service 命令返回：

```
allow NetworkManager_t dnsmasq_unit_file_t : service { start stop status reload kill load } ;
allow NetworkManager_t nscd_unit_file_t : service { start stop status reload kill load } ;
allow NetworkManager_t ntpd_unit_file_t : service { start stop status reload kill load } ;
allow NetworkManager_t pppd_unit_file_t : service { start stop status reload kill load } ;
allow NetworkManager_t polipo_unit_file_t : service { start stop status reload kill load } ;
```

10.2. SELINUX 和 JOURNALD

在 systemd 中，journald 守护进程（也称为 systemd-journal）是 syslog 实用程序的替代选择，后者是收集和存储日志数据的系统服务。它根据从内核收到的日志信息、使用 libc syslog（） 功能的用户进程、系统服务的标准和错误输出，或使用其原生 API，创建和维护结构化和索引化日志。它以安全的方式为每个日志消息隐式收集大量元数据字段。

systemd-journal 服务可与 SELinux 配合使用，以提高安全性。SELinux 控制进程仅允许进程执行它们所设计的操作；有时，根据策略编写者的安全目标，有时更少一些。例如，SELinux 可防止泄露的 ntpd 进程执行除处理网络时间以外的任何操作。但是，ntpd 进程会发送 syslog 消息，以便被入侵的进程能够继续发送这些消息。被泄露 的 ntpd 可能会格式化 syslog 消息以匹配其他守护进程，并可能错误地导致管理员无法正常工作，甚至更糟糕地，会读取 syslog 文件以破坏整个系统。

systemd-journal 守护进程将验证所有日志消息，此外，还为它们添加 SELinux 标签。然后可以轻松地检测日志消息中的不一致情况，并在发生之前防止这种类型的攻击。您可以使用 journalctl 实用程序查询 systemd 日志的日志。如果没有指定命令行参数，则运行此实用程序会列出日志的完整内容，从最旧的条目开始。要查看系统上生成的所有日志，包括系统组件的日志，请以 root 用户身份执行 journalctl。如果您以非 root 用户身份执行它，则输出将仅限于与当前登录用户相关的日志。

例 10.2. 使用 journalctl列出日志

可以使用 journalctl 列出与特定 SELinux 标签相关的所有日志：例如，以下命令列出了 system_u:system_r:policykit_t:s0 标签下记录的所有日志：

```
~]# journalctl _SELINUX_CONTEXT=system_u:system_r:policykit_t:s0
Oct 21 10:22:42 localhost.localdomain polkitd[647]: Started polkitd version 0.112
Oct 21 10:22:44 localhost.localdomain polkitd[647]: Loading rules from directory /etc/polkit-1/rules.d
Oct 21 10:22:44 localhost.localdomain polkitd[647]: Loading rules from directory /usr/share/polkit-1/rules.d
Oct 21 10:22:44 localhost.localdomain polkitd[647]: Finished loading, compiling and executing 5 rules
Oct 21 10:22:44 localhost.localdomain polkitd[647]: Acquired the name
```

```
org.freedesktop.PolicyKit1 on the system bus Oct 21 10:23:10 localhost polkitd[647]:
Registered Authentication Agent for unix-session:c1 (system bus name :1.49, object path
/org/freedesktop/PolicyKit1/AuthenticationAgent, locale en_US.UTF-8) (disconnected from
bus)
Oct 21 10:23:35 localhost polkitd[647]: Unregistered Authentication Agent for unix-
session:c1 (system bus name :1.80 [/usr/bin/gnome-shell --mode=classic], object path
/org/freedesktop/PolicyKit1/AuthenticationAgent, locale en_US.utf8)
```

有关 journalctl 的详情，请查看 journalctl(1) 手册页。

第 11 章 故障排除

以下章节论述了 SELinux 拒绝访问时发生的情况：问题的主要三种原因；在哪里查找有关正确标记的信息；分析 SELinux 拒绝；以及使用 audit2 创建自定义策略模块。

11.1. 拒绝访问时的 HAPPENS

SELinux 决策（如允许或禁止访问）会被缓存。这个缓存被称为 Access Vector Cache（AVC）。当 SELinux 拒绝访问时，会记录拒绝消息。这些拒绝也称为 "AVC 拒绝"，并记录到不同的位置，具体取决于正在运行的守护进程：

Daemon: auditd on

日志位置: /var/log/audit/audit.log

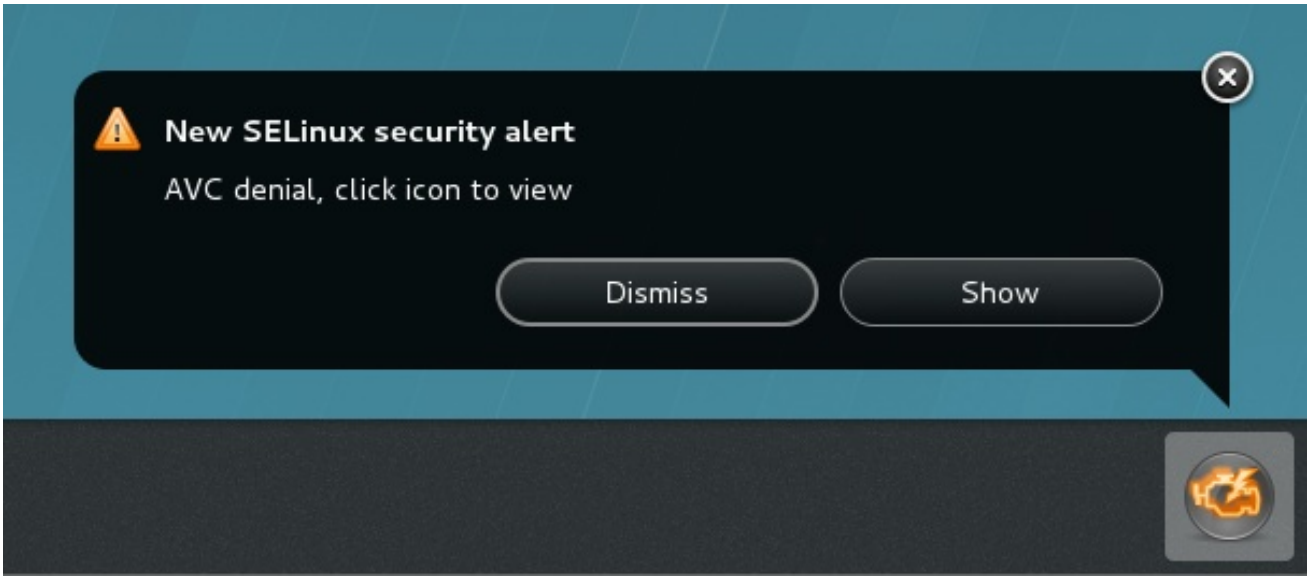
Daemon: auditd off; rsyslogd on

日志位置: /var/log/messages

Daemon: setroubleshootd、rsyslogd 和 auditd on

日志位置: /var/log/audit/audit.log.易于读取的拒绝消息也发送到 /var/log/messages

如果您正在运行 X Window 系统，安装 setroubleshoot 和 setroubleshoot-server 软件包，并且 setroubleshootd 和 auditd 守护进程正在运行，SELinux 拒绝访问时会显示警告：



单击 Show 可以详细分析 SELinux 拒绝访问的原因，以及允许访问的可能解决方案。如果您没有运行 X Window 系统，SELinux 拒绝访问就不太明显。例如，浏览网站的用户可能会收到类似如下的错误：

```
Forbidden

You don't have permission to access file name on this server
```

对于这些情况，如果 DAC 规则（标准 Linux 权限）允许访问，请分别检查 /var/log/messages 和 /var/log/audit/audit.log 的 "SELinux is prevent" 和" denied" 错误。这可以通过以 root 用户身份运行以下命令来完成：

```
~]# grep "SELinux is preventing" /var/log/messages
```

```
~]# grep "denied" /var/log/audit/audit.log
```

11.2. 问题最多的三种原因

以下小节描述了问题的主要三个原因：标记问题、为服务配置布尔值和端口，以及不断发展的 SELinux 规则。

11.2.1. 标记问题

在运行 SELinux 的系统上，所有进程和文件都标有包含安全相关信息的标签。此信息称为 SELinux 上下文。如果这些标签错误，则可能会拒绝访问。标记不正确的应用可能会导致向其进程分配不正确的标签。这会导致 SELinux 拒绝访问，并且进程可能会创建标记错误的文件。

标记问题的常见原因是，当服务使用非标准目录时。例如，管理员不必将 /var/www/html/ 用于网站，而是希望使用 /srv/myweb/。在 Red Hat Enterprise Linux 中，/srv 目录标有 var_t 类型。在 /srv 中创建的文件和目录继承此类型。此外，顶级目录中新创建的对象（如 /myserver）可能会使用 default_t 类型进行标记。SELinux 会阻止 Apache HTTP 服务器(httpd)访问这两种类型。要允许访问，SELinux 必须知道 /srv/myweb/ 中的文件可以被 httpd 访问：

```
~]# semanage fcontext -a -t httpd_sys_content_t "/srv/myweb(/.*)?"
```

此 semanage 命令将 /srv/myweb/ 目录（以及其中所有文件和目录）的上下文添加到 SELinux 文件上下文配置中[8].semanage 实用程序不会更改上下文。以 root 用户身份，运行 restorecon 工具来应用更改：

```
~]# restorecon -R -v /srv/myweb
```

有关在 file-context 配置中添加上下文的更多信息，请参阅 第 4.7.2 节 “持久性更改：semanage fcontext”。

11.2.1.1. 什么是正确的上下文？

matchpathcon 实用程序检查文件路径的上下文，并将其与该路径的默认标签进行比较。以下示例演示了在包含错误标记文件的目录中使用 matchpathcon ：

```
~]$ matchpathcon -V /var/www/html/*
/var/www/html/index.html has context unconfined_u:object_r:user_home_t:s0, should be
system_u:object_r:httpd_sys_content_t:s0
/var/www/html/page1.html has context unconfined_u:object_r:user_home_t:s0, should be
system_u:object_r:httpd_sys_content_t:s0
```

在本例中，index .html 和 page1.html 文件使用 user_home_t 类型进行标记。这种类型用于用户主目录中的文件。使用 st v 命令 从您的主目录中移动文件可能会导致文件使用 user_home_t 类型进行标记。这个类型不应存在于主目录之外。使用 restorecon 实用程序将这些文件恢复到其正确类型：

```
~]# restorecon -v /var/www/html/index.html
restorecon reset /var/www/html/index.html context unconfined_u:object_r:user_home_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

要恢复目录中所有文件的上下文，请使用 -R 选项：

```
~]# restorecon -R -v /var/www/html/
restorecon reset /var/www/html/page1.html context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /var/www/html/index.html context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

有关 matchpathcon 的详情，请查看 第 4.10.3 节 “检查默认 SELinux 上下文”。

11.2.2. 受限服务如何运行？

服务可以以多种方式运行。为了满足此需求，您需要指定如何运行您的服务。这可以通过布尔值来实现，允许在运行时更改 SELinux 策略的部分，而不必知晓编写 SELinux 策略。这允许更改，例如允许服务访问 NFS 卷，而无需重新加载或重新编译 SELinux 策略。此外，在非默认端口号上运行服务需要使用 semanage 命令更新策略配置。

例如，要允许 Apache HTTP 服务器与 MariaDB 通信，启用 httpd_can_network_connect_db 布尔值：

```
~]# setsebool -P httpd_can_network_connect_db on
```

如果特定服务的访问权限被拒绝，请使用 getsebool 和 grep 实用程序查看是否有布尔值可用于允许访问。例如，使用 getsebool -a | grep ftp 命令搜索 FTP 相关的布尔值：

```
~]$ getsebool -a | grep ftp
ftpd_anon_write --> off
ftpd_full_access --> off
ftpd_use_cifs --> off
ftpd_use_nfs --> off

ftpd_connect_db --> off
httpd_enable_ftp_server --> off
tftp_anon_write --> off
```

要列出布尔值列表以及布尔值是 on 或 off，请运行 getsebool -a 命令。对于布尔值列表，说明每个布尔值是什么，以及它们是 on 或 off，以 root 用户身份运行 semanage boolean -l 命令。有关列出和配置布尔值的详情，请查看 第 4.6 节 “布尔值”。

端口号

根据策略配置，服务只能在某些端口号中运行。尝试更改服务在没有更改策略的情况下运行的端口可能会导致服务无法启动。例如，以 root 用户身份运行 semanage port -l | grep http 命令以列出 http 相关端口：

```
~]# semanage port -l | grep http
http_cache_port_t              tcp      3128, 8080, 8118
http_cache_port_t              udp      3130
http_port_t                    tcp      80, 443, 488, 8008, 8009, 8443
pegasus_http_port_t            tcp      5988
pegasus_https_port_t           tcp      5989
```

http_port_t 端口类型定义 Apache HTTP 服务器可以侦听的端口，在本例中为 TCP 端口 80、443、488、8008、8009 和 8443。如果管理员配置了 httpd.conf，以便 httpd 侦听端口 9876(Listen 9876)，但没有更新策略来反映这一点，以下命令会失败：

```
~]# systemctl start httpd.service
Job for httpd.service failed. See 'systemctl status httpd.service' and 'journalctl -xn' for details.
```

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: failed (Result: exit-code) since Thu 2013-08-15 09:57:05 CEST; 59s ago
  Process: 16874 ExecStop=/usr/sbin/httpd $OPTIONS -k graceful-stop (code=exited,
status=0/SUCCESS)
  Process: 16870 ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND (code=exited,
status=1/FAILURE)
```

在 /var/log/audit/audit.log 中记录类似如下的 SELinux 拒绝信息：

```
type=AVC msg=audit(1225948455.061:294): avc:  denied  { name_bind } for  pid=4997
comm="httpd" src=9876 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=system_u:object_r:port_t:s0 tclass=tcp_socket
```

要允许 httpd 侦听 http_port_t 端口类型未列出的端口，请输入 semanage port 命令向策略配置添加端口[9]：

```
~]# semanage port -a -t http_port_t -p tcp 9876
```

a 选项添加新记录； -t 选项定义类型； -p 选项定义协议。最后的参数是要添加的端口号。

11.2.3. 演进规则和损坏的应用程序

应用程序可能会损坏，从而导致 SELinux 拒绝访问。另外，SELinux 规则会不断演变 - SELinux 可能没有了解某个应用程序会以某种特定方式运行，因此即使应用程序按预期工作，也有可能出现拒绝访问的问题。例如，如果 PostgreSQL 的新版本发布，它可能会执行之前没有看到过的当前策略的操作，从而导致访问被拒绝，即使应该允许访问。

对于这种情况，在访问被拒绝后，使用 audit2allow 实用程序创建自定义策略模块以允许访问。有关使用 audit2allow 的详情，请查看 第 11.3.8 节 “允许访问： audit2 允许”。

11.3. 修复问题

以下部分有助于对问题进行故障排除。它们进入：检查 Linux 权限，这些权限在 SELinux 规则之前检查；SELinux 拒绝访问的可能原因；服务的手册页，包含有关标记和布尔值的信息；允许一个进程运行许可而非整个系统的许可域；如何搜索和查看拒绝信息；分析拒绝信息；使用 audit2allow 创建自定义策略模块。

11.3.1. Linux 权限

拒绝访问时，检查标准 Linux 权限。如 第 1 章 简介 所述，大多数操作系统使用自主访问控制(DAC)系统来控制访问，允许用户控制他们拥有的文件的权限。在 DAC 规则后检查 SELinux 策略规则。如果 DAC 规则首先拒绝访问，则不会使用 SELinux 策略规则。

如果访问被拒绝，且没有记录 SELinux 拒绝，请使用以下命令查看标准 Linux 权限：

-

```
~]$ ls -l /var/www/html/index.html
-rw-r----- 1 root root 0 2009-05-07 11:06 index.html
```

在本例中，index .html 归 root 用户和组所有。root 用户具有读取和写入权限(-rw)，root 组的成员具有读取权限(-r-)。其他人没有访问权限(---)。默认情况下，此类权限不允许 httpd 读取此文件。要解决这个问题，请使用 chown 命令更改所有者和组。这个命令必须以 root 用户身份运行：

```
~]# chown apache:apache /var/www/html/index.html
```

这假定为默认配置，其中 httpd 以 Linux Apache 用户身份运行。如果您使用其他用户运行 httpd，请将 apache:apache 替换为该用户。

有关管理 Linux 权限的详情，请查看 Fedora 文档项目"权限" 草案。

11.3.2. 可能的 Silent Denials 原因

在某些情况下，当 SELinux 拒绝访问时，可能不会记录 AVC 拒绝消息。应用程序和系统库函数通常探测到比执行任务所需的更多访问权限。为了在不填写 AVC 拒绝的情况下对应用程序探测保持最小的特权，该策略可以在不使用 dontaudit 规则的情况下静默 AVC 拒绝。这些规则在标准策略中很常见。dontaudit 的缺点是，尽管 SELinux 拒绝访问，但拒绝消息不会被记录，从而使故障排除变得更加困难。

要临时禁用 dontaudit 规则，允许记录所有拒绝，以 root 用户身份输入以下命令：

```
~]# semodule -DB
```

d 选项 禁用 dontaudit 规则； -B 选项重新构建策略。运行 semodule -DB 后，尝试判断遇到权限问题的应用程序，并查看现在是否记录与应用程序相关的 SELinux 拒绝信息。请特别注意决定允许哪些拒绝，因为有些拒绝应该被忽略并由 dontaudit 规则处理。如果不确定，或在搜索指南时，请在 SELinux 列表中联系其他 SELinux 用户和开发人员，如 fedora-selinux-list。

要重建策略并启用 dontaudit 规则，以 root 用户身份输入以下命令：

```
~]# semodule -B
```

这会将策略恢复到其原始状态。要获得 dontaudit 规则的完整列表，请运行 sesearch --dontaudit 命令。使用 -s 域 选项和 grep 命令缩小搜索范围。例如：

```
~]$ sesearch --dontaudit -s smbd_t | grep squid
dontaudit smbd_t squid_port_t : tcp_socket name_bind ;
dontaudit smbd_t squid_port_t : udp_socket name_bind ;
```

有关分析拒绝的详情，请查看 第 11.3.6 节 “原始审计消息” 和 第 11.3.7 节 “sealert 消息”。

11.3.3. 为服务手动页面

服务的 man page 包含有价值的信息，如用于给定情况的文件类型，以及用于更改服务访问权限的布尔值（如 httpd 访问 NFS 卷）。此信息可能位于标准 man page 或 man page 中，可以使用 sepolicy manpage 实用程序从每个服务域的 SELinux 策略自动生成。此类 man page 以 service-name_selinux 格式命名。selinux-policy-doc 软件包还提供此类 man page。

例如： httpd_selinux(8) 手册页包含有关为给定情况使用哪些文件类型的信息，以及布尔值以允许脚本、共享文件、访问用户主目录中的目录等。其他包含服务的 SELinux 信息的 man page 包括：

- Samba：例如：启用 samba_enable_home_dirs 布尔值的 samba_selinux(8) 手册页可让 Samba 共享用户主目录。
- NFS: nfsd_selinux(8) 手册页描述了 SELinux nfsd 策略，允许用户尽可能安全地设置其 nfsd 进程。

man page 中的信息可帮助您配置正确的文件类型和布尔值，从而有助于阻止 SELinux 拒绝访问。

有关 sepolicy manpage 的详情请参考 第 5.4 节 “生成 man page: sepolicy manpage”。

11.3.4. 许可域

当 SELinux 以 permissive 模式运行时，SELinux 不会拒绝访问，但会记录对于在 enforcing 模式运行时会被拒绝的操作。在以前的版本中，无法使单个域许可（记住：进程在域中运行）。在某些情况下，这会导致整个系统对问题进行故障排除。

许可域允许管理员将单个进程（域）配置为运行许可，而不是使整个系统成为许可。仍然对许可域执行 SELinux 检查；但是，内核允许访问，并在 SELinux 拒绝访问的情况下报告 AVC 拒绝。

Permissive 域有以下用法：

- 它们可用于使单个进程（域）运行 permissive 模式来排除问题，而无需通过使整个系统成为许可关系而使其面临风险。

- 它们允许管理员为新应用创建策略。在以前的版本中，建议创建一个最小策略，然后整个机器都进入 permissive 模式，以便应用程序可以运行，但 SELinux 拒绝仍然被记录。audit2allow 随后可用于帮助编写策略。这使得整个系统面临风险。使用 permissive 域时，新策略中只有域可以标记为许可，而不会使整个系统面临风险。

11.3.4.1. 创建域许可

要使域成为许可状态，请运行 semanage permissive -a domain 命令，其中 domain 是 您想要许可的域。例如，以 root 用户身份输入以下命令使 httpd_t 域（运行 Apache HTTP 服务器的域）permissive:

```
~]# semanage permissive -a httpd_t
```

要查看您许可的域列表，请以 root 用户身份运行 semodule -l | grep permissive 命令。例如：

```
~]# semodule -l | grep permissive
permissive_httpd_t    (null)
permissivedomains     (null)
```

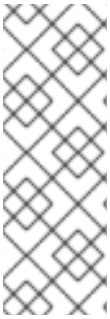
如果您不再希望某个域宽松，请以 root 用户身份运行 semanage permissive -d 域 命令。例如：

```
~]# semanage permissive -d httpd_t
```

11.3.4.2. 禁用许可域

permissivedomains.pp 模块包含系统上呈现的所有 permissive 域声明。要禁用所有 permissive 域，以 root 用户身份输入以下命令：

