Legal Notice

Copyright © 2017 Red Hat, Inc.

This document is licensed by Red Hat under the Creative Commons Attribution-ShareAlike 3.0 Unported License. If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This book assists users and administrators in learning the processes and practices of securing workstations and servers against local and remote intrusion, exploitation, and malicious activity. Focused on Red Hat Enterprise Linux but detailing concepts and techniques valid for all Linux systems, this guide details the planning and the tools involved in creating a secured computing environment for the data center, workplace, and home. With proper administrative knowledge, vigilance, and tools, systems running Linux can be both fully functional and secured from most common intrusion and exploit methods.
# Table of Contents

## CHAPTER 1. OVERVIEW OF SECURITY TOPICS ........................................ 4
1.1. WHAT IS COMPUTER SECURITY? ............................................ 4
1.2. SECURITY CONTROLS .................................................. 5
1.3. VULNERABILITY ASSESSMENT ......................................... 6
1.4. SECURITY THREATS ................................................... 10
1.5. COMMON EXPLOITS AND ATTACKS ..................................... 13

## CHAPTER 2. SECURITY TIPS FOR INSTALLATION ............................... 17
2.1. SECURING BIOS ...................................................... 17
2.2. PARTITIONING THE DISK ............................................ 17
2.3. INSTALLING THE MINIMUM AMOUNT OF PACKAGES REQUIRED .... 18
2.4. Restricting network connectivity during the installation process ....... 19
2.5. POST-INSTALLATION PROCEDURES .................................. 19
2.6. ADDITIONAL RESOURCES ............................................ 19

## CHAPTER 3. KEEPING YOUR SYSTEM UP-TO-DATE ............................... 21
3.1. MAINTAINING INSTALLED SOFTWARE .................................. 21
3.2. USING THE RED HAT CUSTOMER PORTAL ............................ 25
3.3. ADDITIONAL RESOURCES ............................................ 26

## CHAPTER 4. HARDENING YOUR SYSTEM WITH TOOLS AND SERVICES .... 28
4.1. DESKTOP SECURITY ................................................. 28
4.2. CONTROLLING ROOT ACCESS ......................................... 37
4.3. SECURING SERVICES ................................................ 44
4.4. SECURING NETWORK ACCESS ........................................ 65
4.5. SECURING DNS TRAFFIC WITH DNSSEC ............................ 73
4.6. SECURING VIRTUAL PRIVATE NETWORKS (VPNS) .................... 83
4.7. USING OPENSSL ................................................... 94
4.8. USING STUNNEL ................................................... 99
4.9. ENCRYPTION ....................................................... 102
4.10. USING NETWORK-BOUND DISK ENCRYPTION ....................... 118
4.11. CHECKING INTEGRITY WITH AIDE ................................ 125
4.12. USING USBGUARD .................................................. 127
4.13. HARDENING TLS CONFIGURATION .................................. 132
4.14. USING SHARED SYSTEM CERTIFICATES ............................ 140
4.15. USING MACSEC ................................................... 143
4.16. REMOVING DATA SECURELY USING SCRUB ....................... 143

## CHAPTER 5. USING FIREWALLS .................................................. 146
5.1. INTRODUCTION TO FIREWALLD ....................................... 146
5.2. INSTALLING FIREWALLD ............................................. 151
5.3. CONFIGURING FIREWALLD .......................................... 152
5.4. USING THE IPTABLES SERVICE ...................................... 179
5.5. ADDITIONAL RESOURCES ............................................ 187

## CHAPTER 6. SYSTEM AUDITING .................................................. 189
Use Cases ......................................................................... 190
6.1. AUDIT SYSTEM ARCHITECTURE ...................................... 191
6.2. INSTALLING THE AUDIT PACKAGES ................................ 191
6.3. CONFIGURING THE AUDIT SERVICE .................................. 191
6.4. STARTING THE AUDIT SERVICE ...................................... 193
6.5. DEFINING AUDIT RULES ............................................ 194
6.6. UNDERSTANDING AUDIT LOG FILES ............................... 200
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.7. SEARCHING THE AUDIT LOG FILES</td>
<td>206</td>
</tr>
<tr>
<td>6.8. CREATING AUDIT REPORTS</td>
<td>206</td>
</tr>
<tr>
<td>6.9. ADDITIONAL RESOURCES</td>
<td>207</td>
</tr>
<tr>
<td>CHAPTER 7. COMPLIANCE AND VULNERABILITY SCANNING WITH OPENSCAP</td>
<td>209</td>
</tr>
<tr>
<td>7.1. SECURITY COMPLIANCE IN RED HAT ENTERPRISE LINUX</td>
<td>209</td>
</tr>
<tr>
<td>7.2. DEFINING COMPLIANCE POLICY</td>
<td>209</td>
</tr>
<tr>
<td>7.3. USING SCAP WORKBENCH</td>
<td>218</td>
</tr>
<tr>
<td>7.4. USING OSCAP</td>
<td>226</td>
</tr>
<tr>
<td>7.5. USING OPENSCAP WITH DOCKER</td>
<td>233</td>
</tr>
<tr>
<td>7.6. USING OPENSCAP WITH ATOMIC</td>
<td>235</td>
</tr>
<tr>
<td>7.7. USING OPENSCAP WITH RED HAT SATELLITE</td>
<td>237</td>
</tr>
<tr>
<td>7.8. PRACTICAL EXAMPLES</td>
<td>237</td>
</tr>
<tr>
<td>7.9. ADDITIONAL RESOURCES</td>
<td>238</td>
</tr>
<tr>
<td>CHAPTER 8. FEDERAL STANDARDS AND REGULATIONS</td>
<td>240</td>
</tr>
<tr>
<td>8.1. FEDERAL INFORMATION PROCESSING STANDARD (FIPS)</td>
<td>240</td>
</tr>
<tr>
<td>8.2. NATIONAL INDUSTRIAL SECURITY PROGRAM OPERATING MANUAL (NISPOM)</td>
<td>242</td>
</tr>
<tr>
<td>8.3. PAYMENT CARD INDUSTRY DATA SECURITY STANDARD (PCI DSS)</td>
<td>242</td>
</tr>
<tr>
<td>8.4. SECURITY TECHNICAL IMPLEMENTATION GUIDE</td>
<td>242</td>
</tr>
<tr>
<td>APPENDIX A. ENCRYPTION STANDARDS</td>
<td>243</td>
</tr>
<tr>
<td>A.1. SYNCHRONOUS ENCRYPTION</td>
<td>243</td>
</tr>
<tr>
<td>A.2. PUBLIC-KEY ENCRYPTION</td>
<td>243</td>
</tr>
<tr>
<td>APPENDIX B. AUDIT SYSTEM REFERENCE</td>
<td>247</td>
</tr>
<tr>
<td>B.1. AUDIT EVENT FIELDS</td>
<td>247</td>
</tr>
<tr>
<td>B.2. AUDIT RECORD TYPES</td>
<td>253</td>
</tr>
<tr>
<td>APPENDIX C. REVISION HISTORY</td>
<td>262</td>
</tr>
</tbody>
</table>
CHAPTER 1. OVERVIEW OF SECURITY TOPICS

Due to the increased reliance on powerful, networked computers to help run businesses and keep track of our personal information, entire industries have been formed around the practice of network and computer security. Enterprises have solicited the knowledge and skills of security experts to properly audit systems and tailor solutions to fit the operating requirements of their organization. Because most organizations are increasingly dynamic in nature, their workers are accessing critical company IT resources locally and remotely, hence the need for secure computing environments has become more pronounced.

Unfortunately, many organizations (as well as individual users) regard security as more of an afterthought, a process that is overlooked in favor of increased power, productivity, convenience, ease of use, and budgetary concerns. Proper security implementation is often enacted postmortem — after an unauthorized intrusion has already occurred. Taking the correct measures prior to connecting a site to an untrusted network, such as the Internet, is an effective means of thwarting many attempts at intrusion.

NOTE

This document makes several references to files in the /lib directory. When using 64-bit systems, some of the files mentioned may instead be located in /lib64.

1.1. WHAT IS COMPUTER SECURITY?

Computer security is a general term that covers a wide area of computing and information processing. Industries that depend on computer systems and networks to conduct daily business transactions and access critical information regard their data as an important part of their overall assets. Several terms and metrics have entered our daily business vocabulary, such as total cost of ownership (TCO), return on investment (ROI), and quality of service (QoS). Using these metrics, industries can calculate aspects such as data integrity and high-availability (HA) as part of their planning and process management costs. In some industries, such as electronic commerce, the availability and trustworthiness of data can mean the difference between success and failure.

1.1.1. Standardizing Security

Enterprises in every industry rely on regulations and rules that are set by standards-making bodies such as the American Medical Association (AMA) or the Institute of Electrical and Electronics Engineers (IEEE). The same ideals hold true for information security. Many security consultants and vendors agree upon the standard security model known as CIA, or Confidentiality, Integrity, and Availability. This three-tiered model is a generally accepted component to assessing risks of sensitive information and establishing security policy. The following describes the CIA model in further detail:

- Confidentiality — Sensitive information must be available only to a set of pre-defined individuals. Unauthorized transmission and usage of information should be restricted. For example, confidentiality of information ensures that a customer's personal or financial information is not obtained by an unauthorized individual for malicious purposes such as identity theft or credit fraud.

- Integrity — Information should not be altered in ways that render it incomplete or incorrect. Unauthorized users should be restricted from the ability to modify or destroy sensitive information.
Availability — Information should be accessible to authorized users any time that it is needed. Availability is a warranty that information can be obtained with an agreed-upon frequency and timeliness. This is often measured in terms of percentages and agreed to formally in Service Level Agreements (SLAs) used by network service providers and their enterprise clients.

1.2. SECURITY CONTROLS

Computer security is often divided into three distinct master categories, commonly referred to as controls:

- Physical
- Technical
- Administrative

These three broad categories define the main objectives of proper security implementation. Within these controls are sub-categories that further detail the controls and how to implement them.

1.2.1. Physical Controls

Physical control is the implementation of security measures in a defined structure used to deter or prevent unauthorized access to sensitive material. Examples of physical controls are:

- Closed-circuit surveillance cameras
- Motion or thermal alarm systems
- Security guards
- Picture IDs
- Locked and dead-bolted steel doors
- Biometrics (includes fingerprint, voice, face, iris, handwriting, and other automated methods used to recognize individuals)

1.2.2. Technical Controls

Technical controls use technology as a basis for controlling the access and usage of sensitive data throughout a physical structure and over a network. Technical controls are far-reaching in scope and encompass such technologies as:

- Encryption
- Smart cards
- Network authentication
- Access control lists (ACLs)
- File integrity auditing software
1.2.3. Administrative Controls

Administrative controls define the human factors of security. They involve all levels of personnel within an organization and determine which users have access to what resources and information by such means as:

- Training and awareness
- Disaster preparedness and recovery plans
- Personnel recruitment and separation strategies
- Personnel registration and accounting

1.3. VULNERABILITY ASSESSMENT

Given time, resources, and motivation, an attacker can break into nearly any system. All of the security procedures and technologies currently available cannot guarantee that any systems are completely safe from intrusion. Routers help secure gateways to the Internet. Firewalls help secure the edge of the network. Virtual Private Networks safely pass data in an encrypted stream. Intrusion detection systems warn you of malicious activity. However, the success of each of these technologies is dependent upon a number of variables, including:

- The expertise of the staff responsible for configuring, monitoring, and maintaining the technologies.
- The ability to patch and update services and kernels quickly and efficiently.
- The ability of those responsible to keep constant vigilance over the network.

Given the dynamic state of data systems and technologies, securing corporate resources can be quite complex. Due to this complexity, it is often difficult to find expert resources for all of your systems. While it is possible to have personnel knowledgeable in many areas of information security at a high level, it is difficult to retain staff who are experts in more than a few subject areas. This is mainly because each subject area of information security requires constant attention and focus. Information security does not stand still.

A vulnerability assessment is an internal audit of your network and system security; the results of which indicate the confidentiality, integrity, and availability of your network (as explained in Section 1.1.1, “Standardizing Security”). Typically, vulnerability assessment starts with a reconnaissance phase, during which important data regarding the target systems and resources is gathered. This phase leads to the system readiness phase, whereby the target is essentially checked for all known vulnerabilities. The readiness phase culminates in the reporting phase, where the findings are classified into categories of high, medium, and low risk; and methods for improving the security (or mitigating the risk of vulnerability) of the target are discussed.

If you were to perform a vulnerability assessment of your home, you would likely check each door to your home to see if they are closed and locked. You would also check every window, making sure that they closed completely and latch correctly. This same concept applies to systems, networks, and electronic data. Malicious users are the thieves and vandals of your data. Focus on their tools, mentality, and motivations, and you can then react swiftly to their actions.

1.3.1. Defining Assessment and Testing
Vulnerability assessments may be broken down into one of two types: *outside looking in* and *inside looking around*.

When performing an outside-looking-in vulnerability assessment, you are attempting to compromise your systems from the outside. Being external to your company provides you with the cracker's viewpoint. You see what a cracker sees — publicly-routable IP addresses, systems on your DMZ, external interfaces of your firewall, and more. DMZ stands for “demilitarized zone”, which corresponds to a computer or small subnetwork that sits between a trusted internal network, such as a corporate private LAN, and an untrusted external network, such as the public Internet. Typically, the DMZ contains devices accessible to Internet traffic, such as Web (HTTP) servers, FTP servers, SMTP (e-mail) servers and DNS servers.

When you perform an inside-looking-around vulnerability assessment, you are at an advantage since you are internal and your status is elevated to trusted. This is the viewpoint you and your co-workers have once logged on to your systems. You see print servers, file servers, databases, and other resources.

