

```
Give root password for maintenance  
(or press Control-D to continue): redhat  
[root@servera ~]#
```

► 4. 确定当前挂载的文件系统。

```
[root@servera ~]# mount  
...output omitted...  
/dev/vda1 on / type xfs (ro,relatime,seclabel,attr2,inode64,noquota)  
...output omitted...
```

请注意，root文件系统以只读方式挂载。

► 5. 重新挂载root文件系统读/写。

```
[root@servera ~]# mount -o remount,rw /
```

► 6. 使用**mount -a**命令，尝试挂载其他所有文件系统。对于**--all (-a)**选项，该命令将挂载所有在**/etc/fstab**中列出但尚未挂载的文件系统。

```
[root@servera ~]# mount -a  
mount: /RemoveMe: mount point does not exist.
```

► 7. 编辑**/etc/fstab**以修复问题。

7.1. 删除或注释掉不正确的行。

```
[root@servera ~]# vim /etc/fstab  
...output omitted...  
#/dev/sdz1    /RemoveMe    xfs    defaults    0 0
```

7.2. 更新系统的**systemd**以注册新的**/etc/fstab**配置。

```
[root@servera ~]# systemctl daemon-reload  
[root@servera ~]#
```

► 8. 通过尝试挂载所有条目来验证**/etc/fstab**现在是否正确。

```
[root@servera ~]# mount -a  
[root@servera ~]#
```

► 9. 重新启动系统并等待启动完成。现在，系统应当能正常启动。

```
[root@servera ~]# systemctl reboot
```

完成

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

```
[student@workstation ~]$ lab boot-repairing finish
```

本引导式练习到此结束。

▶ 开放研究实验

控制启动过程

任务执行清单

在本实验中，您将重置系统上的 `root` 密码，从错误配置中恢复，并设置默认启动目标。

成果

您应能够：

- 重置丢失的 `root` 密码。
- 诊断并修复启动问题。
- 设置默认 `systemd` 目标。

在你开始之前

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

在 `workstation` 上，运行 `lab boot-review start` 命令。此命令将运行一个起始脚本，它将确定 `serverb` 计算机是否可从网络访问。此外，它还会引入文件系统问题，重置 `root` 密码，为 GRUB2 菜单设置更长的超时，以及重新启动 `serverb`。

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

1. 在 `serverb` 上，将 `root` 密码重置为 `redhat`。

根据您的课堂环境，找到 `serverb` 控制台的图标。从该控制台开始工作。

2. 系统无法启动。启动作业似乎并没有完成。从控制台修复问题。
3. 在 `serverb` 上更改默认 `systemd` 目标，以便在系统启动时自动启动图形界面。
`serverb` 上尚未安装图形界面。对于本练习而言，将仅设置默认目标，而不安装软件包。

评估

在 `workstation` 上，运行 `lab boot-review grade` 脚本来确认是否成功完成本练习。

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

完成

在 `workstation` 上，运行 `lab boot-review finish` 脚本来完成实验。

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

本实验到此结束。

► 解决方案

控制启动过程

任务执行清单

在本实验中，您将重置系统上的 `root` 密码，从错误配置中恢复，并设置默认启动目标。

成果

您应能够：

- 重置丢失的 `root` 密码。
- 诊断并修复启动问题。
- 设置默认 `systemd` 目标。

在你开始之前

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

在 `workstation` 上，运行 `lab boot-review start` 命令。此命令将运行一个起始脚本，它将确定 `serverb` 计算机是否可从网络访问。此外，它还会引入文件系统问题，重置 `root` 密码，为 GRUB2 菜单设置更长的超时，以及重新启动 `serverb`。

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

1. 在 `serverb` 上，将 `root` 密码重置为 `redhat`。

根据您的课堂环境，找到 `serverb` 控制台的图标。从该控制台开始工作。

- 1.1. 使用相关按钮或菜单项向系统发送 `Ctrl+Alt+Del`。

- 1.2. 在显示启动加载器菜单后，按任意键（`Enter` 除外）中断倒计时。

- 1.3. 使用光标键突出显示默认的启动加载器条目。

- 1.4. 按 `e` 编辑当前条目。

- 1.5. 使用光标键导航到以 `linux` 开头的那一行。

- 1.6. 按 `End` 将光标移至行末。

- 1.7. 将 `rd.break` 附加到行末。

- 1.8. 按 `Ctrl+x` 使用修改后的配置进行启动。

- 1.9. 在 `switch_root` 提示符中，以读/写形式重新挂载 `/sysroot` 文件系统，然后使用 `chroot` 进入 `/sysroot` 中的 `chroot` 存放位置。

```
switch_root:/# mount -o remount,rw /sysroot
switch_root:/# chroot /sysroot
```

1.10. 将 **root** 密码设置为 **redhat**。

```
sh-4.4# passwd root  
Changing password for user root.  
New password: redhat  
BAD PASSWORD: The password is shorter than 8 characters  
Retype new password: redhat  
passwd: all authentication tokens updated successfully.
```

1.11. 将系统配置为在启动后自动执行完整 SELinux 重新标记。

```
sh-4.4# touch /.autorelabel
```

1.12. 键入 **exit** 两次，以继续启动系统。由于存在您在下一步中解决的问题，系统无法启动。

2. 系统无法启动。启动作业似乎并没有完成。从控制台修复问题。

2.1. 将系统启动到紧急模式。为此，请使用相关按钮或菜单项向系统发送 **Ctrl+Alt+Del**，以重启 **serverb**。

2.2. 在显示启动加载器菜单后，按任意键（**Enter** 除外）中断倒计时。

2.3. 使用光标键突出显示默认的启动加载器条目。

2.4. 按 **e** 编辑当前条目。

2.5. 使用光标键导航到以 **linux** 开头的那一行。

2.6. 按 **End** 将光标移至行末。

2.7. 将 **systemd.unit=emergency.target** 附加到行末。

2.8. 按 **Ctrl+x** 使用修改后的配置进行启动。

2.9. 登录紧急模式。**root** 密码为 **redhat**。

```
Give root password for maintenance  
(or press Control-D to continue): redhat  
[root@serverb ~]#
```