```
~]# semodule -d permissivedomains
```



注意

通过 **semodule -d** 命令禁用了策略模块后，**semodule -l** 命令的输出中将不再显示它。要查看包括禁用的所有策略模块，以 **root** 用户身份输入以下命令：

```
~]# semodule --list-modules=full
```

11.3.4.3. 拒绝许可域

对于 **permissive** 域，**SYSC ALL** 消息不同。以下是来自 **Apache HTTP** 服务器的 **AVC** 拒绝（及关联的系统调用）示例：

```
type=AVC msg=audit(1226882736.442:86): avc:  denied  { getattr } for  pid=2427 comm="httpd"
path="/var/www/html/file1" dev=dm-0 ino=284133 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file

type=SYSCALL msg=audit(1226882736.442:86): arch=40000003 syscall=196 success=no exit=-13
a0=b9a1e198 a1=bfc2921c a2=54dff4 a3=2008171 items=0 ppid=2425 pid=2427 auid=502 uid=48
gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=4 comm="httpd"
exe="/usr/sbin/httpd" subj=unconfined_u:system_r:httpd_t:s0 key=(null)
```

默认情况下，**httpd _t** 域不是 **permissive**，因此操作被拒绝，**SYSC ALL** 消息包含 **success=no**。以下是相同情况下 **AVC** 拒绝的示例，但运行 **semanage permissive -a httpd_t** 命令以使 **httpd_t** 域许可：

```
type=AVC msg=audit(1226882925.714:136): avc:  denied  { read } for  pid=2512 comm="httpd"
name="file1" dev=dm-0 ino=284133 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file

type=SYSCALL msg=audit(1226882925.714:136): arch=40000003 syscall=5 success=yes exit=11
a0=b962a1e8 a1=8000 a2=0 a3=8000 items=0 ppid=2511 pid=2512 auid=502 uid=48 gid=48
euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=4 comm="httpd"
exe="/usr/sbin/httpd" subj=unconfined_u:system_r:httpd_t:s0 key=(null)
```

在这种情况下，尽管记录了 **AVC** 拒绝，但访问并未被拒绝，如 **SYSCALL** 消息中的 **success=yes** 所示。

有关 **permissive** 域的更多信息，请参阅 **Dan Walsh** 的"许可域" 博客条目。

11.3.5. 搜索和查看地址

本节假定已安装 setroubleshoot、setroubleshoot-server、dbus 和 audit 软件包，并且 auditd、rsyslogd 和 setroubleshootd 守护进程正在运行。有关启动这些守护进程的详情请参考 第 4.2 节 “使用哪个日志文件”。有多个实用程序可用于搜索和查看 SELinux AVC 信息，如 ausearch、aureport 和 sealert。

ausearch

audit 软件包提供了 ausearch 实用程序，可根据不同的搜索条件查询 审计 守护进程日志中的事件。[10] ausearch 工具程序访问 /var/log/audit/audit.log，因此必须以 root 用户身份运行：

搜索: 所有拒绝

命令: ausearch -m avc,user_avc,selinux_err,user_selinux_err

搜索: 今天拒绝这种情况

命令: ausearch -m avc -ts today

搜索: 拒绝来自最后 10 分钟

命令: ausearch -m avc -ts recent

要搜索特定服务的 SELinux AVC 消息，请使用 -c comm-name 选项，其中 comm-name 是可执行文件的名称，例如 httpd 用于 Apache HTTP 服务器，smb d 用于 Samba：

```
~]# ausearch -m avc -c httpd
```

```
~]# ausearch -m avc -c smbd
```

对于每个 ausearch 命令，建议您使用 --interpret (-i)选项来简化可读性，或使用 --raw(-r)选项进行脚本处理。有关更多 ausearch 选项，请参阅 ausearch(8) 手册页。

aureport

audit 软件包提供 anu report 实用程序，用于生成审计系统日志的摘要报告。[11] aureport 实用程序访问 /var/log/audit/audit.log，因此必须以 root 用户身份运行。要查看 SELinux 拒绝消息列表以及每个消息发生的频率，请运行 aureport -a 命令。以下是包含两个拒绝的输出示例：

```
~]# aureport -a

AVC Report
===============================================================
# date time comm subj syscall class permission obj event
===============================================================
1. 05/01/2009 21:41:39 httpd unconfined_u:system_r:httpd_t:s0 195 file getattr
```

> *system_u:object_r:samba_share_t:s0 denied 2*
> *2. 05/03/2009 22:00:25 vsftpd unconfined_u:system_r:ftpd_t:s0 5 file read unconfined_u:object_r:cifs_t:s0 denied 4*

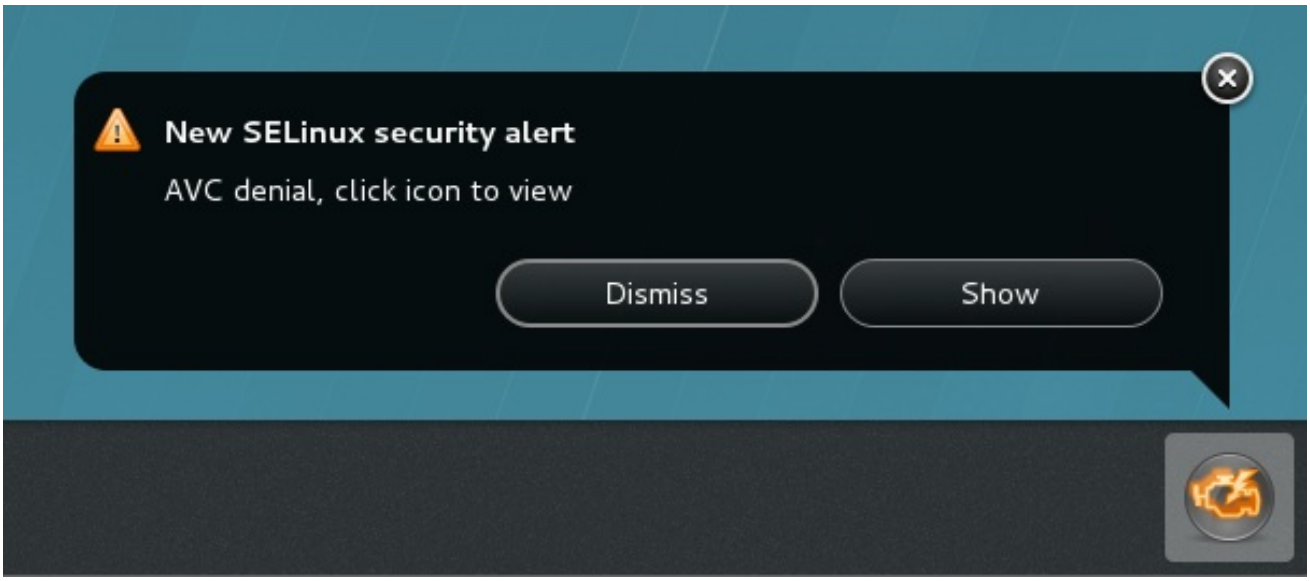
sealert

setroubleshoot-server 软件包提供 sealert 实用程序，它会读取 setroubleshoot-server 转换的拒绝消息。[12] 拒绝分配 ID，如 /var/log/messages 所示。以下是拒绝 信息的示例 ：

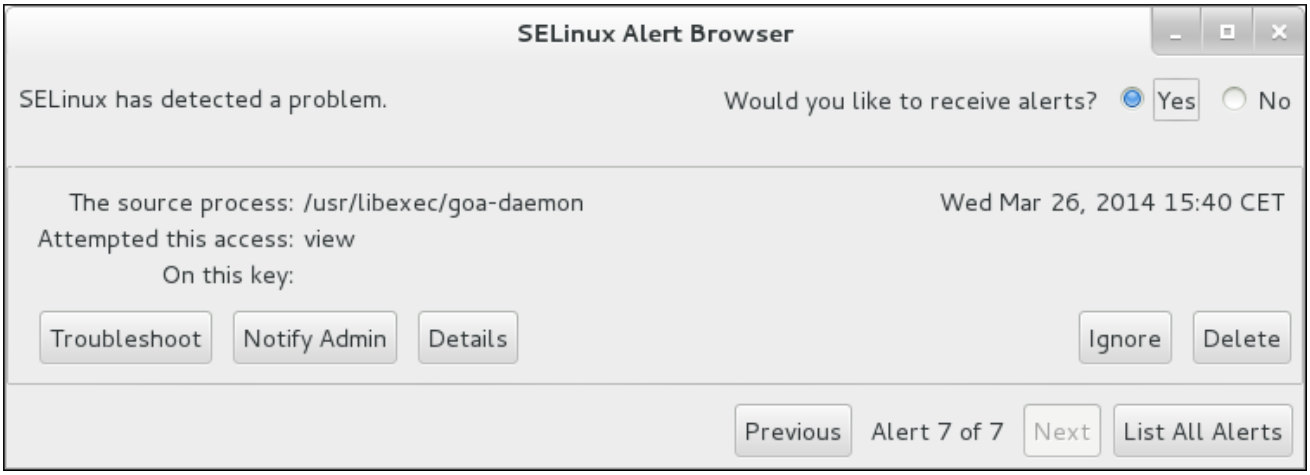
> *setroubleshoot: SELinux is preventing /usr/sbin/httpd from name_bind access on the tcp_socket. For complete SELinux messages. run sealert -l 8c123656-5dda-4e5d-8791-9e3bd03786b7*

在本例中，拒绝 ID 是 8c123656-5dda-4e5d-8791-9e3bd03786b7。l 选项 取 ID 作为参数。运行 sealert -l 8c123656-5dda-4e5d-8791-9e3bd03786b7 命令可以详细分析 SELinux 拒绝访问的原因，以及允许访问的可能解决方案。

如果您正在运行 X Window 系统，安装 setroubleshoot 和 setroubleshoot-server 软件包，并且 setroubleshootd、dbus 和 auditd 守护进程正在运行，SELinux 拒绝访问时会显示警告：



点击 Show 启动 sealert GUI，允许您对问题进行故障排除：



*或者，运行 **sealert -b** 命令以启动 **sealert GUI**。要查看所有拒绝信息的详细分析，请运行 **sealert -l *** 命令。*

11.3.6. 原始审计消息

*原始审计消息记录到 **/var/log/audit/audit.log**。以下是 **Apache HTTP** 服务器（在 **httpd_t** 域中运行）尝试访问 **/var/www/html/file1** 文件（使用 **samba_share_t** 类型标记）时发生的 **AVC** 拒绝消息（以及关联的系统调用）：*

```
type=AVC msg=audit(1226874073.147:96): avc:  denied  { getattr } for  pid=2465 comm="httpd"
path="/var/www/html/file1" dev=dm-0 ino=284133 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file

type=SYSCALL msg=audit(1226874073.147:96): arch=40000003 syscall=196 success=no exit=-13
a0=b98df198 a1=bfec85dc a2=54dff4 a3=2008171 items=0 ppid=2463 pid=2465 auid=502 uid=48
gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=6 comm="httpd"
exe="/usr/sbin/httpd" subj=unconfined_u:system_r:httpd_t:s0 key=(null)
```

{ getattr }

*大括号中的项目表示拒绝的权限。**getattr** 条目指示源进程正在尝试读取目标文件的状态信息。这在读取文件前发生。由于访问的文件带有错误的标签，因此此操作被拒绝。通常看到的权限包括 **getattr**、读取和写入 。*

comm="httpd"

*启动进程的可执行文件。可执行文件的完整路径可在系统调用(SYSCALL)消息的 **exe=** 部分中找到，本例中为 **exe="/usr/sbin/httpd"**。*

path="/var/www/html/file1"

进程试图访问的对象（目标）路径。

scontext="unconfined_u:system_r:httpd_t:s0"

尝试拒绝的操作的进程的 SELinux 上下文。在本例中，这是 Apache HTTP 服务器的 SELinux 上下文，它在 httpd_t 域中运行。

tcontext="unconfined_u:object_r:samba_share_t:s0"

进程试图访问的对象（目标）的 SELinux 上下文。在本例中，这是 file1 的 SELinux 上下文。请注意，在 httpd _t 域中运行的进程无法访问 samba_share _t 类型。

在某些情况下，tcontext 可能与 scontext 匹配，例如，当进程尝试执行将更改该运行进程的特征的系统服务（如用户 ID）时。另外，当进程试图使用超出正常限值允许的资源（如内存）时，t context 可能会与 scontext 匹配，从而进行安全检查以查看是否允许该进程突破这些限制。

在系统调用(SYSCALL)消息中，需要两个项目：

- success=no: 表示是否强制拒绝(AVC)。succes s=no 表示系统调用没有成功（SELinux 被拒绝的访问）。s uccess=yes 表示系统调用是否成功。这可用于 permissive 域或未限制的域，如 unconfined_service_t 和 kernel_t。
- exe="/usr/sbin/httpd" ：启动该进程的可执行文件的完整路径，在本例中为 exe="/usr/sbin/httpd"。

文件类型不正确是 SELinux 拒绝访问的常见原因。要开始进行故障排除，请将源上下文(scontext)与目标上下文(tcontext)进行比较。进程(scontext)是否应该访问这样的对象(tcontext)？例如： Apache HTTP 服务器(httpd_t)应该只访问 httpd_selinux(8) 手册页中指定的类型，如 httpd_sys_content_t、public_content_t 等等。

11.3.7. sealert 消息

拒绝分配 ID，如 /var/log/messages 所示。以下是 Apache HTTP 服务器（在 httpd_t 域中运行）试图访问 /var/www/html/file1 文件（使用 samba_share_t 类型标记）时发生的 AVC 拒绝示例：

```
hostname setroubleshoot: SELinux is preventing httpd (httpd_t) "getattr" to /var/www/html/file1 (samba_share_t). For complete SELinux messages. run sealert -l 32eee32b-21ca-4846-a22f-0ba050206786
```

如建议，运行 sealert -l 32eee32b-21ca-4846-a22f-0ba050206786 命令以查看完整的消息。这个命令只适用于本地机器，并显示与 sealert GUI 相同的信息：

```
~]$ sealert -l 32eee32b-21ca-4846-a22f-0ba050206786
SELinux is preventing httpd from getattr access on the file /var/www/html/file1.

*****  Plugin restorecon (92.2 confidence) suggests   ************************

If you want to fix the label.
/var/www/html/file1 default label should be httpd_sys_content_t.
Then you can run restorecon.
Do
# /sbin/restorecon -v /var/www/html/file1

*****  Plugin public_content (7.83 confidence) suggests   ********************

If you want to treat file1 as public content
Then you need to change the label on file1 to public_content_t or public_content_rw_t.
Do
# semanage fcontext -a -t public_content_t '/var/www/html/file1'
# restorecon -v '/var/www/html/file1'

*****  Plugin catchall (1.41 confidence) suggests   **************************

If you believe that httpd should be allowed getattr access on the file1 file by default.
Then you should report this as a bug.
You can generate a local policy module to allow this access.
Do
allow this access for now by executing:
# ausearch -c 'httpd' --raw | audit2allow -M my-httpd
# semodule -i my-httpd.pp


Additional Information:
Source Context                system_u:system_r:httpd_t:s0
Target Context                unconfined_u:object_r:samba_share_t:s0
Target Objects                /var/www/html/file1 [ file ]
Source                        httpd
Source Path                   httpd
Port                          <Unknown>
Host                          hostname.redhat.com
Source RPM Packages
Target RPM Packages
Policy RPM                    selinux-policy-3.13.1-166.el7.noarch
Selinux Enabled               True
Policy Type                   targeted
Enforcing Mode                Enforcing
Host Name                     hostname.redhat.com
Platform                      Linux hostname.redhat.com
                              3.10.0-693.el7.x86_64 #1 SMP Thu Jul 6 19:56:57
                              EDT 2017 x86_64 x86_64
Alert Count                   2
First Seen                    2017-07-20 02:52:11 EDT
Last Seen                     2017-07-20 02:52:11 EDT
Local ID                      32eee32b-21ca-4846-a22f-0ba050206786

Raw Audit Messages
```

> type=AVC msg=audit(1500533531.140:295): avc: denied { getattr } for pid=24934 comm="httpd" path="/var/www/html/file1" dev="vda1" ino=31457414 scontext=system_u:system_r:httpd_t:s0 tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file
>
> Hash: httpd,httpd_t,samba_share_t,file,getattr

概述

被拒绝操作的简短概述。这与 /var/log/messages 中的拒绝相同。在本例中，httpd 进程 被拒绝访问文件（文件1），该文件使用 samba_share_t 类型进行标记。

详细描述

更为详细的描述。在本例中，file 1 使用 samba_share_t 类型进行标记。此类型用于您要使用 Samba 导出的文件和目录。描述建议将类型更改为 Apache HTTP 服务器和 Samba 可访问的类型（如果需要此类访问）。

允许访问

有关如何允许访问的建议。这可以是重新标记文件、启用布尔值或创建本地策略模块。在这种情况下，建议使用 Apache HTTP 服务器和 Samba 可访问的类型标记该文件。

修复命令

建议将 命令允许访问并解决拒绝。在本例中，它提供了 命令将 file1 类型更改为 public_content_t 命令，后者可供 Apache HTTP 服务器和 Samba 访问。

其它信息

错误报告中有用的信息，如策略软件包名称和版本(selinux-policy-3.13.1-166.el7.noarch)，但可能不适用于解决拒绝的原因。

原始审计消息

与拒绝关联的 /var/log/audit/audit.log 中的原始审计消息。有关 AVC 拒绝中每个项目的信息，请参阅 第 11.3.6 节 “原始审计消息”。

11.3.8. 允许访问： audit2 允许

警告

不要在生产中使用本节中的示例。它仅用于演示 audit2allow 实用程序的使用。

audit2allow 实用程序从被拒绝的操作日志中收集信息，然后生成 SELinux 策略允许规则。[13] 根据 第 11.3.7 节 “sealert 消息” 分析拒绝信息后，如果没有标签更改或布尔值允许访问，请使用 audit2allow 创建本地策略模块。当 SELinux 拒绝访问时，运行 audit2 会生成允许 之前拒绝的访问的 Type Enforcement 规则。

当您看到 SELinux 拒绝时，您不应该使用 audit2allow 生成本地策略模块作为您的第一个选项。故障排除应该先检查是否有标记问题。第二个最常见的情况是您更改了进程配置，并且忘记了要让 SELinux 了解它。如需更多信息，请参阅 SELinux 错误的四个关键原因 白皮书。

以下示例演示了使用 audit2allow 创建策略模块：

1. 拒绝信息和相关系统调用记录到 /var/log/audit/audit.log 文件中：

```
type=AVC msg=audit(1226270358.848:238): avc:  denied  { write } for  pid=13349
comm="certwatch" name="cache" dev=dm-0 ino=218171
scontext=system_u:system_r:certwatch_t:s0 tcontext=system_u:object_r:var_t:s0 tclass=dir

type=SYSCALL msg=audit(1226270358.848:238): arch=40000003 syscall=39 success=no
exit=-13 a0=39a2bf a1=3ff a2=3a0354 a3=94703c8 items=0 ppid=13344 pid=13349
auid=4294967295 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=(none)
ses=4294967295 comm="certwatch" exe="/usr/bin/certwatch"
subj=system_u:system_r:certwatch_t:s0 key=(null)
```

在本例中，certwatch 被拒绝了对标有 var_t 类型的目录的写入访问。根据 第 11.3.7 节 “sealert 消息” 分析拒绝消息。如果没有标签更改或布尔值被允许访问，请使用 audit2allow 创建本地策略模块。

2. 输入以下命令生成人类可读的描述，说明为什么拒绝访问。audit2allow 工具读取 /var/log/audit/audit.log，因此必须以 root 用户身份运行：

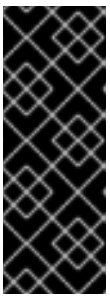
```
~]# audit2allow -w -a
type=AVC msg=audit(1226270358.848:238): avc:  denied  { write } for  pid=13349
comm="certwatch" name="cache" dev=dm-0 ino=218171
```

```
scontext=system_u:system_r:certwatch_t:s0 tcontext=system_u:object_r:var_t:s0 tclass=dir
 Was caused by:
  Missing type enforcement (TE) allow rule.