There are striking distinctions between the two types of vulnerability assessments. Being internal to your company gives you more privileges than an outsider. In most organizations, security is configured to keep intruders out. Very little is done to secure the internals of the organization (such as departmental firewalls, user-level access controls, and authentication procedures for internal resources). Typically, there are many more resources when looking around inside as most systems are internal to a company. Once you are outside the company, your status is untrusted. The systems and resources available to you externally are usually very limited.

Consider the difference between vulnerability assessments and *penetration tests*. Think of a vulnerability assessment as the first step to a penetration test. The information gleaned from the assessment is used for testing. Whereas the assessment is undertaken to check for holes and potential vulnerabilities, the penetration testing actually attempts to exploit the findings.

Assessing network infrastructure is a dynamic process. Security, both information and physical, is dynamic. Performing an assessment shows an overview, which can turn up false positives and false negatives. A false positive is a result, where the tool finds vulnerabilities which in reality do not exist. A false negative is when it omits actual vulnerabilities.

Security administrators are only as good as the tools they use and the knowledge they retain. Take any of the assessment tools currently available, run them against your system, and it is almost a guarantee that there are some false positives. Whether by program fault or user error, the result is the same. The tool may find false positives, or, even worse, false negatives.

Now that the difference between a vulnerability assessment and a penetration test is defined, take the findings of the assessment and review them carefully before conducting a penetration test as part of your new best practices approach.
WARNING

Do not attempt to exploit vulnerabilities on production systems. Doing so can have adverse effects on productivity and efficiency of your systems and network.

The following list examines some of the benefits to performing vulnerability assessments.

- Creates proactive focus on information security.
- Finds potential exploits before crackers find them.
- Results in systems being kept up to date and patched.
- Promotes growth and aids in developing staff expertise.
- Abates financial loss and negative publicity.

1.3.2. Establishing a Methodology for Vulnerability Assessment

To aid in the selection of tools for a vulnerability assessment, it is helpful to establish a vulnerability assessment methodology. Unfortunately, there is no predefined or industry approved methodology at this time; however, common sense and best practices can act as a sufficient guide.

What is the target? Are we looking at one server, or are we looking at our entire network and everything within the network? Are we external or internal to the company? The answers to these questions are important as they help determine not only which tools to select but also the manner in which they are used.

To learn more about establishing methodologies, see the following website:

- https://www.owasp.org/ — The Open Web Application Security Project

1.3.3. Vulnerability Assessment Tools

An assessment can start by using some form of an information-gathering tool. When assessing the entire network, map the layout first to find the hosts that are running. Once located, examine each host individually. Focusing on these hosts requires another set of tools. Knowing which tools to use may be the most crucial step in finding vulnerabilities.

Just as in any aspect of everyday life, there are many different tools that perform the same job. This concept applies to performing vulnerability assessments as well. There are tools specific to operating systems, applications, and even networks (based on the protocols used). Some tools are free; others are not. Some tools are intuitive and easy to use, while others are cryptic and poorly documented but have features that other tools do not.

Finding the right tools may be a daunting task and, in the end, experience counts. If possible, set up a test lab and try out as many tools as you can, noting the strengths and weaknesses of each. Review the README file or man page for the tools. Additionally, look to
the Internet for more information, such as articles, step-by-step guides, or even mailing lists specific to the tools.

The tools discussed below are just a small sampling of the available tools.

1.3.3.1. Scanning Hosts with Nmap

Nmap is a popular tool that can be used to determine the layout of a network. Nmap has been available for many years and is probably the most often used tool when gathering information. An excellent manual page is included that provides detailed descriptions of its options and usage. Administrators can use Nmap on a network to find host systems and open ports on those systems.

Nmap is a competent first step in vulnerability assessment. You can map out all the hosts within your network and even pass an option that allows Nmap to attempt to identify the operating system running on a particular host. Nmap is a good foundation for establishing a policy of using secure services and restricting unused services.

To install Nmap, run the `yum install nmap` command as the root user.

1.3.3.1.1. Using Nmap

Nmap can be run from a shell prompt by typing the `nmap` command followed by the host name or IP address of the machine to scan:

```
nmap <hostname>
```

For example, to scan a machine with host name `foo.example.com`, type the following at a shell prompt:

```
~]$ nmap foo.example.com
```

The results of a basic scan (which could take up to a few minutes, depending on where the host is located and other network conditions) look similar to the following:

```
Interesting ports on foo.example.com:
Not shown: 1710 filtered ports
 PORT   STATE SERVICE
22/tcp open  ssh
53/tcp open  domain
80/tcp open  http
113/tcp closed auth
```

Nmap tests the most common network communication ports for listening or waiting services. This knowledge can be helpful to an administrator who wants to close unnecessary or unused services.

For more information about using Nmap, see the official homepage at the following URL:

http://www.insecure.org/

1.3.3.2. Nessus

Nessus is a full-service security scanner. The plug-in architecture of Nessus allows users to customize it for their systems and networks. As with any scanner, Nessus is only as good
as the signature database it relies upon. Fortunately, **Nessus** is frequently updated and features full reporting, host scanning, and real-time vulnerability searches. Remember that there could be false positives and false negatives, even in a tool as powerful and as frequently updated as **Nessus**.

**NOTE**

The **Nessus** client and server software requires a subscription to use. It has been included in this document as a reference to users who may be interested in using this popular application.

For more information about **Nessus**, see the official website at the following URL:

http://www.nessus.org/

### 1.3.3.3. OpenVAS

**OpenVAS** *(Open Vulnerability Assessment System)* is a set of tools and services that can be used to scan for vulnerabilities and for a comprehensive vulnerability management. The **OpenVAS** framework offers a number of web-based, desktop, and command line tools for controlling the various components of the solution. The core functionality of **OpenVAS** is provided by a security scanner, which makes use of over 33 thousand daily-updated Network Vulnerability Tests (NVT). Unlike **Nessus** (see Section 1.3.3.2, “**Nessus**”), **OpenVAS** does not require any subscription.

For more information about **OpenVAS**, see the official website at the following URL:

http://www.openvas.org/

### 1.3.3.4. Nikto

**Nikto** is an excellent *common gateway interface* (CGI) script scanner. **Nikto** not only checks for CGI vulnerabilities but does so in an evasive manner, so as to elude intrusion-detection systems. It comes with thorough documentation which should be carefully reviewed prior to running the program. If you have web servers serving CGI scripts, **Nikto** can be an excellent resource for checking the security of these servers.

More information about **Nikto** can be found at the following URL:

http://cirt.net/nikto2

### 1.4. SECURITY THREATS

#### 1.4.1. Threats to Network Security

Bad practices when configuring the following aspects of a network can increase the risk of an attack.

**Insecure Architectures**

A misconfigured network is a primary entry point for unauthorized users. Leaving a trust-based, open local network vulnerable to the highly-insecure Internet is much like leaving a door ajar in a crime-ridden neighborhood — nothing may happen for an arbitrary amount of time, but someone exploits the opportunity eventually.
Broadcast Networks
System administrators often fail to realize the importance of networking hardware in their security schemes. Simple hardware, such as hubs and routers, relies on the broadcast or non-switched principle; that is, whenever a node transmits data across the network to a recipient node, the hub or router sends a broadcast of the data packets until the recipient node receives and processes the data. This method is the most vulnerable to address resolution protocol (ARP) or media access control (MAC) address spoofing by both outside intruders and unauthorized users on local hosts.

Centralized Servers
Another potential networking pitfall is the use of centralized computing. A common cost-cutting measure for many businesses is to consolidate all services to a single powerful machine. This can be convenient as it is easier to manage and costs considerably less than multiple-server configurations. However, a centralized server introduces a single point of failure on the network. If the central server is compromised, it may render the network completely useless or worse, prone to data manipulation or theft. In these situations, a central server becomes an open door that allows access to the entire network.

1.4.2. Threats to Server Security
Server security is as important as network security because servers often hold a great deal of an organization's vital information. If a server is compromised, all of its contents may become available for the cracker to steal or manipulate at will. The following sections detail some of the main issues.

Unused Services and Open Ports
A full installation of Red Hat Enterprise Linux 7 contains more than 1000 application and library packages. However, most server administrators do not opt to install every single package in the distribution, preferring instead to install a base installation of packages, including several server applications. See Section 2.3, “Installing the Minimum Amount of Packages Required” for an explanation of the reasons to limit the number of installed packages and for additional resources.

A common occurrence among system administrators is to install the operating system without paying attention to what programs are actually being installed. This can be problematic because unneeded services may be installed, configured with the default settings, and possibly turned on. This can cause unwanted services, such as Telnet, DHCP, or DNS, to run on a server or workstation without the administrator realizing it, which in turn can cause unwanted traffic to the server or even a potential pathway into the system for crackers. See Section 4.3, “Securing Services” for information on closing ports and disabling unused services.

Unpatched Services
Most server applications that are included in a default installation are solid, thoroughly tested pieces of software. Having been in use in production environments for many years, their code has been thoroughly refined and many of the bugs have been found and fixed.

However, there is no such thing as perfect software and there is always room for further refinement. Moreover, newer software is often not as rigorously tested as one might expect, because of its recent arrival to production environments or because it may not be as popular as other server software.

Developers and system administrators often find exploitable bugs in server applications and publish the information on bug tracking and security-related websites such as the Bugtraq mailing list (http://www.securityfocus.com) or the Computer Emergency Response Team (CERT) website (http://www.cert.org). Although these mechanisms are an effective way of
alerting the community to security vulnerabilities, it is up to system administrators to patch their systems promptly. This is particularly true because crackers have access to these same vulnerability tracking services and will use the information to crack unpatched systems whenever they can. Good system administration requires vigilance, constant bug tracking, and proper system maintenance to ensure a more secure computing environment.

See Chapter 3, Keeping Your System Up-to-Date for more information about keeping a system up-to-date.

**Inattentive Administration**

Administrators who fail to patch their systems are one of the greatest threats to server security. According to the SysAdmin, Audit, Network, Security Institute (SANS), the primary cause of computer security vulnerability is “assigning untrained people to maintain security and providing neither the training nor the time to make it possible to learn and do the job.”[1] This applies as much to inexperienced administrators as it does to overconfident or amotivated administrators.

Some administrators fail to patch their servers and workstations, while others fail to watch log messages from the system kernel or network traffic. Another common error is when default passwords or keys to services are left unchanged. For example, some databases have default administration passwords because the database developers assume that the system administrator changes these passwords immediately after installation. If a database administrator fails to change this password, even an inexperienced cracker can use a widely-known default password to gain administrative privileges to the database. These are only a few examples of how inattentive administration can lead to compromised servers.

**Inherently Insecure Services**

Even the most vigilant organization can fall victim to vulnerabilities if the network services they choose are inherently insecure. For instance, there are many services developed under the assumption that they are used over trusted networks; however, this assumption fails as soon as the service becomes available over the Internet — which is itself inherently untrusted.

One category of insecure network services are those that require unencrypted usernames and passwords for authentication. Telnet and FTP are two such services. If packet sniffing software is monitoring traffic between the remote user and such a service usernames and passwords can be easily intercepted.

Inherently, such services can also more easily fall prey to what the security industry terms the *man-in-the-middle* attack. In this type of attack, a cracker redirects network traffic by tricking a cracked name server on the network to point to his machine instead of the intended server. Once someone opens a remote session to the server, the attacker's machine acts as an invisible conduit, sitting quietly between the remote service and the unsuspecting user capturing information. In this way a cracker can gather administrative passwords and raw data without the server or the user realizing it.

Another category of insecure services include network file systems and information services such as NFS or NIS, which are developed explicitly for LAN usage but are, unfortunately, extended to include WANs (for remote users). NFS does not, by default, have any authentication or security mechanisms configured to prevent a cracker from mounting the NFS share and accessing anything contained therein. NIS, as well, has vital information that must be known by every computer on a network, including passwords and file permissions, within a plain text ASCII or DBM (ASCII-derived) database. A cracker who gains access to this database can then access every user account on a network, including the administrator's account.

By default, Red Hat Enterprise Linux 7 is released with all such services turned off.
However, since administrators often find themselves forced to use these services, careful configuration is critical. See Section 4.3, “Securing Services” for more information about setting up services in a safe manner.

### 1.4.3. Threats to Workstation and Home PC Security

Workstations and home PCs may not be as prone to attack as networks or servers, but since they often contain sensitive data, such as credit card information, they are targeted by system crackers. Workstations can also be co-opted without the user's knowledge and used by attackers as "slave" machines in coordinated attacks. For these reasons, knowing the vulnerabilities of a workstation can save users the headache of reinstalling the operating system, or worse, recovering from data theft.

#### Bad Passwords

Bad passwords are one of the easiest ways for an attacker to gain access to a system. For more on how to avoid common pitfalls when creating a password, see Section 4.1.1, “Password Security”.

#### Vulnerable Client Applications

Although an administrator may have a fully secure and patched server, that does not mean remote users are secure when accessing it. For instance, if the server offers Telnet or FTP services over a public network, an attacker can capture the plain text usernames and passwords as they pass over the network, and then use the account information to access the remote user's workstation.

Even when using secure protocols, such as SSH, a remote user may be vulnerable to certain attacks if they do not keep their client applications updated. For instance, v.1 SSH clients are vulnerable to an X-forwarding attack from malicious SSH servers. Once connected to the server, the attacker can quietly capture any keystrokes and mouse clicks made by the client over the network. This problem was fixed in the v.2 SSH protocol, but it is up to the user to keep track of what applications have such vulnerabilities and update them as necessary.

Section 4.1, “Desktop Security” discusses in more detail what steps administrators and home users should take to limit the vulnerability of computer workstations.

### 1.5. COMMON EXPLOITS AND ATTACKS

Table 1.1, “Common Exploits” details some of the most common exploits and entry points used by intruders to access organizational network resources. Key to these common exploits are the explanations of how they are performed and how administrators can properly safeguard their network against such attacks.

