Red Hat Enterprise Linux 9

使用 SELinux

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

Last Updated: 2022-06-15

Red Hat Enterprise Linux 9 使用 SELinux

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

本文档可帮助用户和管理员了解 SELinux 功能的基本性和原则，并描述了设置和配置各种服务的实际任务。

目录

让开源更具包容性

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。详情请查看 CTO Chris Wright 的信息。

对红帽文档提供反馈

我们感谢您对文档提供反馈信息。请让我们了解如何改进文档。

- 关于特定内容的简单评论：
  1. 请确定您使用 *Multi-page HTML* 格式查看文档。另外，确定 **Feedback** 按钮出现在文档页的右上方。
  2. 用鼠标指针高亮显示您想评论的文本部分。
  3. 点在高亮文本上弹出的 **Add Feedback**。
  4. 按照显示的步骤操作。
- 要通过 Bugzilla 提交反馈，请创建一个新的问题单：
  1. 进入 Bugzilla 网站。
  2. 在 Component 中选择 **Documentation**。
  3. 在 **Description** 中输入您要提供的信息。包括文档相关部分的链接。
  4. 点 **Submit Bug**。

第 1 章 SELINUX 入门

SELinux（Security Enhanced Linux）提供了一个额外的系统安全层。SELinux 从根本上解决以下问题：*May <subject> do <action> to <object>?*, 例如：*Web 服务器是否可以访问用户主目录中的文件？*

1.1. SELINUX 简介

系统管理员一般无法通过基于用户、组群和其它权限（称为 Discretionary Access Control，DAC）的标准访问策略生成全面、精细的安全策略。例如，限制特定应用程序只能查看日志文件，而同时允许其他应用程序在日志文件中添加新数据。

Security Enhanced Linux(SELinux)实现强制访问控制(MAC)。每个进程和系统资源都有一个特殊的安全性标签,称为 *SELinux 上下文（context）*。SELinux 上下文有时被称为 *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个一致方法，而且消除了在其他身份识别系统中可能存在的模糊性。例如，某个文件可以在使用绑定挂载的系统中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。

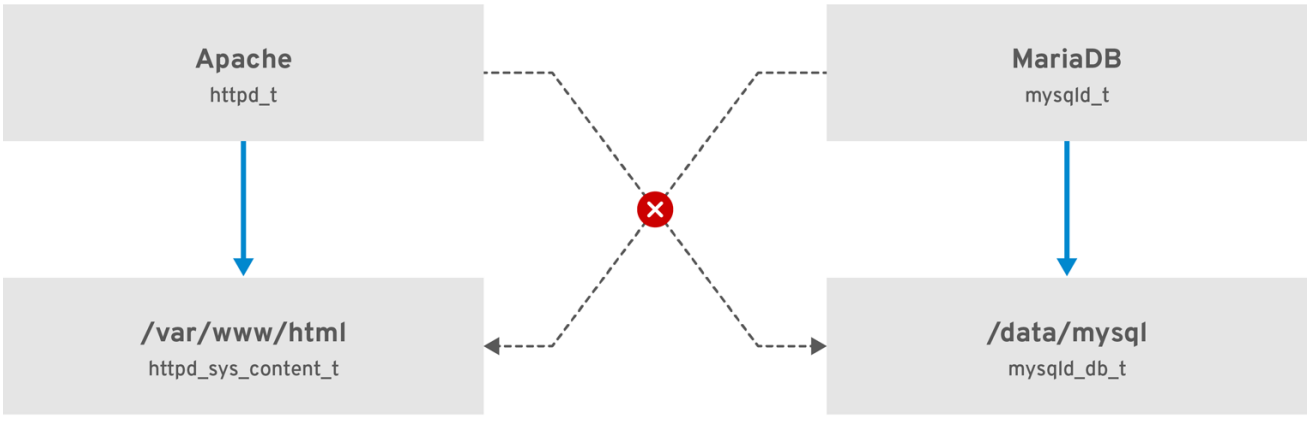
注意

请记住，对 SELinux 策略规则的检查是在 DAC 规则后进行的。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 类型以 **_t** 结尾。例如，Web 服务器的类型名称为 **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**。

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/** 和其他 Web 服务器目录中上下文的文件和目录(**httpd_sys_content_t**)。策略中没有允许规则适用于通常位于 **/tmp** 和 **/var/tmp/** 中的文件，因此不允许访问。因此，当使用 SELinux 时，即使 Apache 被破坏，一个恶意的脚本可以访问它，也无法访问 **/tmp** 目录。

图 1.1. 通过 SELinux 以安全的方式运行 Apache 和 MariaDB 的示例。



如上图所示，SELinux 允许作为 **httpd_t** 运行 Apache 进程访问 **/var/www/html/** 目录，并且拒绝同一进程访问 **/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 进程可以访问 **/data/mysql/** 目录，SELinux 也会正确地拒绝使用 **mysqld_t** 类型的进程来访问标记为 **httpd_sys_content_t** 的 **/var/www/html/** 目录。

其他资源

- **selinux(8)** 手册页和 **apropos selinux** 命令列出的 man page。
- 在安装了 **selinux-policy-doc** 软件包后，**man -k _selinux** 命令会列出 man page。
- The SELinux Coloring Book 可帮助您更好地了解 SELinux 基本概念。。
- SELinux Wiki FAQ

1.2. 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，仍需要遵循好的安全实践，如保持软件更新、使用安全的密码、使用防火墙。

1.3. SELINUX 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，这会导致 Linux 用户无法运行，除非有其他配置的用户 ID(setuid)应用

程序，如 **sudo** 和 **su**。

- 增加进程和数据的分离。SELinux *域（domain）*的概念允许定义哪些进程可以访问某些文件和目录。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 中使用 BIND（Berkeley Internet Name Domain）作为 DNS 服务器运行时，即使管理员没有限制哪些服务器可执行区传输，默认的 SELinux 策略也会阻止区文件 [1] 通过 BIND **named** 守护进程本身或其它进程的区传输被更新。

1.4. SELINUX 构架和软件包

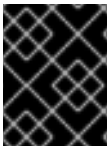
SELinux 是一个内置在 Linux 内核中的 Linux 安全模块（LSM）。内核中的 SELinux 子系统由安全策略驱动，该策略由管理员控制并在引导时载入。系统中所有与安全性相关的、内核级别的访问操作都会被 SELinux 截取，并在加载的安全策略上下文中检查。如果载入的策略允许操作，它将继续进行。否则,操作会被阻断，进程会收到一个错误。

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。原始审计消息会记录到 **/var/log/audit/audit.log**，它们以 **type=AVC** 字符串开头。

在 RHEL 9 中，系统服务由 **systemd** 守护进程控制； **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** 守护进程可以参考 SELinux 策略，检查调用进程标签以及调用者试图管理的单元文件标签，然后询问 SELinux 是否允许调用者的访问。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。

systemd 守护进程也可以作为 SELinux 访问管理器使用。它检索运行 **systemctl** 或向 **systemd** 发送 **D-Bus** 消息的进程标签。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** 就可以从内核中检索信息。这意味着，需要与特定服务交互的 **systemd** 进行交互的应用程序现在可以受 SELinux 限制。策略作者也可以使用这些精细的控制来限制管理员。

如果进程向另一个进程发送 **D-Bus** 消息，如果 SELinux 策略不允许这两个进程的 **D-Bus** 通信，则系统会打印 **USER_AVC** 拒绝消息，以及 D-Bus 通信超时。请注意，两个进程间的 D-Bus 通信会双向运行。



重要

为了避免不正确的 SELinux 标记以及后续问题，请确定使用 **systemctl start** 命令启动服务。

RHEL 9 提供以下用于 SELinux 的软件包：

- 策略：**selinux-policy-targeted**, **selinux-policy-mls**

- 工具： **policycoreutils**,**policycoreutils-gui**,**libselinux-utils**,**policycoreutils-python-utils**,**setools-console**,**checkpolicy**

1.5. SELINUX 状态和模式

SELinux 可使用三种模式之一运行： enforcing（强制）、permissive（宽容）或 disabled（禁用）。

- 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。

- 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。不建议在生产环境系统中使用 permissive 模式，但 permissive 模式对 SELinux 策略开发和调试很有帮助。

- 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。

使用 **setenforce** 实用程序在 enforcing 模式和 permissive 模式之间切换。使用 **setenforce** 所做的更改在重新引导后不会保留。要更改为 enforcing 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** 实用程序查看当前的 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，将独立的域设置为 permissive 模式。例如，使 *httpd_t* 域为 permissive 模式：

```
# semanage permissive -a httpd_t
```

请注意，permissive 域是一个强大的工具，它可能会破坏您系统的安全性。红帽建议谨慎使用 permissive 域，如仅在调试特定情境时使用。

[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。

第 2 章 更改 SELINUX 状态和模式

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。

2.1. SELINUX 状态和模式的更改

如 SELinux 状态和模式 中所述，SELinux 可以被启用或禁用。启用后，SELinux 有两个模式： enforcing 和 permissive。

使用 **getenforce** 或 **sestatus** 命令检查 SELinux 的运行模式。 **getenforce** 命令返回 **Enforcing**、**Permissive** 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
Memory protection checking:     actual (secure)
Max kernel policy version:      31
```

警告

当系统以 permissive 模式运行 SELinux 时，用户和进程可能会错误地标记各种文件系统对象。当禁用 SELinux 时创建的文件系统对象不会被标记。这会在将 SELinux 改为 enforcing 模式时导致问题，因为 SELinux 依赖于正确的文件系统对象标签。

为防止错误标记和未标记的文件造成问题，SELinux 在从 disabled 状态更改为 permissive 或 enforcing 模式时自动重新标记文件系统。以 root 用户身份使用 **fixfiles -F onboot** 命令创建包含 **-F** 选项的 **/.autorelabel** 文件，以确保在下次重启时重新标记文件。

在重新引导系统以进行重新标记之前，请确保系统将以 permissive 模式引导，例如使用 **enforcing=0** 内核选项。在启动 **selinux-autorelabel** 服务前，当系统包括 **systemd** 需要的未被标记的文件时，系统无法引导。如需更多信息，请参阅 RHBZ#2021835。

2.2. 切换到 PERMISSIVE 模式

使用以下步骤将 SELinux 模式永久改为 permissive。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。

先决条件

- **selinux-policy-targeted**、**libselinux-utils** 和 **policycoreutils** 软件包已安装在您的系统中。
- 未使用 **selinux=0** 或 **enforcing=0** 内核参数。

步骤

1. 在您选择的文本编辑器中打开 **/etc/selinux/config** 文件，例如：

```
# vi /etc/selinux/config
```

2. 配置 **SELINUX=permissive** 选项：

```
# 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
```

3. 重启系统：

```
# reboot
```

验证

1. 系统重启后，确认 **getenforce** 命令返回 **Permissive**：

```
$ getenforce
Permissive
```

2.3. 切换到 ENFORCING 模式

使用以下步骤将 SELinux 切换到 enforcing 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 RHEL 中，当系统最初使用 SELinux 安装时，默认启用 enforcing 模式。

先决条件

- **selinux-policy-targeted**、**libselinux-utils** 和 **policycoreutils** 软件包已安装在您的系统中。
- 未使用 **selinux=0** 或 **enforcing=0** 内核参数。

步骤

1. 在您选择的文本编辑器中打开 **/etc/selinux/config** 文件，例如：

```
# vi /etc/selinux/config
```

2. 配置 **SELINUX=enforcing** 选项：

```
# 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
```

3. 保存更改，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。

验证

1. 系统重启后，确认 **getenforce** 命令返回 **Enforcing**:

```
$ getenforce
Enforcing
```

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,USER_SELINUX_ERR -ts today
```

另外，如果安装了 **setroubleshoot-server** 软件包，请输入：

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

如果 SELinux 是活跃的，且 Audit 守护进程(**auditd**)没有在您的系统中运行，在 **dmesg** 命令输出中搜索特定的 SELinux 信息：

```
# dmesg | grep -i -e type=1300 -e type=1400
```

如需更多信息，请参阅 SELinux 故障排除。

2.4. 在之前禁用的系统中启用 SELINUX

为了避免问题，比如系统无法引导或进程失败，请在之前禁用它的系统中启用 SELinux 时按照以下步骤操作。

警告

当系统以 permissive 模式运行 SELinux 时，用户和进程可能会错误地标记各种文件系统对象。当禁用 SELinux 时创建的文件系统对象不会被标记。这会在将 SELinux 改为 enforcing 模式时导致问题，因为 SELinux 依赖于正确的文件系统对象标签。

为防止错误标记和未标记的文件造成问题，SELinux 在从 disabled 状态更改为 permissive 或 enforcing 模式时自动重新标记文件系统。

在重新引导系统以进行重新标记之前，请确保系统将以 permissive 模式引导，例如使用 **enforcing=0** 内核选项。在启动 **selinux-autorelabel** 服务前，当系统包括 **systemd** 需要的未被标记的文件时，系统无法引导。如需更多信息，请参阅 RHBZ#2021835。

步骤

1. 以 permissive 模式启用 SELinux。如需更多信息，请参阅切换到 permissive 模式。
2. 重启您的系统：

```
# reboot
```

3. 检查 SELinux 拒绝信息。如需更多信息，请参阅 识别 SELinux 拒绝操作信息。
4. 确保在下次重启时重新标记文件：

```
# fixfiles -F onboot
```

这会创建包含 **-F** 选项的 **/.autorelabel** 文件。

警告

进入 **fixfiles -F onboot** 命令前，始终切换到 permissive 模式。这可防止系统在系统包含未标记的文件时无法引导。如需更多信息，请参阅 RHBZ#2021835。