 You can use audit2allow to generate a loadable module to allow this access.
```

a 命令行 选项可使所有审计日志被读取。w 选项 生成人类可读的描述。如上所示，缺少 Type Enforcement 规则，因此访问被拒绝。

3. **输入以下命令查看允许拒绝访问的 Type Enforcement 规则：**

```
~]# audit2allow -a


#============= certwatch_t ==============
allow certwatch_t var_t:dir write;
```



重要

缺少 Type Enforcement 规则通常由 SELinux 策略中的错误导致，应在 Red Hat Bugzilla 中报告。对于 Red Hat Enterprise Linux，针对 Red Hat Enterprise Linux 产品创建程序错误，然后选择 selinux-policy 组件。在此类错误报告中，包括 audit2allow -w -a 和 audit2allow -a 命令的输出。

4. **要使用 audit2allow -a 显示的规则，以 root 用户身份输入以下命令来创建自定义模块。在当前工作目录中， 使用 -M 指定的名称创建一个 Type Enforcement 文件(.te)：**

```
~]# audit2allow -a -M mycertwatch
****************** IMPORTANT ***********************
To make this policy package active, execute:

semodule -i mycertwatch.pp
```

5. **此外，audit2 允许将 Type Enforcement 规则编译 到策略软件包(.pp)中：**

```
~]# ls
mycertwatch.pp  mycertwatch.te
```

要安装模块，以 root 用户身份输入以下命令：

```
~]# semodule -i mycertwatch.pp
```

重要

使用 audit2 允许 创建的模块可能会允许超过所需的访问权限。建议通过 audit2allow 创建的策略发布到上游 SELinux 列表中以供审核。 如果您认为策略中存在错误，请在 Red Hat Bugzilla 中创建一个错误。

如果您有多个来自多个进程的拒绝消息，但只想为单个进程创建自定义策略，请使用 grep 实用程序缩小对 audit2allow 的输入范围。以下示例演示了使用 grep 仅通过 audit2allow 发送与 certwatch 相关的拒绝信息：

```
~]# grep certwatch /var/log/audit/audit.log | audit2allow -R -M mycertwatch2
****************** IMPORTANT ***********************
To make this policy package active, execute:

semodule -i mycertwatch2.pp
```

---

[8] /etc/selinux/targeted/contexts/files/ 中的文件定义文件和目录的上下文.restorecon 读取此目录中的文件，并将 文件实用程序设置为将 文件和目录恢复到其默认上下文。

[9] semanage port -a 命令向 /etc/selinux/targeted/modules/active/ports.local 文件中添加条目。请注意，默认情况下，该文件只能由 root 用户查看。

[10] 有关 ausearch 的详情，请查看 ausearch(8) 手册页。

[11] 有关工作端口的详情，请查看 aureport(8) 手册页。

[12] 有关 sealert 的详情，请查看 sealert(8) 手册页。

[13] 有关 audit2allow 的更多信息，请参阅 audit2allow(1) 手册页。

第 12 章 更多信息

12.1. 贡献者

- Dominick Grift - 技术编辑器
- Murray McAllister - 红帽产品安全
- James Morris - 技术编辑器
- Eric Malaysia - 技术编辑器
- Scott Radvan - 红帽客户内容服务
- Daniel Walsh - 红帽安全工程

12.2. 其他资源

Fedora

- 主页： http://fedoraproject.org/wiki/SELinux
- 故障排除： http://fedoraproject.org/wiki/SELinux/Troubleshooting
- Fedora SELinux https://fedoraproject.org/wiki/SELinux_FAQ 常见问题：

国家安全局(NSA)

NSA 是原始的 SELinux 开发人员。NSA 国家信息保障研究实验室(NIARL)的研究人员在 Linux 内核的主要子系统中设计并实施了灵活的强制访问控制，并且实施 Flask 架构提供的新操作系统组件，即安全

服务器和访问电量缓存。

- *主 SELinux 网站： https://www.nsa.gov/what-we-do/research/selinux/*
- *SELinux 邮件列表： https://www.nsa.gov/what-we-do/research/selinux/mailing-list.shtml*
- *SELinux https://www.nsa.gov/what-we-do/research/selinux/documentation/index.shtml 文档：*
- *SELinux 背景： https://www.nsa.gov/what-we-do/research/selinux/related-work/*

Tresys Technology

Tresys Technology 是上游技术，用于：

- *SELinux 用户空间库和工具.*
- *SELinux 参考策略.*

SELinux GitHub 存储库

- *SELinux 项目： https://github.com/SELinuxProject*
- *Tresys 技术： https://github.com/TresysTechnology/*

SELinux Project Wiki

- *主页： http://selinuxproject.org/page/Main_Page*
- *用户资源，包括文档、邮件列表、网站和工具链接： http://selinuxproject.org/page/User_Resources*

SELinux 笔记本 - 基础 - 第 4 版

http://freecomputerbooks.com/books/The_SELinux_Notebook-4th_Edition.pdf

DigitalOcean：CentOS 7 中的 SELinux 简介

https://www.digitalocean.com/community/tutorial_series/an-introduction-to-selinux-on-centos-7

IRC

在 Freenode 上：

- *#selinux*
- *#fedora-selinux*

部分 II. 管理受限服务

本书的这一部分将更多关注实际任务，并提供了如何设置和配置各种服务的信息。对于每一服务，列出了最常用的类型和布尔值。还包括配置这些服务的真实示例，以及 SELinux 如何补充其操作的演示。

当 SELinux 处于 enforcing 模式时，Red Hat Enterprise Linux 中使用的默认策略是目标策略。目标在受限域中运行的进程和不是目标运行的进程在不受限制的域中运行。如需有关目标策略、受限制和不受限制的进程的更多信息，请参阅 第 3 章 目标策略。

第 13 章 APACHE HTTP 服务器

Apache HTTP 服务器提供具有当前 HTTP 标准的开源 HTTP 服务器。[14]

在红帽企业 Linux 中，httpd 软件包 提供了 Apache HTTP 服务器。输入以下命令查看是否安装了 httpd 软件包：

```
~]$ rpm -q httpd
package httpd is not installed
```

如果没有安装，且您想要使用 Apache HTTP 服务器，请以 root 用户身份使用 yum 工具来安装它：

```
~]# yum install httpd
```

13.1. APACHE HTTP 服务器和 SELINUX

启用 SELinux 时，Apache HTTP 服务器(httpd)默认运行限制。受限制的进程在其自己的域中运行，并且与其他受限制的进程隔开。如果一个受攻击者限制的进程受到 SELinux 策略配置的影响，攻击者对资源的访问权限和可能受到的破坏会受到限制。以下示例演示了在自己的域中运行的 httpd 进程。本例假设安装了 httpd、setroubleshoot、setroubleshoot-server 和 policycoreutils-python 软件包：

1. 运行 getenforce 命令确认 SELinux 是否以强制模式运行：

   ```
   ~]$ getenforce
   Enforcing
   ```

   当 SELinux 以强制模式运行时，该命令将返回 强制。

2. 以 root 用户身份输入以下命令启动 httpd ：

   ```
   ~]# systemctl start httpd.service
   ```

   确认 服务正在运行。输出中应包括以下信息（只有时间戳有所不同）：

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since Mon 2013-08-05 14:00:55 CEST; 8s ago
```

3. 要查看 httpd 进程，请执行以下命令：

```
~]$ ps -eZ | grep httpd
system_u:system_r:httpd_t:s0    19780 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19781 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19782 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19783 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19784 ?        00:00:00 httpd
system_u:system_r:httpd_t:s0    19785 ?        00:00:00 httpd
```

与 httpd 进程关联的 SELinux 上下文是 system_u:system_r:httpd_t:s0。上下文的第二部分 httpd_t 是类型。类型定义进程的域以及文件的类型。在本例中，httpd 进程在 httpd _t 域中运行。

SELinux 策略定义在受限域（如 httpd_t）内运行的进程如何与文件、其他进程和系统交互。文件必须正确标记，以允许 httpd 访问这些文件。例如，httpd 可以 读取使用 httpd_sys_content_t 类型标记的文件，但无法写入这些文件，即使 Linux(DAC)权限允许写入访问。必须启用布尔值以允许某些行为，例如允许脚本网络访问、允许 httpd 访问 NFS 和 CIFS 卷，以及允许 httpd 执行通用网关接口(CGI)脚本。

配置 /etc/httpd/conf/httpd.conf 文件后，httpd 侦听 TCP 端口 80、443、488、8008、8009 或 8443 以外的端口，必须使用 semanage port 命令将新端口号添加到 SELinux 策略配置。以下示例演示了将 httpd 配置为侦听 httpd 的 SELinux 策略配置中尚未定义的端口， 因此 httpd 无法启动。本例还演示如何配置 SELinux 系统，以允许 httpd 成功侦听策略中尚未定义的非标准端口。本例假定已安装了 httpd 软件包。以 root 用户身份运行示例中的每个命令：

1. 输入以下命令确认 httpd 没有运行：

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
        Active: inactive (dead)
```

如果输出不同，请停止该进程：

```
~]# systemctl stop httpd.service
```

2. 使用 semanage 工具查看 SELinux 允许 httpd 侦听的端口：

```
~]# semanage port -l | grep -w http_port_t
http_port_t                    tcp      80, 443, 488, 8008, 8009, 8443
```

3. 以 root 用户身份编辑 /etc/httpd/conf/httpd.conf 文件。配置 Listen 选项，使它列出 httpd 的 SELinux 策略配置中未配置的端口。在本例中，httpd 被配置为侦听端口 12345：

```
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses (0.0.0.0)
#
#Listen 12.34.56.78:80
Listen 127.0.0.1:12345
```

4. 输入以下命令启动 httpd ：

```
~]# systemctl start httpd.service
Job for httpd.service failed. See 'systemctl status httpd.service' and 'journalctl -xn' for details.
```

记录类似如下的 SELinux 拒绝信息：

```
setroubleshoot: SELinux is preventing the httpd (httpd_t) from binding to port 12345. For
complete SELinux messages. run sealert -l f18bca99-db64-4c16-9719-1db89f0d8c77
```

5. 为了使 SELinux 允许 httpd 侦听端口 12345，如本例中使用的那样，需要使用以下命令：

```
~]# semanage port -a -t http_port_t -p tcp 12345
```

6. 再次启动 httpd 并使它监听新端口：

```
~]# systemctl start httpd.service
```

7. 现在 SELinux 已配置为允许 httpd 侦听非标准端口（本示例中为 TCP 12345），httpd 在此端口上 成功启动。

8. 要证明 httpd 正在 TCP 端口 12345 上侦听和通信，请打开到指定端口的 telnet 连接并发出 HTTP GET 命令，如下所示：

```
~]# telnet localhost 12345
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET / HTTP/1.0

HTTP/1.1 200 OK
Date: Wed, 02 Dec 2009 14:36:34 GMT
Server: Apache/2.2.13 (Red Hat)
Accept-Ranges: bytes
Content-Length: 3985
Content-Type: text/html; charset=UTF-8
[...continues...]
```

13.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

以下示例在 /var/www/html/ 目录中创建一个新文件，并显示了从父目录(/var/www/html/)继承 httpd_sys_content_t 类型的文件：

1. 输入以下命令查看 /var/www/html/ 的 SELinux 上下文：

```
~]$ ls -dZ /var/www/html
drwxr-xr-x  root root system_u:object_r:httpd_sys_content_t:s0 /var/www/html
```

这表明 /var/www/html/ 已使用 httpd_sys_content_t 类型进行标记。

2. 以 root 用户身份使用 touch 工具创建新文件：

```
~]# touch /var/www/html/file1
```

3. 输入以下命令查看 SELinux 上下文：

```
~]$ ls -Z /var/www/html/file1
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/file1
```

ls -Z 命令显示标有 httpd_sys_content_t 类型 的文件1。SELinux 允许 httpd 读取使用此类型标记的文件，但不允许写入它们，即使 Linux 权限允许写入访问。SELinux 策略定义 httpd_t 域中运行的进程类型（httpd 运行 位置）可以读取和写入到哪些类型。这有助于防止进程访问由另一个进程使用的文件。

例如，httpd 可以访问使用 httpd_sys_content_t 类型（通常用于 Apache HTTP 服务器）标记的文件，但默认情况下无法访问使用 samba_share_t 类型标记的文件（适用于 Samba）。此外，用户主目录中的文件标有 user_home_t 类型： 默认情况下，这可防止 httpd 读取或写入用户主目录中的文件。

下表列出了用于 httpd 的一些类型。不同的类型允许您配置灵活的访问：

httpd_sys_content_t

将此类型用于静态 Web 内容，如静态网站使用的 .html 文件。使用此类型的标记的文件可以访问 httpd 和 httpd 执行脚本（仅读取 ）。默认情况下，httpd 或其他进程无法将标有此类型的文件和目录写入或修改。请注意，默认情况下，在 中创建或复制到 /var/www/html/ 目录中的文件使用 httpd_sys_content_t 类型进行标记。

httpd_sys_script_exec_t

将此类型用于您希望 httpd 执行的脚本。此类型通常用于 /var/www/cgi-bin/ 目录中的通用网关接口(CGI)脚本。默认情况下，SELinux 策略阻止 httpd 执行 CGI 脚本。要允许这种情况，请使用 httpd_sys_script_exec_t 类型标记脚本并启用 httpd_enable_cgi 布尔值。由 httpd 执行时，标有 httpd_sys_script_exec _t 的脚本在 httpd_sys_script_t 域中运行 。httpd_sys_script_t 域可以访问其他系统域，如 postgresql_t 和 mysqld_t。

httpd_sys_rw_content_t

使用此类型标记的文件可以通过使用 httpd_sys_script_exec_t 类型标记的脚本写入到 中，但无法通过标记任何其他类型的脚本进行修改。您必须使用 httpd_sys_rw_content_t 类型标记要从 并通过 httpd_sys_script_exec_t 类型的脚本读取并写入的文件。

httpd_sys_ra_content_t

使用此类型标记的文件可以通过使用 httpd_sys_script_exec_t 类型标记的脚本附加到，但无法通过标记任何其他类型的脚本进行修改。您必须使用 httpd_sys_ra_content_t 类型来 标记要从中读取的文件，并通过标有 httpd_sys_script_exec_t 类型的脚本将其附加到。

httpd_unconfined_script_exec_t

带有此类型标记的脚本在未受到 SELinux 保护的情况下运行。耗尽所有其他选项后，仅将此类型用于复杂的脚本。最好使用这种类型，而不是对 httpd 或整个系统禁用 SELinux 保护。

注意

要查看 httpd 的更多可用类型，请输入以下命令：

```
~]$ grep httpd /etc/selinux/targeted/contexts/files/file_contexts
```

过程 13.1. 更改 SELinux 上下文

可以使用 chcon 命令更改文件和目录的类型。使用 chcon 进行的更改不会在文件系统重新标记或 restorecon 命令后保留。SELinux 策略控制用户是否能够修改任何给定文件的 SELinux 上下文。以下示例演示了创建新目录和供 httpd 使用的 index.html 文件，并标记该文件和目录以允许 httpd 访问它们：

1. 以 root 用户身份使用 mkdir 实用程序创建顶级目录结构以存储供 httpd 使用的文件：

```
~]# mkdir -p /my/website
```

2. 在 file-context 配置中与 模式不匹配的文件和目录可能会使用 default_t 类型进行标记。受限制的服务无法访问此类型：

```
~]$ ls -dZ /my
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /my
```

3. 以 root 身份输入以下命令，将 my/ 目录和子目录的类型更改为 httpd 可访问的类型：现在，在 /my/website/ 下创建的文件会继承 httpd_sys_content_t 类型，而不是 default_t 类型，因此 httpd 可以访问：

```
~]# chcon -R -t httpd_sys_content_t /my/
~]# touch /my/website/index.html
~]# ls -Z /my/website/index.html
-rw-r--r--  root root unconfined_u:object_r:httpd_sys_content_t:s0 /my/website/index.html
```

有关 chcon 的更多信息，请参阅 第 4.7.1 节 “临时更改：chcon”。

使用 semanage fcontext 命令（semanage 由 policycoreutils-python 软件包提供），以在重新标记和 restorecon 命令保留标签更改。此命令会添加对 file-context 配置的更改。然后，运行 restorecon（读取 file-context 配置）以应用标签更改。以下示例演示了创建新目录和供 httpd 使用的 index.html 文件，并永久更改该目录和文件的标签以允许 httpd 访问它们：

1. 以 root 用户身份使用 mkdir 实用程序创建顶级目录结构以存储供 httpd 使用的文件：

```
~]# mkdir -p /my/website
```

2. 以 root 用户身份输入以下命令，将标签更改添加到 file-context 配置中：

```
~]# semanage fcontext -a -t httpd_sys_content_t "/my(/.*)?"
```

"/my(/.*)?" 表达式表示标签更改适用于 my/ 目录及其下的所有文件和目录。

3. 以 root 用户身份使用 touch 工具创建新文件：

```
~]# touch /my/website/index.html
```

4. 以 root 身份输入以下命令来应用标签更改（restorecon 读取 file-context 配置，这由 semanage 命令在第 2 步中修改）：

```
~]# restorecon -R -v /my/
restorecon reset /my context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /my/website context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /my/website/index.html context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

有关 semanage 的详情请参考 第 4.7.2 节 “持久性更改：semanage fcontext”。

13.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。这可以通过布尔值来实现，这些布尔值允许在运行时更改 SELinux 策略的部分，而无需了解 SELinux 策略的编写。这允许更改，例如允许服务访问 NFS 卷，而无需重新加载或重新编译 SELinux 策略。

若要修改布尔值的状态，可使用 setsebool 命令。例如，要启用 httpd_anon_write 布尔值，以 root 用户身份输入以下命令：

```
~]# setsebool -P httpd_anon_write on
```

要禁用布尔值，使用同一个示例，只需 在 命令中切换到 off 即可，如下所示：

```
~]# setsebool -P httpd_anon_write off
```

注意

如果您不希望 setsebool 更改在重新引导后保留，则不要使用 -P 选项。

以下是可用常见布尔值的描述，可满足 httpd 的运行方式：

httpd_anon_write

禁用后，此布尔值仅允许 httpd 对标记为 public_content_rw_t 类型的文件具有读取访问权限。启用此布尔值可允许 httpd 写入标有 public_content_rw_t 类型的文件，例如包含公共文件传输服务文件的公共目录。

httpd_mod_auth_ntlm_winbind

启用此布尔值允许使用 httpd 中的 mod_auth_ntlm_winbind 模块访问 NTLM 和 Winbind 身份验证机制。

httpd_mod_auth_pam

启用此布尔值允许使用 httpd 中的 mod_auth_pam 模块访问 PAM 身份验证机制。

httpd_sys_script_anon_write

此布尔值定义是否允许 HTTP 脚本对标记为 public_content_rw_t 类型的文件进行写入访问，这在公共文件传输服务中使用。

httpd_builtin_scripting

此布尔值定义对 httpd 脚本的访问。PHP 内容通常需要启用此布尔值。

httpd_can_network_connect

禁用后，这个布尔值可防止 HTTP 脚本和模块启动与网络或远程端口的连接。启用此布尔值以允许此访问。

httpd_can_network_connect_db

禁用后，此布尔值可防止 HTTP 脚本和模块启动与数据库服务器的连接。启用此布尔值以允许此访问。

httpd_can_network_relay

当 httpd 用作正向或反向代理时，启用此布尔值。

httpd_can_sendmail

禁用后，此布尔值可防止 HTTP 模块发送邮件。如果 httpd 中找到漏洞，这可以防止垃圾邮件攻击。启用此布尔值以允许 HTTP 模块发送邮件。

httpd_dbus_avahi

禁用后，此布尔值拒绝 httpd 通过D-Bus 访问 avahi 服务。启用此布尔值以允许此访问。

httpd_enable_cgi

禁用后，此布尔值会阻止 httpd 执行 CGI 脚本。启用此布尔值以允许 httpd 执行 CGI 脚本（CGI 脚本必须使用 httpd_sys_script_exec_t 类型标记）。

httpd_enable_ftp_server

启用此布尔值允许 httpd 侦听 FTP 端口并充当 FTP 服务器。

httpd_enable_homedirs

禁用后，此布尔值阻止 httpd 访问用户主目录。启用此布尔值以允许 httpd 访问用户主目录；例如，/home/*/ 中的内容。

httpd_execmem

启用后，此布尔值允许 httpd 执行需要内存地址可执行和可写入的程序。不建议从安全角度启用此布尔值，因为它可减少对缓冲区溢出的保护，但某些模块和应用程序（如 Java 和 Mono 应用）需要此特权。

httpd_ssi_exec

此布尔值定义服务器端是否可以在网页中包含(SSI)元素。

httpd_tty_comm

此布尔值定义是否允许 httpd 访问控制终端。通常不需要这种访问，但在配置 SSL 证书文件等情形中，需要进行终端访问来显示和处理密码提示符。

httpd_unified

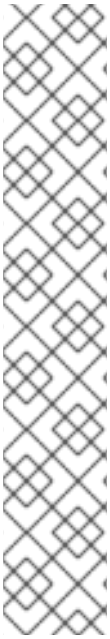
启用后，此布尔值允许 httpd_t 完全访问所有 httpd 类型（即执行、读取或写入 sys_content_t）。禁用后，将区分只读、可写入或可执行的 Web 内容。禁用此布尔值可确保额外的安全级别，但增加了管理开销，即必须根据每个布尔值应具有的文件访问权限单独标记脚本和其他 Web 内容。

httpd_use_cifs

启用此布尔值，以允许 httpd 访问使用 thecifs_t 类型标记的 CIFS 卷上的文件，例如使用 Samba 挂载的文件系统。

httpd_use_nfs

启用此布尔值，以允许 httpd 访问使用 nfs_t 类型标记的 NFS 卷上的文件，例如使用 NFS 挂载的文件系统。



注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

13.4. 配置示例

以下示例展示了 SELinux 如何补充 Apache HTTP 服务器以及如何维护 Apache HTTP 服务器的完整功能。

13.4.1. 运行静态站点

要创建静态网站，请使用 httpd_sys_content_t 类型标记该网站的 .html 文件。默认情况下，Apache HTTP 服务器无法写入标记为 httpd_sys_content_t 类型的文件。以下示例为只读网站创建一个新目录来存储文件：

1. 以 root 用户身份使用 mkdir 实用程序创建顶级目录：

```
~]# mkdir /mywebsite
```

2. 以 root 用户身份，创建 /mywebsite/index.html 文件。将以下内容复制并粘贴到 /mywebsite/index.html 中：

```
<html>
<h2>index.html from /mywebsite/</h2>
</html>
```

3. 要允许 Apache HTTP 服务器只读访问 /mywebsite/ 以及它下的文件和子目录，请将 目录标记为 httpd_sys_content_t 类型。以 root 用户身份输入以下命令，将标签更改添加到 file-context 配置中：

```
~]# semanage fcontext -a -t httpd_sys_content_t "/mywebsite(/.*)?"
```