2.10. 重新挂载 **/** 文件系统读/写。

```
[root@serverb ~]# mount -o remount,rw /
```

2.11. 使用 **mount -a** 命令，尝试挂载其他所有文件系统。

```
[root@serverb ~]# mount -a  
mount: /olddata: can't find UUID=4d5c85a5-8921-4a06-8aff-80567e9689bc,
```

2.12. 编辑 **/etc/fstab** 以删除或注释掉不正确的行。

```
[root@serverb ~]# vim /etc/fstab  
...output omitted...  
#UUID=4d5c85a5-8921-4a06-8aff-80567e9689bc /olddata xfs defaults 0 0
```

2.13.更新系统的 **systemd** 以注册新的 **/etc/fstab** 配置。

```
[root@serverb ~]# systemctl daemon-reload  
[root@serverb ~]#
```

2.14.通过尝试挂载所有条目来验证 **/etc/fstab** 现在是否正确。

```
[root@serverb ~]# mount -a  
[root@serverb ~]#
```

2.15.重新启动系统并等待启动完成。由于您在第一步中创建了 **/autorelabel** 文件，因此在设置 **root** 密码后，系统将会运行 SELinux 重新标记，然后再次自行进行重新启动。现在，系统应当能正常启动。

```
[root@serverb ~]# systemctl reboot
```

3. 在 **serverb** 上更改默认 **systemd** 目标，以便在系统启动时自动启动图形界面。
serverb 上尚未安装图形界面。对于本练习而言，将仅设置默认目标，而不安装软件包。

3.1. 以 **root** 用户身份登录 **serverb**。使用 **redhat** 作为密码。

3.2. 使用 **systemctl set-default** 命令将 **graphical.target** 设置为默认目标。

```
[root@serverb ~]# systemctl set-default graphical.target
```

3.3. 使用 **systemctl get-default** 命令验证您的工作。

```
[root@serverb ~]# systemctl get-default  
graphical.target
```

3.4. 从 **serverb** 注销。

```
[root@serverb ~]# exit
```

评估

在 **workstation** 上，运行 **lab boot-review grade** 脚本来确认是否成功完成本练习。

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

完成

在 **workstation** 上，运行 **lab boot-review finish** 脚本并完成实验。

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

本实验到此结束。

总结

在本章中，您学到了：

- **systemctl reboot** 和 **systemctl poweroff** 可分别重新启动和关闭系统。
- **systemctl isolate target-name.target** 在运行时切换到新目标。
- **systemctl get-default** 和 **systemctl set-default** 可用于查询和设置默认目标。
- 在 **initramfs** 移交控制权前，在内核命令行中使用 **rd.break** 来中断启动过程。root 文件系统以只读形式挂载于 **/sysroot** 下。
- 紧急目标可用于诊断和修复文件系统问题。

章 11

管理网络安全

目标

使用系统防火墙和 SELinux 规则来控制与服务的网络连接。

培训目标

- 使用防火墙规则来接受或拒绝与系统服务的网络连接。
- 通过管理 SELinux 端口标签来控制网络服务是否可以使用特定的网络端口。

章节

- 管理服务器防火墙（及引导式练习）
- 控制 SELinux 端口标记（及引导式练习）

实验

管理服务器防火墙

管理服务器防火墙

培训目标

学完本节后，您应能够使用防火墙规则来接受或拒绝与系统服务的网络连接。

防火墙架构概念

Linux 内核中包含 **netfilter**，它是网络流量操作（如数据包过滤、网络地址转换和端口转换）的框架。通过在内核中实现拦截函数调用和消息的处理程序，**netfilter** 允许其他内核模块直接与内核的网络堆栈进行接口连接。防火墙软件使用这些 Hook 来注册过滤规则和数据包修改功能，以便对经过网络堆栈的每个数据包进行处理。在到达用户空间组件或应用之前，任何传入、传出或转发的网络数据包都可以通过编程方式来检查、修改、丢弃或路由。**Netfilter** 是红帽企业 Linux 8 防火墙的主要组件。

Nftables 增强了 netfilter

Linux 内核中还包含 **nftables**，这是一个新的过滤器和数据包分类子系统，其增强了 **netfilter** 的部分代码，但仍保留了 **netfilter** 的架构，如网络堆栈 Hook、连接跟踪系统及日志记录功能。**nftables** 更新的优势在于更快的数据包处理、更快的规则集更新，以及以相同的规则同时处理 IPv4 和 IPv6。**nftables** 与原始 **netfilter** 之间的另一个主要区别是它们的接口。**Netfilter** 通过多个实用程序框架进行配置，其中包括 **iptables**、**ip6tables**、**arptables** 和 **ebtables**，这些框架现在已被弃用。**Nftables** 则使用单个 **nft** 用户空间实用程序，通过一个接口来管理所有协议，由此消除了以往不同前端和多个 **netfilter** 接口引起的争用问题。

firewalld 简介

Firewalld 是一个动态防火墙管理器，它是 **nftables** 框架的前端（使用 **nft** 命令）。在推出 **nftables** 之前，作为一种改进 **iptables** 服务的替代方案，**firewalld** 曾使用 **iptables** 命令来直接配置 **netfilter**。在 RHEL 8 中，**firewalld** 仍然是推荐的前端，它使用 **nft** 来管理防火墙规则集。**Firewalld** 仍可以读取和管理 **iptables** 配置文件和规则集，并使用 **xtables-nft-multi** 将 **iptables** 对象直接转换为 **nftables** 规则和对象。对于 **nft** 转换过程无法正确处理现有 **iptables** 规则集的复杂用例，您可以对 **firewalld** 进行配置，使之恢复为 **iptables** 后端，虽然强烈建议不要这样做。