5. 如果没有拒绝的操作，切换到 enforcing 模式。如需更多信息，请参阅在引导时进入 SELinux 模式。

验证

1. 系统重启后，确认 **getenforce** 命令返回 **Enforcing**:

```
$ getenforce
Enforcing
```

注意

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。
- 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 部分。

其他资源

- SELinux 状态和模式部分涵盖了模式中的临时更改。

2.5. 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 SELinux 的好处中介绍的好处都将没有。

重要

红帽强烈建议您使用 permissive 模式，而不是永久禁用 SELinux。如需有关 permissive 模式的更多信息，请参阅切换为 permissive 模式。

先决条件

- 已安装 **grubby** 软件包：

```
$ rpm -q grubby
grubby-version
```

步骤

永久禁用 SELinux：

1. 将您的引导装载程序配置为在内核命令行中添加 **selinux=0** ：

```
$ sudo grubby --update-kernel ALL --args selinux=0
```

2. 重启您的系统：

```
$ reboot
```

验证

- 重新引导后，确认 **getenforce** 命令返回 **Disabled** ：

```
$ getenforce
Disabled
```

2.6. 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，并涉及 **/.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
```

如果文件系统中包含大量错误标记的对象，以 permissive 模式启动系统，使 autorelabel 进程成功。

其他资源

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

```
# dnf install kernel-doc
$ less /usr/share/doc/kernel-doc-4.18.0/Documentation/admin-guide/kernel-parameters.txt
```

第 3 章 管理限制和未限制的用户

下面的部分解释了 Linux 用户与 SELinux 用户的映射，描述了基本限制的用户域，并演示了将新用户映射到 SELinux 用户。

3.1. 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。

要在您的系统中查看 SELinux 用户映射，以 root 用户身份使用 **semanage login -l** 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       *
```

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

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

限制的用户受 SELinux 策略中明确定义的 SELinux 规则的限制。无限制的用户只能受到 SELinux 的最小限制。

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
   guest_u
   root
   staff_u
   sysadm_u
   system_u
   unconfined_u
   user_u
   xguest_u
```

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

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。每个受限制的用户都受到受限用户域的限制。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，受限制的用户会受到那个目标限制的域的限制。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。

3.2. SELINUX 用户功能

SELinux 策略将每个 Linux 用户映射到 SELinux 用户。这允许 Linux 用户继承 SELinux 用户的限制。

您可以通过调整策略中的布尔值来自定义 SELinux 策略中受限用户的权限。您可以使用 **semanage boolean -l** 命令确定这些布尔值的当前状态。

表 3.1. SELinux 用户的角色

User	默认角色	其他角色
unconfined_u	**unconfined_r**	**system_r**
guest_u	**guest_r**	
xguest_u	**xguest_r**	
user_u	**user_r**	
staff_u	**staff_r**	**sysadm_r**
		unconfined_r
		system_r
sysadm_u	**sysadm_r**	
root	**staff_r**	**sysadm_r**
		unconfined_r
		system_r
system_u	**system_r**	

请注意，**system_u** 是系统进程和对象的特殊用户身份，**system_r** 是关联的角色。管理员不得将这个 **system_u** 用户和 **system_r** 角色关联到 Linux 用户。另外， **unconfined_u** 和 **root** 是没有限制的用户。因此，与这些 SELinux 用户关联的角色不会包含在下表中 type 和 SELinux 角色的访问中。

每个 SELinux 角色都与 SELinux 类型对应，并提供特定的访问权限。

表 3.2. SELinux 角色的类型和访问

Role	Type	使用 X Window 系统登录	su 和 sudo	在主目录和 /tmp 中执行（默认）	Networking
unconfined_r	**unconfined_t**	是	是	是	是
guest_r	**guest_t**	否	否	是	否

Role	Type	使用 X Window 系统登录	su 和 sudo	在主目录和 /tmp 中执行（默认）	Networking
xguest_r	**xguest_t**	是	否	是	仅限 Web 浏览器（Firefox、GNOME Web）
user_r	**user_t**	是	否	是	是
staff_r	**staff_t**	是	仅 **sudo**	是	是
auditadm_r	**auditadm_t**		是	是	是
secadm_r	**secadm_t**		是	是	是
sysadm_r	**sysadm_t**	仅在 **xdm_sysadm_login** 布尔值为 **on** 时	是	是	是

- **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设置的用户 ID(setuid)应用程序（例如 **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**、**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（通过继承的用户权限）,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。
- **xguest_t** 域里的唯一网络访问 Linux 用户是 Firefox 连接到网页。
- **sysadm_u** 用户无法使用 SSH 直接登录。要为 **sysadm_u** 启用 SSH 登录，请将 **ssh_sysadm_login** 布尔值设置为 **on** ：

```
# setsebool -P ssh_sysadm_login on
```

除了已提到的 SELinux 用户之外，还有特殊的角色，可以使用 **semanage user** 命令映射到这些用户。这些角色决定了 SELinux 允许这些用户可以做什么：

- **dbadm_r** 只能管理与 Apache HTTP 服务器相关的 SELinux 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。
- **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** 只能管理与审计子系统相关的进程。

要列出所有可用角色，请输入 **seinfo -r** 命令：

```
$ seinfo -r
Roles: 14
   auditadm_r
   dbadm_r
   guest_r
   logadm_r
   nx_server_r
   object_r
   secadm_r
   staff_r
   sysadm_r
   system_r
   unconfined_r
   user_r
   webadm_r
   xguest_r
```

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

其他资源

- **seinfo(1)**, **semanage-login(8)**, 和 **xguest_selinux(8)** man pages

3.3. 添加一个新用户会自动映射到 SELINUX UNCONFINED_U 用户

下面的步骤演示了如何在系统中添加新 Linux 用户。用户会自动映射到 SELinux **unconfined_u** 用户。

先决条件

- **root** 用户运行没有限制，这在 Red Hat Enterprise Linux 中默认运行。

步骤

1. 输入以下命令创建一个名为 *example.user* 的新的 Linux 用户：

```
# useradd example.user
```

2. 要为 Linux *example.user* 用户分配密码：

```
# passwd example.user
Changing password for user example.user.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

3. 退出当前会话。

4. 以 Linux *example.user* 用户身份登录。当您登录时，**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。然后会使用这个上下文启动 Linux 用户的 shell。

验证

1. 当以 *example.user* 用户身份登录时，检查 Linux 用户的上下文：

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

其他资源

- **pam_selinux(8)** 手册页.

3.4. 以 SELINUX 限制的用户身份添加新用户

使用以下步骤为该系统添加新的 SELinux 保护用户。这个示例步骤会在创建用户帐户后马上将用户映射到 SELinux **staff_u** 用户。

先决条件

- **root** 用户运行没有限制，这在 Red Hat Enterprise Linux 中默认运行。

步骤

1. 输入以下命令创建一个名为 *example.user* 的新 Linux 用户，并将其映射到 SELinux **staff_u** 用户：

```
# useradd -Z staff_u example.user
```

2. 要为 Linux *example.user* 用户分配密码：

```
# passwd example.user
Changing password for user example.user.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

3. 退出当前会话。

4. 以 Linux *example.user* 用户身份登录。用户的 shell 使用 **staff_u** 上下文启动。

验证

1. 当以 *example.user* 用户身份登录时，检查 Linux 用户的上下文：

```
$ id -Z
uid=1000(example.user) gid=1000(example.user) groups=1000(example.user)
context=staff_u:staff_r:staff_t:s0-s0:c0.c1023
```

其他资源

- **pam_selinux(8)** 手册页.

3.5. 限制常规用户

您可以通过将系统映射到 **user_u** SELinux 用户来限制系统中的所有常规用户。

默认情况下，Red Hat Enterprise Linux 中的所有 Linux 用户（包括管理权限的用户）都会映射到无限制的 SELinux 用户 **unconfined_u**。您可以通过将用户分配给受 SELinux 限制的用户来提高系统安全性。这对遵守 V-71971 安全技术实施指南非常有用。

步骤

1. 显示 SELinux 登录记录列表。这个列表显示了 Linux 用户与 SELinux 用户的映射：

```
# 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      *
```

2. 将 __default__ user（代表所有没有显式映射的用户）映射到 **user_u** SELinux 用户：

```
# semanage login -m -s user_u -r s0 __default__
```

验证

1. 检查 __default__ 用户是否已映射到 **user_u** SELinux 用户：

```
# semanage login -l

Login Name    SELinux User   MLS/MCS Range    Service

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

2. 验证新用户的进程在 **user_u:user_r:user_t:s0** SELinux 上下文中运行。

   a. 创建一个新用户：

   ```
   # adduser example.user
   ```

   b. 定义 *example.user* 的密码：

   ```
   # passwd example.user
   ```

   c. 注销 **root**，然后以新用户身份登录。

   d. 显示用户 ID 的安全上下文：

   ```
   [example.user@localhost ~]$ id -Z
   user_u:user_r:user_t:s0
   ```

   e. 显示用户当前进程的安全上下文：

   ```
   [example.user@localhost ~]$ ps axZ
   LABEL                          PID TTY      STAT   TIME COMMAND
   ```

```
-                              1 ?        Ss     0:05 /usr/lib/systemd/systemd --switched-root --
system --deserialize 18
-                           3729 ?        S      0:00 (sd-pam)
user_u:user_r:user_t:s0        3907 ?        Ss     0:00 /usr/lib/systemd/systemd --user
-                           3911 ?        S      0:00 (sd-pam)
user_u:user_r:user_t:s0        3918 ?        S      0:00 sshd: example.user@pts/0
user_u:user_r:user_t:s0        3922 pts/0    Ss     0:00 -bash
user_u:user_r:user_dbusd_t:s0  3969 ?        Ssl    0:00 /usr/bin/dbus-daemon --session --
address=systemd: --nofork --nopidfile --systemd-activation --syslog-only
user_u:user_r:user_t:s0        3971 pts/0    R+     0:00 ps axZ
```

3.6. 通过映射到 SYSADM_U 来限制管理员

您可以通过将用户直接映射到 **sysadm_u** SELinux 用户来限制具有管理权限的用户。当用户登录时，会话会在 **sysadm_u:sysadm_r:sysadm_t** SELinux 上下文中运行。

默认情况下，Red Hat Enterprise Linux 中的所有 Linux 用户（包括管理权限的用户）都会映射到无限制的 SELinux 用户 **unconfined_u**。您可以通过将用户分配给受 SELinux 限制的用户来提高系统安全性。这对遵守 V-71971 安全技术实施指南非常有用。

先决条件

- **root** 用户运行没有限制。这是 Red Hat Enterprise Linux 的默认设置。

步骤

1. 可选：允许 **sysadm_u** 用户使用 SSH 连接到系统：

```
# setsebool -P ssh_sysadm_login on
```

2. 创建新用户，将用户添加到 **wheel** 用户组，并将用户映射到 **sysadm_u** SELinux 用户：

```
# adduser -G wheel -Z sysadm_u example.user
```

3. 可选：将现有用户映射到 **sysadm_u** SELinux 用户，并将用户添加到 **wheel** 用户组：

```
# usermod -G wheel -Z sysadm_u example.user
```

验证

1. 检查 ***example.user*** 是否已映射到 **sysadm_u** SELinux 用户：

```
# semanage login -l | grep example.user
example.user     sysadm_u    s0-s0:c0.c1023   *
```

2. 以 ***example.user*** 身份登录，例如使用 SSH，并显示用户的安全上下文：

```
[example.user@localhost ~]$ id -Z
sysadm_u:sysadm_r:sysadm_t:s0-s0:c0.c1023
```

3. 切换到 **root** 用户：

```
$ sudo -i
[sudo] password for example.user:
```

4. 验证安全性上下文是否保持不变：

```
# id -Z
sysadm_u:sysadm_r:sysadm_t:s0-s0:c0.c1023
```

5. 尝试管理任务，例如重启 **sshd** 服务：

```
# systemctl restart sshd
```

如果没有输出结果，则代表命令可以成功完成。

如果该命令没有成功完成，它会输出以下信息：

```
Failed to restart sshd.service: Access denied
See system logs and 'systemctl status sshd.service' for details.
```

3.7. 使用 SUDO 和 SYSADM_R 角色限制管理员

您可以将具有管理权限的特定用户映射到 **staff_u** SELinux 用户，并配置 **sudo**，以便用户获取 **sysadm_r** SELinux 管理员角色。这个角色允许用户在不拒绝 SELinux 的情况下执行管理任务。当用户登录时，会话会在 **staff_u:staff_r:staff_t** SELinux 上下文中运行，但当用户使用 **sudo** 进入命令时，会话会更改为 **staff_u:sysadm_r:sysadm_t** 上下文。

默认情况下，Red Hat Enterprise Linux 中的所有 Linux 用户（包括管理权限的用户）都会映射到无限制的 SELinux 用户 **unconfined_u**。您可以通过将用户分配给受 SELinux 限制的用户来提高系统安全性。这对遵守 V-71971 安全技术实施指南非常有用。

先决条件

- **root** 用户运行没有限制。这是 Red Hat Enterprise Linux 的默认设置。

步骤

1. 创建新用户，将用户添加到 **wheel** 用户组，并将用户映射到 **staff_u** SELinux 用户：

```
# adduser -G wheel -Z staff_u example.user
```

2. 可选：将现有用户映射到 **staff_u** SELinux 用户，并将用户添加到 **wheel** 用户组：

```
# usermod -G wheel -Z staff_u example.user
```

3. 要允许 *example.user* 获取 SELinux 管理员角色，请在 **/etc/sudoers.d/** 目录中创建新文件，例如：

```
# visudo -f /etc/sudoers.d/example.user
```