#### Table 1.1. Common Exploits

<table>
<thead>
<tr>
<th>Exploit</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exploit</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Null or Default Passwords</td>
<td>Leaving administrative passwords blank or using a default password set by the product vendor. This is most common in hardware such as routers and firewalls, but some services that run on Linux can contain default administrator passwords as well (though Red Hat Enterprise Linux 7 does not ship with them).</td>
<td>Commonly associated with networking hardware such as routers, firewalls, VPNs, and network attached storage (NAS) appliances. Common in many legacy operating systems, especially those that bundle services (such as UNIX and Windows.) Administrators sometimes create privileged user accounts in a rush and leave the password null, creating a perfect entry point for malicious users who discover the account.</td>
</tr>
<tr>
<td>Default Shared Keys</td>
<td>Secure services sometimes package default security keys for development or evaluation testing purposes. If these keys are left unchanged and are placed in a production environment on the Internet, all users with the same default keys have access to that shared-key resource, and any sensitive information that it contains.</td>
<td>Most common in wireless access points and preconfigured secure server appliances.</td>
</tr>
<tr>
<td>IP Spoofing</td>
<td>A remote machine acts as a node on your local network, finds vulnerabilities with your servers, and installs a backdoor program or Trojan horse to gain control over your network resources.</td>
<td>Spoofing is quite difficult as it involves the attacker predicting TCP/IP sequence numbers to coordinate a connection to target systems, but several tools are available to assist crackers in performing such a vulnerability. Depends on target system running services (such as <code>rsh</code>, <code>telnet</code>, <code>FTP</code> and others) that use <code>source-based</code> authentication techniques, which are not recommended when compared to PKI or other forms of encrypted authentication used in <code>ssh</code> or SSL/TLS.</td>
</tr>
<tr>
<td>Exploit</td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Eavesdropping</td>
<td>Collecting data that passes between two active nodes on a network by eavesdropping on the connection between the two nodes.</td>
<td>This type of attack works mostly with plain text transmission protocols such as Telnet, FTP, and HTTP transfers. Remote attacker must have access to a compromised system on a LAN in order to perform such an attack; usually the cracker has used an active attack (such as IP spoofing or man-in-the-middle) to compromise a system on the LAN. Preventative measures include services with cryptographic key exchange, one-time passwords, or encrypted authentication to prevent password snooping; strong encryption during transmission is also advised.</td>
</tr>
<tr>
<td>Service Vulnerabilities</td>
<td>An attacker finds a flaw or loophole in a service run over the Internet; through this vulnerability, the attacker compromises the entire system and any data that it may hold, and could possibly compromise other systems on the network.</td>
<td>HTTP-based services such as CGI are vulnerable to remote command execution and even interactive shell access. Even if the HTTP service runs as a non-privileged user such as &quot;nobody&quot;, information such as configuration files and network maps can be read, or the attacker can start a denial of service attack which drains system resources or renders it unavailable to other users. Services sometimes can have vulnerabilities that go unnoticed during development and testing; these vulnerabilities (such as buffer overflows, where attackers crash a service using arbitrary values that fill the memory buffer of an application, giving the attacker an interactive command prompt from which they may execute arbitrary commands) can give complete administrative control to an attacker. Administrators should make sure that services do not run as the root user, and should stay vigilant of patches and errata updates for applications from vendors or security organizations such as CERT and CVE.</td>
</tr>
</tbody>
</table>
## Application Vulnerabilities

Attackers find faults in desktop and workstation applications (such as email clients) and execute arbitrary code, implant Trojan horses for future compromise, or crash systems. Further exploitation can occur if the compromised workstation has administrative privileges on the rest of the network.

Safeguards can be implemented such that email client software does not automatically open or execute attachments. Additionally, the automatic update of workstation software using Red Hat Network; or other system management services can alleviate the burdens of multi-seat security deployments.

## Denial of Service (DoS) Attacks

Attacker or group of attackers coordinate against an organization's network or server resources by sending unauthorized packets to the target host (either server, router, or workstation). This forces the resource to become unavailable to legitimate users.

The most reported DoS case in the US occurred in 2000. Several highly-trafficked commercial and government sites were rendered unavailable by a coordinated ping flood attack using several compromised systems with high bandwidth connections acting as zombies, or redirected broadcast nodes.

Source packets are usually forged (as well as rebroadcast), making investigation as to the true source of the attack difficult.

Advances in ingress filtering (IETF rfc2267) using `iptables` and Network Intrusion Detection Systems such as `snort` assist administrators in tracking down and preventing distributed DoS attacks.

CHAPTER 2. SECURITY TIPS FOR INSTALLATION

Security begins with the first time you put that CD or DVD into your disk drive to install Red Hat Enterprise Linux 7. Configuring your system securely from the beginning makes it easier to implement additional security settings later.

2.1. SECURING BIOS

Password protection for the BIOS (or BIOS equivalent) and the boot loader can prevent unauthorized users who have physical access to systems from booting using removable media or obtaining root privileges through single user mode. The security measures you should take to protect against such attacks depends both on the sensitivity of the information on the workstation and the location of the machine.

For example, if a machine is used in a trade show and contains no sensitive information, then it may not be critical to prevent such attacks. However, if an employee’s laptop with private, unencrypted SSH keys for the corporate network is left unattended at that same trade show, it could lead to a major security breach with ramifications for the entire company.

If the workstation is located in a place where only authorized or trusted people have access, however, then securing the BIOS or the boot loader may not be necessary.

2.1.1. BIOS Passwords

The two primary reasons for password protecting the BIOS of a computer are:

1. Preventing Changes to BIOS Settings — If an intruder has access to the BIOS, they can set it to boot from a CD-ROM or a flash drive. This makes it possible for them to enter rescue mode or single user mode, which in turn allows them to start arbitrary processes on the system or copy sensitive data.

2. Preventing System Booting — Some BIOSes allow password protection of the boot process. When activated, an attacker is forced to enter a password before the BIOS launches the boot loader.

Because the methods for setting a BIOS password vary between computer manufacturers, consult the computer’s manual for specific instructions.

If you forget the BIOS password, it can either be reset with jumpers on the motherboard or by disconnecting the CMOS battery. For this reason, it is good practice to lock the computer case if possible. However, consult the manual for the computer or motherboard before attempting to disconnect the CMOS battery.

2.1.1.1. Securing Non-BIOS-based Systems

Other systems and architectures use different programs to perform low-level tasks roughly equivalent to those of the BIOS on x86 systems. For example, the Unified Extensible Firmware Interface (UEFI) shell.

For instructions on password protecting BIOS-like programs, see the manufacturer’s instructions.

2.2. PARTITIONING THE DISK
Red Hat recommends creating separate partitions for the /boot, /, /home/tmp, and /var/tmp/ directories. The reasons for each are different, and we will address each partition.

/boot
This partition is the first partition that is read by the system during boot up. The boot loader and kernel images that are used to boot your system into Red Hat Enterprise Linux 7 are stored in this partition. This partition should not be encrypted. If this partition is included in / and that partition is encrypted or otherwise becomes unavailable then your system will not be able to boot.

/home
When user data (/home) is stored in / instead of in a separate partition, the partition can fill up causing the operating system to become unstable. Also, when upgrading your system to the next version of Red Hat Enterprise Linux 7 it is a lot easier when you can keep your data in the /home partition as it will not be overwritten during installation. If the root partition (/) becomes corrupt your data could be lost forever. By using a separate partition there is slightly more protection against data loss. You can also target this partition for frequent backups.

/tmp and /var/tmp/
Both the /tmp and /var/tmp/ directories are used to store data that does not need to be stored for a long period of time. However, if a lot of data floods one of these directories it can consume all of your storage space. If this happens and these directories are stored within / then your system could become unstable and crash. For this reason, moving these directories into their own partitions is a good idea.

NOTE
During the installation process, an option to encrypt partitions is presented to you. The user must supply a passphrase. This passphrase will be used as a key to unlock the bulk encryption key, which is used to secure the partition’s data. For more information on LUKS, see Section 4.9.1, “Using LUKS Disk Encryption”.

2.3. INSTALLING THE MINIMUM AMOUNT OF PACKAGES REQUIRED

It is best practice to install only the packages you will use because each piece of software on your computer could possibly contain a vulnerability. If you are installing from the DVD media, take the opportunity to select exactly what packages you want to install during the installation. If you find you need another package, you can always add it to the system later.

For more information about installing the Minimal install environment, see the Software Selection chapter of the Red Hat Enterprise Linux 7 Installation Guide. A minimal installation can also be performed by a Kickstart file using the --nobase option. For more information about Kickstart installations, see the Package Selection section from the Red Hat Enterprise Linux 7 Installation Guide.
2.4. RESTRICTING NETWORK CONNECTIVITY DURING THE INSTALLATION PROCESS

When installing Red Hat Enterprise Linux, the installation medium represents a snapshot of the system at a particular time. Because of this, it may not be up-to-date with the latest security fixes and may be vulnerable to certain issues that were fixed only after the system provided by the installation medium was released.

When installing a potentially vulnerable operating system, always limit exposure only to the closest necessary network zone. The safest choice is the “no network” zone, which means to leave your machine disconnected during the installation process. In some cases, a LAN or intranet connection is sufficient while the Internet connection is the riskiest. To follow the best security practices, choose the closest zone with your repository while installing Red Hat Enterprise Linux from a network.

For more information about configuring network connectivity, see the Network & Hostname chapter of the Red Hat Enterprise Linux 7 Installation Guide.

2.5. POST-INSTALLATION PROCEDURES

The following steps are the security-related procedures that should be performed immediately after installation of Red Hat Enterprise Linux.

1. Update your system. enter the following command as root:

   ~]# yum update

2. Even though the firewall service, firewalld, is automatically enabled with the installation of Red Hat Enterprise Linux, there are scenarios where it might be explicitly disabled, for example in the kickstart configuration. In such a case, it is recommended to consider re-enabling the firewall.

   To start firewalld enter the following commands as root:

   ~]# systemctl start firewalld
   ~]# systemctl enable firewalld

3. To enhance security, disable services you do not need. For example, if there are no printers installed on your computer, disable the cups service using the following command:

   ~]# systemctl disable cups

   To review active services, enter the following command:

   ~]$ systemctl list-units | grep service

2.6. ADDITIONAL RESOURCES

For more information about installation in general, see the Red Hat Enterprise Linux 7 Installation Guide.
Since system BIOSes differ between manufacturers, some may not support password protection of either type, while others may support one type but not the other.
CHAPTER 3. KEEPING YOUR SYSTEM UP-TO-DATE

This chapter describes the process of keeping your system up-to-date, which involves planning and configuring the way security updates are installed, applying changes introduced by newly updated packages, and using the Red Hat Customer Portal for keeping track of security advisories.

3.1. MAINTAINING INSTALLED SOFTWARE

As security vulnerabilities are discovered, the affected software must be updated in order to limit any potential security risks. If the software is a part of a package within a Red Hat Enterprise Linux distribution that is currently supported, Red Hat is committed to releasing updated packages that fix the vulnerabilities as soon as possible.

Often, announcements about a given security exploit are accompanied with a patch (or source code) that fixes the problem. This patch is then applied to the Red Hat Enterprise Linux package and tested and released as an erratum update. However, if an announcement does not include a patch, Red Hat developers first work with the maintainer of the software to fix the problem. Once the problem is fixed, the package is tested and released as an erratum update.

If an erratum update is released for software used on your system, it is highly recommended that you update the affected packages as soon as possible to minimize the amount of time the system is potentially vulnerable.

3.1.1. Planning and Configuring Security Updates

All software contains bugs. Often, these bugs can result in a vulnerability that can expose your system to malicious users. Packages that have not been updated are a common cause of computer intrusions. Implement a plan for installing security patches in a timely manner to quickly eliminate discovered vulnerabilities, so they cannot be exploited.

Test security updates when they become available and schedule them for installation. Additional controls need to be used to protect the system during the time between the release of the update and its installation on the system. These controls depend on the exact vulnerability, but may include additional firewall rules, the use of external firewalls, or changes in software settings.

Bugs in supported packages are fixed using the errata mechanism. An erratum consists of one or more RPM packages accompanied by a brief explanation of the problem that the particular erratum deals with. All errata are distributed to customers with active subscriptions through the Red Hat Subscription Management service. Errata that address security issues are called Red Hat Security Advisories.

For more information on working with security errata, see Section 3.2.1, “Viewing Security Advisories on the Customer Portal”. For detailed information about the Red Hat Subscription Management service, including instructions on how to migrate from RHN Classic, see the documentation related to this service: Red Hat Subscription Management.

3.1.1.1. Using the Security Features of Yum

The Yum package manager includes several security-related features that can be used to search, list, display, and install security errata. These features also make it possible to use Yum to install nothing but security updates.
To check for security-related updates available for your system, enter the following command as root:

```bash
~# yum check-update --security
Loaded plugins: langpacks, product-id, subscription-manager
rhel-7-workstation-rpms/x86_64                  | 3.4 kB  00:00:00
No packages needed for security; 0 packages available
```

Note that the above command runs in a non-interactive mode, so it can be used in scripts for automated checking whether there are any updates available. The command returns an exit value of 100 when there are any security updates available and 0 when there are not. On encountering an error, it returns 1.

Analogously, use the following command to only install security-related updates:

```bash
~# yum update --security
```

Use the `updateinfo` subcommand to display or act upon information provided by repositories about available updates. The `updateinfo` subcommand itself accepts a number of commands, some of which pertain to security-related uses. See Table 3.1, “Security-related commands usable with yum updateinfo” for an overview of these commands.