应用会使用 D-Bus 接口查询子系统。**firewalld** 子系统（可通过 **firewalld** RPM 软件包获得）未包含在最小安装中，但包含在基本安装中。借助 **firewalld**，可以将所有网络流量分为多个区域，从而简化防火墙管理。根据数据包源 IP 地址或传入网络接口等条件，流量将转入相应区域的防火墙规则。每个区域都有自己的端口和服务列表，它们处于打开或者关闭状态。



注意

对于笔记本电脑或经常更改网络的其他计算机，可以使用 NetworkManager 自动设置连接的防火墙区域。这些区域使用适于特定连接的规则进行自定义。

当您经常在家庭、办公室和公共无线网络间切换时，此功能尤为实用。当连接到家庭网络和企业网络时，用户可能希望系统的 **sshd** 服务可访问，但当用户连接到当地咖啡馆的公共无线网络时则不然。

Firewalld 会检查进入系统的每个数据包的源地址。如果该源地址被分配给特定区域，则应用该区域的规则。如果该源地址未分配给某个区域，**firewalld** 就会将数据包与传入网络接口的区域相关联，并应用该区域的规则。如果出于某种原因，网络接口未与某个区域关联，则 **firewalld** 会将数据包与默认区域相关联。

默认区域不是一个单独的区域，而是指代现有的区域。最初，**firewalld** 指定 **public** 区域为默认区域，并将 **lo** 回环接口映射至 **trusted** 区域。

大多数区域会允许与特定端口和协议（例如 **631/udp**）或预定义服务（例如 **ssh**）的列表匹配的流量通过防火墙。如果流量不与允许的端口和协议或服务匹配，则通常会被拒绝。（**trusted** 区域默认情况下允许所有流量，它是此规则的一个例外。）

预定义区域

Firewalld 上有一些预定义区域，可分别进行自定义。默认情况下，如果传入流量属于系统启动的通信的一部分，则所有区域都允许这些传入流量和所有传出流量。下表详细介绍了这些初始区域配置。

Firewalld 区域默认配置

| 区域名称 | 默认配置 |
|-----------------|---|
| trusted | 允许所有传入流量。 |
| 家 | 除非与传出流量相关，或与 ssh 、 mdns 、 ipp-client 、 samba-client 或 dhcpv6-client 预定义服务匹配，否则拒绝传入流量。 |
| 内部 | 除非与传出流量相关，或与 ssh 、 mdns 、 ipp-client 、 samba-client 或 dhcpv6-client 预定义服务匹配，否则拒绝传入流量（一开始与 home 区域相同）。 |
| 工作 | 除非与传出流量相关，或与 ssh 、 ipp-client 或 dhcpv6-client 预定义服务匹配，否则拒绝传入流量。 |
| 公共 | 除非与传出流量相关，或与 ssh 或 dhcpv6-client 预定义服务匹配，否则拒绝传入流量。新添加的网络接口的默认区域。 |
| external | 除非与传出流量相关，或与 ssh 预定义服务匹配，否则拒绝传入流量。通过此区域转发的 IPv4 传出流量将进行伪装，以使其看起来像是来自传出网络接口的 IPv4 地址。 |
| dmz | 除非与传出流量相关，或与 ssh 预定义服务匹配，否则拒绝传入流量。 |
| block | 除非与传出流量相关，否则拒绝所有传入流量。 |
| drop | 除非与传出流量相关，否则丢弃所有传入流量（甚至不产生包含 ICMP 错误的响应）。 |

有关可用预定义区域及其预期用途的列表，请参阅 **firewalld.zones(5)**。

预定义服务

Firewalld 上有一些预定义服务。这些服务定义可帮助您识别要配置的特定网络服务。例如，可指定预构建的 **samba-client** 服务来配置正确的端口和协议，而无需研究 **samba-client** 服务的相关端口。下表列出了初始防火墙区域配置中使用的预定义服务。

预定义 Firewalld 服务节选

| 服务名称 | 配置 |
|---------------|--|
| SSH | 本地 SSH 服务器。到 22/tcp 的流量 |
| dhcpv6-client | 本地 DHCPv6 客户端。到 fe80::/64 IPv6 网络中 546/udp 的流量 |
| ipp-client | 本地 IPP 打印。到 631/udp 的流量。 |
| samba-client | 本地 Windows 文件和打印共享客户端。到 137/udp 和 138/udp 的流量。 |
| MDNS | 多播 DNS (mDNS) 本地链路名称解析。到 5353/udp 指向 224.0.0.251 (IPv4) 或 ff02::fb (IPv6) 多播地址的流量。 |



注意

许多预定义服务都包含在 firewalld 软件包中。使用 **firewall-cmd --get-services** 可以列出这些预定义服务。预定义服务的配置文件位于 **/usr/lib/firewalld/services** 中，其格式由 **firewalld.zone(5)** 定义。