4. 在新文件中添加以下行：

```
example.user ALL=(ALL) TYPE=sysadm_t ROLE=sysadm_r ALL
```

验证

1. 检查 ***example.user*** 是否已映射到 **staff_u** SELinux 用户：

```
# semanage login -l | grep example.user
example.user     staff_u    s0-s0:c0.c1023   *
```

2. 以 *example.user* 身份登录，例如使用 SSH 并切换到 **root** 用户：

```
[example.user@localhost ~]$ sudo -i
[sudo] password for example.user:
```

3. 显示 **root** 安全上下文：

```
# id -Z
staff_u:sysadm_r:sysadm_t:s0-s0:c0.c1023
```

4. 尝试管理任务，例如重启 **sshd** 服务：

```
# systemctl restart sshd
```

如果没有输出结果，则代表命令可以成功完成。

如果该命令没有成功完成，它会输出以下信息：

```
Failed to restart sshd.service: Access denied
See system logs and 'systemctl status sshd.service' for details.
```

3.8. 其他资源

- **unconfined_selinux(8)**, **user_selinux(8)**, **staff_selinux(8)**, 和 **sysadm_selinux(8)** man pages
- 如何设置 SELinux 受限用户的系统

第 4 章 为使用非标准配置的应用程序和服务配置 SELINUX

当 SELinux 处于 enforcing 模式时，默认策略是目标（targeted）策略。以下小节提供有关在更改默认配置后为各种服务设置和配置 SELinux 策略的信息，比如端口、数据库位置或者用于进程的文件系统权限。

您将了解如何为非标准端口更改 SELinux 类型，识别并修复默认目录更改的不正确的标签，以及使用 SELinux 布尔值调整策略。

4.1. 在非标准配置中为 APACHE HTTP 服务器自定义 SELINUX 策略

您可以将 Apache HTTP 服务器配置为在不同端口中侦听，并在非默认目录中提供内容。要防止 SELinux 拒绝带来的后果，请按照以下步骤调整系统的 SELinux 策略。

先决条件

- 已安装 **httpd** 软件包，并将 Apache HTTP 服务器配置为侦听 TCP 端口 3131，并使用 **/var/test_www/** 目录而不是默认的 **/var/www/** 目录。
- **policycoreutils-python-utils** 和 **setroubleshoot-server** 软件包已安装在您的系统中。

步骤

1. 启动 **httpd** 服务并检查状态：

```
# systemctl start httpd
# systemctl status httpd
...
httpd[14523]: (13)Permission denied: AH00072: make_sock: could not bind to address
[::]:3131
...
systemd[1]: Failed to start The Apache HTTP Server.
...
```

2. SELinux 策略假设 **httpd** 在端口 80 上运行：

```
# semanage port -l | grep http
http_cache_port_t              tcp      8080, 8118, 8123, 10001-10010
http_cache_port_t              udp      3130
http_port_t                    tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
pegasus_http_port_t            tcp      5988
pegasus_https_port_t           tcp      5989
```

3. 更改 SELinux 类型端口 3131 使其与端口 80 匹配：

```
# semanage port -a -t http_port_t -p tcp 3131
```

4. 再次启动 **httpd** ：

```
# systemctl start httpd
```

5. 但是，内容仍无法访问：

```
# wget localhost:3131/index.html
```

```
...
HTTP request sent, awaiting response... 403 Forbidden
...
```

使用 **sealert** 工具查找原因：

```
# sealert -l "*"
...
SELinux is preventing httpd from getattr access on the file /var/test_www/html/index.html.
...
```

6. 使用 **matchpathcon** 工具比较标准 SELinux 类型以及新路径：

```
# matchpathcon /var/www/html /var/test_www/html
/var/www/html       system_u:object_r:httpd_sys_content_t:s0
/var/test_www/html  system_u:object_r:var_t:s0
```

7. 将新 **/var/test_www/html/** 内容目录的 SELinux 类型改为默认 **/var/www/html** 目录的类型：

```
# semanage fcontext -a -e /var/www /var/test_www
```

8. 递归重新标记 **/var** 目录：

```
# restorecon -Rv /var/
...
Relabeled /var/test_www/html from unconfined_u:object_r:var_t:s0 to
unconfined_u:object_r:httpd_sys_content_t:s0
Relabeled /var/test_www/html/index.html from unconfined_u:object_r:var_t:s0 to
unconfined_u:object_r:httpd_sys_content_t:s0
```

验证

1. 检查 **httpd** 服务是否正在运行：

```
# systemctl status httpd
...
Active: active (running)
...
systemd[1]: Started The Apache HTTP Server.
httpd[14888]: Server configured, listening on: port 3131
...
```

2. 验证 Apache HTTP 服务器提供的内容是否可以访问：

```
# wget localhost:3131/index.html
...
HTTP request sent, awaiting response... 200 OK
Length: 0 [text/html]
Saving to: ‘index.html’
...
```

其他资源

- The **semanage(8)**, **matchpathcon(8)**, 和 **sealert(8)** man pages.

4.2. 调整用于使用 SELINUX 布尔值共享 NFS 和 CIFS 卷的策略

您可以使用布尔值在运行时更改 SELinux 策略部分，即使您不了解 SELinux 策略的编写方式。这启用了更改，比如允许服务访问 NFS 卷而无需重新载入或者重新编译 SELinux 策略。以下流程演示了列出 SELinux 布尔值并进行配置，以实现策略中所需的更改。

在客户端的 NFS 挂载使用 NFS 卷策略定义的默认上下文标记。在 RHEL 中，此默认上下文使用 **nfs_t** 类型。另外，挂载在客户端中的 Samba 共享使用策略定义的默认上下文标记。此默认上下文使用 **cifs_t** 类型。您可以启用或禁用布尔值来控制允许哪些服务访问 **nfs_t** 和 **cifs_t** 类型。

要允许 Apache HTTP 服务器服务(**httpd**)访问和共享 NFS 和 CIFS 卷，请执行以下步骤：

先决条件

- （可选）安装 **selinux-policy-devel** 软件包，以获取 **semanage boolean -l** 命令的输出中 SELinux 布尔值的更清晰和详细描述。

步骤

1. 识别与 NFS、CIFS 和 Apache 相关的 SELinux 布尔值：

```
# semanage boolean -l | grep 'nfs\|cifs' | grep httpd
httpd_use_cifs                 (off  ,  off)  Allow httpd to access cifs file systems
httpd_use_nfs                  (off  ,  off)  Allow httpd to access nfs file systems
```

2. 列出布尔值的当前状态：

```
$ getsebool -a | grep 'nfs\|cifs' | grep httpd
httpd_use_cifs --> off
httpd_use_nfs --> off
```

3. 启用指定的布尔值：

```
# setsebool httpd_use_nfs on
# setsebool httpd_use_cifs on
```

注意

使用 **setsebool** 和 **-P** 选项，使更改在重新启动后仍然有效。**setsebool -P** 命令需要重建整个策略，且可能需要一些时间，具体取决于您的配置。

验证

1. 检查布尔值为 **on**：

```
$ getsebool -a | grep 'nfs\|cifs' | grep httpd
httpd_use_cifs --> on
httpd_use_nfs --> on
```

其他资源

- **semanage-boolean(8)**, **sepolicy-booleans(8)**, **getsebool(8)**, **setsebool(8)**, **booleans(5)**, 和 **booleans(8)** man pages

4.3. 其他资源

- 故障排除与 SELinux 相关的问题

第 5 章 故障排除与 SELINUX 相关的问题

如果您计划在之前禁用 SELinux 的系统中启用 SELinux，或者您以非标准配置运行服务，您可能需要排除 SELinux 可能会阻断的问题。请注意，在多数情况下，SELinux 拒绝通常代表存在错误的配置。

5.1. 识别 SELINUX 拒绝

只执行此流程中的必要步骤；在大多数情况下，您只需要执行第 1 步。

步骤

1. 当您的情况被 SELinux 阻止时，**/var/log/audit/audit.log** 文件是第一个检查有关拒绝的更多信息。要查询审计日志，请使用 **ausearch** 工具。因为 SELinux 决策（如允许或禁止访问）已被缓存，且这个缓存被称为 Access Vector Cache（AVC），所以对消息类型参数使用 **AVC** 和 **USER_AVC** 值，例如：

   ```
   # ausearch -m AVC,USER_AVC,SELINUX_ERR,USER_SELINUX_ERR -ts recent
   ```

   如果没有匹配项，请检查 audit 守护进程是否正在运行。如果没有，请在启动 **auditd** 后重复拒绝的场景，然后再次检查审计日志。

2. 如果 **auditd** 正在运行，但 **ausearch** 的输出中没有匹配项，请检查 **systemd** Journal 提供的消息：

   ```
   # journalctl -t setroubleshoot
   ```

3. 如果 SELinux 处于活跃状态，且 Audit 守护进程没有在您的系统中运行，在 **dmesg** 命令的输出中搜索特定的 SELinux 信息：

   ```
   # dmesg | grep -i -e type=1300 -e type=1400
   ```

4. 即使进行了前面的三个检查后，您仍可能找不到任何结果。在这种情况下，因为 **dontaudit** 规则，可以静默 AVC 拒绝。
   要临时禁用 **dontaudit** 规则，请记录所有拒绝：

   ```
   # semodule -DB
   ```

   在重新运行拒绝的场景并使用前面的步骤查找拒绝信息后，以下命令会在策略中再次启用 **dontaudit** 规则：

   ```
   # semodule -B
   ```

5. 如果您应用了前面所有四个步骤，这个问题仍然无法识别，请考虑 SELinux 是否真正阻止了您的场景：

   - 切换到 permissive 模式：

     ```
     # setenforce 0
     $ getenforce
     Permissive
     ```

   - 重复您的场景。

如果问题仍然存在，则代表 SELinux 以外的系统阻断了您的场景。

5.2. 分析 SELINUX 拒绝信息

在确认 SELinux 会阻止您的场景后，可能需要在进行修复前分析根本原因。

先决条件

- **policycoreutils-python-utils** 和 **setroubleshoot-server** 软件包已安装在您的系统中。

步骤

1. 使用 **sealert** 命令列出有关日志拒绝的详情，例如：

```
$ sealert -l "*"
SELinux is preventing /usr/bin/passwd from write access on the file
/root/test.

*****  Plugin leaks (86.2 confidence) suggests ******************************

If you want to ignore passwd trying to write access the test file,
because you believe it should not need this access.
Then you should report this as a bug.
You can generate a local policy module to dontaudit this access.
Do
# ausearch -x /usr/bin/passwd --raw | audit2allow -D -M my-passwd
# semodule -X 300 -i my-passwd.pp

*****  Plugin catchall (14.7 confidence) suggests **************************

...

Raw Audit Messages
type=AVC msg=audit(1553609555.619:127): avc:  denied  { write } for
pid=4097 comm="passwd" path="/root/test" dev="dm-0" ino=17142697
scontext=unconfined_u:unconfined_r:passwd_t:s0-s0:c0.c1023
tcontext=unconfined_u:object_r:admin_home_t:s0 tclass=file permissive=0

...

Hash: passwd,passwd_t,admin_home_t,file,write
```

2. 如果上一步中的输出没有包含清晰的建议：

   - 启用全路径审核查看访问对象的完整路径，并让其他 Linux Audit 事件字段可见：

     ```
     # auditctl -w /etc/shadow -p w -k shadow-write
     ```

   - 清除 **setroubleshoot** 缓存：

     ```
     # rm -f /var/lib/setroubleshoot/setroubleshoot.xml
     ```

   - 重现问题。

- 重复步骤 1。
  完成这个过程后，禁用全路径审核：

```
# auditctl -W /etc/shadow -p w -k shadow-write
```

3. 如果 **sealert** 只返回 **catchall** 建议，或者建议使用 **audit2allow** 工具添加新规则，请将您的问题与 审计日志中 SELinux 拒绝中列出的 示例匹配。

其他资源

- **sealert(8)** 手册页

5.3. 修复分析的 SELINUX 拒绝问题

在大多数情况下，**sealert** 工具提供的建议将为您提供有关如何修复与 SELinux 策略相关的问题的正确指导。有关如何使用 **sealert** 分析 SELinux 拒绝的信息，请参阅 分析 SELinux 拒绝信息。

当工具建议使用 **audit2allow** 工具进行配置更改时，请小心。当您看到 SELinux 拒绝时，您不应该使用 **audit2allow** 来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。

标记问题

标记问题的常见原因是，当服务使用非标准目录时。例如，管理员可能想要使用 **/srv/myweb/**，而不是在网站中使用 **/var/www/html/**。在 Red Hat Enterprise Linux 中， **/srv** 目录使用 **var_t** 类型进行标记。在 **/srv** 中创建的文件和目录继承这个类型。另外，顶层目录中新创建的对象（如 **/myserver** ）可以使用 **default_t** 类型进行标记。SELinux 会阻止 Apache HTTP 服务器(**httpd**)访问这两个类型。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** 命令会将 **/srv/myweb/** 目录以及其下的所有文件和目录添加到 SELinux file-context 配置的上下文。**semanage** 程序不会更改上下文。以 root 用户身份，使用 **restorecon** 程序应用更改：

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

不正确的上下文

matchpathcon 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** 类型进行标识。这种类型用于用户主目录中的文件。使用 **mv** 命令从您的主目录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
```

以非标准方式配置受限应用程序

服务可以以多种方式运行。要考虑这一点，您需要指定如何运行您的服务。您可以通过 SELinux 布尔值达到此目的，允许在运行时更改 SELinux 策略的部分。这启用了更改，比如允许服务访问 NFS 卷而无需重新载入或者重新编译 SELinux 策略。另外，在非默认端口号中运行服务需要使用 **semanage** 命令来更新策略配置。

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