### Table 3.1. Security-related commands usable with yum updateinfo

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>advisory [advisories]</td>
<td>Displays information about one or more advisories. Replace advisories with an advisory number or numbers.</td>
</tr>
<tr>
<td>cves</td>
<td>Displays the subset of information that pertains to CVE (Common Vulnerabilities and Exposures).</td>
</tr>
<tr>
<td>security or sec</td>
<td>Displays all security-related information.</td>
</tr>
<tr>
<td>severity [severity_level] or sev [severity_level]</td>
<td>Displays information about security-relevant packages of the supplied severity_level.</td>
</tr>
</tbody>
</table>

#### 3.1.2. Updating and Installing Packages

When updating software on a system, it is important to download the update from a trusted source. An attacker can easily rebuild a package with the same version number as the one that is supposed to fix the problem but with a different security exploit and release it on the Internet. If this happens, using security measures, such as verifying files against the original RPM, does not detect the exploit. Thus, it is very important to only download RPMs from trusted sources, such as from Red Hat, and to check the package signatures to verify their integrity.

See the Yum chapter of the Red Hat Enterprise Linux 7 System Administrator’s Guide for detailed information on how to use the Yum package manager.

#### 3.1.2.1. Verifying Signed Packages

All Red Hat Enterprise Linux packages are signed with the Red Hat GPG key. GPG stands
for **GNU Privacy Guard**, or **GnuPG**, a free software package used for ensuring the authenticity of distributed files. If the verification of a package signature fails, the package may be altered and therefore cannot be trusted.

The **Yum** package manager allows for an automatic verification of all packages it installs or upgrades. This feature is enabled by default. To configure this option on your system, make sure the **gpgcheck** configuration directive is set to 1 in the `/etc/yum.conf` configuration file.

Use the following command to manually verify package files on your filesystem:

```
rpmkeys --checksig package_file.rpm
```

See the **Product Signing (GPG) Keys** article on the Red Hat Customer Portal for additional information about Red Hat package-signing practices.

### 3.1.2.2. Installing Signed Packages

To install verified packages (see Section 3.1.2.1, “Verifying Signed Packages” for information on how to verify packages) from your filesystem, use the **yum install** command as the **root** user as follows:

```
yum install package_file.rpm
```

Use a shell glob to install several packages at once. For example, the following commands installs all `.rpm` packages in the current directory:

```
yum install *.rpm
```

**IMPORTANT**

Before installing any security errata, be sure to read any special instructions contained in the erratum report and execute them accordingly. See Section 3.1.3, “Applying Changes Introduced by Installed Updates” for general instructions about applying changes made by errata updates.

### 3.1.3. Applying Changes Introduced by Installed Updates

After downloading and installing security errata and updates, it is important to halt the usage of the old software and begin using the new software. How this is done depends on the type of software that has been updated. The following list itemizes the general categories of software and provides instructions for using updated versions after a package upgrade.

**NOTE**

In general, rebooting the system is the surest way to ensure that the latest version of a software package is used; however, this option is not always required, nor is it always available to the system administrator.
User-space applications are any programs that can be initiated by the user. Typically, such applications are used only when the user, a script, or an automated task utility launch them.

Once such a user-space application is updated, halt any instances of the application on the system, and launch the program again to use the updated version.

**Kernel**

The kernel is the core software component for the Red Hat Enterprise Linux 7 operating system. It manages access to memory, the processor, and peripherals, and it schedules all tasks.

Because of its central role, the kernel cannot be restarted without also rebooting the computer. Therefore, an updated version of the kernel cannot be used until the system is rebooted.

**KVM**

When the qemu-kvm and libvirt packages are updated, it is necessary to stop all guest virtual machines, reload relevant virtualization modules (or reboot the host system), and restart the virtual machines.

Use the `lsmod` command to determine which modules from the following are loaded: `kvm`, `kvm-intel`, or `kvm-amd`. Then use the `modprobe -r` command to remove and subsequently the `modprobe -a` command to reload the affected modules. For example:

```
~$ lsmod | grep kvm
kvm_intel             143031  0
kvm                   460181  1 kvm_intel
~$ modprobe -r kvm-intel
~$ modprobe -r kvm
~$ modprobe -a kvm kvm-intel
```

**Shared Libraries**

Shared libraries are units of code, such as `glibc`, that are used by a number of applications and services. Applications utilizing a shared library typically load the shared code when the application is initialized, so any applications using an updated library must be halted and relaunched.

To determine which running applications link against a particular library, use the `lsof` command:

```
lsof library
```

For example, to determine which running applications link against the `libwrap.so.0` library, type:

```
~$ lsof /lib64/libwrap.so.0
COMMAND     PID USER  FD   TYPE DEVICE SIZE/OFF     NODE NAME
pulseaudi 12363 test mem REG  253,0    42520 34121785
/usr/lib64/libwrap.so.0.7.6
 gnome-set 12365 test mem REG  253,0    42520 34121785
```
This command returns a list of all the running programs that use TCP wrappers for host-access control. Therefore, any program listed must be halted and relaunched when the tcp_wrappers package is updated.

**systemd Services**

systemd services are persistent server programs usually launched during the boot process. Examples of systemd services include `sshd` or `vsftpd`.

Because these programs usually persist in memory as long as a machine is running, each updated systemd service must be halted and relaunched after its package is upgraded. This can be done as the `root` user using the `systemctl` command:

```
systemctl restart service_name
```

Replace `service_name` with the name of the service you want to restart, such as `sshd`.

**Other Software**

Follow the instructions outlined by the resources linked below to correctly update the following applications.


### 3.2. USING THE RED HAT CUSTOMER PORTAL

The Red Hat Customer Portal at [https://access.redhat.com/](https://access.redhat.com/) is the main customer-oriented resource for official information related to Red Hat products. You can use it to find documentation, manage your subscriptions, download products and updates, open support cases, and learn about security updates.

#### 3.2.1. Viewing Security Advisories on the Customer Portal

To view security advisories (errata) relevant to the systems for which you have active subscriptions, log into the Customer Portal at [https://access.redhat.com/](https://access.redhat.com/) and click on the **Download Products & Updates** button on the main page. When you enter the **Software & Download Center** page, continue by clicking on the **Errata** button to see a list of advisories pertinent to your registered systems.

To browse a list of all security updates for all active Red Hat products, go to **Security → Security Updates → Active Products** using the navigation menu at the top of the page.

Click on the erratum code in the left part of the table to display more detailed information about the individual advisories. The next page contains not only a description of the given...
erratum, including its causes, consequences, and required fixes, but also a list of all packages that the particular erratum updates along with instructions on how to apply the updates. The page also includes links to relevant references, such as related CVE.

3.2.2. Navigating CVE Customer Portal Pages

The CVE (Common Vulnerabilities and Exposures) project, maintained by The MITRE Corporation, is a list of standardized names for vulnerabilities and security exposures. To browse a list of CVE that pertain to Red Hat products on the Customer Portal, log into your account at https://access.redhat.com/ and navigate to Security → Resources → CVE Database using the navigation menu at the top of the page.

Click on the CVE code in the left part of the table to display more detailed information about the individual vulnerabilities. The next page contains not only a description of the given CVE but also a list of affected Red Hat products along with links to relevant Red Hat errata.

3.2.3. Understanding Issue Severity Classification

All security issues discovered in Red Hat products are assigned an impact rating by Red Hat Product Security according to the severity of the problem. The four-point scale consists of the following levels: Low, Moderate, Important, and Critical. In addition to that, every security issue is rated using the Common Vulnerability Scoring System (CVSS) base scores.

Together, these ratings help you understand the impact of security issues, allowing you to schedule and prioritize upgrade strategies for your systems. Note that the ratings reflect the potential risk of a given vulnerability, which is based on a technical analysis of the bug, not the current threat level. This means that the security impact rating does not change if an exploit is released for a particular flaw.

To see a detailed description of the individual levels of severity ratings on the Customer Portal, visit the Severity Ratings page.

3.3. ADDITIONAL RESOURCES

For more information about security updates, ways of applying them, the Red Hat Customer Portal, and related topics, see the resources listed below.

Installed Documentation

- yum(8) — The manual page for the Yum package manager provides information about the way Yum can be used to install, update, and remove packages on your systems.
- rpmkeys(8) — The manual page for the rpmkeys utility describes the way this program can be used to verify the authenticity of downloaded packages.

Online Documentation

- Red Hat Enterprise Linux 7 System Administrator's Guide — The System Administrator's Guide for Red Hat Enterprise Linux 7 documents the use of the Yum and rpm commands that are used to install, update, and remove packages on Red Hat Enterprise Linux 7 systems.
Red Hat Customer Portal


- Red Hat Security Blog—Articles about latest security-related issues from Red Hat security professionals.

See Also

- Chapter 2, Security Tips for Installation describes how to configure your system securely from the beginning to make it easier to implement additional security settings later.

- Section 4.9.2, "Creating GPG Keys" describes how to create a set of personal GPG keys to authenticate your communications.
CHAPTER 4. HARDENING YOUR SYSTEM WITH TOOLS AND SERVICES

4.1. DESKTOP SECURITY

Red Hat Enterprise Linux 7 offers several ways for hardening the desktop against attacks and preventing unauthorized accesses. This section describes recommended practices for user passwords, session and account locking, and safe handling of removable media.

4.1.1. Password Security

Passwords are the primary method that Red Hat Enterprise Linux 7 uses to verify a user's identity. This is why password security is so important for protection of the user, the workstation, and the network.

For security purposes, the installation program configures the system to use Secure Hash Algorithm 512 (SHA512) and shadow passwords. It is highly recommended that you do not alter these settings.

If shadow passwords are deselected during installation, all passwords are stored as a one-way hash in the world-readable /etc/passwd file, which makes the system vulnerable to offline password cracking attacks. If an intruder can gain access to the machine as a regular user, he can copy the /etc/passwd file to his own machine and run any number of password cracking programs against it. If there is an insecure password in the file, it is only a matter of time before the password cracker discovers it.

Shadow passwords eliminate this type of attack by storing the password hashes in the file /etc/shadow, which is readable only by the root user.

This forces a potential attacker to attempt password cracking remotely by logging into a network service on the machine, such as SSH or FTP. This sort of brute-force attack is much slower and leaves an obvious trail as hundreds of failed login attempts are written to system files. Of course, if the cracker starts an attack in the middle of the night on a system with weak passwords, the cracker may have gained access before dawn and edited the log files to cover his tracks.

In addition to format and storage considerations is the issue of content. The single most important thing a user can do to protect his account against a password cracking attack is create a strong password.

**NOTE**

Red Hat recommends using a central authentication solution, such as Red Hat Identity Management (IdM). Using a central solution is preferred over using local passwords. For details, see:

- Introduction to Red Hat Identity Management
- Defining Password Policies

4.1.1.1. Creating Strong Passwords

When creating a secure password, the user must remember that long passwords are stronger than short and complex ones. It is not a good idea to create a password of just
eight characters, even if it contains digits, special characters and uppercase letters. Password cracking tools, such as John The Ripper, are optimized for breaking such passwords, which are also hard to remember by a person.

In information theory, entropy is the level of uncertainty associated with a random variable and is presented in bits. The higher the entropy value, the more secure the password is. According to NIST SP 800-63-1, passwords that are not present in a dictionary comprised of 50000 commonly selected passwords should have at least 10 bits of entropy. As such, a password that consists of four random words contains around 40 bits of entropy. A long password consisting of multiple words for added security is also called a passphrase, for example:

```
randomword1 randomword2 randomword3 randomword4
```

If the system enforces the use of uppercase letters, digits, or special characters, the passphrase that follows the above recommendation can be modified in a simple way, for example by changing the first character to uppercase and appending "1!". Note that such a modification does not increase the security of the passphrase significantly.

Another way to create a password yourself is using a password generator. The `pwmake` is a command-line tool for generating random passwords that consist of all four groups of characters – uppercase, lowercase, digits and special characters. The utility allows you to specify the number of entropy bits that are used to generate the password. The entropy is pulled from `/dev/urandom`. The minimum number of bits you can specify is 56, which is enough for passwords on systems and services where brute force attacks are rare. 64 bits is adequate for applications where the attacker does not have direct access to the password hash file. For situations when the attacker might obtain the direct access to the password hash or the password is used as an encryption key, 80 to 128 bits should be used. If you specify an invalid number of entropy bits, `pwmake` will use the default of bits. To create a password of 128 bits, enter the following command:

```
pwmake 128
```

While there are different approaches to creating a secure password, always avoid the following bad practices:

- Using a single dictionary word, a word in a foreign language, an inverted word, or only numbers.
- Using less than 10 characters for a password or passphrase.
- Using a sequence of keys from the keyboard layout.
- Writing down your passwords.
- Using personal information in a password, such as birth dates, anniversaries, family member names, or pet names.
- Using the same passphrase or password on multiple machines.

While creating secure passwords is imperative, managing them properly is also important, especially for system administrators within larger organizations. The following section details good practices for creating and managing user passwords within an organization.

### 4.1.1.2. Forcing Strong Passwords
If an organization has a large number of users, the system administrators have two basic options available to force the use of strong passwords. They can create passwords for the user, or they can let users create their own passwords while verifying the passwords are of adequate strength.

Creating the passwords for the users ensures that the passwords are good, but it becomes a daunting task as the organization grows. It also increases the risk of users writing their passwords down, thus exposing them.

For these reasons, most system administrators prefer to have the users create their own passwords, but actively verify that these passwords are strong enough. In some cases, administrators may force users to change their passwords periodically through password aging.

When users are asked to create or change passwords, they can use the `passwd` command-line utility, which is PAM-aware (Pluggable Authentication Modules) and checks to see if the password is too short or otherwise easy to crack. This checking is performed by the `pam_pwquality.so` PAM module.

**NOTE**

In Red Hat Enterprise Linux 7, the `pam_pwquality` PAM module replaced `pam_cracklib`, which was used in Red Hat Enterprise Linux 6 as a default module for password quality checking. It uses the same back end as `pam_cracklib`.