可以使用预定义服务，或者直接指定所需的端口和协议。可以使用 Web 控制台图形界面来查看预定义服务和定义更多服务。

配置防火墙

系统管理员可通过三种主要方式与 firewalld 交互：

- 直接编辑 **/etc/firewalld/** 中的配置文件（本章中不讨论）
- Web 控制台图形界面
- **firewall-cmd** 命令行工具

使用 Web 控制台配置防火墙服务

要使用 Web 控制台来配置防火墙服务，请单击 Reuse my password for privileged tasks 选项以特权访问权限进行登录。这将允许用户以 sudo 权限执行修改防火墙服务的命令。

图 11.1: Web 控制台特权登录

单击左侧导航菜单中的 Networking 选项，以显示主网络页面中的 Firewall 部分。单击 Firewall 链接，以访问允许的服务列表。

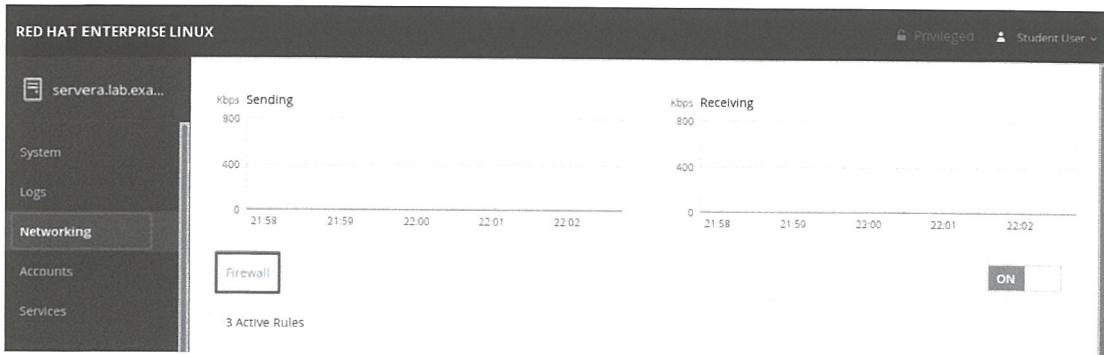


图 11.2: Web 控制台网络

允许的服务列表是指防火墙当前所允许的那些服务。单击服务名称左侧的箭头 (>) 可查看服务详细信息。要添加服务, 请单击 Firewall Allowed Services 页面右上角的 Add Services... 按钮。