```
# setsebool -P httpd_can_network_connect_db on
```

请注意，**-P** 选项可使系统重启后设置具有持久性。

如果特定服务无法访问，请使用 **getsebool** 和 **grep** 实用程序查看是否有布尔值是否可用于访问。例如，使用 **getsebool -a | grep ftp** 命令搜索 FTP 相关布尔值：

```
$ 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
```

要获得布尔值列表并找出是否启用或禁用它们，请使用 **getsebool -a** 命令。要获得包括布尔值的列表，并找出它们是否启用或禁用，请安装 **selinux-policy-devel** 软件包并以 root 用户身份使用 **semanage boolean -l** 命令。

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 端口类型定义了 Apache HTTP 服务器可以侦听的端口，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应这一点，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)
```

类似于以下内容的 SELinux 拒绝消息会记录到 **/var/log/audit/audit.log** ：

```
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** 侦听没有为 **http_port_t** 端口类型列出的端口，请使用 **semanage port** 命令为端口分配不同的标签：

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

a 选项添加新的记录； **-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。

个别情况、演变或损坏的应用程序以及被破坏的系统

应用程序可能会包含程序漏洞，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，当一个 PostgreSQL 的新版本发布后，它可能会执行一些当前策略无法处理的操作，从而导致访问被拒绝，即使应该允许访问。

对于这样的情形，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。您可以在 Red Hat Bugzilla 中报告 SELinux 策略中缺少的规则。对于 Red Hat Enterprise Linux 9，针对 **Red Hat Enterprise Linux 9** 产品创建错误，然后选择 **selinux-policy** 组件。在此类程序漏洞报告中包含 **audit2allow -w -a** 和 **audit2allow -a** 命令的输出。

如果应用程序请求主要的安全特权，这可能代表，应用程序可能已被破坏。使用入侵检测工具检查此类行为。

红帽客户门户网站中的 Solution Engine 也以文章的形式提供了相关的指导信息。它包括了您遇到的相同或非常类似的问题的解决方案。选择相关的产品和版本，并使用与 SELinux 相关的关键字，如 *selinux* 或 *avc*，以及您阻断的服务或应用程序的名称，例如： **selinux samba**。

5.4. 审计日志中的 SELINUX 拒绝

Linux Audit 系统默认将日志条目存储在 **/var/log/audit/audit.log** 文件中。

要仅列出与 SELinux 相关的记录，请使用 **ausearch** 命令，并将 message type 参数设置为 **AVC** 和 **AVC_USER**，例如：

```
# ausearch -m AVC,USER_AVC,SELINUX_ERR,USER_SELINUX_ERR
```

审计日志文件中的 SELinux 拒绝条目类似如下：

-

```
type=AVC msg=audit(1395177286.929:1638): avc:  denied  { read } for  pid=6591 comm="httpd"
name="webpages" dev="0:37" ino=2112 scontext=system_u:system_r:httpd_t:s0
tcontext=system_u:object_r:nfs_t:s0 tclass=dir
```

这个条目最重要的部分是：

- **avc: denied** - SELinux 执行的操作，并在 AVC 中记录
- **{ read }** - 被拒绝的操作
- **pid=6591** - 试图执行被拒绝操作的主体的进程识别符
- **comm="httpd"** - 用于调用分析进程的命令名称
- **httpd_t** - 进程的 SELinux 类型
- **nfs_t** - 受进程操作影响的对象的 SELinux 类型
- **tclass=dir** - 目标对象类

以前的日志条目可转换为：

*SELinux 拒绝 PID 为 6591、以及从带有 **nfs_t** 类型的目录进行读取的 **httpd_t** 类型的 **httpd** 进程*

当 Apache HTTP 服务器试图访问使用 Samba 套件类型标记的目录时，会出现以下 SELinux 拒绝信息：

```
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
```

- **{ getattr }** - **getattr** 条目表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。SELinux 会拒绝这个操作，因为进程会访问该文件，且没有适当的标签。通常的权限包括 **getattr**, **read**, 和 **write**。
- **path="/var/www/html/file1"** - 该进程试图访问的对象（目标）的路径。
- **scontext="unconfined_u:system_r:httpd_t:s0"** - 试图拒绝操作的进程（源）的 SELinux 上下文。在这种情况下，它是 Apache HTTP 服务器的 SELinux 上下文，它使用 **httpd_t** 类型运行。
- **tcontext="unconfined_u:object_r:samba_share_t:s0"** - 试图访问的对象（目标）的 SELinux 上下文。在这种情况下，它是 **file1** 的 SELinux 上下文。

这个 SELinux 拒绝信息可以被解释为：

*SELinux 拒绝了 PID 为 2465 的 **httpd** 进程访问带有 **samba_share_t** 类型的 **/var/www/html/file1** 文件。除非有其他配置，在 **httpd_t** 域中运行的进程无法访问该文件。*

其他资源

- **auditd(8)** 和 **ausearch(8)** man page

5.5. 其他资源

- CLI 中的基本 SELinux 故障排除
- SELinux 试图告诉我什么？SELinux 错误的 4 个关键原因

第 6 章 使用多级别安全（MLS）

多级别安全（Multi-Level Security，简称 MLS）策略使用许可*级别*的概念，这个概念首先由美国国防人员设计。MLS 满足一组非常严格的安全要求，这基于在严格控制的环境中管理的信息（如军事）。

使用 MLS 非常复杂，不适合于一般用例场景。

6.1. 多级别安全（MLS）

多级别安全(MLS)技术使用信息安全级别将数据分为分级分类，例如：

- [Low] 非保密
- [low] 保密
- [high] 机密
- [Highest] 顶端机密

默认情况下，MLS SELinux 策略使用 16 敏感度级别：

- **s0** 是最敏感的。
- **s15** 是最敏感的。

MLS 使用特定的术语来解决敏感度级别：

- 用户和进程称为 **主题（subjects）**，其敏感度级别被称为 **安全权限（clearance）**。
- 系统文件、设备和其他被动组件称为 **对象（objects）**，其敏感度级别被称为**安全级别（classification）**。

为了实施 MLS，SELinux 使用 **Bell-La Padula Model** (BLP)模型。这个模型根据附加到每个主体和对象的标签指定系统中如何进行信息流。

BLP 的基本原则是"**不能从上面读取，不能向下面写入**"。这意味着用户只能读取自己的敏感度级别及更低级别的文件，数据只能从较低级别流入到更高级别，且不会从高级别流向低级别。

MLS SELinux 策略（在 RHEL 上实现 MLS）会应用名为 **Bell-La Padula 的修改原则，其写入相等**。这意味着，用户可以在自己的敏感度级别和更低级别中读取文件，但只能在自己的级别上写入。例如，这可以防止不明确的用户将内容写入 top-secret 文件中。

例如，默认情况下，具有明确级别 **s2** 的用户：

- 可以读取具有敏感度级别 **s0**、 **s1** 和 **s2** 的文件。
- 无法读取具有敏感度级别 **s3** 及更高等级的文件。
- 可以使用精确的 **s2** 来修改文件。
- 无法修改比 **s2** 不同的敏感度级别的文件。

注意

安全管理员可以通过修改系统的 SELinux 策略来调整此行为。例如，他们可以允许用户以较低级别修改文件，这会增加文件的敏感度级别。

实际上，用户通常会被分配到一系列清晰的级别，如 **s1-s2**。用户可以读取级别低于用户的最大文件，并写入该范围内的任何文件。

例如，默认情况下，用户有一个明确范围的 **s1-s2** ：

- 可以读取具有敏感度级别 **s0** 和 **s1** 的文件。
- 不能读取级别 **s2** 及更高等级的文件。
- 可以修改具有敏感度级别 **s1** 的文件。
- 无法修改比 **s1** 不同的敏感度级别的文件。
- 可以将自己的明确级别更改为 **s2**。

MLS 环境中非特权用户的安全上下文是：

```
user_u:user_r:user_t:s1
```

其中：

user_u
是 SELinux 用户。

user_r
是 SELinux 角色。

user_t
是 SELinux 类型。

s1
是 MLS 敏感度级别的范围。

系统始终将 MLS 访问规则与传统的文件访问权限合并。例如，如果具有安全级别 "Secret" 的用户使用 Discretionary Access Control(DAC)来阻止其他用户对文件的访问，即使"Top Secret"用户无法访问该文件。较高的安全许可不会自动允许用户浏览整个文件系统。

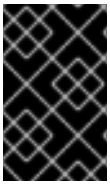
拥有顶级别的用户不会自动获得多级系统的管理权限。虽然他们可能对系统的所有敏感信息的访问权限，但这与拥有管理权限不同。

此外，管理权限不提供对敏感信息的访问权限。例如，即使某人以 **root** 身份登录，它们仍然无法读取 top-secret 信息。

您可以使用类别进一步调整 MLS 系统中的访问。使用多类别安全性(MCS)，您可以定义项目或部门等类别，用户只能访问为其分配的类别中的文件。如需更多信息，请参阅 使用多类别 Security(MCS)获取数据保密性。

6.2. MLS 中的 SELINUX 角色

SELinux 策略将每个 Linux 用户映射到 SELinux 用户。这允许 Linux 用户继承 SELinux 用户的限制。



重要

MLS 策略不包含 **unconfined** 模块，包括无限制的用户、类型和角色。因此，用户将不受限制（包括 **root**）的用户无法访问每个对象，并执行他们在目标策略中可以进行的每个操作。

您可以通过调整策略中的布尔值来自定义 SELinux 策略中受限用户的权限。您可以使用 **semanage boolean -l** 命令确定这些布尔值的当前状态。

表 6.1. MLS 中 SELinux 用户的角色

User	默认角色	其他角色
guest_u	**guest_r**	
xguest_u	**xguest_r**	
user_u	**user_r**	
staff_u	**staff_r**	**auditadm_r**
		secadm_r
		sysadm_r
		staff_r
sysadm_u	**sysadm_r**	
root	**staff_r**	**auditadm_r**
		secadm_r
		sysadm_r
		system_r
system_u	**system_r**	

请注意，**system_u** 是系统进程和对象的特殊用户身份，**system_r** 是关联的角色。管理员不得将这个 **system_u** 用户和 **system_r** 角色关联到 Linux 用户。另外，**unconfined_u** 和 **root** 是没有限制的用户。因此，与这些 SELinux 用户关联的角色不会包含在下表中 type 和 SELinux 角色的访问中。

每个 SELinux 角色都与 SELinux 类型对应，并提供特定的访问权限。

表 6.2. MLS 中 SELinux 角色的类型和访问

Role	Type	使用 X 窗口系统登录	su 和 sudo	在主目录和 /tmp 中执行（默认）	Networking
guest_r	**guest_t**	否	否	是	否

Role	Type	使用 X 窗口系统登录	su 和 sudo	在主目录和 /tmp 中执行（默认）	Networking
xguest_r	**xguest_t**	是	否	是	仅限 Web 浏览器（Firefox、GNOME Web）
user_r	**user_t**	是	否	是	是
staff_r	**staff_t**	是	仅 **sudo**	是	是
auditadm_r	**auditadm_t**		是	是	是
secadm_r	**secadm_t**		是	是	是
sysadm_r	**sysadm_t**	仅在 **xdm_sysadm_login** 布尔值为 **on** 时	是	是	是

- 默认情况下，syssys_r 角色具有 **secadm_r** 角色的权限，这意味着具有 **sysadm_r** 角色的用户可以管理安全策略。如果这没有与您的用例对应，您可以通过在策略中禁用 **sysadm_secadm** 模块来分离这两个角色。如需更多信息，请参阅 MLS 中的安全管理 9 月进行系统管理
- 非登录角色 **dbadm_r**、**logadm_r** 和 **webadm_r** 可用于管理任务的子集。默认情况下，这些角色不与任何 SELinux 用户关联。

6.3. 将 SELINUX 策略切换到 MLS

使用以下步骤将 SELinux 策略从 targeted 切换到多级别安全（MLS）。

重要

红帽不推荐在运行 X 窗口系统的系统中使用 MLS 策略。另外，当您使用 MLS 标签重新标记文件系统时，系统可能会阻止受限制的域访问，这会阻止您的系统正确启动。因此请确定您在重新标记文件前将 SELinux 切换到 permissive 模式。在大多数系统中，您会看到在切换到 MLS 后出现了很多 SELinux 拒绝信息，且其中很多都不容易修复。

步骤

1. 安装 **selinux-policy-mls** 软件包：

```
# dnf install selinux-policy-mls
```

2. 在您选择的文本编辑器中打开 **/etc/selinux/config** 文件，例如：

```
# vi /etc/selinux/config
```

3. 将 SELinux 模式从 enforcing 改为 permissive，并从 targeted 策略切换到 MLS：

```
SELINUX=permissive
SELINUXTYPE=mls
```

保存更改，退出编辑器。

4. 在启用 MLS 策略前，您必须使用 MLS 标签重新标记文件系统中的每个文件：

```
# fixfiles -F onboot
System will relabel on next boot
```

5. 重启系统：

```
# reboot
```

6. 检查 SELinux 拒绝信息：

```
# ausearch -m AVC,USER_AVC,SELINUX_ERR,USER_SELINUX_ERR -ts recent -i
```

因为前面的命令没有涵盖所有情况，请参阅 SELinux 故障排除中有关识别、分析以及修复 SELinux 拒绝的指导。

7. 在确定您的系统中没有与 SELinux 相关的问题后，通过更改 **/etc/selinux/config** 中的对应选项将 SELinux 切换回 enforcing 模式：