The `pam_pwquality` module is used to check a password's strength against a set of rules. Its procedure consists of two steps: first it checks if the provided password is found in a dictionary. If not, it continues with a number of additional checks. `pam_pwquality` is stacked alongside other PAM modules in the `password` component of the `/etc/pam.d/passwd` file, and the custom set of rules is specified in the `/etc/security/pwquality.conf` configuration file. For a complete list of these checks, see the `pwquality.conf (8)` manual page.

**Example 4.1. Configuring password strength-checking in pwquality.conf**

To enable using `pam_quality`, add the following line to the `password` stack in the `/etc/pam.d/passwd` file:

```bash
password required pam_pwquality.so retry=3
```

Options for the checks are specified one per line. For example, to require a password with a minimum length of 8 characters, including all four classes of characters, add the following lines to the `/etc/security/pwquality.conf` file:

```bash
minlen = 8
minclass = 4
```

To set a password strength-check for character sequences and same consecutive characters, add the following lines to `/etc/security/pwquality.conf`:

```bash
maxsequence = 3
maxrepeat = 3
```
In this example, the password entered cannot contain more than 3 characters in a monotonic sequence, such as \texttt{abcd}, and more than 3 identical consecutive characters, such as \texttt{1111}.

\textbf{NOTE}

As the root user is the one who enforces the rules for password creation, they can set any password for themselves or for a regular user, despite the warning messages.

\subsection*{4.1.1.3. Configuring Password Aging}

Password aging is another technique used by system administrators to defend against bad passwords within an organization. Password aging means that after a specified period (usually 90 days), the user is prompted to create a new password. The theory behind this is that if a user is forced to change his password periodically, a cracked password is only useful to an intruder for a limited amount of time. The downside to password aging, however, is that users are more likely to write their passwords down.

To specify password aging under Red Hat Enterprise Linux 7, make use of the \texttt{chage} command.

\textbf{IMPORTANT}

In Red Hat Enterprise Linux 7, shadow passwords are enabled by default. For more information, see the \textit{Red Hat Enterprise Linux 7 System Administrator's Guide}.

The \texttt{-M} option of the \texttt{chage} command specifies the maximum number of days the password is valid. For example, to set a user’s password to expire in 90 days, use the following command:

\begin{verbatim}
chage -M 90 username
\end{verbatim}

In the above command, replace \texttt{username} with the name of the user. To disable password expiration, use the value of \texttt{-1} after the \texttt{-M} option.

For more information on the options available with the \texttt{chage} command, see the table below.

\begin{table}[h]
\centering
\caption{chage command line options}
\begin{tabular}{|l|l|}
\hline
\textbf{Option} & \textbf{Description} \\
\hline
\texttt{-d days} & Specifies the number of days since January 1, 1970 the password was changed. \\
\hline
\texttt{-E date} & Specifies the date on which the account is locked, in the format YYYY-MM-DD. Instead of the date, the number of days since January 1, 1970 can also be used. \\
\hline
\end{tabular}
\end{table}
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-I days</td>
<td>Specifies the number of inactive days after the password expiration before locking the account. If the value is 0, the account is not locked after the password expires.</td>
</tr>
<tr>
<td>-l</td>
<td>Lists current account aging settings.</td>
</tr>
<tr>
<td>-m days</td>
<td>Specify the minimum number of days after which the user must change passwords. If the value is 0, the password does not expire.</td>
</tr>
<tr>
<td>-M days</td>
<td>Specify the maximum number of days for which the password is valid. When the number of days specified by this option plus the number of days specified with the -d option is less than the current day, the user must change passwords before using the account.</td>
</tr>
<tr>
<td>-W days</td>
<td>Specifies the number of days before the password expiration date to warn the user.</td>
</tr>
</tbody>
</table>

You can also use the `chage` command in interactive mode to modify multiple password aging and account details. Use the following command to enter interactive mode:

```
chage <username>
```

The following is a sample interactive session using this command:

```
~]# chage juan
Changing the aging information for juan
Enter the new value, or press ENTER for the default
Minimum Password Age [0]: 10
Maximum Password Age [99999]: 90
Last Password Change (YYYY-MM-DD) [2006-08-18]:
Password Expiration Warning [7]:
Password Inactive [-1]:
Account Expiration Date (YYYY-MM-DD) [1969-12-31]:
```

You can configure a password to expire the first time a user logs in. This forces users to change passwords immediately.

1. Set up an initial password. To assign a default password, enter the following command at a shell prompt as `root`:

```
pwd username
```
The `passwd` utility has the option to set a null password. Using a null password, while convenient, is a highly insecure practice, as any third party can log in and access the system using the insecure user name. Avoid using null passwords wherever possible. If it is not possible, always make sure that the user is ready to log in before unlocking an account with a null password.

2. Force immediate password expiration by running the following command as `root`:

```
chage -d 0 username
```

This command sets the value for the date the password was last changed to the epoch (January 1, 1970). This value forces immediate password expiration no matter what password aging policy, if any, is in place.

Upon the initial log in, the user is now prompted for a new password.

### 4.1.2. Account Locking

In Red Hat Enterprise Linux 7, the `pam_faillock` PAM module allows system administrators to lock out user accounts after a specified number of failed attempts. Limiting user login attempts serves mainly as a security measure that aims to prevent possible brute force attacks targeted to obtain a user’s account password.

With the `pam_faillock` module, failed login attempts are stored in a separate file for each user in the `/var/run/faillock` directory.

**NOTE**

The order of lines in the failed attempt log files is important. Any change in this order can lock all user accounts, including the `root` user account when the `even_deny_root` option is used.

Follow these steps to configure account locking:

1. To lock out any non-root user after three unsuccessful attempts and unlock that user after 10 minutes, add two lines to the `auth` section of the `/etc/pam.d/system-auth` and `/etc/pam.d/password-auth` files. After your edits, the entire `auth` section in both files should look like this:

   ```
   1 auth        required      pam_env.so
   2 auth        required      pam_faillock.so preauth silent audit
deny=3 unlock_time=600
   3 auth        sufficient    pam_unix.so nullok try_first_pass
   4 auth        [default=die] pam_faillock.so authfail audit deny=3
     unlock_time=600
   ```
5 auth requisite pam_succeed_if.so uid >= 1000 quiet_success
6 auth required pam_deny.so

Lines number 2 and 4 have been added.

2. Add the following line to the account section of both files specified in the previous step:

```plaintext
account required pam_faillock.so
```

3. To apply account locking for the root user as well, add the even_deny_root option to the pam_faillock entries in the /etc/pam.d/system-auth and /etc/pam.d/password-auth files:

```plaintext
auth required pam_faillock.so preauth silent audit deny=3 even_deny_root unlock_time=600
auth sufficient pam_unix.so nullok try_first_pass
auth [default=die] pam_faillock.so authfail audit deny=3 even_deny_root unlock_time=600
account required pam_faillock.so
```

When user john attempts to log in for the fourth time after failing to log in three times previously, his account is locked upon the fourth attempt:

```plaintext
[yruseva@localhost ~]$ su - john
Account locked due to 3 failed logins
su: incorrect password
```

To prevent the system from locking users out even after multiple failed logins, add the following line just above the line where pam_faillock is called for the first time in both /etc/pam.d/system-auth and /etc/pam.d/password-auth. Also replace user1, user2, and user3 with the actual user names.

```plaintext
auth [success=1 default=ignore] pam_succeed_if.so user in user1:user2:user3
```

To view the number of failed attempts per user, run, as root, the following command:

```plaintext
[root@localhost ~]# faillock
john:
When       Type    Source
Valid 2013-03-05 11:44:14 TTY pts/0
V
```

To unlock a user’s account, run, as root, the following command:

```plaintext
faillock --user <username> --reset
```

**Keeping Custom Settings with authconfig**

When modifying authentication configuration using the authconfig utility, the system-auth
and **password-auth** files are overwritten with the settings from the **authconfig** utility. This can be avoided by creating symbolic links in place of the configuration files, which **authconfig** recognizes and does not overwrite. In order to use custom settings in the configuration files and **authconfig** simultaneously, configure account locking using the following steps:

1. Check whether the **system-auth** and **password-auth** files are already symbolic links pointing to **system-auth-ac** and **password-auth-ac** (this is the system default):

   ```
   ~]# ls -l /etc/pam.d/{password,system}-auth
   ```

   If the output is similar to the following, the symbolic links are in place, and you can skip to step number 3:

   ```
   lrwxrwxrwx. 1 root root 16 24. Feb 09.29 /etc/pam.d/password-auth -> password-auth-ac
   lrwxrwxrwx. 1 root root 28 24. Feb 09.29 /etc/pam.d/system-auth -> system-auth-ac
   ```

   If the **system-auth** and **password-auth** files are not symbolic links, continue with the next step.

2. Rename the configuration files:

   ```
   ~]# mv /etc/pam.d/system-auth /etc/pam.d/system-auth-ac
   ~]# mv /etc/pam.d/password-auth /etc/pam.d/password-auth-ac
   ```

3. Create configuration files with your custom settings:

   ```
   ~]# vi /etc/pam.d/system-auth-local
   ```

   The **/etc/pam.d/system-auth-local** file should contain the following lines:

   ```
   auth required pam_faillock.so preauth silent audit deny=3 unlock_time=600
   auth include system-auth-ac
   auth [default=die] pam_faillock.so authfail silent audit deny=3 unlock_time=600
   account required pam_faillock.so
   account include system-auth-ac
   password include system-auth-ac
   session include system-auth-ac
   ```

   ```
   ~]# vi /etc/pam.d/password-auth-local
   ```

   The **/etc/pam.d/password-auth-local** file should contain the following lines:

   ```
   auth required pam_faillock.so preauth silent audit deny=3 unlock_time=600
   auth include password-auth-ac
   ```
auth        [default=die]  pam_faillock.so authfail silent audit
deny=3  unlock_time=600
account     required       pam_faillock.so
account     include        password-auth-ac
password    include        password-auth-ac
session     include        password-auth-ac

4. Create the following symbolic links:

```
~]# ln -sf /etc/pam.d/system-auth-local /etc/pam.d/system-auth
~]# ln -sf /etc/pam.d/password-auth-local /etc/pam.d/password-auth
```

For more information on various pam_faillock configuration options, see the pam_faillock(8) manual page.

### 4.1.3. Session Locking

Users may need to leave their workstation unattended for a number of reasons during everyday operation. This could present an opportunity for an attacker to physically access the machine, especially in environments with insufficient physical security measures (see Section 1.2.1, “Physical Controls”). Laptops are especially exposed since their mobility interferes with physical security. You can alleviate these risks by using session locking features which prevent access to the system until a correct password is entered.

**NOTE**

The main advantage of locking the screen instead of logging out is that a lock allows the user's processes (such as file transfers) to continue running. Logging out would stop these processes.

#### 4.1.3.1. Locking Virtual Consoles Using vlock

Users may also need to lock a virtual console. This can be done using a utility called vlock.

To install this utility, execute the following command as root:

```
~]# yum install vlock
```

After installation, any console session can be locked using the vlock command without any additional parameters. This locks the currently active virtual console session while still allowing access to the others. To prevent access to all virtual consoles on the workstation, execute the following:

```
vlock -a
```

In this case, vlock locks the currently active console and the -a option prevents switching to other virtual consoles.

See the vlock(1) man page for additional information.

### 4.1.4. Enforcing Read-Only Mounting of Removable Media
To enforce read-only mounting of removable media (such as USB flash disks), the administrator can use a udev rule to detect removable media and configure them to be mounted read-only using the blockdev utility. This is sufficient for enforcing read-only mounting of physical media.

Using blockdev to Force Read-Only Mounting of Removable Media
To force all removable media to be mounted read-only, create a new udev configuration file named, for example, 80-readonly-removables.rules in the /etc/udev/rules.d/ directory with the following content:

```
SUBSYSTEM=="block",ATTRS{removable}=="1",RUN{program}="/sbin/blockdev --setro %N"
```

The above udev rule ensures that any newly connected removable block (storage) device is automatically configured as read-only using the blockdev utility.

Applying New udev Settings
For these settings to take effect, the new udev rules need to be applied. The udev service automatically detects changes to its configuration files, but new settings are not applied to already existing devices. Only newly connected devices are affected by the new settings. Therefore, you need to unmount and unplug all connected removable media to ensure that the new settings are applied to them when they are next plugged in.

To force udev to re-apply all rules to already existing devices, enter the following command as root:

```
~# udevadm trigger
```

Note that forcing udev to re-apply all rules using the above command does not affect any storage devices that are already mounted.

To force udev to reload all rules (in case the new rules are not automatically detected for some reason), use the following command:

```
~# udevadm control --reload
```

### 4.2. CONTROLLING ROOT ACCESS

When administering a home machine, the user must perform some tasks as the root user or by acquiring effective root privileges using a setuid program, such as sudo or su. A setuid program is one that operates with the user ID (UID) of the program’s owner rather than the user operating the program. Such programs are denoted by an s in the owner section of a long format listing, as in the following example:

```
%]$ ls -l /bin/su
-rwsr-xr-x. 1 root root 34904 Mar 10 2011 /bin/su
```

**NOTE**

The s may be upper case or lower case. If it appears as upper case, it means that the underlying permission bit has not been set.

For the system administrator of an organization, however, choices must be made as to how
much administrative access users within the organization should have to their machines. Through a PAM module called `pam_console.so`, some activities normally reserved only for the root user, such as rebooting and mounting removable media, are allowed for the first user that logs in at the physical console. However, other important system administration tasks, such as altering network settings, configuring a new mouse, or mounting network devices, are not possible without administrative privileges. As a result, system administrators must decide how much access the users on their network should receive.

### 4.2.1. Disallowing Root Access

If an administrator is uncomfortable allowing users to log in as root for these or other reasons, the root password should be kept secret, and access to runlevel one or single user mode should be disallowed through boot loader password protection (see Section 4.2.5, “Securing the Boot Loader” for more information on this topic.)