4. 以 root 用户身份使用 restorecon 工具进行标签更改：

```
~]# restorecon -R -v /mywebsite
restorecon reset /mywebsite context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /mywebsite/index.html context unconfined_u:object_r:default_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

5. 在本例中，以 root 用户身份编辑 /etc/httpd/conf/httpd.conf 文件。注释掉现有的 DocumentRoot 选项。添加 DocumentRoot "/mywebsite" 选项。编辑后，这些选项应如下所示：

```
#DocumentRoot "/var/www/html"
DocumentRoot "/mywebsite"
```

6. 以 root 身份输入以下命令，查看 Apache HTTP 服务器的状态。如果服务器停止，启动它：

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: inactive (dead)
```

```
~]# systemctl start httpd.service
```

如果服务器正在运行，请以 root 身份执行以下命令来重新启动服务（这也会应用对 httpd.conf 所做的任何更改）：

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: active (running) since Wed 2014-02-05 13:16:46 CET; 2s ago
```

```
~]# systemctl restart httpd.service
```

7. 使用 Web 浏览器导航到 此时会显示以下内容：

```
index.html from /mywebsite/
```

13.4.2. 共享 NFS 和 CIFS 卷

默认情况下，客户端上的 NFS 挂载使用 NFS 卷策略定义的默认上下文标记。在常见策略中，此默认上下文使用 nfs_t 类型。此外，默认情况下，挂载在客户端上的 Samba 共享使用策略定义的默认上下文标记。在常见策略中，此默认上下文使用 thecifs_t 类型。

根据策略配置，服务可能无法读取使用 nfs_t or cifs_t 类型标记的文件。这可能会导致标记为这些类型的文件系统被挂载，然后被其他服务读取或导出。可以启用或禁用布尔值，以控制哪些服务可以访问 nfs_t 和 cifs_t 类型。

启用 httpd_use_nfs 布尔值，以允许 httpd 访问和共享 NFS 卷（使用 nfs_t 类型标记）：

```
~]# setsebool -P httpd_use_nfs on
```

启用 httpd_use_cifs 布尔值，以允许 httpd 访问和共享 CIFS 卷（使用 cifs_t 类型标记）：

```
~]# setsebool -P httpd_use_cifs on
```

注意

如果您不希望 setsebool 更改在重新引导后保留，则不要使用 -P 选项。

13.4.3. 在服务间共享文件

类型强制可帮助阻止进程访问供其他进程使用的文件。例如，默认情况下，Samba 无法读取使用 httpd_sys_content_t 类型标记的文件，这些文件旨在供 Apache HTTP 服务器使用。如果所需的文件标有 public_content_t 或 public_content_ rw_t 类型，则可以在 Apache HTTP 服务器、FTP、rsync 和 Samba 之间共享文件。

以下示例创建了目录和文件，并允许通过 Apache HTTP 服务器、FTP、rsync 和 Samba 共享（只读）目录和文件：

1. **以 root 用户身份使用 mkdir 工具创建新的顶级目录，以在多个服务间共享文件：**

```
~]# mkdir /shares
```

2. **在 file-context 配置中与 模式不匹配的文件和目录可能会使用 default_t 类型进行标记。受限制的服务无法访问此类型：**

```
~]$ ls -dZ /shares
drwxr-xr-x  root root unconfined_u:object_r:default_t:s0 /shares
```

3. **以 root 用户身份，创建 /shares/index.html 文件。将以下内容复制并粘贴到 /shares/index.html 中：**

```
<html>
<body>
<p>Hello</p>
</body>
</html>
```

4. **使用 public_content_t 类型标记 /shares/ 允许 Apache HTTP 服务器、FTP、rsync 和 Samba 的只读访问。以 root 用户身份输入以下命令，将标签更改添加到 file-context 配置中：**

```
~]# semanage fcontext -a -t public_content_t "/shares(/.*)?"
```

5. 以 root 用户身份使用 restorecon 工具应用标签更改：

```
~]# restorecon -R -v /shares/
restorecon reset /shares context unconfined_u:object_r:default_t:s0-
>system_u:object_r:public_content_t:s0
restorecon reset /shares/index.html context unconfined_u:object_r:default_t:s0-
>system_u:object_r:public_content_t:s0
```

通过 Samba 共享 /shares/ ：

1. 确认已安装了 samba 、 samba-common 和 samba-client 软件包（版本号可能有所不同）：

```
~]$ rpm -q samba samba-common samba-client
samba-3.4.0-0.41.el6.3.i686
samba-common-3.4.0-0.41.el6.3.i686
samba-client-3.4.0-0.41.el6.3.i686
```

如果没有安装任何这些软件包，请以 root 用户身份运行以下命令安装它们：

```
~]# yum install package-name
```

2. 以 root 用户身份编辑 /etc/samba/smb.conf 文件。在该文件的底部添加以下条目，以通过 Samba 共享 /shares/ 目录：

```
[shares]
comment = Documents for Apache HTTP Server, FTP, rsync, and Samba
path = /shares
public = yes
writable = no
```

3. 需要 Samba 帐户来挂载 Samba 文件系统。以 root 身份输入以下命令来创建 Samba 帐户，其中 username 是现有 Linux 用户。例如， smbpasswd -a testuser 为 Linux testuser 用户创建一个 Samba 帐户：

```
~]# smbpasswd -a testuser
New SMB password: Enter a password
Retype new SMB password: Enter the same password again
Added user testuser.
```

如果您运行上述命令，指定系统上不存在的帐户的用户名，这会导致 Cannot locate Unix 帐户出现 'username'! 错误。

4. 启动 Samba 服务：

```
~]# systemctl start smb.service
```

5. 输入以下命令列出可用的共享，其中 username 是第 3 步中添加的 Samba 帐户。提示输入密码时，在第 3 步中输入分配给 Samba 帐户的密码（版本号可能有所不同）：

```
~]$ smbclient -U username -L localhost
Enter username's password:
Domain=[HOSTNAME] OS=[Unix] Server=[Samba 3.4.0-0.41.el6]

Sharename       Type      Comment
---------       ----      -------
shares          Disk      Documents for Apache HTTP Server, FTP, rsync, and Samba
IPC$            IPC       IPC Service (Samba Server Version 3.4.0-0.41.el6)
username        Disk      Home Directories
Domain=[HOSTNAME] OS=[Unix] Server=[Samba 3.4.0-0.41.el6]

Server               Comment
---------            -------

Workgroup            Master
---------            -------
```

6. 用户 mkdir 实用程序来创建新目录。该目录将用于挂载 共享 Samba 共享：

```
~]# mkdir /test/
```

7. 以 root 身份输入以下命令将 共享 Samba 共享 挂载到 /test/，将 username 替换为第 3 步中的用户名：

```
~]# mount //localhost/shares /test/ -o user=username
```

输入 username 的密码，这是在第 3 步中配置的。

8. 查看通过 Samba 共享的 文件的内容：

```
~]$ cat /test/index.html
```

```
<html>
<body>
<p>Hello</p>
</body>
</html>
```

通过 Apache HTTP 服务器共享 /shares/ ：

1. **确认已安装 httpd 软件包（版本号可能有所不同）：**

```
~]$ rpm -q httpd
httpd-2.2.11-6.i386
```

如果没有安装这个软件包，以 root 用户身份使用 yum 工具安装它：

```
~]# yum install httpd
```

2. **更改到 /var/www/html/ 目录。以 root 用户身份输入以下命令来创建 /shares/ 目录的链接（名为 共享）：**

```
html]# ln -s /shares/ shares
```

3. **启动 Apache HTTP 服务器：**

```
~]# systemctl start httpd.service
```

4. **使用 Web 浏览器导航到 显示 /shares/index.html 文件。**

默认情况下，Apache HTTP 服务器会读取 index.html 文件（如果存在）。如果 /shares/ 没有 index.html，且有 file1、file2 和 file3，则访问 http://localhost/shares 时会出现目录列表：

1. **删除 index.html 文件：**

```
~]# rm -i /shares/index.html
```

2. 以 root 用户身份使用 touch 实用程序在 /shares/ 中创建三个文件：

```
~]# touch /shares/file{1,2,3}
~]# ls -Z /shares/
-rw-r--r--  root root system_u:object_r:public_content_t:s0 file1
-rw-r--r--  root root unconfined_u:object_r:public_content_t:s0 file2
-rw-r--r--  root root unconfined_u:object_r:public_content_t:s0 file3
```

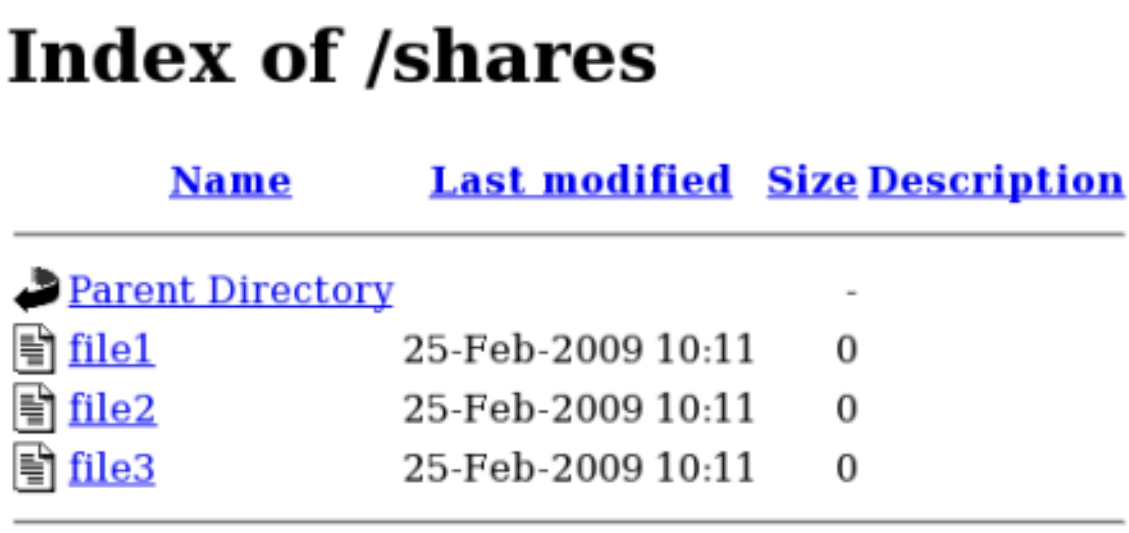
3. 以 root 用户身份输入以下命令查看 Apache HTTP 服务器的状态：

```
~]# systemctl status httpd.service
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
   Active: inactive (dead)
```

如果服务器停止，启动它：

```
~]# systemctl start httpd.service
```

4. 使用 Web 浏览器导航到 此时会显示目录列表：



13.4.4. 更改端口号

根据策略配置，服务只能在某些端口号中运行。尝试更改服务在没有更改策略的情况下运行的端口可能会导致服务无法启动。以 root 用户身份使用 semanage 工具列出 SELinux 允许 httpd 侦听的端口：

```
~]# semanage port -l | grep -w http_port_t
http_port_t                    tcp      80, 443, 488, 8008, 8009, 8443
```

默认情况下，SELinux 允许 httpd 侦听 TCP 端口 80、443、488、8008、8009 或 8443。如果配置了 /etc/httpd/conf/httpd.conf，以便 httpd 侦听未列出的 http_port_t 端口，httpd无法启动。

将 httpd 配置为在 TCP 端口 80、443、488、8008、8009 或 8443 以外的端口中运行：

1. 以 root 用户身份编辑 /etc/httpd/conf/httpd.conf 文件，以便 Listen 选项列出 httpd 的 SELinux 策略中未配置的端口。以下示例将 httpd 配置为侦听 10.0.0.1 IP 地址和 TCP 端口 12345：

```
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses (0.0.0.0)
#
#Listen 12.34.56.78:80
Listen 10.0.0.1:12345
```

2. 以 root 用户身份输入以下命令，将端口添加到 SELinux 策略配置中：

```
~]# semanage port -a -t http_port_t -p tcp 12345
```

3. 确认添加了端口：

```
~]# semanage port -l | grep -w http_port_t
http_port_t                    tcp      12345, 80, 443, 488, 8008, 8009, 8443
```

如果您不再在端口 12345 上运行 httpd，请以 root 用户身份使用 semanage 工具从策略配置中删除端口：

```
~]# semanage port -d -t http_port_t -p tcp 12345
```

---

[14] 如需更多信息，请参阅《系统管理员指南》中的名为 Apache HTTP Sever 的章节。

第 14 章 SAMBA

Samba 是服务器消息块(SMB)和通用 Internet 文件系统(CIFS)协议的开源实施，跨各种操作系统在客户端之间提供文件和打印服务。[15]

在 Red Hat Enterprise Linux 中，samba 软件包提供 Samba 服务器。输入以下命令查看是否安装了 samba 软件包：

```
~]$ rpm -q samba
package samba is not installed
```

如果没有安装，且您想要使用 Samba，请以 root 用户身份使用 yum 工具来安装它：

```
~]# yum install samba
```

14.1. SAMBA 和 SELINUX

启用 SELinux 后，Samba 服务器(smbd)默认运行限制。受限制的服务在自己的域中运行，与其他受限服务分离。以下示例演示了在自己的域中运行的 smbd 进程。本例假设安装了 samba 软件包：

1. 运行 getenforce 命令确认 SELinux 是否以强制模式运行：

   ```
   ~]$ getenforce
   Enforcing
   ```

   当 SELinux 以强制模式运行时，该命令将返回 强制。

2. 以 root 用户身份输入以下命令启动 smbd ：

   ```
   ~]# systemctl start smb.service
   ```

   确认 服务正在运行。输出中应包括以下信息（只有时间戳有所不同）：

   ```
   ~]# systemctl status smb.service
   smb.service - Samba SMB Daemon
     Loaded: loaded (/usr/lib/systemd/system/smb.service; disabled)
     Active: active (running) since Mon 2013-08-05 12:17:26 CEST; 2h 22min ago
   ```

-

3. 要查看 smbd 进程，请执行以下命令：

```
~]$ ps -eZ | grep smb
system_u:system_r:smbd_t:s0      9653 ?        00:00:00 smbd
system_u:system_r:smbd_t:s0      9654?         00:00:00 smbd
```

与 smbd 进程关联的 SELinux 上下文为 system_u:system_r:smbd_t:s0。上下文的第二部分 smbd_t 是类型。类型定义进程的域以及文件的类型。在本例中，smb d 进程 在 smbd_t 域中运行。

文件必须正确标记，以允许 smbd 访问和共享这些文件。例如，smbd 可以读取和写入标有 samba_share_t 类型的文件，但默认情况下，无法访问使用 Apache HTTP 服务器使用 httpd_sys_content_t 类型的文件。必须启用布尔值以允许某些行为，例如允许通过 Samba 导出主目录和 NFS 卷，以及允许 Samba 充当域控制器。

14.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

使用 samba_share_t 类型标记文件，以允许 Samba 共享它们。仅标记您创建的文件，且不要使用 samba_share_t 类型重新标记系统文件：可以启用布尔值以共享此类文件和目录。只要相应地设置了 /etc/samba/smb.conf 文件，SELinux 允许 Samba 写入使用 samba_share_t 类型标记的文件。

samba_etc_t 类型用于 /etc/samba/ 目录中的某些文件，如 smb.conf。不要使用 samba_etc_t 类型手动标记文件。如果此目录中的文件没有正确标记，请以 root 用户身份输入 restorecon -R -v /etc/samba 命令，将这些文件恢复到其默认上下文。如果 /etc/samba/smb.conf 没有使用 samba_etc_t 类型标记，启动 Samba 服务可能会失败，并且可能会记录 SELinux 拒绝消息。以下是 /etc/samba/smb.conf 使用 httpd_sys_content_t 类型标记时的拒绝信息示例：

```
setroubleshoot: SELinux is preventing smbd (smbd_t) "read" to ./smb.conf (httpd_sys_content_t). For
complete SELinux messages. run sealert -l deb33473-1069-482b-bb50-e4cd05ab18af
```

14.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

smbd_anon_write

启用此布尔值后，smb d 便可写入公共目录，例如为其他情况下没有特殊访问限制的常用文件保留的区域。

samba_create_home_dirs

启用此布尔值后，Samba 便可独立创建新的主目录。这通常通过 PAM 等机制执行。

samba_domain_controller

启用后，此布尔值允许 Samba 充当域控制器，并授予其执行 useradd、groupadd 和 passwd 等相关命令的权限。

samba_enable_home_dirs

启用此布尔值可让 Samba 共享用户的主目录。

samba_export_all_ro

导出任何文件或目录，允许只读权限。这样，未使用 samba_share_t 类型标记的文件和目录可以通过 Samba 共享。当启用了 samba_export_all_ro 布尔值，但 samba_export_all_rw 布尔值被禁用时，对 Samba 共享的写入访问将被拒绝，即使在 /etc/samba/smb.conf 中配置了写入访问权限，也允许写入访问权限。

samba_export_all_rw

导出任何文件或目录，允许读取和写入权限。这允许通过 Samba 导出没有使用 samba_share_t 类型标记的文件和目录。必须将 /etc/samba/smb.conf 中的权限和 Linux 权限配置为允许写入访问。

samba_run_unconfined

启用此布尔值后，Samba 就可以运行 /var/lib/samba/scripts/ 目录中未限制的脚本。

samba_share_fusefs

必须启用此布尔值，以便 Samba 共享 fusefs 文件系统。

samba_share_nfs

禁用此布尔值可防止 smbd 通过 Samba 对 NFS 共享具有完全访问权限。启用此布尔值将允许 Samba 共享 NFS 卷。

use_samba_home_dirs

启用此布尔值，将远程服务器用于 Samba 主目录。

virt_use_samba

允许虚拟机访问 CIFS 文件.

注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

14.4. 配置示例

以下示例提供了 SELinux 如何补充 Samba 服务器以及如何维护 Samba 服务器功能的实际演示。

14.4.1. 共享您创建的目录

以下示例创建新目录，并通过 Samba 共享该目录：

1. 确认已安装 samba 、 samba-common 和 samba-client 软件包：

```
~]$ rpm -q samba samba-common samba-client
package samba is not installed
package samba-common is not installed
package samba-client is not installed
```

如果没有安装这些软件包，以 root 用户身份使用 yum 工具安装它们：

```
~]# yum install package-name
```

2. 以 root 用户身份使用 mkdir 实用程序创建一个新的顶级目录，以通过 Samba 共享文件：

```
~]# mkdir /myshare
```

3. 使用 touch 实用程序 root 创建空文件。此文件稍后用于验证 Samba 共享正确挂载：

```
~]# touch /myshare/file1
```

4. 只要相应地设置了 /etc/samba/smb.conf 文件，SELinux 允许 Samba 读取和写入标记有 samba_share_t 类型的文件。以 root 用户身份输入以下命令，将标签更改添加到 file-context 配置中：

```
~]# semanage fcontext -a -t samba_share_t "/myshare(/.*)?"
```

5. 以 root 用户身份使用 restorecon 工具应用标签更改：

```
~]# restorecon -R -v /myshare
restorecon reset /myshare context unconfined_u:object_r:default_t:s0-
>system_u:object_r:samba_share_t:s0
restorecon reset /myshare/file1 context unconfined_u:object_r:default_t:s0-
>system_u:object_r:samba_share_t:s0
```

6. 以 root 用户身份编辑 /etc/samba/smb.conf。在该文件的底部添加以下内容，以通过 Samba 共享 /myshare/ 目录：

```
[myshare]
comment = My share
path = /myshare
public = yes
writable = no
```

7. 需要 Samba 帐户来挂载 Samba 文件系统。以 root 身份输入以下命令来创建 Samba 帐户，其中 username 是现有 Linux 用户。例如，smbpasswd -a testuser 为 Linux testuser 用户创建一个 Samba 帐户：

```
~]# smbpasswd -a testuser
New SMB password: Enter a password
Retype new SMB password: Enter the same password again
Added user testuser.
```

如果您输入以上命令并指定系统中不存在的帐户的用户名，这会导致 Cannot locate Unix 帐户出现 'username'! 错误。

8. 启动 **Samba 服务：**

```
~]# systemctl start smb.service
```

9. **输入以下命令列出可用的共享，其中 username 是第 7 步中添加的 Samba 帐户。提示输入密码时，在第 7 步中输入分配给 Samba 帐户的密码（版本号可能有所不同）：**

```
~]$ smbclient -U username -L localhost
Enter username's password:
Domain=[HOSTNAME] OS=[Unix] Server=[Samba 3.4.0-0.41.el6]

Sharename       Type      Comment
---------       ----      -------
myshare         Disk      My share
IPC$            IPC       IPC Service (Samba Server Version 3.4.0-0.41.el6)
username        Disk      Home Directories
Domain=[HOSTNAME] OS=[Unix] Server=[Samba 3.4.0-0.41.el6]

Server               Comment
---------            -------

Workgroup            Master
---------            -------
```

10. **以 root 用户身份使用 mkdir 实用程序来创建新目录。该目录将用于挂载 myshare Samba 共享：**

```
~]# mkdir /test/
```

11. **以 root 身份输入以下命令将 myshare Samba 共享挂载到 /test/，将 username 替换为第 7 步中的用户名：**

```
~]# mount //localhost/myshare /test/ -o user=username
```

输入 username 的密码，该密码在第 7 步中配置。

12. 输入以下命令查看在第 3 步中创建的 file1 文件：

```
~]$ ls /test/
file1
```

14.4.2. 共享网站

可能无法使用 samba_share_t 类型标记文件，例如，当想要在 /var/www/html/ 目录中共享网站时。对于这样的情形，请使用 samba_export_all_ro 布尔值共享任何文件或目录（不考虑当前标签）、允许只读权限，或者使用 samba_export_all_rw 布尔值共享任何文件或目录（无当前标签除外），允许读取和写入权限。

以下示例在 /var/www/html/ 中为网站创建一个文件，然后通过 Samba 共享该文件，允许读取和写入权限。本例假定安装了 httpd、samba、samba-common、samba-client 和 wget 软件包：

1. 以 root 用户身份，创建 /var/www/html/file1.html 文件。将以下内容复制并粘贴到该文件中：