```
SELINUX=enforcing
```

8. 重启系统：

```
# reboot
```

重要

如果您的系统没有启动，或者您无法在切换到 MLS 后登录，请将 **enforcing=0** 参数添加到内核命令行。如需更多信息，请参阅在引导时更改 SELinux 模式。

另请注意，在 MLS 中，以 **root** 用户身份通过 SSH 登录映射到 **sysadm_r** SELinux 角色，它与作为 **staff_r** 中的 **root** 登录不同。在 MLS 中首次启动系统前，请考虑通过将 **ssh_sysadm_login** SELinux 布尔值设置为 **1** 来允许以 **sysadm_r** 身份登录 SSH 登录。要稍后才启用 **ssh_sysadm_login**（已在 MLS 中），您需要在 **staff_r** 中以 **root** 身份登陆，使用 **newrole -r sysadm_r** 命令切换到 **sysadm_r** 中的 **root**，然后将布尔值设置为 **1**。

验证

1. 验证 SELinux 是否在 enforcing 模式下运行：

```
# getenforce
Enforcing
```

2. 检查 SELinux 的状态是否返回 **mls** 值：

```
# sestatus | grep mls
Loaded policy name:             mls
```

其他资源

- The **fixfiles(8)**, **setsebool(8)**, 和 **ssh_selinux(8)** man pages.

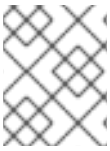
6.4. 在 MLS 中建立用户明确

将 SELinux 策略切换到 MLS 后，必须通过将 SELinux 策略映射到受限的 SELinux 用户来为用户分配安全清晰级别。默认情况下，具有给定安全缺陷的用户：

- 不能读取具有更高敏感度级别的对象。
- 无法写入到不同敏感度级别的对象。

先决条件

- SELinux 策略被设置为 **mls**。
- SELinux 模式设置为 **enforcing**。
- 已安装 **policycoreutils-python-utils** 软件包。
- 分配给 SELinux 受限用户的用户：
  - 对于非授权用户，分配给 **user_u** （以下流程中的*example_user* ）。
  - 对于特权用户，分配给 **staff_u** （以下流程中的 *staff* ）。



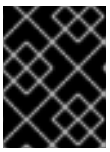
注意

确保 MLS 策略处于活动状态时已创建该用户。MLS 中无法使用在其他 SELinux 策略中创建的用户。

步骤

1. 可选：要防止向 SELinux 策略添加错误，切换到 **permissive** SELinux 模式，这有助于进行故障排除：

```
# setenforce 0
```



重要

在 permissive 模式中，SELinux 不强制执行活跃策略，而是只记录 Access Vector Cache(AVC)消息，然后可用于故障排除和调试。

2. 为 **staff_u** SELinux 用户定义清晰的范围。例如，这个命令会将安全权限范围设置为 **s1** 到 **s15**，**s1** 是默认的安全权限级别：

```
# semanage user -m -L s1 -r s1-s15 _staff_u
```

3. 为用户主目录生成 SELinux 文件上下文配置条目：

```
# genhomedircon
```

4. 将文件安全上下文恢复到默认值：

```
# restorecon -R -F -v /home/
Relabeled /home/staff from staff_u:object_r:user_home_dir_t:s0 to
staff_u:object_r:user_home_dir_t:s1
Relabeled /home/staff/.bash_logout from staff_u:object_r:user_home_t:s0 to
staff_u:object_r:user_home_t:s1
Relabeled /home/staff/.bash_profile from staff_u:object_r:user_home_t:s0 to
staff_u:object_r:user_home_t:s1
Relabeled /home/staff/.bashrc from staff_u:object_r:user_home_t:s0 to
staff_u:object_r:user_home_t:s1
```

5. 为用户分配安全权限级别：

```
# semanage login -m -r s1 example_user
```

其中 ***s1*** 是分配给用户的安全权限级别。

6. 将用户的主目录重新标记到用户的明确级别：

```
# chcon -R -l s1 /home/example_user
```

7. 可选：如果您之前切换到 **permissive** SELinux 模式，并在验证所有内容可以正常工作后，切换回 **enforcing** SELinux 模式：

```
# setenforce 1
```

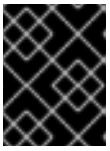
验证步骤

1. 验证用户是否已映射到正确的 SELinux 用户，并分配了正确的级别：

```
# semanage login -l
Login Name      SELinux User         MLS/MCS Range        Service
__default__     user_u               s0-s0                *
example_user    user_u               s1                   *
...
```

2. 以 MLS 内的用户身份登录。

3. 验证用户的安全级别是否正常工作：



重要

如果配置不正确，您用于验证的文件不应包含任何敏感信息，并且用户实际上可以访问未经授权的文件。

a. 验证用户无法读取具有更高级别敏感性的文件。

b. 验证用户可以写入具有相同敏感性的文件。

c. 验证用户可以读取具有较低级别的敏感性的文件。

其他资源

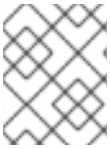
- 第 6.3 节 “将 SELinux 策略切换到 MLS” .
- 第 3.4 节 “以 SELinux 限制的用户身份添加新用户” .
- 第 2 章 *更改 SELinux 状态和模式* .
- 第 5 章 *故障排除与 SELinux 相关的问题* .
- CLI 基本故障排除 知识库文章。

6.5. 在 MLS 中定义的安全范围内更改用户清晰的级别

作为多级安全性(MLS)中的用户，您可以在管理员分配给您的范围内更改当前的明确级别。您永远不会超过范围的上限，或低于您范围内的下级。例如，您可以修改较低的敏感性文件，而不提高其敏感度级别最高的级别。

例如，作为分配给范围 **s1-s3** 的用户：

- 您可以切换到级别 **s1**、**s 2** 和 **s3**。
- 您可以切换到范围 **s1-s2** 和 **s2-s3**。
- 您不能切换到范围 **s0-s3** 或 **s1-s4**。



注意

切换到其他级别会打开具有不同清晰的新 shell。这意味着您无法返回原始清晰级别与减少它的方式相同。但是，通过输入 **exit** 总是可以返回到之前的 shell。

先决条件

- SELinux 策略设置为 **mls**。
- SELinux 模式 **设置为强制模式**。
- 您可以作为分配给一系列 MLS clearance 级别的用户身份登录。

步骤

1. 以 用户身份登录安全终端。

注意

安全终端在 **/etc/selinux/mls/contexts/securetty_types** 文件中定义。默认情况下，控制台是一个安全终端，但 SSH 不是。

2. 检查当前用户的安全上下文：

```
$ id -Z
user_u:user_r:user_t:s0-s2
```

在本例中，该用户被分配给 **user_u** SELinux 用户、**user_r** 角色、**user_t** 类型，以及 MLS 安全范围 **s0-s2**。

3. 检查当前用户的安全上下文：

```
$ id -Z
user_u:user_r:user_t:s1-s2
```

4. 切换到用户清晰范围内的不同安全清晰范围：

```
$ newrole -l s1
```

您可以切换到其最大值较低或等于您的分配范围的任何范围。输入单个级别范围会更改所分配范围的较低限制。例如，输入 **newrole -l s1** 作为具有 **s0-s2** 范围的用户，相当于输入 **newrole -l s1-s2**。

验证

1. 显示当前用户的安全上下文：

```
$ id -Z
user_u:user_r:user_t:s1-s2
```

2. 通过终止当前 shell，返回到之前带有原始范围的 shell：

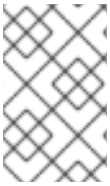
```
$ exit
```

其他资源

- using-multi-level-security-mls_using-selinux.xml
- **newrole(1)** man page
- **securetty_types(5)** man page

6.6. 在 MLS 中增大文件敏感级别

默认情况下，多级别安全(MLS)用户无法增加文件敏感度级别。但是，安全管理员(**secadm_r**)可以更改此默认行为，通过向系统 SELinux 策略添加本地模块 **mlsfilewrite** 来增加文件的敏感度。然后，分配至策略模块中定义的 SELinux 类型的用户可以通过修改 文件来增加文件分类级别。每当用户修改文件时，文件的敏感度级别都会提高用户当前安全范围的值。



注意

安全管理员以分配给 **secadm_r** 角色的用户身份登录时，可以使用 **chcon -l *s0 /path/to/file*命令更改文件的安全级别**。如需更多信息，请参阅 第 6.7 节 “在 MLS 中更改文件敏感度”。

先决条件

- SELinux 策略被设置为 **mls**。
- SELinux 模式设置为 **enforcing**。
- 已安装 **policycoreutils-python-utils** 软件包。

- **mlsfilewrite** local 模块安装在 SELinux MLS 策略中。
- 您以 MLS 中的用户登录，如下：
  - 分配给定义的安全范围。本例演示了一个安全性范围为 **s0-s2** 的用户。
  - 分配给 **mlsfilewrite** 模块中定义的同一 SELinux 类型。这个示例需要 （**typeattributeset mlsfilewrite(user_t)**） 模块。

步骤

1. 可选：显示当前用户的安全上下文：

```
$ id -Z
user_u:user_r:user_t:s0-s2
```

2. 将用户的 MLS clearance 范围的较低级别改为您要分配给该文件的级别：

```
$ newrole -l s1-s2
```

3. 可选：显示当前用户的安全上下文：

```
$ id -Z
user_u:user_r:user_t:s1-s2
```

4. 可选：显示文件的安全上下文：

```
$ ls -Z /path/to/file
user_u:object_r:user_home_t:s0 /path/to/file
```

5. 通过修改该文件，将文件的敏感度级别改为用户清晰的较低级别：

```
$ touch /path/to/file
```

重要

如果系统上使用了 **restorecon** 命令，则分类级别恢复为默认值。

6. 可选：退出 shell 以返回到用户的前一个安全范围：

```
$ exit
```

验证

- 显示文件的安全上下文：

```
$ ls -Z /path/to/file
user_u:object_r:user_home_t:s1 /path/to/file
```

其他资源

- 第 6.10 节 “允许 MLS 用户在较低级别上编辑文件”.

6.7. 在 MLS 中更改文件敏感度

在 MLS SELinux 策略中，用户只能修改自己的敏感度级别的文件。这是为了防止以较低明确的级别向用户公开任何高度敏感信息，同时防止不明确的用户创建高敏感文件。不过，管理员可以手动增加文件的分类，例如要在更高级别处理该文件。

先决条件

- SELinux 策略设置为 **mls**。
- SELinux 模式被设置为 enforcing。
- 您有安全管理权限，这意味着您要分配给其中之一：
  - **secadm_r** 角色。
  - 如果启用了 **sysadm_secadm** 模块，进入 **sysadm_r** 角色。**sysadm_secadm** 模块默认启用。
- 已安装 **policycoreutils-python-utils** 软件包。
- 分配给任何安全权限级别的用户。如需更多信息，请参阅 MLS 中建立用户清除级别。
  在本例中，***User1*** 已有 **s1** 级别的权限。
- 分配了安全级别的、您有全访问的文件。
  在本例中，***/path/to/file*** 具有级别 **s1** 权限。

步骤

1. 检查该文件的安全级别：

```
# ls -lZ /path/to/file
-rw-r-----. 1 User1 User1 user_u:object_r:user_home_t:s1 0 12. Feb 10:43 /path/to/file
```

2. 更改文件的默认安全级别：

```
# semanage fcontext -a -r s2 /path/to/file
```

3. 强制重新标记文件的 SELinux 上下文：

```
# restorecon -F -v /path/to/file
Relabeled /path/to/file from user_u:object_r:user_home_t:s1 to
user_u:object_r:user_home_t:s2
```

验证

1. 检查该文件的安全级别：

```
# ls -lZ /path/to/file
-rw-r-----. 1 User1 User1 user_u:object_r:user_home_t:s2 0 12. Feb 10:53 /path/to/file
```

2. 可选：验证具有较低级别权限的用户是否无法读取该文件：

```
$ cat /path/to/file
cat: file: Permission denied
```

其他资源

- 第 6.4 节 “在 MLS 中建立用户明确” .

6.8. 在 MLS 中将系统管理与安全管理分离

默认情况下，syssys_r 角色具有 **secadm_r** 角色的权限，这意味着具有 **sysadm_r** 角色的用户可以管理安全策略。如果需要对安全授权进行更多控制，您可以通过将 Linux 用户分配给 **secadm_r** 角色并在 SELinux 策略中禁用 **sysadm_secadm** 模块将系统管理与安全管理分开。

先决条件

- SELinux 策略被设置为 **mls**。
- SELinux 模式设置为 **enforcing**。
- 已安装 **policycoreutils-python-utils** 软件包。
- 分配给 **secadm_r** 角色的 Linux 用户：
  - 该用户被分配给 **staff_u** SELinux 用户
  - 定义了此用户的密码。

警告

确保您可以以用户 身份登录，这将分配给 **secadm** 角色。如果不能，您可以防止以后修改系统的 SELinux 策略。

步骤

1. 在 **/etc/ sudoers.d** 目录中为用户创建新的 sudoers 文件：

   ```
   # visudo -f /etc/sudoers.d/<sec_adm_user>
   ```

   为保持 **sudoers** 文件的组织，请 ***<sec_adm_user>*** 替换为将分配给 **secadm** 角色的 Linux 用户。

2. 在 **/etc/sudoers.d/*<sec_adm_user>*** 文件中添加以下内容：

   ```
   <sec_adm_user> ALL=(ALL) TYPE=secadm_t ROLE=secadm_r ALL
   ```

   此行授权所有主机上的 ***<secadmuser>*** 用户执行所有命令，并默认将用户映射到 **secadm** SELinux 类型和角色。

3. 以 *<sec_adm_user>* 用户身份登录：

注意

为确保 SELinux 上下文（由 SELinux 用户、角色和类型组成），使用 **ssh**、控制台或 **xdm** 登陆。**su** 和 **sudo** 等其他方法无法更改整个 SELinux 上下文。

4. 验证用户的安全上下文：

```
$ id
uid=1000(<sec_adm_user>) gid=1000(<sec_adm_user>) groups=1000(<sec_adm_user>)
context=staff_u:staff_r:staff_t:s0-s15:c0.c1023
```

5. 为 root 用户运行交互式 shell：