The following are four different ways that an administrator can further ensure that root logins are disallowed:

#### Changing the root shell

To prevent users from logging in directly as root, the system administrator can set the root account's shell to `/sbin/nologin` in the `/etc/passwd` file.

**Table 4.2. Disabling the Root Shell**

<table>
<thead>
<tr>
<th>Effects</th>
<th>Does Not Affect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prevents access to a root shell and logs any such attempts. The following programs are prevented from accessing the root account:</td>
<td></td>
</tr>
<tr>
<td>✓ login</td>
<td></td>
</tr>
<tr>
<td>✓ gdm</td>
<td></td>
</tr>
<tr>
<td>✓ kdm</td>
<td></td>
</tr>
<tr>
<td>✓ xdm</td>
<td></td>
</tr>
<tr>
<td>✓ su</td>
<td></td>
</tr>
<tr>
<td>✓ ssh</td>
<td></td>
</tr>
<tr>
<td>✓ scp</td>
<td></td>
</tr>
<tr>
<td>✓ sftp</td>
<td></td>
</tr>
<tr>
<td>Programs that do not require a shell, such as FTP clients, mail clients, and many setuid programs. The following programs are not prevented from accessing the root account:</td>
<td></td>
</tr>
<tr>
<td>✓ sudo</td>
<td></td>
</tr>
<tr>
<td>✓ FTP clients</td>
<td></td>
</tr>
<tr>
<td>✓ Email clients</td>
<td></td>
</tr>
</tbody>
</table>

#### Disabling root access using any console device (tty)

To further limit access to the root account, administrators can disable root logins at the console by editing the `/etc/securetty` file. This file lists all devices the root user is allowed to log into. If the file does not exist at all, the root user can log in through any communication device on the system, whether through the console or a raw network interface. This is dangerous, because a user can log in to their machine as root using Telnet, which transmits the password in plain text over the network.

By default, Red Hat Enterprise Linux 7’s `/etc/securetty` file only allows the root user
to log in at the console physically attached to the machine. To prevent the root user from logging in, remove the contents of this file by typing the following command at a shell prompt as root:

```bash
echo > /etc/securetty
```

To enable securetty support in the KDM, GDM, and XDM login managers, add the following line:

```bash
auth [user_unknown=ignore success=ok ignore=ignore default=bad]
pam_securetty.so
```

to the files listed below:

- `/etc/pam.d/gdm`
- `/etc/pam.d/gdm-autologin`
- `/etc/pam.d/gdm-fingerprint`
- `/etc/pam.d/gdm-password`
- `/etc/pam.d/gdm-smartcard`
- `/etc/pam.d/kdm`
- `/etc/pam.d/kdm-np`
- `/etc/pam.d/xdm`

**WARNING**

A blank `/etc/securetty` file does not prevent the root user from logging in remotely using the OpenSSH suite of tools because the console is not opened until after authentication.

**Table 4.3. Disabling Root Logins**
Effects | Does Not Affect
--- | ---
Prevents access to the root account using the console or the network. The following programs are prevented from accessing the root account:
- login
- gdm
- kdm
- xdm
- Other network services that open a tty | Programs that do not log in as root, but perform administrative tasks through setuid or other mechanisms. The following programs are not prevented from accessing the root account:
- su
- sudo
- ssh
- scp
- sftp

Disabling root SSH logins
To prevent root logins through the SSH protocol, edit the SSH daemon's configuration file, /etc/ssh/sshd_config, and change the line that reads:

```sh
#PermitRootLogin yes
```

to read as follows:

```sh
PermitRootLogin no
```

Table 4.4. Disabling Root SSH Logins

<table>
<thead>
<tr>
<th>Effects</th>
<th>Does Not Affect</th>
</tr>
</thead>
</table>
| Prevents root access using the OpenSSH suite of tools. The following programs are prevented from accessing the root account:
- ssh
- scp
- sftp | Programs that are not part of the OpenSSH suite of tools. |

Using PAM to limit root access to services
PAM, through the /lib/security/pam_listfile.so module, allows great flexibility in denying specific accounts. The administrator can use this module to reference a list of users who are not allowed to log in. To limit root access to a system service, edit the file for the target service in the /etc/pam.d/ directory and make sure the pam_listfile.so module is required for authentication.
The following is an example of how the module is used for the vsftpd FTP server in the `/etc/pam.d/vsftpd` PAM configuration file (the `\` character at the end of the first line is not necessary if the directive is on a single line):

```
auth required /lib/security/pam_listfile.so item=user 
  sense=deny file=/etc/vsftpd.ftpusers onerr=succeed
```

This instructs PAM to consult the `/etc/vsftpd.ftpusers` file and deny access to the service for any listed user. The administrator can change the name of this file, and can keep separate lists for each service or use one central list to deny access to multiple services.

If the administrator wants to deny access to multiple services, a similar line can be added to the PAM configuration files, such as `/etc/pam.d/pop` and `/etc/pam.d/imap` for mail clients, or `/etc/pam.d/ssh` for SSH clients.


### Table 4.5. Disabling Root Using PAM

<table>
<thead>
<tr>
<th>Effects</th>
<th>Does Not Affect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prevents <code>root</code> access to network services that are PAM aware. The following services are prevented from accessing the <code>root</code> account:</td>
<td>Programs and services that are not PAM aware.</td>
</tr>
<tr>
<td>• login</td>
<td></td>
</tr>
<tr>
<td>• gdm</td>
<td></td>
</tr>
<tr>
<td>• kdm</td>
<td></td>
</tr>
<tr>
<td>• xdm</td>
<td></td>
</tr>
<tr>
<td>• ssh</td>
<td></td>
</tr>
<tr>
<td>• scp</td>
<td></td>
</tr>
<tr>
<td>• sftp</td>
<td></td>
</tr>
<tr>
<td>• FTP clients</td>
<td></td>
</tr>
<tr>
<td>• Email clients</td>
<td></td>
</tr>
<tr>
<td>• Any PAM aware services</td>
<td></td>
</tr>
</tbody>
</table>

### 4.2.2. Allowing Root Access

If the users within an organization are trusted and computer-literate, then allowing them `root` access may not be an issue. Allowing `root` access by users means that minor activities, like adding devices or configuring network interfaces, can be handled by the individual users, leaving system administrators free to deal with network security and other important issues.
On the other hand, giving root access to individual users can lead to the following issues:

- **Machine Misconfiguration** — Users with root access can misconfigure their machines and require assistance to resolve issues. Even worse, they might open up security holes without knowing it.

- **Running Insecure Services** — Users with root access might run insecure servers on their machine, such as FTP or Telnet, potentially putting usernames and passwords at risk. These services transmit this information over the network in plain text.

- **Running Email Attachments As Root** — Although rare, email viruses that affect Linux do exist. A malicious program poses the greatest threat when run by the root user.

- **Keeping the audit trail intact** — Because the root account is often shared by multiple users, so that multiple system administrators can maintain the system, it is impossible to figure out which of those users was root at a given time. When using separate logins, the account a user logs in with, as well as a unique number for session tracking purposes, is put into the task structure, which is inherited by every process that the user starts. When using concurrent logins, the unique number can be used to trace actions to specific logins. When an action generates an audit event, it is recorded with the login account and the session associated with that unique number. Use the aulast command to view these logins and sessions. The --proof option of the aulast command can be used suggest a specific ausearch query to isolate auditable events generated by a particular session. For more information about the Audit system, see Chapter 6, System Auditing.

### 4.2.3. Limiting Root Access

Rather than completely denying access to the root user, the administrator may want to allow access only through setuid programs, such as su or sudo. For more information on su and sudo, see the Gaining Privileges chapter in Red Hat Enterprise Linux 7 System Administrator's Guide, and the su(1) and sudo(8) man pages.

### 4.2.4. Enabling Automatic Logouts

When the user is logged in as root, an unattended login session may pose a significant security risk. To reduce this risk, you can configure the system to automatically log out idle users after a fixed period of time.

1. As root, add the following line at the beginning of the/etc/profile file to make sure the processing of this file cannot be interrupted:

   ```
   trap "" 1 2 3 15
   ```

2. As root, insert the following lines to the/etc/profile file to automatically log out after 120 seconds:

   ```
   export TMOUT=120
   readonly TMOUT
   ```

   The TMOUT variable terminates the shell if there is no activity for the specified number of seconds (set to 120 in the above example). You can change the limit according to the needs of the particular installation.
4.2.5. Securing the Boot Loader

The primary reasons for password protecting a Linux boot loader are as follows:

1. Preventing Access to Single User Mode— If attackers can boot the system into single user mode, they are logged in automatically as root without being prompted for the root password.

   **WARNING**
   
   Protecting access to single user mode with a password by editing the SINGLE parameter in the /etc/sysconfig/init file is not recommended. An attacker can bypass the password by specifying a custom initial command (using the init= parameter) on the kernel command line in GRUB 2. It is recommended to password-protect the GRUB 2 boot loader, as described in the Protecting GRUB 2 with a Password chapter in Red Hat Enterprise Linux 7 System Administrator's Guide.

2. Preventing Access to the GRUB 2 Console— If the machine uses GRUB 2 as its boot loader, an attacker can use the GRUB 2 editor interface to change its configuration or to gather information using the cat command.

3. Preventing Access to Insecure Operating Systems— If it is a dual-boot system, an attacker can select an operating system at boot time, for example DOS, which ignores access controls and file permissions.

Red Hat Enterprise Linux 7 includes the GRUB 2 boot loader on the Intel 64 and AMD64 platform. For a detailed look at GRUB 2, see the Working With the GRUB 2 Boot Loader chapter in Red Hat Enterprise Linux 7 System Administrator's Guide.

4.2.5.1. Disabling Interactive Startup

Pressing the I key at the beginning of the boot sequence allows you to start up your system interactively. During an interactive startup, the system prompts you to start up each service one by one. However, this may allow an attacker who gains physical access to your system to disable the security-related services and gain access to the system.

To prevent users from starting up the system interactively, as root, disable the PROMPT parameter in the /etc/sysconfig/init file:

```
PROMPT=no
```

4.2.6. Protecting Hard and Symbolic Links

To prevent malicious users from exploiting potential vulnerabilities caused by unprotected hard and symbolic links, Red Hat Enterprise Linux 7 includes a feature that only allows links to be created or followed provided certain conditions are met.

In case of hard links, one of the following needs to be true:
The user owns the file to which they link.

The user already has read and write access to the file to which they link.

In case of symbolic links, processes are only permitted to follow links when outside of world-writeable directories with sticky bits, or one of the following needs to be true:

- The process following the symbolic link is the owner of the symbolic link.
- The owner of the directory is the same as the owner of the symbolic link.

This protection is turned on by default. It is controlled by the following options in the /usr/lib/sysctl.d/50-default.conf file:

```bash
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
```

To override the default settings and disable the protection, create a new configuration file called, for example, 51-no-protect-links.conf in the /etc/sysctl.d/ directory with the following content:

```bash
fs.protected_hardlinks = 0
fs.protected_symlinks = 0
```

**NOTE**

Note that in order to override the default system settings, the new configuration file needs to have the .conf extension, and it needs to be read after the default system file (the files are read in lexicographic order, therefore settings contained in a file with a higher number at the beginning of the file name take precedence).

See the sysctl.d(5) manual page for more detailed information about the configuration of kernel parameters at boot using the sysctl mechanism.

### 4.3. SECURING SERVICES

While user access to administrative controls is an important issue for system administrators within an organization, monitoring which network services are active is of paramount importance to anyone who administers and operates a Linux system.

Many services under Red Hat Enterprise Linux 7 are network servers. If a network service is running on a machine, then a server application (called a daemon), is listening for connections on one or more network ports. Each of these servers should be treated as a potential avenue of attack.

#### 4.3.1. Risks To Services

Network services can pose many risks for Linux systems. Below is a list of some of the primary issues:

- **Denial of Service Attacks (DoS)** — By flooding a service with requests, a denial of service attack can render a system unusable as it tries to log and answer each request.
- **Distributed Denial of Service Attack (DDoS)** — A type of DoS attack which uses multiple compromised machines (often numbering in the thousands or more) to direct a coordinated attack on a service, flooding it with requests and making it unusable.

- **Script Vulnerability Attacks** — If a server is using scripts to execute server-side actions, as Web servers commonly do, an attacker can target improperly written scripts. These script vulnerability attacks can lead to a buffer overflow condition or allow the attacker to alter files on the system.

- **Buffer Overflow Attacks** — Services that want to listen on ports 1 through 1023 must start either with administrative privileges or the `CAP_NET_BIND_SERVICE` capability needs to be set for them. Once a process is bound to a port and is listening on it, the privileges or the capability are often dropped. If the privileges or the capability are not dropped, and the application has an exploitable buffer overflow, an attacker could gain access to the system as the user running the daemon. Because exploitable buffer overflows exist, crackers use automated tools to identify systems with vulnerabilities, and once they have gained access, they use automated rootkits to maintain their access to the system.

**NOTE**

The threat of buffer overflow vulnerabilities is mitigated in Red Hat Enterprise Linux 7 by ExecShield, an executable memory segmentation and protection technology supported by x86-compatible uni- and multi-processor kernels. ExecShield reduces the risk of buffer overflow by separating virtual memory into executable and non-executable segments. Any program code that tries to execute outside of the executable segment (such as malicious code injected from a buffer overflow exploit) triggers a segmentation fault and terminates.

Execshield also includes support for No eXecute (NX) technology on AMD64 platforms and Intel® 64 systems. These technologies work in conjunction with ExecShield to prevent malicious code from running in the executable portion of virtual memory with a granularity of 4KB of executable code, lowering the risk of attack from buffer overflow exploits.

**IMPORTANT**

To limit exposure to attacks over the network, all services that are unused should be turned off.