```
<html>
<h2>File being shared through the Apache HTTP Server and Samba.</h2>
</html>
```

2. 输入以下命令查看 file1.html 的 SELinux 上下文：

```
~]$ ls -Z /var/www/html/file1.html
-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/file1.html
```

该文件标有 httpd_sys_content_t 标签。默认情况下，Apache HTTP 服务器可以访问此类型，但 Samba 不能。

3. 启动 Apache HTTP 服务器：

```
~]# systemctl start httpd.service
```

4. 更改到用户具有写入权限的目录，并输入以下命令。除非对默认配置进行了更改，否则这个

命令会成功：

```
~]$ wget http://localhost/file1.html
Resolving localhost... 127.0.0.1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 84 [text/html]
Saving to: `file1.html.1'

100%[======================>] 84          --.-K/s   in 0s

`file1.html.1' saved [84/84]
```

5. 以 root 用户身份编辑 /etc/samba/smb.conf。在该文件的底部添加以下内容，通过 Samba 共享 /var/www/html/ 目录：

```
[website]
comment = Sharing a website
path = /var/www/html/
public = no
writable = no
```

6. /var/www/html/ 目录标有 httpd_sys_content_t 类型。默认情况下，Samba 无法访问使用此类型标记的文件和目录，即使 Linux 权限允许这样做。要允许 Samba 访问，请启用 samba_export_all_ro 布尔值：

```
~]# setsebool -P samba_export_all_ro on
```

如果您不希望更改在重新引导后保留，则不要使用 -P 选项。请注意，启用 samba_export_all_ro 布尔值可允许 Samba 访问任何类型。

7. 启动 Samba 服务：

```
~]# systemctl start smb.service
```

---

[15] 如需更多信息，请参阅《系统管理员指南》中的 Samba 部分。

第 15 章 文件传输协议

文件传输协议(FTP)是当今 Internet 上最旧且最常用的协议之一。其用途是在网络上的计算机主机之间可靠地传输文件，而无需用户直接登录远程主机或了解如何使用远程系统。它允许用户使用一组标准简单的命令来访问远程系统上的文件。

高安全性 FTP 后台程序(vsftpd)设计为快速、稳定且最重要的是安全。它能够高效且安全地处理大量连接，因此 vsftpd 是唯一随 Red Hat Enterprise Linux 分发的独立 FTP 的原因。

在 Red Hat Enterprise Linux 中， vsftpd 软件包提供 高安全性 FTP 后台程序。输入以下命令查看是否安装了 vsftpd:

```
~]$ rpm -q vsftpd
package vsftpd is not installed
```

如果您希望 FTP 服务器和 vsftpd 软件包没有安装，以 root 用户身份使用 yum 工具安装它：

```
~]# yum install vsftpd
```

15.1. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

默认情况下，匿名用户在使用 FTP 登录时，对 /var/ftp/ 目录中的文件具有读取访问权限。此目录标有 public_content_t 类型，仅允许读取访问权限，即使 /etc/vsftpd/vsftpd.conf 中配置了写入访问权限。public_content_t 类型可供其他服务访问，如 Apache HTTP 服务器、Samba 和 NFS。

使用以下类型之一通过 FTP 共享文件：

public_content_t

使用 public_content_t 类型标记您创建的文件和目录，以便通过 vsftpd 以只读方式共享它们。其他服务（如 Apache HTTP 服务器、Samba 和 NFS）也有权访问使用此类型标记的文件。标签有 public_content_t 类型的文件无法写入，即使 Linux 权限允许写入访问。如果需要写入访问权限，请使用 public_content_rw_t 类型。

public_content_rw_t

使用 public_content_rw_t 类型标记您创建的文件和目录，以通过 vsftpd 使用读写权限共享它们。其他服务（如 Apache HTTP 服务器、Samba 和 NFS）也有权访问使用此类型标记的文件。请记住，必须为每个服务启用布尔值，然后才能写入标有此类型的文件。

15.2. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

ftpd_anon_write

禁用后，此布尔值可防止 vsftpd 写入标有 public_content_rw_t 类型的文件和目录。启用此布尔值，以允许用户使用 FTP 上传文件。将文件上传到 的目录必须使用 public_content_rw_t 类型进行标记，并且必须相应地设置 Linux 权限。

ftpd_full_access

启用此布尔值后，只有 Linux(DAC)权限用于控制访问权限，经过身份验证的用户也可以读取和写入没有使用 public_content_t 或 public_content_ rw_t 类型标记的文件。

ftpd_use_cifs

启用此布尔值后，vsftpd 可以访问标有 thecifs_t 类型的文件和目录；因此，启用此布尔值后，您可以共享通过 vsftpd 使用 Samba 挂载的文件系统。

ftpd_use_nfs

启用此布尔值后，vsftpd 可以访问标有 nfs_t 类型的文件和目录；因此，这个布尔值允许您共享通过 vsftpd 使用 NFS 挂载的文件系统。

ftpd_connect_db

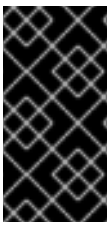
允许 FTP 后台程序启动与数据库的连接.

httpd_enable_ftp_server

允许 httpd 守护进程侦听 FTP 端口并充当 FTP 服务器。

tftp_anon_write

启用此布尔值后，TFTP 可以访问公共目录，例如为其他情况下没有特殊访问限制的常用文件保留区域。



重要

Red Hat Enterprise Linux 7.7 不提供 ftp_home_dir 布尔值。如需更多信息，请参阅 Red Hat Enterprise Linux 7.3 发行注记 文档。

注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

第 16 章 网络文件系统

网络文件系统(NFS)允许远程主机通过网络挂载文件系统，并像在本地挂载这些文件系统一样与这些文件系统交互。这使系统管理员能够将资源整合到网络上的集中式服务器上。[16]

在 Red Hat Enterprise Linux 中，需要 nfs-utils 软件包才能获得完整的 NFS 支持。输入以下命令查看是否安装了 nfs-utils ：

```
~]$ rpm -q nfs-utils
package nfs-utils is not installed
```

如果没有安装，且您想要使用 NFS，使用 yum 工具作为 root 来安装它：

```
~]# yum install nfs-utils
```

16.1. NFS 和 SELINUX

运行 SELinux 时，默认情况下 NFS 后台程序会受到限制，但 nfsd 进程除外，该进程使用 unconfined kernel_t 域类型进行标记。默认情况下，SELinux 策略允许 NFS 共享文件。此外，还支持在客户端和服务器之间传递 SELinux 标签，这可以为访问 NFS 卷的受限制域提供更好的安全控制。例如，在 NFS 卷上设置主目录时，可以指定只能访问主目录而非卷中其他目录的受限制域。同样，安全虚拟化等应用可以在 NFS 卷上设置镜像文件的标签，从而提高虚拟机的隔离级别。

默认禁用对标记 NFS 的支持。要启用它，请参阅 第 16.4.1 节 “启用 SELinux 标记的 NFS 支持”。

16.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

默认情况下，客户端中挂载的 NFS 卷使用 NFS 策略定义的默认上下文标记。在常见策略中，此默认上下文使用 nfs_t 类型。root 用户可以使用 mount -context 选项覆盖默认类型。以下类型用于 NFS：不同的类型允许您配置灵活的访问：

var_lib_nfs_t

此类型用于复制到 /var/lib/nfs/ 目录中的现有文件和新文件。在正常操作中不需要更改此类型。

要恢复对默认设置的更改，以 root 用户身份运行 restorecon -R -v /var/lib/nfs 命令。

nfsd_exec_t

/usr/sbin/rpc.nfsd 文件标有 nfsd_exec_t 标签，以及其他系统可执行文件和与 NFS 相关的库也是如此。用户不应为此类型标记任何文件。nfs d_exec_t 将过渡到 nfsd_t。

16.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

ftpd_use_nfs

启用后，此布尔值允许 ftpd 守护进程访问 NFS 卷。

cobbler_use_nfs

启用后，此布尔值允许 cobblerd 守护进程访问 NFS 卷。

git_system_use_nfs

启用后，此布尔值允许 Git 系统守护进程读取 NFS 卷上的系统共享存储库。

httpd_use_nfs

启用后，此布尔值允许 httpd 守护进程访问 NFS 卷中存储的文件。

samba_share_nfs

启用后，此布尔值允许 smbd 守护进程共享 NFS 卷。禁用后，此布尔值可防止 smbd 使用 Samba 对 NFS 共享具有完全访问权限。

sanlock_use_nfs

启用后，这个布尔值允许 sanlock 守护进程管理 NFS 卷。

sge_use_nfs

启用后，此布尔值允许 sge 调度程序访问 NFS 卷。

use_nfs_home_dirs

启用后，这个布尔值添加了对 NFS 主目录的支持。

virt_use_nfs

启用后，此布尔值允许自信虚拟客户机管理 NFS 卷上的文件。

xen_use_nfs

启用后，此布尔值允许 Xen 管理 NFS 卷上的文件。

git_cgi_use_nfs

启用后，此布尔值允许 Git 通用网关接口(CGI)访问 NFS 卷。

注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

16.4. 配置示例

16.4.1. 启用 SELinux 标记的 NFS 支持

以下示例演示了如何启用 SELinux 标记的 NFS 支持。本例假定已安装了 nfs-utils 软件包，使用了 SELinux targeted 策略，并且 SELinux 处于强制模式。

注意

应在 NFS 服务器 nfs-srv 上执行第 1-3 步。

1. 如果 NFS 服务器正在运行，请停止它：

```
[nfs-srv]# systemctl stop nfs
```

确认服务器已停止：

```
[nfs-srv]# systemctl status nfs
nfs-server.service - NFS Server
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; disabled)
   Active: inactive (dead)
```

2. 编辑 /etc/sysconfig/nfs 文件，将 RPCNFSDARGS 标志设置为 "-V 4.2:

```
# Optional arguments passed to rpc.nfsd. See rpc.nfsd(8)
RPCNFSDARGS="-V 4.2"
```

3. 再次启动服务器，并确认它正在运行。输出将包含以下信息，只有时间戳会有所不同：

```
[nfs-srv]# systemctl start nfs
```

```
[nfs-srv]# systemctl status nfs
nfs-server.service - NFS Server
   Loaded: loaded (/usr/lib/systemd/system/nfs-server.service; disabled)
   Active: active (exited) since Wed 2013-08-28 14:07:11 CEST; 4s ago
```

4. 在客户端中挂载 NFS 服务器：

```
[nfs-client]# mount -o v4.2 server:mntpoint localmountpoint
```

5. 现在，所有 SELinux 标签都从服务器成功传递给客户端：

```
[nfs-srv]$ ls -Z file
-rw-rw-r--. user user unconfined_u:object_r:svirt_image_t:s0 file
[nfs-client]$ ls -Z file
```

```
-rw-rw-r--. user user unconfined_u:object_r:svirt_image_t:s0 file
```

注意

如果您启用对主目录或其他内容的带标记 NFS 支持，则内容将被标记为 EXT 文件系统上的内容相同。另请注意，挂载具有不同 NFS 版本或尝试挂载不支持标记 NFS 的服务器可能会导致返回错误。

---

[16] 如需更多信息，请参阅《存储管理指南》中的 网络文件系统(NFS) 章节。

第 17 章 BERKELEY INTERNET 名称域

BIND 使用 指定的 守护进程执行名称解析服务。BIND 允许用户按名称而非数字地址查找计算机资源和服务。

在 Red Hat Enterprise Linux 中， bind 软件包提供 DNS 服务器。输入以下命令查看是否安装了 bind 软件包：

```
~]$ rpm -q bind
package bind is not installed
```

如果没有安装， 请以 root 用户身份使用 yum 工具安装它：

```
~]# yum install bind
```

17.1. BIND 和 SELINUX

/var/named/slaves/、/ var/named/dynamic/ 和 /var/named/data/ 目录的默认权限允许使用区域传送和动态 DNS 更新来更新区域文件。/var/named/ 中的文件使用 named_zone_t 类型标记，该类型用于主区域文件。

对于从属服务器，将 /etc/named.conf 文件配置为将从属区域放置在 /var/named/slaves/ 中。以下是在 /var/named/slaves/ 中存储 testdomain.com 的区域文件的从属 DNS 服务器中的域条目示例：

```
zone "testdomain.com" {
   type slave;
   masters { IP-address; };
   file "/var/named/slaves/db.testdomain.com";
        };
```

如果区域文件标记为 named_zone_t，则必须启用 named_write_master_zones 布尔值，以允许区域传送和动态 DNS 更新区域文件。此外，必须更改父目录的模式，以允许 指定用户 或组具有读取、写入和执行访问权限。

如果 /var/named/ 中的区域文件标有 named_cache_t 类型，则文件系统重新标记或运行 restorecon -R /var/ 将将其类型更改为 named_zone_t。

17.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

以下类型与 BIND 一起使用：不同的类型允许您配置灵活的访问：

named_zone_t

用于主区域文件.其他服务无法修改此类型的文件。只有启用了 named _write_master_zones 布尔值时，指定守护进程才能修改此类型的文件。

named_cache_t

默认情况下，named 可以写入使用此类型标记的文件，而不设置额外的布尔值。在 /var/named/slaves/、/var/ named/dynamic/ 和 /var/ named/data/ 目录中 复制或创建的文件会自动标记为 named_cache_t 类型。

named_var_run_t

在 /var/run/bind/、/ var/ run/named/ 和 /var/ run/unbound/ 目录中 复制或创建的文件会自动标记为 named_var_run_t 类型。

named_conf_t

与 BIND 相关的配置文件通常存储在 /etc 目录中，使用 named_conf_t 类型自动标记。

named_exec_t

与 BIND 相关的可执行文件通常存储在 /usr/sbin/ 目录中，使用 named_exec_t 类型自动标记。

named_log_t

与 BIND 相关的日志文件通常存储在 /var/log/ 目录中，使用 named_log_t 类型自动标记。

named_unit_file_t

/usr/lib/systemd/system/ 目录中的可执行 BIND 相关文件会自动使用 named_unit_file_t 类型进行标记。

17.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

named_write_master_zones

禁用后，此布尔值可防止 named 写入标记为 named_zone_t 类型的区域文件或目录。守护进程通常不需要写入区域文件；但是，如果需要，或者次要服务器需要写入区域文件，请启用此布尔值以允许执行此操作。

named_tcp_bind_http_port

启用后，此布尔值允许 BIND 绑定 Apache 端口。

注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

17.4. 配置示例

17.4.1. 动态 DNS

BIND 允许主机动态更新 DNS 和区域文件中的记录。这在主机计算机的 IP 地址频繁更改并且 DNS 记录需要实时修改时使用。

对于您要通过动态 DNS 更新的区域文件，使用 /var/named/dynamic/ 目录。在 中创建或复制到此目录中创建的文件将继承 Linux 权限，允许 named 写入它们。由于这些文件使用 named_cache_t 类型标记，SELinux 允许 named 向它们写入。

如果 /var/named/dynamic/ 中的区域文件标有 named_zone_t 类型，则动态 DNS 更新可能在特定时间段内不成功，因为更新需要在合并之前首先写入到日志中。如果在日志尝试合并时使用 named_zone_t 类型标记区域文件，则会记录如下错误：

```
named[PID]: dumping master file: rename: /var/named/dynamic/zone-name: permission denied
```

另外，还记录了以下 SELinux 拒绝信息：

```
setroubleshoot: SELinux is preventing named (named_t) "unlink" to zone-name (named_zone_t)
```

要解决这个问题，以 root 用户身份使用 restorecon 工具：

```
~]# restorecon -R -v /var/named/dynamic
```

第 18 章 并发版本系统

Concurrent Versioning System(CVS)是一个免费的修订版本控制系统。它用于监控和跟踪对通常由多个不同用户访问的一组中央文件进行的修改。程序员通常使用它来管理源代码存储库，并广泛供开源开发人员使用。

在红帽企业 Linux 中，c vs 软件包提供了 CVS。输入以下命令查看是否安装了 cvs 软件包：

```
~]$ rpm -q cvs
package cvs is not installed
```

如果没有安装，且您想要使用 CVS，请以 root 用户身份使用 yum 工具来安装它：

```
~]# yum install cvs
```

18.1. CVS 和 SELINUX

cvs 守护进程使用 cvs_t 类型运行。默认情况下，在 Red Hat Enterprise Linux 中，CVS 只允许读取和写入某些目录。标签 cvs_data_t 定义 cvs 对哪些区域具有读取和写入访问权限。将 CVS 与 SELinux 搭配使用时，必须分配正确的标签，以便客户端能够完全访问为 CVS 数据保留的区域。

18.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

以下类型与 CVS 一起使用：不同的类型允许您配置灵活的访问：

cvs_data_t

此类型用于 CVS 存储库中的数据。CVS 只有当数据具有此类型时，才可获得对数据的完整访问权限。

cvs_exec_t

此类型用于 /usr/bin/cvs 二进制文件。

18.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

cvs_read_shadow

此布尔值允许 cvs 守护进程访问 /etc/shadow 文件以进行用户身份验证。

注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

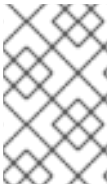
请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

18.4. 配置示例

18.4.1. 设置 CVS

这个示例描述了一个简单的 CVS 设置以及允许远程访问的 SELinux 配置。本例中使用了两个主机：主机名为 cvs-srv 的 CVS 服务器，IP 地址为 192.168.1.1，主机名为 cvs-client 的客户端，以及 IP 地址 192.168.1.100。两个主机在同一子网(192.168.1.0/24)中。这只是一个示例，假设已安装了 cvs 和 xinetd 软件包，使用了 SELinux targeted 策略，并且 SELinux 正以强制模式运行。

此示例将显示，即使具有完全 DAC 权限，SELinux 仍然可以根据文件标签强制执行策略规则，并且只允许访问已明确由 CVS 标识访问的特定区域。



注意

应在 CVS 服务器上执行第 1-9 步，c vs-srv.

1. 这个示例需要 **cvs** 和 **xinetd** 软件包。确认已安装了软件包：

```
[cvs-srv]$ rpm -q cvs xinetd
package cvs is not installed
package xinetd is not installed
```

如果没有安装，以 **root** 用户身份使用 **yum** 工具安装它：

```
[cvs-srv]# yum install cvs xinetd
```

2. 以 **root** 用户身份输入以下命令，创建一个名为 **CVS** 的组：

```
[cvs-srv]# groupadd CVS
```

这也可通过使用 **system-config-users** 实用程序来完成。

3. 创建一个用户名为 **cvsuser** 的用户，并使该用户成为 **CVS** 组的成员。这可以通过 **system-config-users** 完成。

4. 编辑 **/etc/services** 文件，并确保 **CVS** 服务器具有类似如下的未注释条目：

```
cvspserver 2401/tcp   # CVS client/server operations
cvspserver 2401/udp   # CVS client/server operations
```

5. 在文件系统的根区域创建 **CVS** 存储库。在使用 **SELinux** 时，最好在 **root** 文件系统中包含存储库，以便为它指定递归标签，而不影响任何其他子目录。例如，以 **root** 用户身份创建一个 **/cvs/** 目录来存放存储库：

```
[root@cvs-srv]# mkdir /cvs
```

6. 为所有用户授予 **/cvs/** 目录的完整权限：

```
[root@cvs-srv]# chmod -R 777 /cvs
```

警告

这只是一个示例，不应在生产系统中使用这些权限。

7. 编辑 /etc/xinetd.d/cvs 文件，并确保 CVS 部分未注释并配置为使用 /cvs/ 目录。该文件应类似于：

```
service cvspserver
{
 disable = no
 port   = 2401
 socket_type  = stream
 protocol  = tcp
 wait   = no
 user   = root
 passenv   = PATH
 server   = /usr/bin/cvs
 env   = HOME=/cvs
 server_args  = -f --allow-root=/cvs pserver
# bind   = 127.0.0.1
```

8. 启动 **xinetd** 守护进程：

```
[cvs-srv]# systemctl start xinetd.service
```

9. 添加一条规则，允许使用 **system-config-firewall** 实用程序通过端口 **2401** 上的 **TCP** 进行入站连接。

10. 在客户端中，以 **cvsuser** 用户身份输入以下命令：

```
[cvsuser@cvs-client]$ cvs -d /cvs init
```

11. 此时，**CVS** 已配置好，但 **SELinux** 仍会拒绝登录和文件访问。要演示这一点，请在 **cvs-client** 上设置 **$CVSROOT** 变量并尝试远程登录。以下步骤应该在 **cvs-client** 上执行：

```
[cvsuser@cvs-client]$ export CVSROOT=:pserver:cvsuser@192.168.1.1:/cvs
[cvsuser@cvs-client]$
```

```
[cvsuser@cvs-client]$ cvs login
Logging in to :pserver:cvsuser@192.168.1.1:2401/cvs
CVS password: ********
cvs [login aborted]: unrecognized auth response from 192.168.100.1: cvs pserver: cannot
open /cvs/CVSROOT/config: Permission denied
```

SELinux 已阻止访问。为了让 SELinux 允许此访问，应该在 cvs-srv 上执行以下步骤：

12. **更改 /cvs/ 目录的上下文，以便以递归方式标记 /cvs/ 目录中的任何现有和新数据，为其提供 cvs_data_t 类型：**