```
$ sudo -i
[sudo] password for <sec_adm_user>:
```

6. 验证当前用户的安全上下文：

```
# id
uid=0(root) gid=0(root) groups=0(root) context=staff_u:secadm_r:secadm_t:s0-s15:c0.c1023
```

7. 从策略中禁用 **sysadm_secadm** 模块：

```
# semodule -d sysadm_secadm
```

重要

使用 **semodule -d** 命令，而不是使用 **semodule -r** 命令删除系统策略模块。**semodule -r** 命令从您的系统存储中删除模块，这意味着无法重新安装 **selinux-policy-mls** 软件包。

验证

1. 作为分配给 **secadm** 角色的用户，并在 root 用户的交互式 shell 中验证您可以访问安全策略数据：

```
# seinfo -xt secadm_t

Types: 1
   type secadm_t, can_relabelto_shadow_passwords, (...) userdomain;
```

2. 从 root shell 注销：

```
# logout
```

3. 登出 ***<sec_adm_user>*** 用户：

```
$ logout
Connection to localhost closed.
```

4. 显示当前安全上下文：

```
# id
uid=0(root) gid=0(root) groups=0(root) context=root:sysadm_r:sysadm_t:s0-s15:c0.c1023
```

5. 尝试启用 **sysadm_secadm** 模块。该命令应该失败：

```
# semodule -e sysadm_secadm
SELinux:  Could not load policy file /etc/selinux/mls/policy/policy.31:  Permission denied
/sbin/load_policy:  Can't load policy:  Permission denied
libsemanage.semanage_reload_policy: load_policy returned error code 2. (No such file or
directory).
SELinux:  Could not load policy file /etc/selinux/mls/policy/policy.31:  Permission denied
/sbin/load_policy:  Can't load policy:  Permission denied
libsemanage.semanage_reload_policy: load_policy returned error code 2. (No such file or
directory).
semodule:  Failed!
```

6. 尝试显示有关 **sysadm_t** SELinux 类型的详情。该命令应该失败：

```
# seinfo -xt sysadm_t
[Errno 13] Permission denied: '/sys/fs/selinux/policy'
```

6.9. 在 MLS 中定义安全终端

SELinux 策略会检查用户从中连接的终端类型，并允许运行某些 SELinux 应用程序，例如 **newrole**，只从安全终端检查。从非安全终端尝试此操作会产生错误：**错误： 不允许更改非安全终端的级别；**

/etc/selinux/mls/contexts/securetty_types 文件定义了多级别安全(MLS)策略的安全终端。

该文件的默认内容：

```
console_device_t
sysadm_tty_device_t
user_tty_device_t
staff_tty_device_t
auditadm_tty_device_t
secureadm_tty_device_t
```

警告

将终端类型添加到安全终端列表中，可使您的系统暴露于安全风险。

先决条件

- SELinux 策略设置为 **mls**。
- 您已从已经安全的终端连接，或者 SELinux 处于 permissive 模式。
- 您有安全管理权限，这意味着您要分配给其中之一：
  - **sysadm_r** 角色

- **secadm_r** 角色。
- 如果启用了 **sysadm_secadm** 模块，进入 **sysadm_r** 角色。**sysadm_secadm** 模块默认启用。

- 已安装 **policycoreutils-python-utils** 软件包。

步骤

1. 确定当前的终端类型：

```
# ls -Z `tty`
root:object_r:user_devpts_t:s0 /dev/pts/0
```

在本例中，**user_devpts_t** 是当前的终端类型。

2. 在 **/etc/selinux/mls/contexts/securetty_types** 文件中的新行中添加相关的 SELinux 类型。
3. 可选：将 SELinux 切换到 enforcing 模式：

```
# setenforce 1
```

验证

- 从之前不安全的终端中登录到 **/etc/selinux/mls/contexts/securetty_types** 文件。

其他资源

- **securetty_types(5)** man page

6.10. 允许 MLS 用户在较低级别上编辑文件

默认情况下，MLS 用户无法写入在明确范围内低值下具有敏感等级的文件。如果您的场景需要允许用户在较低级别上编辑文件，可以通过创建本地 SELinux 模块来实现。但是，写入文件会使用户当前范围内的低值提高其敏感度级别。

先决条件

- SELinux 策略被设置为 **mls**。
- SELinux 模式设置为 **enforcing**。
- 已安装 **policycoreutils-python-utils** 软件包。
- **setools-console** 和 **audit** 软件包进行验证。

步骤

1. 可选：切换到 permissive 模式以方便故障排除。

```
# setenforce 0
```

2. 使用文本编辑器打开新的 **.cil** 文件，如 **~/local_mlsfilewrite.cil**，并插入以下自定义规则：

```
(typeattributeset mlsfilewrite (_staff_t_))
```

您可以将 ***staff_t*** 替换为不同的 SELinux 类型。通过在此处指定 SELinux 类型，您可以控制哪些 SELinux 角色可以编辑低级别文件。

要让您的本地模块更好地组织，请在本地 SELinux 策略模块的名称中使用 **local_** 前缀。

3. 安装策略模块：

```
# semodule -i ~/local_mlsfilewrite.cil
```

注意

要删除本地策略模块，请使用 **semodule -r ~/*local_mlsfilewrite***。请注意，您必须引用不带 **.cil** 后缀的模块名称。

4. 可选：如果您之前切换到 permissive 模式，返回 enforcing 模式：

```
# setenforce 1
```

验证

1. 在安装的 SELinux 模块列表中找到本地模块：

```
# semodule -lfull | grep "local_mls"
400 local_mlsfilewrite  cil
```

由于本地模块具有优先级 **400**，所以您也可以使用 **semodule -lfull | grep -v ^100** 命令列出它们。

2. 以分配给自定义规则中定义的类型的用户身份登录，例如 ***staff_t***。

3. 尝试写入到具有较低敏感度级别的文件。这会增加文件的分类级别到用户的清晰度级别。

重要

如果配置不正确，您用于验证的文件不应包含任何敏感信息，并且用户实际上可以访问未经授权的文件。

第 7 章 使用多类别安全(MCS)进行数据保密性

您可以使用 MCS 通过分类数据来增强系统数据的机密性，然后授予某些进程和用户对特定类别的访问权限

7.1. 多类别安全性(MCS)

多类别 Security(MCS)是一个访问控制机制，它使用分配给进程和文件的类别。然后，文件只能由分配到相同类别的进程访问。MCS 的目的是维护您系统上的数据保密性。

MCS 类别由 **c0** 到 **c1023** 的值定义，但您也可以为每个类别或类别组合定义一个文本标签，如"Personnel"、"ProjectX"或"ProjectX.Personnel"。MCS 转换服务(**mcstrans**)随后将 category 值替换为系统输入和输出中的相应标签，以便用户可以使用这些标签而不是 category 值。

当用户分配给类别时，他们可以为他们分配的任何类别标记其任何文件。

MCS 适用于一个简单的原则：要访问文件，必须将用户分配给分配给该文件的所有类别。MCS 检查在常规 Linux Discretionary Access Control(DAC)和 SELinux Type Enforcement(TE)规则后应用，因此它只能进一步限制现有的安全配置。

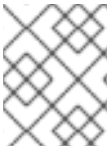
多级别安全中的 MCS

您可以将自己上的 MCS 用作非层次系统，也可以将其与多级别安全(MLS)结合使用，作为分层系统中的非层次结构层。

一个 MLS 中的 MCS 示例是，保密性科研组织，其中文件被分类如下：

表 7.1. 安全级别和类别组合示例

安全级别	类别			
	未指定	项目 X	项目 Y	项目 Z
未分类	**s0**	**s0:c0**	**s0:c1**	**s0:c2**
机密	**s1**	**s1:c0**	**s1:c1**	**s1:c2**
Secret	**s2**	**s2:c0**	**s2:c1**	**s2:c2**
Top secret	**s3**	**s3:c0**	**s3:c1**	**s3:c2**



注意

拥有范围 **s0:c0.1023** 的用户可以访问分配给 **s0** 级别的所有类别的所有文件，除非访问被其他安全机制禁止，如 DAC 或类型执行策略规则。

文件或进程生成的安全上下文是以下组合：

- SELinux 用户
- SELinux 角色
- SELinux 类型

- MLS 敏感度级别
- MCS 类别

例如，在 MLS/MCS 环境中具有访问级别 1 和类别 2 的非授权用户可能具有以下 SELinux 上下文：

```
user_u:user_r:user_t:s1:c2
```

其他资源

- 使用多级别安全(MLS)

7.2. 为数据机密配置多类别安全性

默认情况下，MCS 在 **targeted** 和 **mls** SELinux 策略中处于活跃状态，但没有为用户配置。在 **targeted** 策略中，仅针对以下内容配置 MCS：

- OpenShift
- virt
- sandbox
- 网络标记
- containers (**container-selinux**)

您可以通过创建本地 SELinux 模块并将 **user_t** SELinux 类型限制在类型强制的情况下，将 MCS 规则配置为分类用户的 MCS 规则。

警告

更改某些文件的类别可能会导致某些服务无法正常运行。如果您并不是相关系统的专家，请联系红帽销售代表并请求咨询服务。

先决条件

- SELinux 模式设置为 **enforcing**。
- SELinux 策略被设置为 **targeted** 或 **mls**。
- 已安装 **policycoreutils-python-utils** 和 **setools-console** 软件包。

步骤

1. 创建一个新文件，例如名为 **local_mcs_user.cil** ：

```
# vim local_mcs_user.cil
```

2. 插入以下规则：

```
(typeattributeset mcs_constrained_type (user_t))
```

3. 安装策略模块：

```
# semodule -i local_mcs_user.cil
```

验证

- 对于每个用户域，显示所有组件的更多详情：

```
# seinfo -xt user_t

Types: 1
type user_t, application_domain_type, nsswitch_domain, corenet_unlabeled_type, domain,
kernel_system_state_reader, mcs_constrained_type, netlabel_peer_type, privfd,
process_user_target, scsi_generic_read, scsi_generic_write, syslog_client_type,
pcmcia_typeattr_1, user_usertype, login_userdomain, userdomain, unpriv_userdomain,
userdom_home_reader_type, userdom_filetrans_type, xdmhomewriter, x_userdomain,
x_domain, dridomain, xdrawable_type, xcolormap_type;
```

其他资源

- 创建本地 SELinux 策略模块
- 有关容器上下文中的 MCS 的更多信息，请参阅博客文章 如何使用多级安全性实现 SELinux 独立的容器，并 为什么您应该为您的 Linux 容器使用多类别安全性。

7.3. 在 MCS 中定义类别标签

您可以通过编辑 **setrans.conf** 文件来管理和维护 MCS 类别标签，或使用 MLS 级别的 MCS 类别组合。在这个文件中，SELinux 在内部敏感度和类别级别及其人类可读标签之间保持映射。

注意

类别标签只让用户更易于使用类别。无论您定义标签或是否定义标签，MCS 的效果都相同。

先决条件

- SELinux 模式设置为 **enforcing**。
- SELinux 策略被设置为 **targeted** 或 **mls**。
- 已安装 **policycoreutils-python-utils** 和 **mcstrans** 软件包。

步骤

1. 通过编辑文本编辑器中的 **/etc/selinux/*<selinuxpolicy>*/setrans.conf** 文件来修改现有类别或创建新类别。根据您使用的 SELinux 策略，将 *<selinuxpolicy>* 替换为 **targeted** 或 **mls**。例如：

```
# vi /etc/selinux/targeted/setrans.conf
```

2. 在策略的 **setrans.conf** 文件中，使用语法 **s_<security level>_:c_<category number>_=*<category.name>*** 来定义您的场景所需的类别组合，例如：

```
s0:c0=Marketing
s0:c1=Finance
s0:c2=Payroll
s0:c3=Personnel
```

   - 您可以使用 **c0** 到 **c1023** 中的类别号。
   - 在 **targeted** 策略中，使用 **s0** 安全级别。
   - 在 **mls** 策略中，您可以标记各个敏感度级别和类别的组合。

3. 可选：在 **setrans.conf** 文件中，您还可以标记 MLS 敏感度级别。
4. 保存并退出 文件。
5. 要使更改有效，重启 MCS 翻译服务：

```
# systemctl restart mcstrans
```

验证

- 显示当前类别：

```
# chcat -L
```

上面的示例会产生以下输出：

```
s0:c0                   Marketing
s0:c1                   Finance
s0:c2                   Payroll
s0:c3                   Personnel
s0
s0-s0:c0.c1023              SystemLow-SystemHigh
s0:c0.c1023                 SystemHigh
```

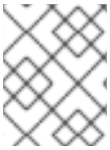
其他资源

- **setrans.conf(5)** 手册页。

7.4. 为 MCS 中的用户分配类别

您可以通过为 Linux 用户分配类别来定义用户授权。分配了类别的用户可以访问和修改用户类别子集的文件。用户也可以为他们自己分配到的类别分配文件。

无法将 Linux 用户分配给在为相关 SELinux 用户定义的安全范围之外的类别。



注意

类别访问权限在登录期间分配。因此，在用户再次登录前，用户无法访问新分配的类别。同样，如果您撤销了对类别的访问权限，这仅在用户再次登录后有效。

先决条件

- SELinux 模式设置为 **enforcing**。
- SELinux 策略被设置为 **targeted** 或 **mls**。
- 已安装 **policycoreutils-python-utils** 软件包。
- Linux 用户被分配给 SELinux 受限用户：
  - 非特权用户将分配给 **user_u**.
  - 特权用户被分配给 **staff_u**。

步骤

1. 定义 SELinux 用户的安全范围。