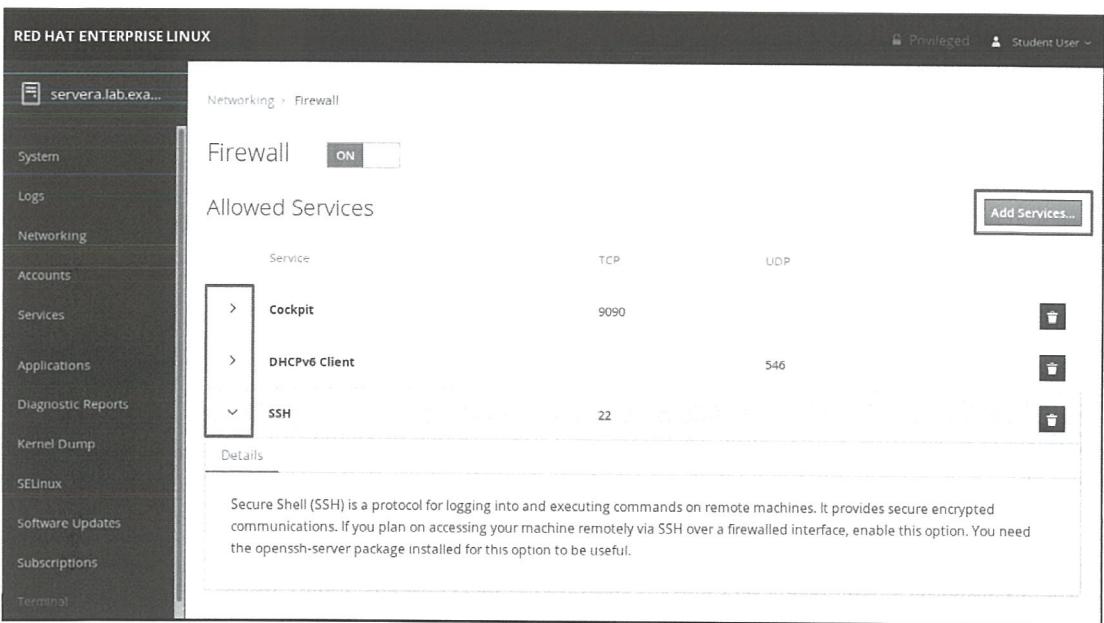


图 11.3: Web 控制台防火墙允许的服务列表

Add Services 页面显示可用的预定义服务。

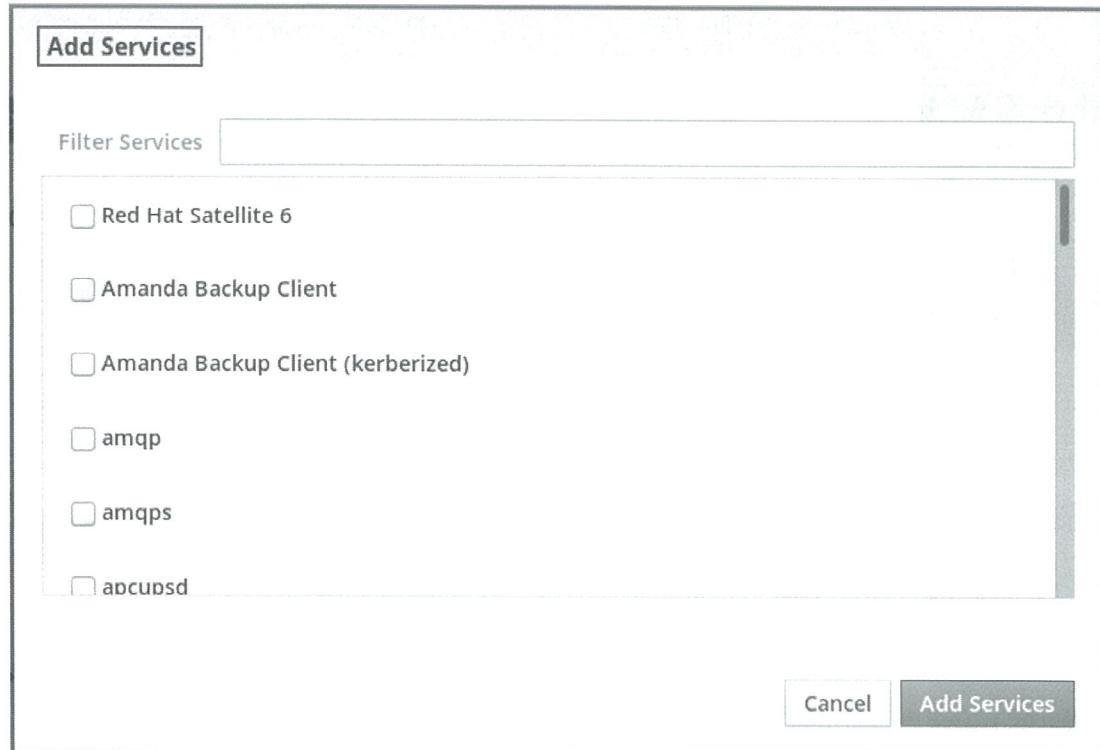


图 11.4: Web 控制台添加服务界面

要选择服务，请滚动列表或在 Filter Services 文本框中输入选择内容。在以下示例中，向搜索文本框中输入字符串 **http**，以查找包含该字符串的服务，即 Web 相关服务。选中服务左侧的复选框，以允许这些服务通过防火墙。单击 Add Services 按钮完成此过程。



图 11.5: Web 控制台服务筛选搜索

界面将返回到 Firewall Allowed Services 页面，在这里可以查看更新后允许的服务列表。

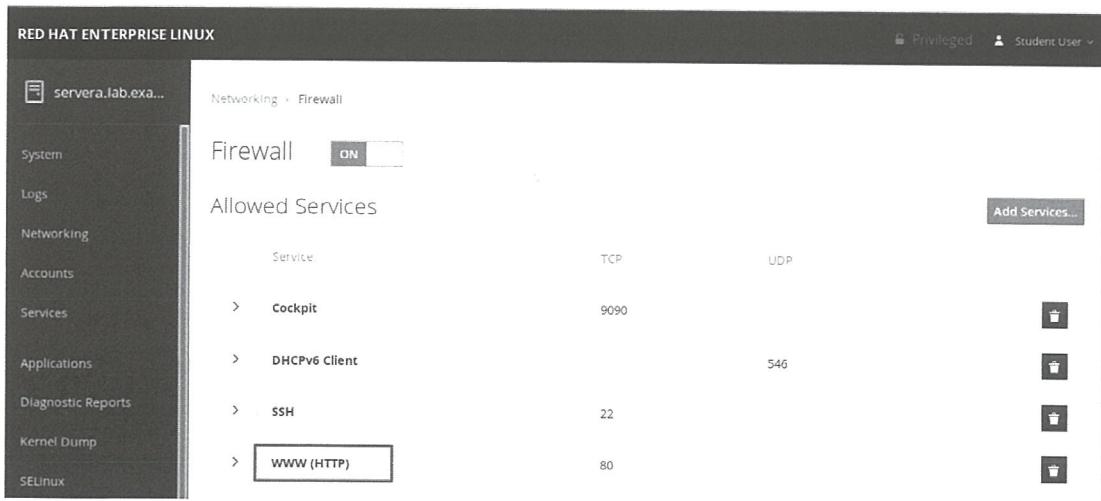


图 11.6: Web 控制台服务列表

从命令行配置防火墙

firewall-cmd 命令将会与 **firewalld** 动态防火墙管理器进行交互。它是作为主 firewalld 软件包的一部分安装的，可用于倾向于使用命令行的管理员，在没有图形环境的系统上工作，或编写有关防火墙设置的脚本。

下表列出一些常用 **firewall-cmd** 命令及其说明。请注意，除非另有指定，否则几乎所有命令都作用于运行时配置，当指定 **--permanent** 选项时除外。如果指定了 **--permanent** 选项，还必须通过运行 **firewall-cmd --reload** 命令来激活设置，它将读取当前的永久配置并将其作为新的运行时配置来应用。列出的许多命令都采用 **--zone=ZONE** 选项来确定所影响的区域。如果需要子网掩码，请使用 CIDR 表示法，如 192.168.1/24。

| FIREWALL-CMD 命令 | 说明 |
|--|---|
| --get-default-zone | 查询当前默认区域。 |
| --set-default-zone=ZONE | 设置默认区域。此命令会同时更改运行时配置和永久配置。 |
| --get-zones | 列出所有可用区域。 |
| --get-active-zones | 列出当前正在使用的所有区域（具有关联的接口或源）及其接口和源信息。 |
| --add-source=CIDR [--zone=ZONE] | 将来自 IP 地址或网络/子网掩码的所有流量路由到指定区域。如果未提供 --zone= 选项，则使用默认区域。 |
| --remove-source=CIDR [--zone=ZONE] | 从区域中删除用于路由来自 IP 地址或网络/子网掩码的所有流量的规则。如果未提供 zone= 选项，则使用默认区域。 |
| --add-interface=INTERFACE [--zone=ZONE] | 将来自 INTERFACE 的所有流量路由到指定区域。如果未提供 --zone= 选项，则使用默认区域。 |