### 4.3.2. Identifying and Configuring Services

To enhance security, most network services installed with Red Hat Enterprise Linux 7 are turned off by default. There are, however, some notable exceptions:

- **cups** — The default print server for Red Hat Enterprise Linux 7.

- **cups-lpd** — An alternative print server.

- **xinetd** — A super server that controls connections to a range of subordinate servers, such as gssftp and telnet.

- **sshd** — The OpenSSH server, which is a secure replacement for Telnet.
When determining whether to leave these services running, it is best to use common sense and avoid taking any risks. For example, if a printer is not available, do not leave cups running. The same is true for portreserve. If you do not mount NFSv3 volumes or use NIS (the ypbind service), then rpcbind should be disabled. Checking which network services are available to start at boot time is not sufficient. It is recommended to also check which ports are open and listening. Refer to Section 4.4.2, “Verifying Which Ports Are Listening” for more information.

4.3.3. Insecure Services

Potentially, any network service is insecure. This is why turning off unused services is so important. Exploits for services are routinely revealed and patched, making it very important to regularly update packages associated with any network service. See Chapter 3, Keeping Your System Up-to-Date for more information.

Some network protocols are inherently more insecure than others. These include any services that:

- **Transmit Usernames and Passwords Over a Network Unencrypted** — Many older protocols, such as Telnet and FTP, do not encrypt the authentication session and should be avoided whenever possible.

- **Transmit Sensitive Data Over a Network Unencrypted** — Many protocols transmit data over the network unencrypted. These protocols include Telnet, FTP, HTTP, and SMTP. Many network file systems, such as NFS and SMB, also transmit information over the network unencrypted. It is the user's responsibility when using these protocols to limit what type of data is transmitted.

Examples of inherently insecure services include rlogin, rsh, telnet, and vsftpd.

All remote login and shell programs (rlogin, rsh, and telnet) should be avoided in favor of SSH. See Section 4.3.11, “Securing SSH” for more information about ss hd.

FTP is not as inherently dangerous to the security of the system as remote shells, but FTP servers must be carefully configured and monitored to avoid problems. See Section 4.3.9, “Securing FTP” for more information about securing FTP servers.

Services that should be carefully implemented and behind a firewall include:

- auth
- nfs-server
- smb and nbm (Samba)
- yppasswdd
- ypserv
- ypxfrd

More information on securing network services is available in Section 4.4, “Securing Network Access”.

4.3.4. Securing rpcbind

The rpcbind service is a dynamic port assignment daemon for RPC services such as NIS and...
NFS. It has weak authentication mechanisms and has the ability to assign a wide range of ports for the services it controls. For these reasons, it is difficult to secure.

**NOTE**

Securing `rpcbind` only affects NFSv2 and NFSv3 implementations, since NFSv4 no longer requires it. If you plan to implement an NFSv2 or NFSv3 server, then `rpcbind` is required, and the following section applies.

If running RPC services, follow these basic rules.

### 4.3.4.1. Protect rpcbind With TCP Wrappers

It is important to use TCP Wrappers to limit which networks or hosts have access to the `rpcbind` service since it has no built-in form of authentication.

Further, use only IP addresses when limiting access to the service. Avoid using host names, as they can be forged by DNS poisoning and other methods.

### 4.3.4.2. Protect rpcbind With firewalld

To further restrict access to the `rpcbind` service, it is a good idea to add `firewalld` rules to the server and restrict access to specific networks.

Below are two example `firewalld` rich language commands. The first allows TCP connections to the port 111 (used by the `rpcbind` service) from the 192.168.0.0/24 network. The second allows TCP connections to the same port from the localhost. All other packets are dropped.

```
~$ firewall-cmd --add-rich-rule='rule family="ipv4" port port="111" protocol="tcp" source address="192.168.0.0/24" invert="True" drop'
~$ firewall-cmd --add-rich-rule='rule family="ipv4" port port="111" protocol="tcp" source address="127.0.0.1" accept'
```

To similarly limit UDP traffic, use the following command:

```
~$ firewall-cmd --add-rich-rule='rule family="ipv4" port port="111" protocol="udp" source address="192.168.0.0/24" invert="True" drop'
```

**NOTE**

Add `--permanent` to the `firewalld` rich language commands to make the settings permanent. See *Chapter 5, Using Firewalls* for more information about implementing firewalls.

### 4.3.5. Securing rpc.mountd

The `rpc.mountd` daemon implements the server side of the NFS MOUNT protocol, a protocol used by NFS version 2 (*RFC 1904*) and NFS version 3 (*RFC 1813*).

If running RPC services, follow these basic rules.
4.3.5.1. Protect rpc.mountd With TCP Wrappers

It is important to use TCP Wrappers to limit which networks or hosts have access to the `rpc.mountd` service since it has no built-in form of authentication.

Further, use only IP addresses when limiting access to the service. Avoid using host names, as they can be forged by DNS poisoning and other methods.

4.3.5.2. Protect rpc.mountd With `firewalld`

To further restrict access to the `rpc.mountd` service, add `firewalld` rich language rules to the server and restrict access to specific networks.

Below are two example `firewalld` rich language commands. The first allows `mountd` connections from the `192.168.0.0/24` network. The second allows `mountd` connections from the local host. All other packets are dropped.

```
~# firewall-cmd --add-rich-rule 'rule family="ipv4" source NOT address="192.168.0.0/24" service name="mountd" drop'
~# firewall-cmd --add-rich-rule 'rule family="ipv4" source address="127.0.0.1" service name="mountd" accept'
```

**NOTE**

Add `--permanent` to the `firewalld` rich language commands to make the settings permanent. See Chapter 5, Using Firewalls for more information about implementing firewalls.

4.3.6. Securing NIS

The Network Information Service (NIS) is an RPC service, called `ypserv`, which is used in conjunction with `rpcbind` and other related services to distribute maps of user names, passwords, and other sensitive information to any computer claiming to be within its domain.

A NIS server is comprised of several applications. They include the following:

- `/usr/sbin/rpc.yppasswdd` — Also called the `yppasswdd` service, this daemon allows users to change their NIS passwords.

- `/usr/sbin/rpc.ypxfrd` — Also called the `ypxfrd` service, this daemon is responsible for NIS map transfers over the network.

- `/usr/sbin/ypserv` — This is the NIS server daemon.

NIS is somewhat insecure by today’s standards. It has no host authentication mechanisms and transmits all of its information over the network unencrypted, including password hashes. As a result, extreme care must be taken when setting up a network that uses NIS. This is further complicated by the fact that the default configuration of NIS is inherently insecure.

It is recommended that anyone planning to implement a NIS server first secure the `rpcbind` service as outlined in Section 4.3.4, “Securing rpcbind”, then address the following issues, such as network planning.
4.3.6.1. Carefully Plan the Network

Because NIS transmits sensitive information unencrypted over the network, it is important the service be run behind a firewall and on a segmented and secure network. Whenever NIS information is transmitted over an insecure network, it risks being intercepted. Careful network design can help prevent severe security breaches.

4.3.6.2. Use a Password-like NIS Domain Name and Hostname

Any machine within a NIS domain can use commands to extract information from the server without authentication, as long as the user knows the NIS server's DNS host name and NIS domain name.

For instance, if someone either connects a laptop computer into the network or breaks into the network from outside (and manages to spoof an internal IP address), the following command reveals the `/etc/passwd` map:

```
ypcat -d <NIS_domain> -h <DNS_hostname> passwd
```

If this attacker is a root user, they can obtain the `/etc/shadow` file by typing the following command:

```
ypcat -d <NIS_domain> -h <DNS_hostname> shadow
```

**NOTE**

If Kerberos is used, the `/etc/shadow` file is not stored within a NIS map.

To make access to NIS maps harder for an attacker, create a random string for the DNS host name, such as `o7hfawtgmhwg.domain.com`. Similarly, create a different randomized NIS domain name. This makes it much more difficult for an attacker to access the NIS server.

4.3.6.3. Edit the `/var/yp/securenets` File

If the `/var/yp/securenets` file is blank or does not exist (as is the case after a default installation), NIS listens to all networks. One of the first things to do is to put netmask/network pairs in the file so that `ypserv` only responds to requests from the appropriate network.

Below is a sample entry from a `/var/yp/securenets` file:

```
255.255.255.0     192.168.0.0
```

**WARNING**

Never start a NIS server for the first time without creating the `/var/yp/securenets` file.
This technique does not provide protection from an IP spoofing attack, but it does at least place limits on what networks the NIS server services.

### 4.3.6.4. Assign Static Ports and Use Rich Language Rules

All of the servers related to NIS can be assigned specific ports except for `rpc.yppasswd` — the daemon that allows users to change their login passwords. Assigning ports to the other two NIS server daemons, `rpc.ypxfrd` and `ypserv`, allows for the creation of firewall rules to further protect the NIS server daemons from intruders.

To do this, add the following lines to `/etc/sysconfig/network`:

```bash
YPSERV_ARGS="-p 834"
YPXFRD_ARGS="-p 835"
```

The following rich language `firewalld` rules can then be used to enforce which network the server listens to for these ports:

```bash
~# firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.0/24" invert="True" port port="834-835" protocol="tcp" drop'
~# firewall-cmd --add-rich-rule='rule family="ipv4" source address="192.168.0.0/24" invert="True" port port="834-835" protocol="udp" drop'
```

This means that the server only allows connections to ports 834 and 835 if the requests come from the `192.168.0.0/24` network. The first rule is for `TCP` and the second for `UDP`.

**NOTE**

See Chapter 5, *Using Firewalls* for more information about implementing firewalls with `iptables` commands.

### 4.3.6.5. Use Kerberos Authentication

One of the issues to consider when NIS is used for authentication is that whenever a user logs into a machine, a password hash from the `/etc/shadow` map is sent over the network. If an intruder gains access to a NIS domain and sniffs network traffic, they can collect user names and password hashes. With enough time, a password cracking program can guess weak passwords, and an attacker can gain access to a valid account on the network.

Since Kerberos uses secret-key cryptography, no password hashes are ever sent over the network, making the system far more secure. See the Logging into IdM Using Kerberos section in the Linux Domain Identity, Authentication, and Policy Guide for more information about Kerberos.

### 4.3.7. Securing NFS

```bash
YPSERV_ARGS="-p 834"
YPXFRD_ARGS="-p 835"
```
IMPORTANT

NFS traffic can be sent using TCP in all versions, it should be used with NFSv3, rather than UDP, and is required when using NFSv4. All versions of NFS support Kerberos user and group authentication, as part of the RPCSEC_GSS kernel module. Information on rpcbind is still included, since Red Hat Enterprise Linux 7 supports NFSv3 which utilizes rpcbind.

4.3.7.1. Carefully Plan the Network

NFSv2 and NFSv3 traditionally passed data insecurely. All versions of NFS now have the ability to authenticate (and optionally encrypt) ordinary file system operations using Kerberos. Under NFSv4 all operations can use Kerberos; under v2 or v3, file locking and mounting still do not use it. When using NFSv4.0, delegations may be turned off if the clients are behind NAT or a firewall. For information on the use of NFSv4.1 to allow delegations to operate through NAT and firewalls, see the pNFS section of the Red Hat Enterprise Linux 7 Storage Administration Guide.

4.3.7.2. Securing NFS Mount Options

The use of the `mount` command in the `/etc/fstab` file is explained in the Using the mount Command chapter of the Red Hat Enterprise Linux 7 Storage Administration Guide. From a security administration point of view it is worthwhile to note that the NFS mount options can also be specified in `/etc/nfsmount.conf`, which can be used to set custom default options.

4.3.7.2.1. Review the NFS Server

WARNING

Only export entire file systems. Exporting a subdirectory of a file system can be a security issue. It is possible in some cases for a client to "break out" of the exported part of the file system and get to unexported parts (see the section on subtree checking in the exports(5) man page.

Use the `ro` option to export the file system as read-only whenever possible to reduce the number of users able to write to the mounted file system. Only use the `rw` option when specifically required. See the man exports(5) page for more information. Allowing write access increases the risk from symlink attacks for example. This includes temporary directories such as `/tmp` and `/usr/tmp`.

Where directories must be mounted with the `rw` option avoid making them world-writable whenever possible to reduce risk. Exporting home directories is also viewed as a risk as some applications store passwords in clear text or weakly encrypted. This risk is being reduced as application code is reviewed and improved. Some users do not set passwords on their SSH keys so this too means home directories present a risk. Enforcing the use of passwords or using Kerberos would mitigate that risk.
Restrict exports only to clients that need access. Use the `showmount -e` command on an NFS server to review what the server is exporting. Do not export anything that is not specifically required.

Do not use the `no_root_squash` option and review existing installations to make sure it is not used. See Section 4.3.7.4, “Do Not Use the no_root_squash Option” for more information.

The `secure` option is the server-side export option used to restrict exports to “reserved” ports. By default, the server allows client communication only from “reserved” ports (ports numbered less than 1024), because traditionally clients have only allowed “trusted” code (such as in-kernel NFS clients) to use those ports. However, on many networks it is not difficult for anyone to become root on some client, so it is rarely safe for the server to assume that communication from a reserved port is privileged. Therefore the restriction to reserved ports is of limited value; it is better to rely on Kerberos, firewalls, and restriction of exports to particular clients.

Most clients still do use reserved ports when possible. However, reserved ports are a limited resource, so clients (especially those with a large number of NFS mounts) may choose to use higher-numbered ports as well. Linux clients may do this using the “noresvport” mount option. If you want to allow this on an export, you may do so with the “insecure” export option.

It is good practice not to allow users to login to a server. While reviewing the above settings on an NFS server conduct a review of who and what can access the server.

### 4.3.7.2.2. Review the NFS Client

Use the `nosuid` option to disallow the use of `setuid` program. The `nosuid` option disables the `set-user-identifier` or `set-group-identifier` bits. This prevents remote users from gaining higher privileges by running a setuid program. Use this option on the client and the server side.