```
# semanage user -m -rs0:c0,c1-s0:c0.c9 <user_u>
```

使用 **setrans.conf** 文件中的类别号 **c0** 到 **c1023** 或 category 标签。如需更多信息，请参阅 MCS 中的定义类别标签。

2. 为 Linux 用户分配 MCS 类别。您只能指定相关 SELinux 用户定义的范围内的范围：

```
# semanage login -m -rs0:c1 <Linux.user1>
```

注意

您可以使用 **chcat** 命令从 Linux 用户添加或删除类别。以下示例添加 ***<category1>*** 并从 ***<Linux.user1>*** 和 ***<Linux.user2>*** 中删除 ***<category2>***：

```
# chcat -l -- +<category1>,-<category2> <Linux.user1>,<Linux.user2>
```

请注意，在使用 **-*<category>*** 语法前，您必须在命令行中指定 **--**。否则，**chcat** 命令会错误地将类别删除作为命令选项进行解译。

验证

- 列出分配给 Linux 用户的类别：

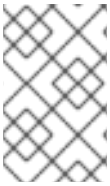
```
# chcat -L -l <Linux.user1>,<Linux.user2>
<Linux.user1>: <category1>,<category2>
<Linux.user2>: <category1>,<category2>
```

其他资源

- **chcat(8)** 手册页。

7.5. 为 MCS 中的文件分配类别

您需要具有管理特权才能为用户分配类别。然后用户可以为文件分配类别。要修改文件的类别，用户必须拥有该文件的访问权限。用户只能为为其分配的类别分配一个文件。



注意

系统将类别访问规则与传统的文件访问权限合并。例如，如果具有 **bigfoot** 类别的用户使用 Discretionary Access Control(DAC)来阻止其他用户对文件的访问，则其他 **bigfoot** 用户可以访问该文件。分配给所有可用类别的用户仍无法访问整个文件系统。

先决条件

- SELinux 模式设置为 **enforcing**。
- SELinux 策略被设置为 **targeted** 或 **mls**。
- 已安装 **policycoreutils-python-utils** 软件包。
- 对 Linux 用户的访问权限和权限：
  - 分配给 SELinux 用户。
  - 分配给要为其分配该文件的类别。如需更多信息，请参阅 MCS 中的用户分配类别。
- 访问您要添加到该类别中的文件的访问权限和权限。
- 为进行验证：对尚未分配给此类别的 Linux 用户访问和权限

步骤

- 为文件添加类别：

```
$ chcat -- +<category1>,+<category2> <path/to/file1>
```

使用 **setrans.conf** 文件中的类别号 **c0** 到 **c1023** 或 category 标签。如需更多信息，请参阅 MCS 中的定义类别标签。

您可以使用相同的语法从文件中删除类别：

```
$ chcat -- -<category1>,-<category2> <path/to/file1>
```

注意

删除类别时，必须在使用 ***-<category>*** 语法前在命令行中指定 **--**。否则，**chcat** 命令可能会错误地认为类别被删除作为一个命令选项。

验证

1. 显示文件的安全上下文，以验证它具有正确的类别：

```
$ ls -lZ <path/to/file>
-rw-r--r--  <LinuxUser1> <Group1> root:object_r:user_home_t:_<sensitivity>_:_<category>_
<path/to/file>
```

文件的特定安全上下文可能有所不同。

2. 可选：当作为未分配给与该文件相同的类别的 Linux 用户登录时，尝试访问该文件：

```
$ cat <path/to/file>
cat: <path/to/file>: Permission Denied
```

其他资源

- **semanage(8)** 手册页。
- **chcat(8)** 手册页。

第 8 章 编写自定义 SELINUX 策略

本节介绍了如何编写和使用可让您运行受 SELinux 限制的应用程序的自定义策略。

8.1. 自定义 SELINUX 策略和相关工具

SELinux 安全策略是 SELinux 规则的集合。策略是 SELinux 的核心组件，它由 SELinux 用户空间工具载入内核。内核强制使用 SELinux 策略来评估系统中的访问请求。默认情况下，SELinux 拒绝所有请求，但与载入策略中指定的规则对应的请求除外。

每个 SELinux 策略规则都描述了进程和系统资源间的交互：

```
ALLOW apache_process apache_log:FILE READ;
```

您可以按如下所示读取此示例规则：***Apache 进程可以读取其日志文件***。在此规则中，**apache_process** 和 **apache_log** 是 **labels**。SELinux 安全策略为进程分配标签并定义与系统资源的关系。这样，策略可将操作系统实体映射到 SELinux 层。

SELinux 标签作为文件系统的扩展属性保存，如 **ext2**。您可以使用 **getfattr** 实用程序或 **ls -Z** 命令列出它们，例如：

```
$ ls -Z /etc/passwd
system_u:object_r:passwd_file_t:s0 /etc/passwd
```

其中 **system_u** 是 SELinux 用户， **object_r** 是 SELinux 角色的示例， **passwd_file_t** 是 SELinux 域。

selinux-policy 软件包提供的默认 SELinux 策略包含作为 Red Hat Enterprise Linux 9 一部分且由软件包在存储库中提供的应用程序和守护进程规则。没有被这个发布策略中的规则描述的应用程序不会被 SELinux 限制。要更改它，您必须使用包含额外定义和规则的 policy 模块来修改策略。

在 Red Hat Enterprise Linux 9 中，您可以查询已安装的 SELinux 策略并使用 **sepolicy** 工具生成新策略模块。**sepolicy** 生成的脚本以及 policy 模块始终包含一个使用 **restorecon** 实用程序的命令。这个工具是修复文件系统中所选部分问题的基本工具。

其他资源

- **sepolicy(8)** 和 **getfattr(1)** man page

8.2. 为自定义应用程序创建并强制 SELINUX 策略

这个示例步骤提供了通过 SELinux 保护简单守护进程的步骤。将守护进程替换为您自定义应用程序，并根据应用程序和安全策略的要求修改示例中的规则。

先决条件

- **policycoreutils-devel** 软件包及其依赖项安装在您的系统中。

步骤

1. 在本例中，准备一个简单的守护进程，它将打开 **/var/log/messages** 文件进行写入：
   a. 创建一个新文件，然后在您选择的文本编辑器中打开：

```
$ vi mydaemon.c
```

b. 插入以下代码：

```
#include <unistd.h>
#include <stdio.h>

FILE *f;

int main(void)
{
while(1) {
f = fopen("/var/log/messages","w");
        sleep(5);
        fclose(f);
    }
}
```

c. 编译文件：

```
$ gcc -o mydaemon mydaemon.c
```

d. 为您的守护进程创建一个 **systemd** 单元文件：

```
$ vi mydaemon.service
[Unit]
Description=Simple testing daemon

[Service]
Type=simple
ExecStart=/usr/local/bin/mydaemon

[Install]
WantedBy=multi-user.target
```

e. 安装并启动守护进程：

```
# cp mydaemon /usr/local/bin/
# cp mydaemon.service /usr/lib/systemd/system
# systemctl start mydaemon
# systemctl status mydaemon
● mydaemon.service - Simple testing daemon
   Loaded: loaded (/usr/lib/systemd/system/mydaemon.service; disabled; vendor preset:
disabled)
   Active: active (running) since Sat 2020-05-23 16:56:01 CEST; 19s ago
 Main PID: 4117 (mydaemon)
    Tasks: 1
   Memory: 148.0K
   CGroup: /system.slice/mydaemon.service
           └─4117 /usr/local/bin/mydaemon

May 23 16:56:01 localhost.localdomain systemd[1]: Started Simple testing daemon.
```

f. 检查新守护进程是否没有被 SELinux 限制：

```
$ ps -efZ | grep mydaemon
system_u:system_r:unconfined_service_t:s0 root 4117    1  0 16:56 ?        00:00:00
/usr/local/bin/mydaemon
```

2. 为守护进程生成自定义策略：

```
$ sepolicy generate --init /usr/local/bin/mydaemon
Created the following files:
/home/example.user/mysepol/mydaemon.te # Type Enforcement file
/home/example.user/mysepol/mydaemon.if # Interface file
/home/example.user/mysepol/mydaemon.fc # File Contexts file
/home/example.user/mysepol/mydaemon_selinux.spec # Spec file
/home/example.user/mysepol/mydaemon.sh # Setup Script
```

3. 使用上一命令创建的设置脚本使用新策略模块重建系统策略：

```
# ./mydaemon.sh
Building and Loading Policy
+ make -f /usr/share/selinux/devel/Makefile mydaemon.pp
Compiling targeted mydaemon module
Creating targeted mydaemon.pp policy package
rm tmp/mydaemon.mod.fc tmp/mydaemon.mod
+ /usr/sbin/semodule -i mydaemon.pp
...
```

请注意，设置脚本使用 **restorecon** 命令重新标记文件系统的对应部分：

```
restorecon -v /usr/local/bin/mydaemon /usr/lib/systemd/system
```

4. 重启守护进程，检查它现在被 SELinux 限制：

```
# systemctl restart mydaemon
$ ps -efZ | grep mydaemon
system_u:system_r:mydaemon_t:s0 root       8150       1  0 17:18 ?        00:00:00
/usr/local/bin/mydaemon
```

5. 由于守护进程现在受 SELinux 限制，SELinux 也阻止它访问 **/var/log/messages**。显示对应的拒绝信息：

```
# ausearch -m AVC -ts recent
...
type=AVC msg=audit(1590247112.719:5935): avc:  denied  { open } for  pid=8150
comm="mydaemon" path="/var/log/messages" dev="dm-0" ino=2430831
scontext=system_u:system_r:mydaemon_t:s0 tcontext=unconfined_u:object_r:var_log_t:s0
tclass=file permissive=1
...
```

6. 您还可以使用 **sealert** 工具获取更多信息：

```
$ sealert -l "*"
SELinux is preventing mydaemon from open access on the file /var/log/messages.

 Plugin catchall (100. confidence) suggests *
```

```
If you believe that mydaemon should be allowed open access on the messages 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 'mydaemon' --raw | audit2allow -M my-mydaemon
# semodule -X 300 -i my-mydaemon.pp

Additional Information:
Source Context                system_u:system_r:mydaemon_t:s0
Target Context                unconfined_u:object_r:var_log_t:s0
Target Objects                /var/log/messages [ file ]
Source                        mydaemon

...
```

7. **使用 audit2allow 工具推荐更改：**

```
$ ausearch -m AVC -ts recent | audit2allow -R

require {
 type mydaemon_t;
}

#============= mydaemon_t ==============
logging_write_generic_logs(mydaemon_t)
```

8. **因为 audit2allow 所推荐的规则在某些情况下可能不正确，所以只使用其输出的一部分来查找对应的策略接口：**

```
$ grep -r "logging_write_generic_logs" /usr/share/selinux/devel/include/ | grep .if
/usr/share/selinux/devel/include/system/logging.if:interface(`logging_write_generic_logs',`
```

9. **检查接口的定义：**

```
$ cat /usr/share/selinux/devel/include/system/logging.if
...
interface(`logging_write_generic_logs',`
        gen_require(`
                type var_log_t;
        ')

        files_search_var($1)
        allow $1 var_log_t:dir list_dir_perms;
        write_files_pattern($1, var_log_t, var_log_t)
')
...
```

10. **在这个示例中，您可以使用推荐的接口。在您的类型强制文件中添加对应的规则：**

```
$ echo "logging_write_generic_logs(mydaemon_t)" >> mydaemon.te
```

另外，您可以添加这个规则而不是使用接口：

```
$ echo "allow mydaemon_t var_log_t:file { open write getattr };" >> mydaemon.te
```

11. 重新安装策略：

```
# ./mydaemon.sh
Building and Loading Policy
+ make -f /usr/share/selinux/devel/Makefile mydaemon.pp
Compiling targeted mydaemon module
Creating targeted mydaemon.pp policy package
rm tmp/mydaemon.mod.fc tmp/mydaemon.mod
+ /usr/sbin/semodule -i mydaemon.pp
...
```

验证

1. 检查您的应用程序是否受 SELinux 限制，例如：

```
$ ps -efZ | grep mydaemon
system_u:system_r:mydaemon_t:s0 root       8150       1  0 17:18 ?        00:00:00
/usr/local/bin/mydaemon
```

2. 验证您的自定义应用程序不会导致任何 SELinux 拒绝：

```
# ausearch -m AVC -ts recent
<no matches>
```

其他资源

- **sepolgen(8)**, **ausearch(8)**, **audit2allow(1)**, **audit2why(1)**, **sealert(8)**, 和 **restorecon(8)** man pages

8.3. 创建本地 SELINUX 策略模块

在活跃的 SELinux 策略中添加特定的 SELinux 策略模块可以修复 SELinux 策略的某些问题。您可以使用此流程修复 红帽发行注记 中介绍的特定已知问题，或实施特定的 红帽解决方案。

警告

只用红帽提供的规则。红帽不支持使用自定义规则创建 SELinux 策略模块，因为这不会超出 产品支持覆盖范围。如果您并不是相关系统的专家，请联系红帽销售代表并请求咨询服务。

先决条件

- **setools-console** 和 **audit** 软件包进行验证。

步骤

1. 使用文本编辑器打开新的 .cil 文件，例如：

```
# vim <local_module>.cil
```

要让您的本地模块更好地组织，请在本地 SELinux 策略模块的名称中使用 local_ 前缀。

2. 从已知问题或红帽解决方案中插入自定义规则。

重要

不要自己编写规则。仅使用特定已知问题或红帽解决方案中提供的规则。

例如，要实现 SELinux 拒绝 cups-lpd 对 RHEL 解决方案中的 cups.sock 的读访问权限，请插入以下规则：