| FIREWALL-CMD 命令 | 说明 |
|---|---|
| <code>--change-interface=INTERFACE [--zone=ZONE]</code> | 将接口与 ZONE 而非其当前区域关联。如果未提供 <code>--zone=</code> 选项，则使用默认区域。 |
| <code>--list-all [--zone=ZONE]</code> | 列出 ZONE 的所有已配置接口、源、服务和端口。如果未提供 <code>--zone=</code> 选项，则使用默认区域。 |
| <code>--list-all-zones</code> | 检索所有区域的所有信息（接口、源、端口、服务）。 |
| <code>--add-service=SERVICE [--zone=ZONE]</code> | 允许到 SERVICE 的流量。如果未提供 <code>--zone=</code> 选项，则使用默认区域。 |
| <code>--add-port=PORT/PROTOCOL [--zone=ZONE]</code> | 允许到 PORT/PROTOCOL 端口的流量。如果未提供 <code>--zone=</code> 选项，则使用默认区域。 |
| <code>--remove-service=SERVICE [--zone=ZONE]</code> | 从区域的允许列表中删除 SERVICE。如果未提供 <code>--zone=</code> 选项，则使用默认区域。 |
| <code>--remove-port=PORT/PROTOCOL [--zone=ZONE]</code> | 从区域的允许列表中删除 PORT/PROTOCOL 端口。如果未提供 <code>--zone=</code> 选项，则使用默认区域。 |
| <code>--reload</code> | 丢弃运行时配置并应用持久配置。 |

以下命令示例会将默认区域设置为 dmz，将来自 192.168.0.0/24 网络的所有流量都分配给 internal 区域，并在 internal 区域上打开用于 mysql 服务的网络端口。

```
[root@host ~]# firewall-cmd --set-default-zone=dmz
[root@host ~]# firewall-cmd --permanent --zone=internal \
--add-source=192.168.0.0/24
[root@host ~]# firewall-cmd --permanent --zone=internal --add-service=mysql
[root@host ~]# firewall-cmd --reload
```



注意

对于 `firewalld` 的基本语法不够的情况，还可以添加富规则（一种更具表达力的语法）来编写复杂的规则。如果富规则语法也不够，您还可以使用直接配置规则，这是与 `firewalld` 规则混合使用的原始 `nft` 语法。

这些高级模式超出了本章的讨论范围。



参考文献

`firewall-cmd(1)`、`firewalld(1)`、`firewalld.zone(5)`、`firewalld.zones(5)` 和 `nft(8)` man page

► 指导练习

管理服务器防火墙

在本练习中，您将通过调整系统防火墙规则（使用 `firewalld`）来控制对系统服务的访问。

成果

您应能够配置防火墙规则来控制对服务的访问。

在你开始之前

在 `workstation` 上，以 `student` 用户身份并使用密码 `student` 进行登录。

从 `workstation`，运行 `lab netsecurity-firewalls start` 命令。该命令将运行一个起始脚本，以确定 `servera` 主机是否可从网络访问。

```
[student@workstation ~]$ lab netsecurity-firewalls start
```

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

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

- 2. 在 `servera` 系统上，确保 `httpd` 和 `mod_ssl` 软件包都已安装。这些软件包用于提供您将使用防火墙保护的 Apache Web 服务器，以及使该 Web 服务器能够通过 SSL 提供内容的必要扩展。

```
[student@servera ~]$ sudo yum install httpd mod_ssl  
[sudo] password for student: student  
...output omitted...  
Is this ok [y/N]: y  
...output omitted...  
Complete!
```

- 3. 以 `servera` 上的 `student` 用户身份，创建 `/var/www/html/index.html` 文件。添加一行文本，内容如下：`I am servera.`。

```
[student@servera ~]$ sudo bash -c \  
"echo 'I am servera.' > /var/www/html/index.html"
```

- 4. 在 `servera` 系统上，启动并启用 `httpd` 服务。

```
[student@servera ~]$ sudo systemctl enable --now httpd  
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/  
lib/systemd/system/httpd.service.
```

► 5. 从 servera 退出。

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

► 6. 从 workstation 尝试同时使用明文端口 80/TCP 和 SSL 封装端口 443/TCP 来访问 servera 上的 Web 服务器。两种尝试都应以失败告终。

6.1. 此命令应失败：

```
[student@workstation ~]$ curl -k http://servera.lab.example.com  
curl: (7) Failed to connect to servera.lab.example.com port 80: No route to host
```

6.2. 此命令也应失败：

```
[student@workstation ~]$ curl -k https://servera.lab.example.com  
curl: (7) Failed to connect to servera.lab.example.com port 443: No route to host
```

► 7. 以 student 用户身份登录 servera。

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

► 8. 在 servera 上，确保 nftables 服务被屏蔽，而 firewalld 服务处于启用和运行状态。

8.1. 确定 nftables 服务的状态是否为 masked。

```
[student@servera ~]$ sudo systemctl status nftables  
[sudo] password for student: student  
● nftables.service - Netfilter Tables  
  Loaded: loaded (/usr/lib/systemd/system/nftables.service; disabled; vendor  
  preset: disabled)  
  Active: inactive (dead)  
    Docs: man:nft(8)
```