```
[root@cvs-srv]# semanage fcontext -a -t cvs_data_t '/cvs(/.*)?'
[root@cvs-srv]# restorecon -R -v /cvs
```

13. **客户端 cvs-client 现在能够登录并访问这个仓库中的所有 CVS 资源：**

```
[cvsuser@cvs-client]$ export CVSROOT=:pserver:cvsuser@192.168.1.1:/cvs
[cvsuser@cvs-client]$
[cvsuser@cvs-client]$ cvs login
Logging in to :pserver:cvsuser@192.168.1.1:2401/cvs
CVS password: ********
[cvsuser@cvs-client]$
```

第 19 章 SQUID 缓存代理

Squid 是 Web 客户端的高性能代理缓存服务器，支持 FTP、Gopher 和 HTTP 数据对象。它通过缓存和重复使用频繁请求的网页来缩短带宽并缩短响应时间。[17]

在 Red Hat Enterprise Linux 中，s quid 软件包提供了 Squid 缓存代理。输入以下命令查看是否安装了 squid 软件包：

```
~]$ rpm -q squid
package squid is not installed
```

如果没有安装，且您想要使用 squid，请以 root 用户身份使用 yum 工具来安装它：

```
~]# yum install squid
```

19.1. SQUID 缓存代理和 SELINUX

启用 SELinux 后，Squid 默认会受到限制。受限制的进程在其自己的域中运行，并且与其他受限制的进程隔开。如果一个受攻击者限制的进程受到 SELinux 策略配置的影响，攻击者对资源的访问权限和可能受到的破坏会受到限制。以下示例演示了在自己的域中运行的 Squid 进程。本例假设安装了 squid 软件包：

1. 运行 getenforce 命令确认 SELinux 是否以强制模式运行：

   ```
   ~]$ getenforce
   Enforcing
   ```

   当 SELinux 以强制模式运行时，该命令将返回 强制。

2. 以 root 用户身份输入以下命令启动 squid 守护进程：

   ```
   ~]# systemctl start squid.service
   ```

   确认 服务正在运行。输出中应包括以下信息（只有时间戳有所不同）：

   ```
   ~]# systemctl status squid.service
   squid.service - Squid caching proxy
   ```

```
   Loaded: loaded (/usr/lib/systemd/system/squid.service; disabled)
   Active: active (running) since Mon 2013-08-05 14:45:53 CEST; 2s ago
```

3. 输入以下命令查看 **squid** 进程 ：

```
~]$ ps -eZ | grep squid
system_u:system_r:squid_t:s0    27018 ?        00:00:00 squid
system_u:system_r:squid_t:s0    27020 ?        00:00:00 log_file_daemon
```

与 **squid** 进程关联的 **SELinux** 上下文为 **system_u:system_r:squid_t:s0**。上下文的第二部分 **squid_t** 是类型。类型定义进程的域以及文件的类型。在本例中，**Squid** 进程在 **squid_t** 域中运行。

SELinux 策略定义了在受限域中运行进程的方式，如 **squid_t**，它们通常与文件、其他进程和系统交互。文件必须正确标记，以允许对文件进行 **squid** 访问。

当 **/etc/squid/squid.conf** 文件被配置为 **squid** 侦听默认 **TCP** 端口 **3128**、**3401** 或 **4827** 以外的端口时，必须使用 **semanage port** 命令将所需的端口号添加到 **SELinux** 策略配置中。以下示例演示了将 **squid** 配置为侦听 **SELinux** 策略配置中最初未为其定义的端口，因此，服务器无法启动。本例还演示如何配置 **SELinux** 系统，以允许 守护进程成功侦听策略中尚未定义的非标准端口。本例假定已安装了 **squid** 软件包。以 **root** 用户身份运行示例中的每个命令：

1. 确认 **squid** 守护进程没有运行：

```
~]# systemctl status squid.service
squid.service - Squid caching proxy
   Loaded: loaded (/usr/lib/systemd/system/squid.service; disabled)
   Active: inactive (dead)
```

如果输出不同，请停止该进程：

```
~]# systemctl stop squid.service
```

2. 输入以下命令查看 **SELinux** 允许 **squid** 侦听的端口：

```
~]# semanage port -l | grep -w -i squid_port_t
squid_port_t                   tcp      3401, 4827
squid_port_t                   udp      3401, 4827
```

3. 以 root 用户身份编辑 /etc/squid/squid.conf。配置 http_port 选项，使其列出 squid 的 SELinux 策略配置中未配置的端口。在这个示例中，守护进程被配置为侦听端口 10000:

```
# Squid normally listens to port 3128
http_port 10000
```

4. 运行 setsebool 命令，确保 squid_connect_any 布尔值设置为 off。这样可保证 squid 只允许在特定端口上操作：

```
~]# setsebool -P squid_connect_any 0
```

5. 启动 squid 守护进程：

```
~]# systemctl start squid.service
Job for squid.service failed. See 'systemctl status squid.service' and 'journalctl -xn' for details.
```

记录类似如下的 SELinux 拒绝信息：

```
localhost setroubleshoot: SELinux is preventing the squid (squid_t) from binding to port
10000. For complete SELinux messages. run sealert -l 97136444-4497-4fff-a7a7-
c4d8442db982
```

6. 为了使 SELinux 允许 squid 侦听端口 10000，如本例中使用的那样，需要使用以下命令：

```
~]# semanage port -a -t squid_port_t -p tcp 10000
```

7. 再次启动 squid 并使其监听新端口：

```
~]# systemctl start squid.service
```

8. 现在 SELinux 已配置为允许 Squid 侦听非标准端口（本示例中为 TCP 10000），它在此端口上成功启动。

19.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则

定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

以下类型与 Squid 一起使用：不同的类型允许您配置灵活的访问：

httpd_squid_script_exec_t

此类型用于诸如 cachemgr.cgi 等实用程序，它提供有关 Squid 及其配置的各种统计信息。

squid_cache_t

将此类型用于由 Squid 缓存的数据，如 /etc/squid/squid.conf 中的 cache_dir 指令所定义。默认情况下，在 中创建或复制到 /var/cache/squid/ 和 /var/ spool/squid/ 目录中的文件使用 squid_cache_t 类型进行标记。用于在 中创建或复制到 /var/squid Guard / 目录的 squid Guard URL 重定向器插件的文件也使用 squid_cache_t 类型标记。Squid 只能将标记为此类型的文件和目录用于其缓存的数据。

squid_conf_t

此类型用于 Squid 配置的目录和文件。现有文件或者在 中创建或复制到 /etc/squid/ 和 /usr/ share/squid/ 目录的文件均使用此类型进行标记，包括错误消息和图标。

squid_exec_t

此类型用于 squid 二进制文件 /usr/sbin/squid。

squid_log_t

此类型用于日志。现有文件或者在 中创建或复制到 /var/log/squid/ 或 /var/ log/squidGuard/ 的文件 必须使用此类型进行标记。

squid_initrc_exec_t

此类型用于启动 squid （位于 /etc/rc.d/init.d/squid ）所需的初始化文件。

squid_var_run_t

此类型供 /var/run/ 目录中的文件使用，特别是名为 /var/run/squid.pid 的进程 ID(PID)，该进程在运行时由 Squid 创建。

19.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

squid_connect_any

启用后，此布尔值允许 Squid 发起与任何端口上的远程主机的连接。

squid_use_tproxy

启用后，此布尔值允许 Squid 作为透明代理运行。

注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

19.4. 配置示例

19.4.1. Squid 连接到非标准端口

以下示例提供了 SELinux 如何通过强制上述布尔值补充 Squid 的真实演示，默认情况下仅允许访问某些端口。此示例随后将演示如何更改 布尔值并演示允许访问。

请注意，这只是一个示例，并演示 SELinux 如何影响简单的 Squid 配置。Squid 综合文档已超出本文档的讨论范畴。详情请查看官方 Squid 文档。此示例假定 Squid 主机有两个网络接口，即 Internet 访问，并且任何防火墙已配置为允许使用 Squid 侦听的默认 TCP 端口(TCP 3128)对内部接口进行访问。

1. 确认已安装了 squid ：

```
~]$ rpm -q squid
package squid is not installed
```

如果没有安装该软件包，以 root 用户身份使用 yum 工具安装它：

```
~]# yum install squid
```

2. 编辑主配置文件 /etc/squid/squid.conf，并确认 cache_dir 指令没有被注释，如下所示：

```
cache_dir ufs /var/spool/squid 100 16 256
```

此行指定要在本示例中使用的 cache_dir 指令的默认设置；它由 Squid 存储格式(ufs)、缓存所在系统上的目录(/var/spool/squid)组成，以兆字节为单位的磁盘空间大小（分别为16 和 256），最后包含 Squid 存储格式(ufs)。

3. 在同一配置文件中，确保 http_access allow localnet 指令已被取消注释。这允许来自 localnet ACL 的流量（在 Red Hat Enterprise Linux 上的 Squid 默认安装中自动配置）。它将允许任何现有 RFC1918 网络上的客户端计算机通过代理进行访问，这足以满足这个简单示例。

4. 在同一配置文件中，确保 see _hostname 指令未注释，并且已配置为计算机的主机名。该值应该是主机的完全限定域名(FQDN)：

```
visible_hostname squid.example.com
```

5. 以 root 身份，输入以下命令启动 squid 守护进程。由于这是 squid 首次启动时，这个命令会按照 cache_dir 指令上方指定的缓存目录初始化，然后启动守护进程：

```
~]# systemctl start squid.service
```

确保 squid 启动成功。输出将包括以下信息，只有时间戳会有所不同：

```
~]# systemctl status squid.service
squid.service - Squid caching proxy
   Loaded: loaded (/usr/lib/systemd/system/squid.service; disabled)
   Active: active (running) since Thu 2014-02-06 15:00:24 CET; 6s ago
```

6. 确认 squid 进程 ID(PID)已作为受限服务启动，如 squid_var_run_t 值所示：

```
~]# ls -lZ /var/run/squid.pid
-rw-r--r--. root squid unconfined_u:object_r:squid_var_run_t:s0 /var/run/squid.pid
```

7. 此时，连接到先前配置的 localnet ACL 的客户端计算机能够将此主机的内部接口用作其代理。这可以在适用于所有常见 Web 浏览器或系统范围的设置中配置。Squid 现在侦听目标计算机的默认端口(TCP 3128)，但目标计算机将仅允许通过通用端口与 Internet 上的其他服务的传出连接。这是由 SELinux 本身定义的策略。SELinux 将拒绝对非标准端口的访问，如下一步中所示：

8. 当客户端通过 Squid 代理（如侦听 TCP 端口 10000 的网站）使用非标准端口发出请求时，会记录类似如下的拒绝：

```
SELinux is preventing the squid daemon from connecting to network port 10000
```

9. 要允许此访问，必须修改 squid_connect_any 布尔值，因为它默认是禁用的：

```
~]# setsebool -P squid_connect_any on
```

注意

如果您不希望 setsebool 更改在重新引导后保留，则不要使用 -P 选项。

10. 现在，客户端可以访问 Internet 上的非标准端口，因为现在允许 Squid 代表其客户端发起到任何端口的连接。

---

[17] 如需更多信息，请参阅 Squid 缓存代理 项目页面。

第 20 章 MARIADB（MARIADB 的替代）

MariaDB 数据库是一个多用户、多线程 SQL 数据库服务器，由 MariaDB 服务器守护进程(mysqld)和许多客户端程序和库组成。[18]

在红帽企业 Linux 中，mariad b-server 软件包提供 MariaDB。输入以下命令查看是否安装了 mariadb-server 软件包：

```
~]$ rpm -q mariadb-server
package mariadb-server is not installed
```

如果没有安装，请以 root 用户身份使用 yum 工具安装它：

```
~]# yum install mariadb-server
```

20.1. MARIADB AND SELINUX

启用 MariaDB 时，它会默认限制运行。受限制的进程在其自己的域中运行，并且与其他受限制的进程隔开。如果一个受攻击者限制的进程受到 SELinux 策略配置的影响，攻击者对资源的访问权限和可能受到的破坏会受到限制。以下示例演示了在自己的域中运行的 MariaDB 进程。本例假设安装了 mariadb-server 软件包：

1. 运行 getenforce 命令确认 SELinux 是否以强制模式运行：

   ```
   ~]$ getenforce
   Enforcing
   ```

   当 SELinux 以强制模式运行时，该命令将返回 强制。

2. 以 root 用户身份输入以下命令启动 mariadb:

   ```
   ~]# systemctl start mariadb.service
   ```

   确认 服务正在运行。输出中应包括以下信息（只有时间戳有所不同）：

   ```
   ~]# systemctl status mariadb.service
   mariadb.service - MariaDB database server
   ```

```
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; disabled)
   Active: active (running) since Mon 2013-08-05 11:20:11 CEST; 3h 28min ago
```

3. 输入以下命令查看 mysqld 进程 ：

```
~]$ ps -eZ | grep mysqld
system_u:system_r:mysqld_safe_t:s0 12831 ?     00:00:00 mysqld_safe
system_u:system_r:mysqld_t:s0   13014 ?        00:00:00 mysqld
```

与 mysqld 进程关联的 SELinux 上下文为 system_u:system_r:mysqld_t:s0。上下文的第二部分 mysqld_t 是类型。类型定义进程的域以及文件的类型。在本例中，mysql d 进程 在 mysqld_t 域中运行。

20.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

以下类型用于 mysqld ：不同的类型允许您配置灵活的访问：

mysqld_db_t

此类型用于 MariaDB 数据库的位置。在 Red Hat Enterprise Linux 中，数据库的默认位置是 /var/lib/mysql/ 目录，但可以更改此目录。如果 MariaDB 数据库的位置发生更改，则必须使用此类型标记新位置。有关如何更改默认数据库位置以及如何相应地标记新部分的说明，请参阅 第 20.4.1 节 “MariaDB 更改数据库位置” 中的示例。

mysqld_etc_t

此类型用于 MariaDB 主配置文件 /etc/my.cnf，以及 /etc/mysql/ 目录中的任何其他配置文件。

mysqld_exec_t

此类型用于位于 /usr/libexec/mysqld 的 mysqld 二进制文件，这是红帽企业 Linux 上 MariaDB 二进制文件的默认位置。其他系统可能会在 /usr/sbin/mysqld 找到此二进制文件，该二进制文件也应使用此类型进行标记。

mysqld_unit_file_t

默认情况下，此类型用于 Red Hat Enterprise Linux 中的 /usr/lib/systemd/system/ 目录中的可

执行 MariaDB 相关文件。

mysqld_log_t

MariaDB 的日志需要使用此类型进行标记才能正常运行。与 mysql.* 通配符匹配的 /var/log/ 目录中所有日志文件都必须使用此类型进行标记。

mysqld_var_run_t

此类型供 /var/run/mariadb/ 目录中的文件使用，特别是名为 /var/run/mariadb/mariadb.pid 的进程 ID(PID)，该进程在运行时由 mysqld 守护进程创建。此类型也用于相关的套接字文件，如 /var/lib/mysql/mysql.sock。此类文件必须正确标记，才能作为受限制的服务进行正常操作。

20.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

selinuxuser_mysql_connect_enabled

启用后，此布尔值允许用户连接到本地 MariaDB 服务器。

exim_can_connect_db

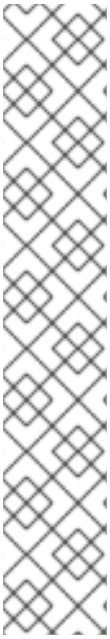
启用后，此布尔值允许 exim 邮件发送程序发起与数据库服务器的连接。

ftpd_connect_db

启用后，此布尔值允许 ftp 守护进程启动与数据库服务器的连接。

httpd_can_network_connect_db

Web 服务器需要启用此布尔值，才能与数据库服务器通信。



注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

20.4. 配置示例

20.4.1. MariaDB 更改数据库位置

在使用红帽企业 Linux 时，MariaDB 存储其数据库的默认位置是 /var/lib/mysql/。这是 SELinux 默认预期的位置，因此已使用 mysqld_db_t 类型正确标记了此区域。

数据库的存储位置可以根据个人环境要求或首选项来更改，但务必要让 SELinux 知道这个新位置；它应相应地进行标记。这个示例解释了如何更改 MariaDB 数据库的位置，以及如何标记新位置，以便 SELinux 仍然可以根据其内容向新区域提供保护机制。

请注意，这只是一个示例，并演示 SELinux 如何影响 MariaDB。MariaDB 的综合文档不在本文的讨论范围之内。详情请查看官方 MariaDB 文档。本例假定已安装了 mariadb-server 和 setroubleshoot-server 软件包，auditd 服务正在运行，并且 /var/lib/mysql/ 的默认位置有一个有效的数据库。

1. 查看 mysql 的默认数据库位置的 SELinux 上下文：

```
~]# ls -lZ /var/lib/mysql
drwx------. mysql mysql system_u:object_r:mysqld_db_t:s0 mysql
```

这将显示 mysqld_db_t，它是数据库文件的位置的默认上下文元素。此上下文必须手动应用到本示例中将使用的新数据库位置，才能正常工作。

2. 输入以下命令并输入 mysqld root 密码以显示可用的数据库：

```
~]# mysqlshow -u root -p
Enter password: *******
+--------------------+
|     Databases      |
+--------------------+
| information_schema |
| mysql              |
| test               |
| wikidb             |
+--------------------+
```

3. 停止 mysqld 守护进程：

```
~]# systemctl stop mariadb.service
```

4. 为数据库的新位置创建一个新目录。本例中使用了 /mysql/ ：

```
~]# mkdir -p /mysql
```

5. 将数据库文件复制到新位置：

```
~]# cp -R /var/lib/mysql/* /mysql/
```

6. 更改此位置的所有权，以允许 mysql 用户和组访问。这会设置 SELinux 仍然会观察到的传统 Unix 权限：

```
~]# chown -R mysql:mysql /mysql
```

7. 输入以下命令查看新目录的初始上下文：

```
~]# ls -lZ /mysql
drwxr-xr-x. mysql mysql unconfined_u:object_r:usr_t:s0   mysql
```

此新创建的目录的 context usr_t 目前不适用于 SELinux，作为 MariaDB 数据库文件的位置。上下文更改后，MariaDB 将能够在此领域正常工作。

8. 使用文本编辑器打开主 MariaDB 配置文件 /etc/my.cnf，再修改 datadir 选项，使其引用新位置。在本例中，应输入的值为 /mysql ：

```
[mysqld]
datadir=/mysql
```

保存此文件并退出。

9. 启动 **mysqld**。该服务应该无法启动，拒绝信息会记录到 **/var/log/messages** 文件中：

```
~]# systemctl start mariadb.service
Job for mariadb.service failed. See 'systemctl status postgresql.service' and 'journalctl -xn'
for details.
```

但是，如果 **audit** 守护进程与 **setroubleshoot** 服务一起运行，拒绝将记录到 **/var/log/audit/audit.log** 文件中：

```
SELinux is preventing /usr/libexec/mysqld "write" access on /mysql. For complete SELinux
messages. run sealert -l b3f01aff-7fa6-4ebe-ad46-abaef6f8ad71
```

此拒绝的原因是 **MariaDB** 数据文件未正确标记 **/mysql/**。**SELinux** 正在阻止 **MariaDB** 访问标记为 **usr_t** 的内容。执行以下步骤解决这个问题：

10. 输入以下命令为 **/mysql/** 添加上下文映射：请注意，默认情况下不会安装 **semanage** 工具。如果您的系统中没有它，请安装 **policycoreutils-python** 软件包。

```
~]# semanage fcontext -a -t mysqld_db_t "/mysql(/.*)?"
```

11. 这个映射被写入到 **/etc/selinux/targeted/contexts/files/file_contexts.local** 文件中：

```
~]# grep -i mysql /etc/selinux/targeted/contexts/files/file_contexts.local

/mysql(/.*)?    system_u:object_r:mysqld_db_t:s0
```

12. 现在，使用 **restorecon** 工具将这个上下文映射到正在运行的系统：

```
~]# restorecon -R -v /mysql
```

13. 现在，**/mysql/** 位置已标记为 **MariaDB** 的正确上下文，**mysql d** 启动：

```
~]# systemctl start mariadb.service
```

14. 确认 /**mysql**/ 的上下文已更改：

```
~]$ ls -lZ /mysql
drwxr-xr-x. mysql mysql system_u:object_r:mysqld_db_t:s0 mysql
```

15. 该位置已被更改并标记，**mysqld** 已成功启动。此时，应测试所有正在运行的服务，以确认正常运行。

---

[18] 如需更多信息，请参阅 MariaDB 项目页面。

第 21 章 POSTGRESQL

PostgreSQL 是对象存储的数据库管理系统(DBMS)。[19]

在红帽企业 Linux 中，postgresql -server 软件包提供 PostgreSQL。输入以下命令查看是否安装了 postgresql-server 软件包：

```
~]# rpm -q postgresql-server
```

如果没有安装，请以 root 用户身份使用 yum 工具安装它：

```
~]# yum install postgresql-server
```

21.1. POSTGRESQL 和 SELINUX

启用 PostgreSQL 时，它会默认限制运行。受限制的进程在其自己的域中运行，并且与其他受限制的进程隔开。如果一个受攻击者限制的进程受到 SELinux 策略配置的影响，攻击者对资源的访问权限和可能受到的破坏会受到限制。以下示例演示了在自己的域中运行的 PostgreSQL 进程。本例假设安装了 postgresql-server 软件包：

1. 运行 getenforce 命令确认 SELinux 是否以强制模式运行：

```
~]$ getenforce
Enforcing
```

当 SELinux 以强制模式运行时，该命令将返回 强制。

2. 以 root 用户身份输入以下命令启动 postgresql ：