```
(allow cupsd_lpd_t cupsd_var_run_t (sock_file (read)))
```

请注意，您可以使用两个 SELinux 规则语法之一： Common Intermediate Language(CIL)和 m4。例如，CIL 中的 (allow cupsd_lpd_t cupsd_var_run_t (sock_file (read))) 等同于 m4 中的以下内容：

```
module local_cupslpd-read-cupssock 1.0;

require {
    type cupsd_var_run_t;
    type cupsd_lpd_t;
    class sock_file read;
}

#============= cupsd_lpd_t ==============
allow cupsd_lpd_t cupsd_var_run_t:sock_file read;
```

3. 保存并关闭该文件。

4. 安装策略模块：

```
# semodule -i <local_module>.cil
```

注意

当使用 # semodule -i 来删除您创建的本地策略模块时，请参考不带 .cil 后缀的模块名称。要删除本地策略模块，请使用 # semodule -r *<local_module>*。

5. 重启与规则相关的任何服务：

```
# systemctl restart <service-name>
```

验证

1. 列出 SELinux 策略中安装的本地模块：

```
# semodule -lfull | grep "local_"
400 local_module  cil
```

注意

由于本地模块具有优先级 400，所以您也可以通过使用该值来从列表中过滤它们，例如使用 semodule -lfull | grep -v ^100 命令。

2. **在 SELinux 策略中搜索相关的允许规则：**

```
# sesearch -A --source=<SOURCENAME> --target=<TARGETNAME> --
class=<CLASSNAME> --perm=<P1>,<P2>
```

其中 *<SOURCENAME>* 是源 SELinux 类型，*<TARGETNAME>* 是目标 SELinux 类型，*<CLASSNAME>* 是安全类或对象类名称，*<P1>* 和 *<P2>* 规则的特定权限。

例如， SELinux denies cups-lpd read access to cups.sock in RHEL**解决方案：**

```
# sesearch -A --source=cupsd_lpd_t --target=cupsd_var_run_t --class=sock_file --
perm=read
allow cupsd_lpd_t cupsd_var_run_t:sock_file { append getattr open read write };
```

最后一行现在应包含 read 操作。

3. **验证相关服务受 SELinux 限制：**

   a. **确定与相关服务相关的进程：**

```
$ systemctl status <service-name>
```

   b. **检查上一命令输出中列出的进程的 SELinux 上下文：**

```
$ ps -efZ | grep <process-name>
```

4. **验证该服务是否不会导致任何 SELinux 拒绝：**

```
# ausearch -m AVC -ts recent
<no matches>
```

其他资源

- ***第 5 章 故障排除与 SELinux 相关的问题***

8.4. 其他资源

- SELinux Policy Workshop

第 9 章 为容器创建 SELINUX 策略

Red Hat Enterprise Linux 9 提供了使用 udica 软件包为容器生成 SELinux 策略的工具。通过udica，您可以创建一个定制的安全策略来更好地控制容器如何访问主机系统资源，如存储、设备和网络。这可让强化容器部署以避免出现安全问题，并简化了规范合规性的实现和维护。

9.1. UDICA SELINUX 策略生成器介绍

为了简化为自定义容器创建新 SELinux 策略，RHEL 9 提供了 udica 工具。您可以使用此工具基于容器的 JavaScript Object Notation（JSON）文件创建策略，该文件包含 Linux 功能、挂载点和端口定义。因此，该工具将使用检查结果生成的规则与从指定 SELinux 通用中间语言（CIL）块继承的规则合并。

使用 udica 为容器生成 SELinux 策略的过程有三个主要部分：

1. 以 JSON 格式解析容器规格文件
2. 根据第一部分的结果查找合适的允许规则
3. 生成最终 SELinux 策略

在解析阶段，udica 会查找 Linux 功能、网络端口和挂载点。

根据结果，udica 检测到容器需要哪些 Linux 功能，并创建一个允许所有这些功能的 SELinux 规则。如果容器绑定到一个特定端口，udica 使用 SELinux 用户空间库来获取通过检查容器使用的端口的正确 SELinux 标签。

之后，udica 检测到哪些目录被挂载到主机中的容器文件系统名称空间中。

CIL 的块继承功能允许 udica 创建 SELinux 模板，*允许规则*专注于特定操作，例如：

- *允许访问主目录*
- *允许访问日志文件*
- *允许访问与 Xserver 的通讯*。

这些模板称为块，最终 SELinux 策略通过合并这些块来创建。

其他资源

- 使用 udica 红帽博客为容器生成 SELinux 策略

9.2. 为自定义容器创建和使用 SELINUX 策略

要为自定义容器生成 SELinux 安全策略，请按照以下步骤执行。

先决条件

- 已安装用于管理容器的 podman 工具。如果没有，使用 dnf install podman 命令。
- 一个自定义 Linux 容器 - 本例中是 *ubi8*。

步骤

1. 安装 udica 软件包：

```
# dnf install -y udica
```

或者，安装 container-tools 模块，它提供一组容器软件包，包括 udica ：

```
# dnf module install -y container-tools
```

2. 启动 *ubi8* **容器，它使用只读权限挂载 /home 目录，以及具有读取和写入的权限的 /var/spool 目录。容器会公开端口 *21*。**

```
# podman run --env container=podman -v /home:/home:ro -v /var/spool:/var/spool:rw -p
21:21 -it ubi8 bash
```

请注意，现在容器使用 container_t SELinux 类型运行。这个类型是 SELinux 策略中所有容器的通用域。针对于您的具体情况，这可能太严格或太宽松。

3. **打开一个新终端，并输入 podman ps 命令以获取容器的 ID：**

```
# podman ps
CONTAINER ID  IMAGE                                COMMAND  CREATED        STATUS
PORTS  NAMES
37a3635afb8f  registry.access.redhat.com/ubi8:latest  bash     15 minutes ago  Up 15
minutes ago          heuristic_lewin
```

4. **创建容器 JSON 文件，并使用 udica 根据 JSON 文件中的信息创建策略模块：**

```
# podman inspect 37a3635afb8f > container.json
# udica -j container.json my_container
Policy my_container with container id 37a3635afb8f created!
[...]
```

或者：

```
# podman inspect 37a3635afb8f | udica my_container
Policy my_container with container id 37a3635afb8f created!

Please load these modules using:
# semodule -i my_container.cil
/usr/share/udica/templates/{base_container.cil,net_container.cil,home_container.cil}

Restart the container with: "--security-opt label=type:my_container.process" parameter
```

5. **如上一步中的 udica 输出所建议，加载策略模块：**

```
# semodule -i my_container.cil
/usr/share/udica/templates/{base_container.cil,net_container.cil,home_container.cil}
```

6. **停止容器并使用 --security-opt label=type:my_container.process 选项再次启动它：**

```
# podman stop 37a3635afb8f
# podman run --security-opt label=type:my_container.process -v /home:/home:ro -v
/var/spool:/var/spool:rw -p 21:21 -it ubi8 bash
```

验证

1. 检查容器使用 **my_container.process** 类型运行：

```
# ps -efZ | grep my_container.process
unconfined_u:system_r:container_runtime_t:s0-s0:c0.c1023 root 2275 434  1 13:49 pts/1
00:00:00 podman run --security-opt label=type:my_container.process -v /home:/home:ro -v
/var/spool:/var/spool:rw -p 21:21 -it ubi8 bash
system_u:system_r:my_container.process:s0:c270,c963 root 2317 2305  0 13:49 pts/0
00:00:00 bash
```

2. 验证 SELinux 现在允许访问 **/home** 和 **/var/spool** 挂载点：

```
[root@37a3635afb8f /]# cd /home
[root@37a3635afb8f home]# ls
username
[root@37a3635afb8f ~]# cd /var/spool/
[root@37a3635afb8f spool]# touch test
[root@37a3635afb8f spool]#
```

3. 检查 SELinux 是否只允许绑定到端口 21：

```
[root@37a3635afb8f /]# dnf install nmap-ncat
[root@37a3635afb8f /]# nc -lvp 21
...
Ncat: Listening on :::21
Ncat: Listening on 0.0.0.0:21
^C
[root@37a3635afb8f /]# nc -lvp 80
...
Ncat: bind to :::80: Permission denied. QUITTING.
```

其他资源

- **udica(8)** 和 **podman(1)** man page
- 构建、运行和管理容器

9.3. 其他资源

- udica - 为容器生成 SELinux 策略

第 10 章 在多个系统中部署相同的 SELINUX 配置

这部分提供了在多个系统中部署验证的 SELinux 配置的建议方法：

- 使用 RHEL 系统角色和 Ansible
- 在脚本中使用 **semanage** export 和导入命令

10.1. SELINUX 系统角色简介

RHEL 系统角色是 Ansible 角色和模块的集合，为远程管理多个 RHEL 系统提供一致的配置界面。SELinux 系统角色启用以下操作：

- 清理与 SELinux 布尔值、文件上下文、端口和登录相关的本地策略修改。
- 设置 SELinux 策略布尔值、文件上下文、端口和登录。
- 在指定文件或目录中恢复文件上下文。
- 管理 SELinux 模块。

下表提供了 SELinux 系统角色中可用的输入变量概述。

表 10.1. SELinux 系统角色变量

角色变量	描述	CLI 的替代方案
selinux_policy	选择保护目标进程或多级别安全保护的策略。	**/etc/selinux/config**中的 **SELINUXTYPE**
selinux_state	切换 SELinux 模式。	**/etc/selinux/config** 中的 **setenforce** 和 **SELINUX**.
selinux_booleans	启用和禁用 SELinux 布尔值。	**setsebool**
selinux_fcontexts	添加或删除 SELinux 文件上下文映射。	**semanage fcontext**
selinux_restore_dirs	在文件系统树中恢复 SELinux 标签。	**restorecon -R**
selinux_ports	在端口上设置 SELinux 标签。	**semanage port**
selinux_logins	将用户设置为 SELinux 用户映射。	**semanage login**
selinux_modules	安装、启用、禁用或删除 SELinux 模块。	**semodule**

rhel-system-roles 软件包安装的 **/usr/share/doc/rhel-system-roles/selinux/example-selinux-playbook.yml** 示例 playbook 演示了如何在 enforcing 模式下设置目标策略。playbook 还应用多个本地策略修改，并在 **/tmp/test_dir/** 目录中恢复文件上下文。

有关 SELinux 角色变量的详细参考，请安装 **rhel-system-roles** 软件包，并参阅 **/usr/share/doc/rhel-system-roles/selinux/** 目录中的 **README.md** 或 **README.html** 文件。

其他资源

- RHEL 系统角色简介。

10.2. 使用 SELINUX 系统角色在多个系统上应用 SELINUX 设置

按照以下步骤，在已验证的 SELinux 设置中准备并应用 Ansible playbook。

先决条件

- 对一个或多个 *受管节点* 的访问和权限，这些节点是您要使用 SELinux 系统角色进行配置的系统。
- 对 *控制节点* 的访问和权限，这是 Red Hat Ansible Core 配置其他系统的系统。
  在控制节点上：
  - **ansible-core** 和 **rhel-system-roles** 软件包已安装 。
  - 列出受管节点的清单文件。

重要

RHEL 8.0-8.5 提供对一个独立的 Ansible 存储库的访问权限，该存储库包含基于 Ansible 的自动化 Ansible Engine 2.9。Ansible Engine 包含命令行实用程序，如 **ansible**、**ansible-playbook**、连接器（如 **docker** 和 **podman** ）以及许多插件和模块。有关如何获取并安装 Ansible Engine 的详情，请参考 如何下载并安装 Red Hat Ansible Engine 知识库文章。

RHEL 8.6 和 9.0 引入了 Ansible Core（作为 **ansible-core** 软件包提供），其中包含 Ansible 命令行工具、命令以及小型内置 Ansible 插件。RHEL 通过 AppStream 软件仓库提供此软件包，它有一个有限的支持范围。如需更多信息，请参阅 RHEL 9 和 RHEL 8.6 及更新的 AppStream 软件仓库文档中的 Ansible Core 软件包的支持范围。

- 列出受管节点的清单文件。

步骤

1. 准备您的 playbook。您可以从头开始，或修改作为 **rhel-system-roles** 软件包的一部分安装的示例 playbook：

```
# cp /usr/share/doc/rhel-system-roles/selinux/example-selinux-playbook.yml my-selinux-playbook.yml
# vi my-selinux-playbook.yml
```

2. 更改 playbook 的内容，使其适合您的场景。例如，以下部分确保系统安装并启用 **selinux-local-1.pp** SELinux 模块：

```
selinux_modules:
- { path: "selinux-local-1.pp", priority: "400" }
```

3. 保存更改，然后退出文本编辑器。

4. 在 *host1*、*host2* 和 *host3* 系统上运行您的 playbook：

```
# ansible-playbook -i host1,host2,host3 my-selinux-playbook.yml
```

其他资源

- 如需更多信息，请安装 **rhel-system-roles** 软件包，并查看 **/usr/share/doc/rhel-system-roles/selinux/** 和 **/usr/share/ansible/roles/rhel-system-roles.selinux/** 目录。

10.3. 使用 SEMANAGE 将 SELINUX 设置传送到另一个系统中

使用以下步骤在基于 RHEL 9 的系统间传输自定义和验证的 SELinux 设置。

先决条件

- **policycoreutils-python-utils** 软件包安装在您的系统中。

步骤

1. 导出验证的 SELinux 设置：

```
# semanage export -f ./my-selinux-settings.mod
```

2. 使用设置将该文件复制到新系统：

```
# scp ./my-selinux-settings.mod new-system-hostname:
```

3. 登录新系统：

```
$ ssh root@new-system-hostname
```

4. 在新系统中导入设置：

```
new-system-hostname# semanage import -f ./my-selinux-settings.mod
```

其他资源

- **semanage-export(8)** 和 **semanage-import(8)** man page