结果显示，nftables 处于禁用和非活动状态，但未被屏蔽。运行以下命令以屏蔽服务。

```
[student@servera ~]$ sudo systemctl mask nftables  
Created symlink /etc/systemd/system/nftables.service → /dev/null.
```

8.2. 验证 nftables 服务的状态是否为 masked。

```
[student@servera ~]$ sudo systemctl status nftables
● nftables.service
  Loaded: masked (Reason: Unit nftables.service is masked.)
  Active: inactive (dead)
```

8.3. 验证 firewalld 服务的状态是否为已启用且正在运行。

```
[student@servera ~]$ sudo systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
  Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor
preset: enabled)
  Active: active (running) since Wed 2019-05-22 15:36:02 CDT; 5min ago
    Docs: man:firewalld(1)
 Main PID: 703 (firewalld)
   Tasks: 2 (limit: 11405)
  Memory: 29.8M
    CGroup: /system.slice/firewalld.service
           └─703 /usr/libexec/platform-python -s /usr/sbin/firewalld --nofork --
nopid

May 22 15:36:01 jegui.ilt.example.com systemd[1]: Starting firewalld - dynamic
firewall daemon...
May 22 15:36:02 jegui.ilt.example.com systemd[1]: Started firewalld - dynamic
firewall daemon.
```

8.4. 从 servera 退出。

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

► 9. 从 workstation 打开 Firefox 并登录到 servera 上运行的 Web 控制台，以便将 httpd 服务添加至 public 网络区域。

- 9.1. 打开 Firefox 并浏览到 <https://servera.lab.example.com:9090>，以访问 Web 控制台。将 servera 使用的自签名证书添加为例外，接受它。
- 9.2. 选中 Reuse my password for privileged tasks 旁边的复选框，以确保具有管理权限。以 student 用户身份并使用 student 作为密码登录。
- 9.3. 单击左侧导航栏中的 Networking。
- 9.4. 单击主 Networking 页面中的 Firewall 链接。
- 9.5. 单击 Firewall 页面右上角的 Add Services... 按钮。
- 9.6. 在 Add Services 用户界面中，向下滚动或使用 Filter Services 找到并选中 Secure WWW (HTTPS) 服务旁边的复选框。
- 9.7. 单击 Add Services 用户界面右下角的 Add Services 按钮。

- 10. 返回 workstation 上的终端并尝试查看 servera 的 Web 服务器内容，以此来验证您的工作。

10.1. 此命令应失败：

```
[student@workstation ~]$ curl -k http://servera.lab.example.com  
curl: (7) Failed to connect to servera.lab.example.com port 80: No route to host
```

10.2. 此命令应成功：

```
[student@workstation ~]$ curl -k https://servera.lab.example.com  
I am servera.
```



注意

如果使用 Firefox 连接 Web 服务器，它将在成功绕过防火墙时提示您对主机证书进行验证。

完成

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

```
[student@workstation ~]$ lab netsecurity-firewalls finish
```

本引导式练习到此结束。

控制 SELINUX 端口标记

培训目标

学完本节后，您应能够验证网络端口是否具有正确的 SELinux 类型，以便服务能够与其绑定。

SELINUX 端口标记

SELinux 不仅仅是进行文件和进程标记。SELinux 策略还严格实施网络流量。SELinux 用来控制网络流量的其中一种方法是标记网络端口；例如，在 **targeted** 策略中，端口 **22/TCP** 具有标签 **ssh_port_t** 与其相关联。默认 HTTP 端口 **80/TCP** 和 **443/TCP** 具有标签 **http_port_t** 与其相关联。

当某个进程希望侦听端口时，SELinux 将检查是否允许与该进程（域）相关联的标签绑定该端口标签。这可以阻止恶意服务控制本应由其他（合法）网络服务使用的端口。

管理 SELINUX 端口标记

如果您决定在非标准端口上运行服务，SELinux 几乎肯定会拦截此流量。在这种情况下，您必须更新 SELinux 端口标签。在某些情况下，**targeted** 策略已经通过可以使用的类型标记了端口；例如，由于端口 **8008/TCP** 通常用于 Web 应用程序，此端口已使用 **http_port_t**（Web 服务器的默认端口类型）进行标记。

列出端口标签

要获取所有当前端口标签分配的概述，请运行 **semanage port -l** 命令。**-l** 选项将以下列形式列出所有当前分配：

```
port_label_t      tcp|udp      comma-separated,list,of,ports
```

输出示例：

```
[root@host ~]# semanage port -l
...output omitted...
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
...output omitted...
```

要优化搜索，请使用 **grep** 命令：

```
[root@host ~]# semanage port -l | grep ftp
ftp_data_port_t        tcp     20
ftp_port_t              tcp     21, 989, 990
ftp_port_t              udp     989, 990
tftp_port_t             udp     69
```

请注意，一个端口标签可能会在输出中出现两次，一次是针对 TCP，一次是针对 UDP。

管理端口标签

使用 **semanage** 命令可以分配新端口标签、删除端口标签或修改现有端口标签。



重要

Linux 发行版中的大多数标准服务都提供了一个 SELinux 策略模块，用于在端口上设置标签。您无法使用 **semanage** 来更改这些端口上的标签；要更改这些标签，您需要替换策略模块。编写和生成策略模块不属于本课程的范围。

要向现有端口标签（类型）中添加端口，请使用以下语法。**-a** 将添加新端口标签，**-t** 表示类型，**-p** 表示协议。

```
[root@host ~]# semanage port -a -t port_label -p tcp|udp PORTNUMBER
```