```
~]# systemctl start postgresql.service
```

确认 服务正在运行。输出中应包括以下信息（只有时间戳有所不同）：

```
~]# systemctl start postgresql.service
postgresql.service - PostgreSQL database server
   Loaded: loaded (/usr/lib/systemd/system/postgresql.service; disabled)
   Active: active (running) since Mon 2013-08-05 14:57:49 CEST; 12s
```

3. 输入以下命令查看 **postgresql** 进程 ：

```
~]$ ps -eZ | grep postgres
system_u:system_r:postgresql_t:s0 395 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 397 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 399 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 400 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 401 ?    00:00:00 postmaster
system_u:system_r:postgresql_t:s0 402 ?    00:00:00 postmaster
```

与 **postgresql** 进程关联的 **SELinux** 上下文为 **system_u:system_r:postgresql_t:s0**。上下文的第二部分 **postgresql_t** 是类型。类型定义进程的域以及文件的类型。在本例中，**postgresql** 进程在 **postgresql _t** 域中运行。

21.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 **Type Enforcement**。所有文件和进程都设置了类型标签：**type** 为进程定义 **SELinux** 域，以及文件的 **SELinux** 类型。**SELinux** 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 **SELinux** 策略规则时才允许访问。

以下类型用于 **postgresql** ：不同的类型允许您配置灵活的访问。请注意，在下面的列表中，使用多个正则表达式来匹配所有可能的位置：

postgresql_db_t

此类型用于多个位置。使用这个类型的标记位置用于 **PostgreSQL** 的数据文件：

- **/usr/lib/pgsql/test/regres**
- **/usr/share/jonas/pgsql**
- **/var/lib/pgsql/data**
- **/var/lib/postgres(ql)?**

postgresql_etc_t

此类型用于 /etc/postgresql/ 目录中的配置文件。

postgresql_exec_t

此类型用于多个位置。使用这个类型标记的位置用于 PostgreSQL 的二进制文件：

- */usr/bin/initdb(.sepgsql)?*
- */usr/bin/(se)?postgres*
- */usr/lib(64)?/postgresql/bin/.**
- */usr/lib(64)?/pgsql/test/regress/pg_regress*

systemd_unit_file_t

此类型用于 /usr/lib/systemd/system/ 目录中的可执行 PostgreSQL 相关文件。

postgresql_log_t

此类型用于多个位置。使用这个类型标记的位置用于日志文件：

- */var/lib/pgsql/logfile*
- */var/lib/pgsql/pgstartup.log*
- */var/lib/sepgsql/pgstartup.log*

- /var/log/postgresql
- /var/log/postgres.log.*
- /var/log/rhdb/rhdb
- /var/log/sepostgresql.log.*

postgresql_var_run_t

此类型用于 PostgreSQL 的运行时文件，如 /var/run/postgresql/ 目录中的进程 ID(PID)。

21.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

selinuxuser_postgresql_connect_enabled

启用此布尔值后，任何用户域（如 PostgreSQL 定义）都允许与数据库服务器建立连接。

注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

21.4. 配置示例

21.4.1. PostgreSQL 更改数据库位置

在使用 Red Hat Enterprise Linux 时，PostgreSQL 存储其数据库的默认位置是 /var/lib/pgsql/data/。这是 SELinux 默认预期的位置，因此已使用 postgresql_db_t 类型正确标记了此区域。

数据库所在区域可根据个人环境要求或首选项进行更改，但务必要让 SELinux 知道此新位置；应相应地对其进行标记。本例介绍如何更改 PostgreSQL 数据库的位置，以及如何标记新位置，以便 SELinux 仍然能够根据其内容向新区域提供保护机制。

请注意，这只是一个示例，并演示 SELinux 如何影响 PostgreSQL。PostgreSQL 的综合文档不在本文档的讨论范围之内。详情请查看官方 PostgreSQL 文档。本例假定已安装了 postgresql-server 软件包。

1. 查看 postgresql 默认数据库位置的 SELinux 上下文：

```
~]# ls -lZ /var/lib/pgsql
drwx------. postgres postgres system_u:object_r:postgresql_db_t:s0 data
```

这将显示 postgresql_db_t，它是数据库文件位置的默认上下文元素。此上下文必须手动应用到本示例中将使用的新数据库位置，才能正常工作。

2. 为数据库的新位置创建一个新目录。本例中使用了 /opt/postgresql/data/。如果您使用不同

的位置，请将以下步骤中的文本替换为您的位置：

```
~]# mkdir -p /opt/postgresql/data
```

3. *执行新位置的目录列表。请注意，新目录的初始上下文是 usr_t。该上下文不足以让 SELinux 为 PostgreSQL 提供保护机制。当上下文改变后，它将能够在新领域正常工作。*

```
~]# ls -lZ /opt/postgresql/
drwxr-xr-x. root root unconfined_u:object_r:usr_t:s0   data
```

4. *更改新位置的所有权，以允许 postgres 用户和组访问。这会设置 SELinux 仍然会观察到的传统 Unix 权限。*

```
~]# chown -R postgres:postgres /opt/postgresql
```

5. *使用文本编辑器打开 /etc/systemd/system/postgresql.service 文件，并修改 PGDATA 和 PGLOG 变量以指向新位置：*

```
~]# vi /etc/systemd/system/postgresql.service
PGDATA=/opt/postgresql/data
PGLOG=/opt/postgresql/data/pgstartup.log
```

保存此文件并退出文本编辑器。

如果 /etc/systemd/system/postgresql.service 文件不存在，请创建该文件并插入以下内容：

```
.include /lib/systemd/system/postgresql.service
[Service]

# Location of database directory
Environment=PGDATA=/opt/postgresql/data
Environment=PGLOG=/opt/postgresql/data/pgstartup.log
```

6. *在新位置初始化数据库：*

```
~]$ su - postgres -c "initdb -D /opt/postgresql/data"
```

7. 更改数据库位置后，启动服务此时将失败：

```
~]# systemctl start postgresql.service
Job for postgresql.service failed. See 'systemctl status postgresql.service' and 'journalctl -xn'
for details.
```

SELinux 导致服务无法启动。这是因为新位置没有正确标记。以下步骤解释了如何标记新位置(/opt/postgresql/)并正确启动 postgresql 服务：

8. 使用 semanage 工具为 /opt/postgresql/ 以及 其中的任何其他目录/文件添加上下文映射：

```
~]# semanage fcontext -a -t postgresql_db_t "/opt/postgresql(/.*)?"
```

9. 这个映射被写入到 /etc/selinux/targeted/contexts/files/file_contexts.local 文件中：

```
~]# grep -i postgresql /etc/selinux/targeted/contexts/files/file_contexts.local

/opt/postgresql(/.*)?    system_u:object_r:postgresql_db_t:s0
```

10. 现在，使用 restorecon 工具将这个上下文映射到正在运行的系统：

```
~]# restorecon -R -v /opt/postgresql
```

11. 现在，/opt/postgresql/ 位置已标记为 PostgreSQL 的正确上下文，postgresql 服务 将成功启动：

```
~]# systemctl start postgresql.service
```

12. 确认 /opt/postgresql/ 的上下文正确：

```
~]$ ls -lZ /opt
drwxr-xr-x. root root system_u:object_r:postgresql_db_t:s0 postgresql
```

13. 使用 ps 命令检查 postgresql 进程是否 显示新位置：

```
~]# ps aux | grep -i postmaster
```

```
postgres 21564  0.3  0.3  42308  4032 ?        S    10:13   0:00 /usr/bin/postmaster -p 5432 -D
/opt/postgresql/data/
```

14. 该位置已被更改并标记，postgresql 已成功启动。此时，应测试所有正在运行的服务，以确认正常运行。

---

[19] 如需更多信息，请参阅 PostgreSQL 项目页面。

第 22 章 RSYNC

rsync 实用程序执行快速文件传输，用于在系统间同步数据。[20]

在使用红帽企业 Linux 时，r sync 软件包提供了 rsync。输入以下命令查看是否安装了 rsync 软件包：

```
~]$ rpm -q rsync
package rsync is not installed
```

如果没有安装，请以 root 用户身份使用 yum 工具安装它：

```
~]# yum install rsync
```

22.1. RSYNC AND SELINUX

SELinux 要求文件具有扩展属性来定义文件类型。策略控制守护进程对这些文件的访问。如果要使用 rsync 守护进程共享文件，您必须使用 public_content_t 类型标记文件和目录。与大多数服务一样，SELinux 需要正确的标记才能通过 rsync 执行其保护机制。[21]

22.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

以下类型与 rsync 一起使用：要配置灵活访问的所有类型：

public_content_t

这是一种通用类型，用于通过 rsync 共享的文件的位置（以及实际文件）。如果创建了一个特殊目录来存放要与 rsync 共享的文件，则该目录及其内容需要应用此标签。

rsync_exec_t

此类型用于 /usr/bin/rsync 系统二进制文件。

rsync_log_t

默认情况下，此类型用于位于 /var/log/rsync.log 的 rsync 日志文件。要将文件 rsync 日志的位置更改为，可在运行时对 rsync 命令使用 --log-file=FILE 选项。

rsync_var_run_t

此类型用于 rsyncd 锁定文件，该文件位于 /var/run/rsyncd.lock。此锁定文件供 rsync 服务器用于管理连接限制。

rsync_data_t

此类型用于您要用作 rsync 域的文件和目录，并将它们与其他服务的访问范围隔离开来。此外，public_content_t 是一种常规 SELinux 上下文类型，当文件或目录与多个服务（例如，FTP 和 NFS 目录作为 rsync 域）交互时，可以使用它。

rsync_etc_t

此类型用于 /etc 目录中与 rsync 相关的文件。

22.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

rsync_anon_write

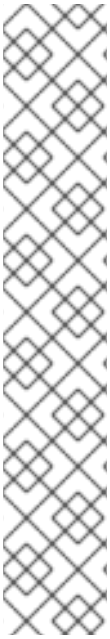
启用此布尔值后，r sync 可以在 rsync_t 域中管理类型为 public_content_rw_t 的文件、链接和目录。这些通常是用于公共文件传输服务的公共文件。必须对此类型进行标记。

rsync_client

启用此布尔值后，r sync 可以启动到定义为 rsync_port_t 的端口的连接，并允许守护进程管理类型为 rsync_data_t 的文件、链接和目录。请注意，rsync 必须 位于 rsync_t 域中，以便 SELinux 能够对它进行控制。本章中的配置示例演示了在 rsync _t 域中运行的 rsync。

rsync_export_all_ro

启用此布尔值后，r sync 域中的 rsync 可以导出对客户端具有只读访问权限的 NFS 和 CIFS 卷。



注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

22.4. 配置示例

22.4.1. rsync 作为守护进程

在使用红帽企业 Linux 时，rsync 可用作守护进程，让多个客户端可以直接将其通信为中央服务器，从而存储中央文件并保持它们同步。以下示例将演示通过正确的域中的网络套接字将 rsync 作为守护进程运行，以及 SELinux 如何期望此守护进程在预定义（SELinux 策略中）TCP 端口上运行。这个示例将介绍如何修改 SELinux 策略，以允许 rsync 守护进程在非标准端口上正常运行。

此示例将在单个系统上执行，以演示 SELinux 策略及其对本地后台程序和进程的控制。请注意，这只是一个示例，并演示了 SELinux 如何影响 rsync。rsync 的综合文档不在本文的讨论范围之内。详情请查看官方的 rsync 文档。本例假定安装了 rsync、setroubleshoot-server 和 audit 软件包，使用了 SELinux targeted 策略，并且 SELinux 处于强制模式。

过程 22.1. 获取 rsync 作为 rsync_t 启动

1. 运行 getenforce 命令确认 SELinux 是否以强制模式运行：

```
~]$ getenforce
Enforcing
```

当 SELinux 以强制模式运行时，该命令将返回 强制。

2. 运行 这个命令 确认 rsync 二进制文件位于系统路径中：

```
~]$ which rsync
/usr/bin/rsync
```

3. 当将 rsync 作为守护进程运行时，配置文件应使用并保存为 /etc/rsyncd.conf。请注意，本例中使用的以下配置文件非常简单，并不表示所有可用的选项，而是足以演示 rsync 守护进程：

```
log file = /var/log/rsync.log
pid file = /var/run/rsyncd.pid
lock file = /var/run/rsync.lock
[files]
 path = /srv/rsync
        comment = file area
        read only = false
        timeout = 300
```

4. 现在，rsync 有一个简单的配置文件可在守护进程模式下运行，您可以运行以下命令来启动该文件：

```
~]# systemctl start rsyncd.service
```

确保 rsyncd 已成功启动（输出应当类似于下方所示），只有时间戳会有所不同：

```
~]# systemctl status rsyncd.service
rsyncd.service - fast remote file copy program daemon
   Loaded: loaded (/usr/lib/systemd/system/rsyncd.service; disabled)
   Active: active (running) since Thu 2014-02-27 09:46:24 CET; 2s ago
 Main PID: 3220 (rsync)
   CGroup: /system.slice/rsyncd.service
           └─3220 /usr/bin/rsync --daemon --no-detach
```

SELinux 现在可以在 rsync 守护进程上强制实施保护机制，因为它现在在 rsync_t 域中运行：

```
~]$ ps -eZ | grep rsync
system_u:system_r:rsync_t:s0     3220 ?        00:00:00 rsync
```

本例演示了如何在 rsync _t 域中运行 rsync d。rsync 也可以作为套接字激活的服务运行。在这种情况下，只有客户端尝试连接服务后才会执行 rsyncd。要启用 rsyncd 作为套接字激活的服务运行，请按照上述步骤操作。要将 rsyncd 作为套接字激活的服务启动，以 root 用户身份输入以下命令：

```
~]# systemctl start rsyncd.socket
```

下一个示例演示了如何在非默认端口上成功运行此守护进程。下一个示例中使用 TCP 端口 10000。

过程 22.2. 在非默认端口上运行 rsync 守护进程

1. 修改 /etc/rsyncd.conf 文件，并在全局配置区域（即定义任何文件区域之前）的文件顶部添加 port = 10000 行。新配置文件将如下所示：

```
log file = /var/log/rsyncd.log
pid file = /var/run/rsyncd.pid
lock file = /var/run/rsync.lock
port = 10000
[files]
        path = /srv/rsync
        comment = file area
        read only = false
 timeout = 300
```

2. 使用这个新设置启动 rsync 守护进程后，SELinux 会记录类似如下的拒绝信息：

```
Jul 22 10:46:59 localhost setroubleshoot: SELinux is preventing the rsync (rsync_t) from
binding to port 10000. For complete SELinux messages, run sealert -l c371ab34-639e-45ae-
9e42-18855b5c2de8
```

3. 使用 semanage 工具将 TCP 端口 10000 添加到 rsync_port_t 中的 SELinux 策略中：

```
~]# semanage port -a -t rsync_port_t -p tcp 10000
```

4. 现在，对于 rsync_port_t 的 TCP 端口 10000 已添加到 SELinux 策略中，rsyncd 将在此端口上启动并正常运行：

```
~]# systemctl start rsyncd.service
```

```
~]# netstat -lnp | grep 10000
tcp        0      0 0.0.0.0:10000   0.0.0.0:*      LISTEN      9910/rsync
```

SELinux 已修改策略，现在允许 rsyncd 在 TCP 端口 10000 上运行。

[20]
如需更多信息，请参阅 Rsync 项目页面。

[21]
有关 rsync 和 SELinux 的更多信息，请参阅 rsync_selinux(8) 手册页。

第 23 章 POSTFIX

Postfix 是一个开源邮件传输代理(MTA)，它支持 LDAP、SMTP AUTH(SASL)和 TLS 等协议。[22]

在 Red Hat Enterprise Linux 中， postfix 软件包提供 Postfix。输入以下命令查看是否安装了 postfix 软件包：

```
~]$ rpm -q postfix
package postfix is not installed
```

如果没有安装，使用 yum 工具 root 安装它：

```
~]# yum install postfix
```

23.1. POSTFIX 和 SELINUX

启用 Postfix 后，它将默认受到限制。受限制的进程在其自己的域中运行，并且与其他受限制的进程隔开。如果一个受攻击者限制的进程受到 SELinux 策略配置的影响，攻击者对资源的访问权限和可能受到的破坏会受到限制。以下示例演示了在自己的域中运行的 Postfix 和相关进程。本例假设已安装 postfix 软件包，并且 Postfix 服务已经启动：

1. 运行 getenforce 命令确认 SELinux 是否以强制模式运行：

   ```
   ~]$ getenforce
   Enforcing
   ```

   当 SELinux 以强制模式运行时，该命令将返回 强制。

2. 以 root 用户身份输入以下命令启动 postfix ：

   ```
   ~]# systemctl start postfix.service
   ```

   确认 服务正在运行。输出中应包括以下信息（只有时间戳有所不同）：

   ```
   ~]# systemctl status postfix.service
   postfix.service - Postfix Mail Transport Agent
     Loaded: loaded (/usr/lib/systemd/system/postfix.service; disabled)
   ```

```
   Active: active (running) since Mon 2013-08-05 11:38:48 CEST; 3h 25min ago
```

3. 运行以下命令来查看 **postfix** 进程 ：

```
~]$ ps -eZ | grep postfix
system_u:system_r:postfix_master_t:s0 1651 ?   00:00:00 master
system_u:system_r:postfix_pickup_t:s0 1662 ?   00:00:00 pickup
system_u:system_r:postfix_qmgr_t:s0 1663 ?     00:00:00 qmgr
```

在上面的输出中，与 **Postfix** 主 进程关联的 **SELinux** 上下文是 **system_u:system_r:postfix_master_t:s0**。上下文的第二部分 **postfix_master_t** 是此进程的类型。类型定义进程的域以及文件的类型。在本例中，**master** 进程在 **postfix_master_t** 域中运行。

23.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 **Type Enforcement**。所有文件和进程都设置了类型标签：**type** 为进程定义 **SELinux** 域，以及文件的 **SELinux** 类型。**SELinux** 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 **SELinux** 策略规则时才允许访问。

以下类型与 **Postfix** 一起使用：要配置灵活访问的所有类型：

postfix_etc_t

此类型用于 **/etc/postfix/** 目录中 **Postfix** 的配置文件。

postfix_data_t

此类型用于 **/var/lib/postfix/** 目录中的 **Postfix** 数据文件。

postfix_var_run_t

此类型用于存储在 **/run/** 目录中的 **Postfix** 文件。

postfix_initrc_exec_t

Postfix 可执行文件使用 **postfix_initrc_exec_t** 类型进行标记。执行时，它们过渡到 **postfix_initrc_t** 域。

postfix_spool_t

此类型用于存储在 /var/spool/ 目录中的 Postfix 文件。

注意

要查看 Postfix 的文件及其类型的完整列表，请输入以下命令：

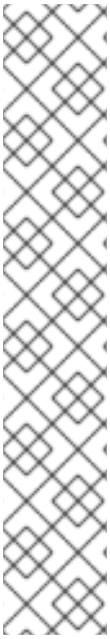
```
~]$ grep postfix /etc/selinux/targeted/contexts/files/file_contexts
```

23.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

postfix_local_write_mail_spool

启用此布尔值后，Postfix 可以写入系统上的本地邮件假脱机。使用本地假脱机时，Postfix 要求启用此布尔值以进行正常操作。



注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

23.4. 配置示例

23.4.1. SpamAssassin 和 Postfix

SpamAssasin 是一个开源邮件过滤器，它提供了一种从传入电子邮件过滤主动电子邮件（垃圾邮件）的方法。[23]

使用红帽企业 Linux 时，spam assassin 软件包提供了 SpamAssassin。输入以下命令查看是否安装了 spamassassin 软件包：

```
~]$ rpm -q spamassassin
package spamassassin is not installed
```

如果没有安装，请以 root 用户身份使用 yum 工具安装它：

```
~]# yum install spamassassin
```

SpamAssassin 与 Postfix 等邮件发送者一起工作，以提供垃圾邮件过滤功能。要让 SpamAssassin 有效拦截、分析和过滤邮件，它必须侦听网络接口。SpamAssassin 的默认端口为 TCP/783，但可以更改此端口。以下示例提供了 SELinux 如何通过仅允许访问特定端口来补充 SpamAssassin 的真实演示。本示例随后将演示如何更改端口，并让 SpamAssassin 在非默认端口上运行。

请注意，这只是一个示例，并演示 SELinux 如何影响简单的 SpamAssassin 配置。SpamAssassin 的综合文档不在本文的讨论范围之内。详情请查看官方 SpamAssassin 文档。这个示例假设已安装 spamassassassin，所有防火墙都已配置为允许访问正在使用的端口上，使用了 SELinux 目标策略，并且 SELinux 以强制模式运行：

过程 23.1. 在非默认端口上运行 SpamAssassin

1. 以 root 用户身份使用 semanage 工具显示 SELinux 允许 spamd 守护进程默认侦听的端口：

```
~]# semanage port -l | grep spamd
spamd_port_t  tcp 783
```

此输出显示在 spamd_port_t 中定义了 TCP/783，作为要操作的 SpamAssassin 的端口。

2. 编辑 /etc/sysconfig/spamassin 配置文件并进行修改，使其在示例端口 TCP/10000 上启动 SpamAssassin：

```
# Options to spamd
SPAMDOPTIONS="-d -p 10000 -c m5 -H"
```

此行现在指定 SpamAssassin 在端口 10000 上运行。本例的其余部分将显示如何修改 SELinux 策略以允许打开此套接字。

3. 启动 SpamAssassin，并显示类似如下的错误消息：

```
~]# systemctl start spamassassin.service
Job for spamassassin.service failed. See 'systemctl status spamassassin.service' and
'journalctl -xn' for details.
```

此输出意味着 SELinux 已阻止对此端口的访问。

4. SELinux 会记录类似如下的拒绝信息：