例如，要允许 gopher 服务侦听端口 **71/TCP**：

```
[root@host ~]# semanage port -a -t gopher_port_t -p tcp 71
```

要查看对默认策略的本地更改，管理员可以在 **semanage** 命令中添加 **-C** 选项。

| SELinux Port Type | Proto | Port Number |
|-------------------|-------|-------------|
| gopher_port_t | tcp | 71 |



注意

targeted 策略随附了大量端口类型。

在 selinux-policy-doc 软件包中可以找到特定于服务的 SELinux man page，其中包含了有关 SELinux 类型、布尔值和端口类型的文档。如果您的系统上尚未安装这些 man page，请遵循以下过程：

```
[root@host ~]# yum -y install selinux-policy-doc
[root@host ~]# man -k _selinux
```

删除端口标签

删除自定义端口标签的语法与添加端口标签的语法相同，但不是使用 **-a** 选项（表示添加），而是使用 **-d** 选项（表示删除）。

例如，要删除端口 **71/TCP** 与 **gopher_port_t** 的绑定：

```
[root@host ~]# semanage port -d -t gopher_port_t -p tcp 71
```

修改端口绑定

要更改端口绑定（可能是因为需求发生改变），请使用 **-m**（修改）选项。这种流程比删除旧绑定并添加新绑定更高效。

例如，要将端口 71/TCP 从 **gopher_port_t** 修改为 **http_port_t**，管理员可以使用以下命令：

```
[root@server ~]# semanage port -m -t http_port_t -p tcp 71
```

和以前一样，使用 **semanage** 命令可以查看修改内容。

```
[root@server ~]# semanage port -l -C
SELinux Port Type          Proto    Port Number
http_port_t                 tcp      71
[root@server ~]# 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      71, 80, 81, 443, 488, 8008, 8009, 8443,
                               9000
pegasus_http_port_t         tcp      5988
pegasus_https_port_t        tcp      5989
```



参考文献

semanage(8)、**semanage-port(8)** 和 ***_selinux(8)** man page

► 指导练习

控制 SELINUX 端口标记

在本实验中，您要将 `servera` 系统配置为在非标准端口上允许 HTTP 访问。

成果：

您将配置在 `servera` 上运行的 Web 服务器，以便成功地在非标准端口上提供内容。

在你开始之前

在 `workstation` 上，以 `student` 用户身份并使用密码 `student` 进行登录。

在 `workstation` 上，运行 `lab netsecurity-ports start` 命令。此命令将运行一个起始脚本，它将确定 `servera` 计算机是否可从网络访问。此外，它还将安装 `httpd` 服务并配置 `servera` 的防火墙，以允许进行 http 连接。

```
[student@workstation ~]$ lab netsecurity-ports start
```

您的组织正在部署一个新的自定义 Web 应用程序。此 Web 应用程序在非标准端口上运行；在本例中为 **82/TCP**。

一位初级管理员已在您的 `servera` 上对此应用进行了配置。但是，Web 服务器内容还是无法访问。

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

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

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

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

- 3. 尝试通过重启 `httpd` 服务来修复 Web 内容的问题。

- 3.1. 使用 `systemctl` 命令重启 `httpd.service`。此命令预计会失败。

```
[root@servera ~]# systemctl restart httpd.service  
Job for httpd.service failed because the control process exited with error code.  
See "systemctl status httpd.service" and "journalctl -xe" for details.
```

3.2. 使用 `systemctl status -l` 命令显示 httpd 服务的状态。请注意 `permission denied` 错误。

```
[root@servera ~]# systemctl status -l httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: failed (Result: exit-code) since Mon 2019-04-08 14:23:29 CEST; 3min 33s ago
     Docs: man:httpd.service(8)
   Process: 28078 ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND (code=exited, status=1/FAILURE)
   Main PID: 28078 (code=exited, status=1/FAILURE)
     Status: "Reading configuration..."

Apr 08 14:23:29 servera.lab.example.com systemd[1]: Starting The Apache HTTP Server...
Apr 08 14:23:29 servera.lab.example.com httpd[28078]: (13)Permission denied: AH00072: make_sock: could not bind to address [::]:82
Apr 08 14:23:29 servera.lab.example.com httpd[28078]: (13)Permission denied: AH00072: make_sock: could not bind to address 0.0.0.0:82
Apr 08 14:23:29 servera.lab.example.com httpd[28078]: no listening sockets available, shutting down
Apr 08 14:23:29 servera.lab.example.com httpd[28078]: AH00015: Unable to open logs
Apr 08 14:23:29 servera.lab.example.com systemd[1]: httpd.service: Main process exited, code=exited, status=1/FAILURE
Apr 08 14:23:29 servera.lab.example.com systemd[1]: httpd.service: Failed with result 'exit-code'.
Apr 08 14:23:29 servera.lab.example.com systemd[1]: Failed to start The Apache HTTP Server.
```

3.3. 使用 `sealert` 命令检查 SELinux 是否在阻止 httpd 绑定到端口 82/TCP。

```
[root@servera ~]# sudo sealert -a /var/log/audit/audit.log
100% done
found 1 alerts in /var/log/audit/audit.log
-----
SELinux is preventing /usr/sbin/httpd from name_bind access on the tcp_socket port 82.

***** Plugin bind_ports (99.5 confidence) suggests *****

If you want to allow /usr/sbin/httpd to bind to network port 82
Then you need to modify the port type.
Do
# semanage port -a -t PORT_TYPE -p tcp 82
  where PORT_TYPE is one of the following: http_cache_port_t, http_port_t,
  jboss_management_port_t, jboss.messaging_port_t, ntop_port_t, puppet_port_t,
...output omitted...
```