```
SELinux is preventing the spamd (spamd_t) from binding to port 10000.
```

5. 以 root 用户身份运行 semanage 以修改 SELinux 策略，以允许 SpamAssassin 在示例端口(TCP/10000)上运行：

```
~]# semanage port -a -t spamd_port_t -p tcp 10000
```

6. 确认 SpamAssassin 现在将启动并在 TCP 端口 10000 上运行：

```
~]# systemctl start spamassassin.service

~]# netstat -lnp | grep 10000
tcp 0 0 127.0.0.1:10000 0.0.0.0:* LISTEN 2224/spamd.pid
```

7. 此时，垃圾邮件在 TCP 端口 10000 上正确运行，因为 SELinux 策略允许访问该端口。

---

[22] 如需更多信息，请参阅《系统管理员指南》中的 Postfix 部分。

[23] 如需更多信息，请参阅《系统管理员指南》中的 Spam Filters 部分。

第 24 章 DHCP

dhcpd 守护进程在 Red Hat Enterprise Linux 中用于动态提供和配置客户端的第 3 层 TCP/IP 详细信息。

dhcp 软件包提供 DHCP 服务器和 dhcpd 守护进程。输入以下命令查看是否安装了 dhcp 软件包：

```
~]# rpm -q dhcp
package dhcp is not installed
```

如果没有安装，请以 root 用户身份使用 yum 工具安装它：

```
~]# yum install dhcp
```

24.1. DHCP 和 SELINUX

启用 dhcpd 时，它将默认限制。受限制的进程在其自己的域中运行，并且与其他受限制的进程隔开。如果一个受攻击者限制的进程受到 SELinux 策略配置的影响，攻击者对资源的访问权限和可能受到的破坏会受到限制。以下示例演示了在自己的域中运行的 dhcpd 和相关进程。本例假设已安装了 dhcp 软件包，并且 dhcpd 服务已经启动：

1. 运行 getenforce 命令确认 SELinux 是否以强制模式运行：

   ```
   ~]$ getenforce
   Enforcing
   ```

   当 SELinux 以强制模式运行时，该命令将返回 强制。

2. 以 root 用户身份输入以下命令启动 dhcpd ：

   ```
   ~]# systemctl start dhcpd.service
   ```

   确认 服务正在运行。输出中应包括以下信息（只有时间戳有所不同）：

   ```
   ~]# systemctl status dhcpd.service
   dhcpd.service - DHCPv4 Server Daemon
     Loaded: loaded (/usr/lib/systemd/system/dhcpd.service; disabled)
   ```

```
Active: active (running) since Mon 2013-08-05 11:49:07 CEST; 3h 20min ago
```

3. 运行以下命令来查看 **dhcpd** 进程 ：

```
~]$ ps -eZ | grep dhcpd
system_u:system_r:dhcpd_t:s0 5483 ?        00:00:00 dhcpd
```

与 **dhcpd** 进程关联的 **SELinux** 上下文是 **system_u:system_r:dhcpd_t:s0**。

24.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 **Type Enforcement**。所有文件和进程都设置了类型标签：**type** 为进程定义 **SELinux** 域，以及文件的 **SELinux** 类型。**SELinux** 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 **SELinux** 策略规则时才允许访问。

以下类型与 **DHCP** 搭配使用：

dhcp_etc_t

此类型主要用于 **/etc** 目录中的文件，包括配置文件。

dhcpd_var_run_t

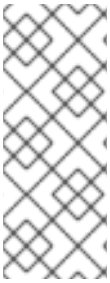
此类型用于 **/var/run/** 目录中 **dhcpd** 的 **PID** 文件。

dhcpd_exec_t

这种类型用于将 **DHCP** 可执行文件转换到 **dhcpd_t** 域。

dhcpd_initrc_exec_t

此类型用于将 **DHCP** 可执行文件转换到 **dhcpd_initrc_t** 域。



注意

*要查看 **dhcpd** 的文件及其类型的完整列表，请输入以下命令：*

```
~]$ grep dhcp /etc/selinux/targeted/contexts/files/file_contexts
```

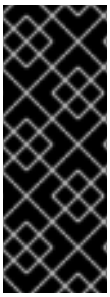
第 25 章 OPENSHIFT BY RED HAT

红帽 OpenShift 是一种平台即服务(PaaS)，使开发人员能够构建和部署 Web 应用程序。OpenShift 提供了广泛的编程语言和框架，包括 Java、Ruby 和 PHP。它还提供集成的开发人员工具，以支持应用生命周期，包括 Eclipse 集成、JBoss Developer Studio 和 Jenkins。OpenShift 使用开源生态系统为移动应用、数据库服务等提供平台。[24]

在 Red Hat Enterprise Linux 中，openshift-clients 软件包提供了 OpenShift 客户端工具。输入以下命令查看是否安装了它：

```
~]$ rpm -q openshift-clients
package openshift-clients is not installed
```

如果没有安装 openshift-clients 软件包，请参阅《OpenShift 企业客户端工具安装指南》 和 OpenShift 在线客户端工具安装指南 以了解 OpenShift 客户端工具安装过程的详细信息。



重要

在以前的版本中，rh c 软件包提供了 OpenShift 客户端工具。在最新的 OpenShift 版本中，这个软件包已被弃用，红帽不再支持这个软件包。因此，在 OpenShift 版本 2 后，rhc 软件包 被替换为 openshift-clients 软件包，它提供用于支持的 OpenShift 版本的 OpenShift 客户端工具。

25.1. OPENSHIFT AND SELINUX

SELinux 提供对使用 OpenShift 的应用提供更好的安全控制，因为所有进程都按照 SELinux 策略进行标记。因此，SELinux 可保护 OpenShift 不受同一节点上运行不同工具内可能的恶意攻击。

有关 SELinux 和 OpenShift 的更多信息，请参阅 Dan Walsh 的演示。

25.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

以下类型与 OpenShift 搭配使用：不同的类型允许您配置灵活的访问：

进程类型

openshift_t

OpenShift 进程与 openshift_t SELinux 类型关联。

可执行文件上的类型

openshift_cgroup_read_exec_t

SELinux 允许这种类型的文件将可执行文件转换到 openshift_cgroup_read_t 域。

openshift_cron_exec_t

SELinux 允许这种类型的文件将可执行文件转换到 openshift_cron_t 域。

openshift_initrc_exec_t

SELinux 允许这种类型的文件将可执行文件转换到 openshift_initrc_t 域。

可写入类型

openshift_cgroup_read_tmp_t

这种类型允许 OpenShift 控制组(cgroup)在 /tmp 目录中读取和访问临时文件。

openshift_cron_tmp_t

此类型允许将 OpenShift cron 作业临时文件存储在 /tmp 中。

openshift_initrc_tmp_t

此类型允许将 OpenShift initrc 临时文件存储在 /tmp 中。

openshift_log_t

使用此类型的文件被视为 OpenShift 日志数据，通常存储在 /var/log/ 目录下。

openshift_rw_file_t

OpenShift 有权读取并写入使用此类型标记的文件。

openshift_tmp_t

此类型用于将 OpenShift 临时文件存储到 /tmp 中。

openshift_tmpfs_t

这种类型允许将 OpenShift 数据存储在 tmpfs 文件系统上。

openshift_var_lib_t

此类型允许将 OpenShift 文件存储在 /var/lib/ 目录中。

openshift_var_run_t

此类型允许将 OpenShift 文件存储在 /run/ 或 /var/run/ 目录中。

25.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

openshift_use_nfs

启用此布尔值后，可以在 NFS 共享上安装 OpenShift。

注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

25.4. 配置示例

25.4.1. 更改默认的 OpenShift 目录

默认情况下，OpenShift 将其数据存储在 /var/lib/openshift/ 目录中，该目录标有 openshift_var_lib_t SELinux 类型。要允许 OpenShift 将数据存储到其他目录中，请使用适当的 SELinux 上下文标记新目录。

以下流程演示了如何将数据存储的默认 OpenShift 目录更改为 /srv/openshift/ ：

过程 25.1. 更改存储数据的默认 OpenShift 目录

1. 以 root 用户身份，在 /srv 目录中创建一个新的 openshift/ 目录。新目录使用 var_t 类型标记：

```
~]# mkdir /srv/openshift
```

```
~]$ ls -Zd /srv/openshift
drwxr-xr-x. root root unconfined_u:object_r:var_t:s0   openshift/
```

2. 以 root 用户身份，使用 semanage 实用程序将 /srv/openshift/ 映射到正确的 SELinux 上下文：

```
~]# semanage fcontext -a -e /var/lib/openshift /srv/openshift
```

3. 然后，以 root 用户身份使用 restorecon 工具应用更改：

```
~]# restorecon -R -v /srv/openshift
```

4. /srv/openshift/ 目录现在使用正确的 openshift_var_lib_t 类型标记：

```
~]$ ls -Zd /srv/openshift
drwxr-xr-x. root root unconfined_u:object_r:openshift_var_lib_t:s0   openshift/
```

---

[24] 如需了解更多有关 OpenShift 的信息，请参阅 OpenShift Container Platform 产品文档和 OpenShift Online 产品文档。

第 26 章 IDENTITY MANAGEMENT

身份管理(IdM)为标准定义的常见网络服务提供了一个统一环境，包括 PAM、LDAP、Kerberos、DNS、NTP 和证书服务。IdM 允许 Red Hat Enterprise Linux 系统充当域控制器。[25]

在 Red Hat Enterprise Linux 中，ipa-server 软件包提供 IdM 服务器。输入以下命令查看是否安装了 ipa-server 软件包：

```
~]$ rpm -q ipa-server
package ipa-server is not installed
```

如果没有安装，以 root 用户身份输入以下命令安装它：

```
~]# yum install ipa-server
```

26.1. IDENTITY MANAGEMENT 和 SELINUX

身份管理可以将 IdM 用户映射到每个主机所配置的 SELinux 角色，从而能够为 IdM 访问权限指定 SELinux 上下文。在用户登录过程中，系统安全服务守护进程(SSSD)查询为特定 IdM 用户定义的访问权限。然后，pam_selinux 模块会向内核发送一个请求，以根据 IdM 访问权限使用正确的 SELinux 上下文启动用户进程，如 guest_u:guest_r:guest_t:s0。

有关身份管理和 SELinux 的更多信息，请参阅 Red Hat Enterprise Linux 7 的 Linux 域、身份、身份验证和策略指南。

26.1.1. 对 Active Directory 域的信任

在以前的 Red Hat Enterprise Linux 版本中，身份管理使用 WinSync 工具来允许 Active Directory(AD)域中的用户访问 IdM 域中存储的数据。为此，WinSync 必须将 AD 服务器中的用户和组数据复制到本地服务器，并保持数据同步。

在 Red Hat Enterprise Linux 7 中，S SSD 守护进程已被改进来使用 AD，用户可以在 IdM 和 AD 域之间创建信任的关系。用户和组数据直接从 AD 服务器读取。另外，提供 Kerberos 跨域信任，允许 AD 和 IdM 域间的单点登录(SSO)身份验证。如果设置了 SSO，来自 AD 域的用户可以访问存储在 IdM 域中无需密码的 Kerberos 保护的数据。

默认情况下不安装此功能。要使用它，请安装额外的 **ipa-server-trust-ad** 软件包。

26.2. 配置示例

26.2.1. 将 SELinux 用户映射到 IdM 用户

以下步骤演示了如何创建新 SELinux 映射以及如何将新的 IdM 用户添加到此映射中。

过程 26.1. 如何将用户添加到 SELinux 映射

1. 要创建新的 SELinux 映射，输入以下命令，其中 **SELinux_mapping** 是新 SELinux 映射的名称，**-- selinuxuser** 选项指定特定的 SELinux 用户：

```
~]$ ipa selinuxusermap-add SELinux_mapping --selinuxuser=staff_u:s0-s0:c0.c1023
```

2. 输入以下命令将 **tuser** 用户名的 IdM 用户添加到 SELinux 映射中：

```
~]$ ipa selinuxusermap-add-user --users=tuser SELinux_mapping
```

3. 要在 SELinux 映射中添加名为 **ipaclient.example.com** 的新主机，请输入以下命令：

```
~]$ ipa selinuxusermap-add-host --hosts=ipaclient.example.com SELinux_mapping
```

4. 当登录到 **ipaclient.example.com** 主机时，**tuser** 用户会获得 **staff_u:s0:c0.c1023** 标签：

```
[tuser@ipa-client]$ id -Z
staff_u:staff_r:staff_t:s0-s0:c0.c1023
```

[25]
有关身份管理的更多信息，请参阅 ***Red Hat Enterprise*** *Linux 7 的 Linux 域、身份、身份验证和策略指南。*

第 27 章 RED HAT GLUSTER STORAGE

红帽 Gluster 存储为企业提供灵活且经济的非结构化数据存储。Gluster 的一个关键构建块 GlusterFS 是基于可堆栈的用户空间设计，通过网络聚合各种存储服务器，并将它们从一个庞大的并行网络文件系统互连。兼容 POSIX 的 GlusterFS 服务器（使用 XFS 文件系统格式在磁盘上存储数据）可以使用行业标准访问协议（包括 NFS 和 CIFS）进行访问。

如需更多信息，请参阅红帽 Gluster 存储指南的产品文档。

glusterfs-server 软件包提供红帽 Gluster 存储。有关其安装过程的详细信息，请参见《红帽 Gluster 存储 安装指南 》。

27.1. RED HAT GLUSTER STORAGE 和 SELINUX

启用后，SELinux 充当额外的安全层，为 glusterd （GlusterFS 管理服务）和 glusterfsd （NFS 服务器）进程提供灵活的强制访问控制，作为红帽 Gluster 存储的一部分。这些进程具有高级进程隔离，未绑定 glusterd_t SELinux 类型。

27.2. 类型

SELinux 目标策略中用于提供高级进程隔离的主要权限控制方法是 Type Enforcement。所有文件和进程都设置了类型标签：type 为进程定义 SELinux 域，以及文件的 SELinux 类型。SELinux 策略规则定义类型如何相互访问，无论是访问某一类型的域还是访问其他域的域。只有在存在允许访问的特定 SELinux 策略规则时才允许访问。

下列类型与红帽 Gluster 存储搭配使用：不同的类型允许您配置灵活的访问：

进程类型

glusterd_t

Gluster 进程与 glusterd_t SELinux 类型关联。

可执行文件上的类型

glusterd_initrc_exec_t

适用于 Gluster 初始化脚本文件的 SELinux 特定脚本类型上下文。

glusterd_exec_t

Gluster 可执行文件的特定于 SELinux 的可执行文件上下文。

端口类型

gluster_port_t

为 glusterd 定义此类型。默认情况下，glu ster d 使用 204007-24027 和 38465-38469 TCP 端口。

文件上下文

glusterd_brick_t

此类型用于威胁为 glusterd brick 数据 的文件。

glusterd_conf_t

此类型与 glusterd 配置数据关联，通常存储在 /etc 目录中。

glusterd_log_t

具有此类型的文件被视为 glusterd 日志数据，通常存储在 /var/log/ 目录下。

glusterd_tmp_t

此类型用于存储 /tmp 目录中的 glusterd 临时文件。

glusterd_var_lib_t

此类型允许将 glusterd 文件存储在 /var/lib/ 目录中。

glusterd_var_run_t

此类型允许将 glusterd 文件存储在 /run/ 或 /var/run/ 目录中。

27.3. 布尔值

SELinux 基于服务运行所需的最低访问权限级别。服务可以以多种方式运行；因此，您需要指定如何运行您的服务。使用以下布尔值设置 SELinux：

gluster_export_all_ro

启用此布尔值后，glu sterfsd 即可以只读方式共享文件和目录。此布尔值默认为禁用。

gluster_export_all_rw

启用此布尔值后，glu sterfsd 将允许 glusterfsd 共享具有读写访问权限的文件和目录。此布尔值默认为启用。

gluster_anon_write

启用此布尔值后，glu sterfsd 可以修改标有 public_content_rw_t SELinux 类型的公共文件。

注意

由于 SELinux 策略的持续开发，以上列表可能不包含与 服务相关的所有布尔值。要列出它们，请输入以下命令：

```
~]$ getsebool -a | grep service_name
```

输入以下命令查看特定布尔值的描述：

```
~]$ sepolicy booleans -b boolean_name
```

请注意，需要额外的 policycoreutils-devel 软件包来提供 sepolicy 实用程序，这个命令才能正常工作。

27.4. 配置示例

27.4.1. 标记 Gluster Bricks

Gluster brick 是受信存储池中服务器上的导出目录。如果 brick 没有标记为正确的 SELinux 上下文 glusterd_brick_t，SELinux 会拒绝某些文件访问操作并生成各种 AVC 消息。

下列步骤演示了如何使用正确的 SELinux 上下文标记 Gluster brick。该过程假定您之前创建和格式化了逻辑卷，如 /dev/rhgs/gluster，以用作 Gluster brick。

有关 Gluster brick 的详细信息，请参见《红帽 Gluster 存储管理指南》 中的"红帽 Gluster 存储卷"一章。

过程 27.1. 如何标记 Gluster Brick

1. 创建目录以挂载之前格式化的逻辑卷。例如：

```
~]# mkdir /mnt/brick1
```

2. 在这种情况下，将逻辑卷 /dev/vg-group/gluster 挂载到上一步中创建的 /mnt/brick1/ 目录。

```
~]# mount /dev/vg-group/gluster /mnt/brick1/
```

请注意，mount 命令仅临时挂载设备。要永久挂载该设备，请在 /etc/fstab 文件中添加类似如下的条目：

```
/dev/vg-group/gluster    /mnt/brick1  xfs rw,inode64,noatime,nouuid      1 2
```

如需更多信息，请参阅 fstab(5) 手册页。

3. 检查 /mnt/brick1/ 的 SELinux 上下文：

```
~]$ ls -lZd /mnt/brick1/
drwxr-xr-x. root root system_u:object_r:unlabeled_t:s0 /mnt/brick1/
```

目录标有 unlabeled_t SELinux 类型。

4. 将 SELinux 类型 /mnt/brick1/ 更改为 glusterd_brick_t SELinux 类型：

```
~]# semanage fcontext -a -t glusterd_brick_t "/mnt/brick1(/.*)?"
```

5. 使用 restorecon 程序应用更改：

```
~]# restorecon -Rv /mnt/brick1
```

6. ***最后，验证上下文是否已成功更改：***

```
~]$ ls -lZd /mnt/brick1
drwxr-xr-x. root root system_u:object_r:glusterd_brick_t:s0 /mnt/brick1/
```

第 28 章 参考

以下参考指与 SELinux 相关的其他信息，但超出了本指南的范围。请注意，由于 SELinux 的快速发展，其中一些材料可能仅适用于 Red Hat Enterprise Linux 的特定版本。

books

SELinux 按示例

Mayer、MacMillan 和 Caplan

2007 年 Prentice 酒店.

SELinux：NSA 的开源安全增强型 Linux

Bill McCarty

O'Reilly Media Inc., 2004

教程和帮助

Russell Coker 的教程和对话

http://www.coker.com.au/selinux/talks/ibmtu-2004/

Dan Walsh 杂志

http://danwalsh.livejournal.com/

红帽知识库

https://access.redhat.com/site/

常规信息

NSA SELinux 主网站

https://www.nsa.gov/What-We-Do/Research/SELinux/

NSA SELinux FAQ

https://www.nsa.gov/What-We-Do/Research/SELinux/FAQs/

邮件列表

NSA SELinux 邮件列表

https://www.nsa.gov/What-We-Do/Research/SELinux/Mailing-List/

Fedora SELinux 邮件列表

http://www.redhat.com/mailman/listinfo/fedora-selinux-list

社区

SELinux Project Wiki

http://selinuxproject.org/page/Main_Page

SELinux 社区页面

http://selinux.sourceforge.net/

IRC

IRC.freenode.net, #selinux

附录 A. 修订历史记录

修订 0.3-06 7.7 GA 发行的版本.	Fri Aug 9 2019	Mirek Jahoda
修订 0.3-05 7.6 GA 发行的版本.	Sat Oct 20 2018	Mirek Jahoda
修订 0.3-03 7.5 GA 发行的版本.	Tue Apr 3 2018	Mirek Jahoda
修订 0.3-01 7.4 GA 发行的版本.	Thu Jul 13 2017	Mirek Jahoda
修订 0.2-18 7.3 GA 出版物版本。	Wed Nov 2 2016	Mirek Jahoda
修订 0.2-11 带有修复的异步发行版本。	Sun Jun 26 2016	Mirek Jahoda
修订 0.2-10 带有修复的异步发行版本。	Sun Feb 14 2016	Robert Krátký
修订 0.2-9 添加了红帽 Gluster 存储章节。	Thu Dec 10 2015	Barbora Ančincová
修订 0.2-8 本书的红帽企业 Linux 7.2 GA 版本.	Thu Nov 11 2015	Barbora Ančincová
修订 0.2-7 红帽企业 Linux 7.2 本书试用版.	Thu Aug 13 2015	Barbora Ančincová
修订 0.2-6 红帽企业 Linux 7.1 GA 版本书.	Wed Feb 18 2015	Barbora Ančincová
修订 0.2-5 更新 以在红帽客户门户上排序顺序.	Fri Dec 05 2014	Barbora Ančincová
修订 0.2-4 红帽企业 Linux 7.1 测试版本书.	Thu Dec 04 2014	Barbora Ančincová
修订 0.1-41 重新构建风格变更.	Tue May 20 2014	Tomáš Čapek
修订 0.1-1 最初创建 Red Hat Enterprise Linux 7 手册	Tue Jan 17 2013	Tomáš Čapek