The `noexec` option disables all executable files on the client. Use this to prevent users from inadvertently executing files placed in the file system being shared. The `nosuid` and `noexec` options are standard options for most, if not all, file systems.

Use the `nodev` option to prevent “device-files” from being processed as a hardware device by the client.

The `resvport` option is a client-side mount option and `secure` is the corresponding server-side export option (see explanation above). It restricts communication to a "reserved port". The reserved or "well known" ports are reserved for privileged users and processes such as the root user. Setting this option causes the client to use a reserved source port to communicate with the server.

All versions of NFS now support mounting with Kerberos authentication. The mount option to enable this is: `sec=krb5`.

NFSv4 supports mounting with Kerberos using `krb5i` for integrity and `krb5p` for privacy protection. These are used when mounting with `sec=krb5`, but need to be configured on the NFS server. See the man page on exports (`man 5 exports`) for more information.

The NFS man page (`man 5 nfs`) has a “SECURITY CONSIDERATIONS” section which explains the security enhancements in NFSv4 and contains all the NFS specific mount options.
4.3.7.3. Beware of Syntax Errors

The NFS server determines which file systems to export and which hosts to export these directories to by consulting the `/etc/exports` file. Be careful not to add extraneous spaces when editing this file.

For instance, the following line in the `/etc/exports` file shares the directory `/tmp/nfs/` to the host `bob.example.com` with read/write permissions.

```
/tmp/nfs/     bob.example.com(rw)
```

The following line in the `/etc/exports` file, on the other hand, shares the same directory to the host `bob.example.com` with read-only permissions and shares it to the `world` with read/write permissions due to a single space character after the host name.

```
/tmp/nfs/     bob.example.com (rw)
```

It is good practice to check any configured NFS shares by using the `showmount` command to verify what is being shared:

```
showmount -e <hostname>
```

4.3.7.4. Do Not Use the no_root_squash Option

By default, NFS shares change the root user to the `nfsnobody` user, an unprivileged user account. This changes the owner of all root-created files to `nfsnobody`, which prevents uploading of programs with the setuid bit set.

If `no_root_squash` is used, remote root users are able to change any file on the shared file system and leave applications infected by Trojans for other users to inadvertently execute.

4.3.7.5. NFS Firewall Configuration

NFSv4 is the default version of NFS for Red Hat Enterprise Linux 7 and it only requires port 2049 to be open for TCP. If using NFSv3 then four additional ports are required as explained below.

**Configuring Ports for NFSv3**

The ports used for NFS are assigned dynamically by rpcbind, which can cause problems when creating firewall rules. To simplify this process, use the `/etc/sysconfig/nfs` file to specify which ports are to be used:

- `MOUNTD_PORT` — TCP and UDP port for mountd (rpc.mountd)
- `STATD_PORT` — TCP and UDP port for status (rpc.statd)
- `LOCKD_TCP_PORT` — TCP port for nlockmgr (rpc.lockd)
- `LOCKD_UDP_PORT` — UDP port nlockmgr (rpc.lockd)

Port numbers specified must not be used by any other service. Configure your firewall to allow the port numbers specified, as well as TCP and UDP port 2049 (NFS).

Run the `rpcinfo -p` command on the NFS server to see which ports and RPC programs are being used.
4.3.7.6. Secure NFS with Red Hat Identity Management

Kerberos-aware NFS setup can be greatly simplified in an environment that is using Red Hat Identity Management which is included in Red Hat Enterprise Linux.

Follow the Red Hat Enterprise Linux 7 Linux Domain Identity, Authentication, and Policy Guide, in particular Setting up a Kerberos-aware NFS Server to learn how to secure NFS with Kerberos when using Red Hat Identity Management.

4.3.8. Securing the Apache HTTP Server

The Apache HTTP Server is one of the most stable and secure services in Red Hat Enterprise Linux 7. A large number of options and techniques are available to secure the Apache HTTP Server — too numerous to delve into deeply here. The following section briefly explains good practices when running the Apache HTTP Server.

Always verify that any scripts running on the system work as intended before putting them into production. Also, ensure that only the root user has write permissions to any directory containing scripts or CGIs. To do this, enter the following commands as the root user:

```bash
chown root <directory_name>
chmod 755 <directory_name>
```

System administrators should be careful when using the following configuration options (configured in `/etc/httpd/conf/httpd.conf`):

**FollowSymLinks**

This directive is enabled by default, so be sure to use caution when creating symbolic links to the document root of the Web server. For instance, it is a bad idea to provide a symbolic link to `/`.

**Indexes**

This directive is enabled by default, but may not be desirable. To prevent visitors from browsing files on the server, remove this directive.

**UserDir**

The **UserDir** directive is disabled by default because it can confirm the presence of a user account on the system. To enable user directory browsing on the server, use the following directives:

```plaintext
UserDir enabled
UserDir disabled root
```

These directives activate user directory browsing for all user directories other than `/root/`. To add users to the list of disabled accounts, add a space-delimited list of users on the **UserDir disabled** line.

**ServerTokens**

The **ServerTokens** directive controls the server response header field which is sent back to clients. It includes various information which can be customized using the following parameters:
ServerTokens Full (default option) — provides all available information (OS type and used modules), for example:

```
Apache/2.0.41 (Unix) PHP/4.2.2 MyMod/1.2
```

ServerTokens Prod or ServerTokens ProductOnly — provides the following information:

```
Apache
```

ServerTokens Major — provides the following information:

```
Apache/2
```

ServerTokens Minor — provides the following information:

```
Apache/2.0
```

ServerTokens Min or ServerTokens Minimal — provides the following information:

```
Apache/2.0.41
```

ServerTokens OS — provides the following information:

```
Apache/2.0.41 (Unix)
```

It is recommended to use the ServerTokens Prod option so that a possible attacker does not gain any valuable information about your system.

**IMPORTANT**

Do not remove the `IncludesNoExec` directive. By default, the `Server-Side Includes` (SSI) module cannot execute commands. It is recommended that you do not change this setting unless absolutely necessary, as it could, potentially, enable an attacker to execute commands on the system.

Removing httpd Modules

In certain scenarios, it is beneficial to remove certain `httpd` modules to limit the functionality of the HTTP Server. To do so, simply comment out the entire line which loads the module you want to remove in the `/etc/httpd/conf/httpd.conf` file. For example, to remove the proxy module, comment out the following line by prepending it with a hash sign:

```
#LoadModule proxy_module modules/mod_proxy.so
```

Note that the `/etc/httpd/conf.d/` directory contains configuration files which are used to load modules as well.

httpd and SELinux
4.3.9. Securing FTP

The File Transfer Protocol (FTP) is an older TCP protocol designed to transfer files over a network. Because all transactions with the server, including user authentication, are unencrypted, it is considered an insecure protocol and should be carefully configured.

Red Hat Enterprise Linux 7 provides two FTP servers:

- **Red Hat Content Accelerator** (tux) — A kernel-space Web server with FTP capabilities.
- **vsftpd** — A standalone, security oriented implementation of the FTP service.

The following security guidelines are for setting up the **vsftpd** FTP service.

4.3.9.1. FTP Greeting Banner

Before submitting a user name and password, all users are presented with a greeting banner. By default, this banner includes version information useful to crackers trying to identify weaknesses in a system.

To change the greeting banner for **vsftpd**, add the following directive to the `/etc/vsftpd/vsftpd.conf` file:

```
ftpd_banner=<insert_greeting_here>
```

Replace `<insert_greeting_here>` in the above directive with the text of the greeting message.

For multi-line banners, it is best to use a banner file. To simplify management of multiple banners, place all banners in a new directory called `/etc/banners/`. The banner file for FTP connections in this example is `/etc/banners/ftp.msg`. Below is an example of what such a file may look like:

```
######### Hello, all activity on ftp.example.com is logged. #########
```

<table>
<thead>
<tr>
<th>NOTE</th>
</tr>
</thead>
</table>
| It is not necessary to begin each line of the file with `220` as specified in Section 4.4.1, “Securing Services With TCP Wrappers and xinetd”.
|

To reference this greeting banner file for **vsftpd**, add the following directive to the `/etc/vsftpd/vsftpd.conf` file:

```
banner_file=/etc/banners/ftp.msg
```

It also is possible to send additional banners to incoming connections using TCP Wrappers as described in Section 4.4.1.1, “TCP Wrappers and Connection Banners”.

4.3.9.2. Anonymous Access
The presence of the `/var/ftp/` directory activates the anonymous account.

The easiest way to create this directory is to install the `vsftpd` package. This package establishes a directory tree for anonymous users and configures the permissions on directories to read-only for anonymous users.

By default the anonymous user cannot write to any directories.

**WARNING**

If enabling anonymous access to an FTP server, be aware of where sensitive data is stored.

### 4.3.9.2.1. Anonymous Upload

To allow anonymous users to upload files, it is recommended that a write-only directory be created within `/var/ftp/pub/`. To do this, enter the following command as root:

```
~]# mkdir /var/ftp/pub/upload
```

Next, change the permissions so that anonymous users cannot view the contents of the directory:

```
~]# chmod 730 /var/ftp/pub/upload
```

A long format listing of the directory should look like this:

```
~]# ls -ld /var/ftp/pub/upload
```

Administrators who allow anonymous users to read and write in directories often find that their servers become a repository of stolen software.

Additionally, under `vsftpd`, add the following line to the `/etc/vsftpd/vsftpd.conf` file:

```
anon_upload_enable=YES
```

### 4.3.9.3. User Accounts

Because FTP transmits unencrypted user names and passwords over insecure networks for authentication, it is a good idea to deny system users access to the server from their user accounts.

To disable all user accounts in `vsftpd`, add the following directive to `/etc/vsftpd/vsftpd.conf`:

```
local_enable=NO
```
4.3.9.3.1. Restricting User Accounts

To disable FTP access for specific accounts or specific groups of accounts, such as the root user and those with sudo privileges, the easiest way is to use a PAM list file as described in Section 4.2.1, “Disallowing Root Access”. The PAM configuration file for vsftpd is /etc/pam.d/vsftpd.

It is also possible to disable user accounts within each service directly.

To disable specific user accounts in vsftpd, add the user name to /etc/vsftpd/ftpusers

4.3.9.4. Use TCP Wrappers To Control Access

Use TCP Wrappers to control access to either FTP daemon as outlined in Section 4.4.1, “Securing Services With TCP Wrappers and xinetd”.

4.3.10. Securing Postfix

Postfix is a Mail Transfer Agent (MTA) that uses the Simple Mail Transfer Protocol (SMTP) to deliver electronic messages between other MTAs and to email clients or delivery agents. Although many MTAs are capable of encrypting traffic between one another, most do not, so sending email over any public networks is considered an inherently insecure form of communication. Postfix replaces Sendmail as the default MTA in Red Hat Enterprise Linux 7.

It is recommended that anyone planning to implement a Postfix server address the following issues.

4.3.10.1. Limiting a Denial of Service Attack

Because of the nature of email, a determined attacker can flood the server with mail fairly easily and cause a denial of service. The effectiveness of such attacks can be limited by setting limits of the directives in the /etc/postfix/main.cf file. You can change the value of the directives which are already there or you can add the directives you need with the value you want in the following format:

||
|<directive> = <value>

The following is a list of directives that can be used for limiting a denial of service attack:

- **smtpd_client_connection_rate_limit** — The maximum number of connection attempts any client is allowed to make to this service per time unit (described below). The default value is 0, which means a client can make as many connections per time unit as Postfix can accept. By default, clients in trusted networks are excluded.

- **anvil_rate_time_unit** — This time unit is used for rate limit calculations. The default value is 60 seconds.

- **smtpd_client_event_limit_exceptions** — Clients that are excluded from the connection and rate limit commands. By default, clients in trusted networks are excluded.

- **smtpd_client_message_rate_limit** — The maximum number of message deliveries a client is allowed to request per time unit (regardless of whether or not Postfix actually accepts those messages).
- **default_process_limit** — The default maximum number of Postfix child processes that provide a given service. This limit can be overruled for specific services in the `master.cf` file. By default the value is 100.

- **queue_minfree** — The minimum amount of free space in bytes in the queue file system that is needed to receive mail. This is currently used by the Postfix SMTP server to decide if it will accept any mail at all. By default, the Postfix SMTP server rejects `MAIL FROM` commands when the amount of free space is less than 1.5 times the `message_size_limit`. To specify a higher minimum free space limit, specify a `queue_minfree` value that is at least 1.5 times the `message_size_limit`. By default the `queue_minfree` value is 0.

- **header_size_limit** — The maximum amount of memory in bytes for storing a message header. If a header is larger, the excess is discarded. By default the value is 102400.

- **message_size_limit** — The maximum size in bytes of a message, including envelope information. By default the value is 10240000.

### 4.3.10.2. NFS and Postfix

Never put the mail spool directory, `/var/spool/postfix/`, on an NFS shared volume. Because NFSv2 and NFSv3 do not maintain control over user and group IDs, two or more users can have the same UID, and receive and read each other's mail.

**NOTE**

With NFSv4 using Kerberos, this is not the case, since the `SECRPC_GSS` kernel module does not utilize UID-based authentication. However, it is still considered good practice not to put the mail spool directory on NFS shared volumes.

### 4.3.10.3. Mail-only Users

To help prevent local user exploits on the Postfix server, it is best for mail users to only access the Postfix server using an email program. Shell accounts on the mail server should not be allowed and all user shells in the `/etc/passwd` file should be set to `/sbin/nologin` (with the possible exception of the root user).

### 4.3.10.4. Disable Postfix Network Listening

By default, Postfix is set up to only listen to the local loopback address. You can verify this by viewing the file `/etc/postfix/main.cf`.

View the file `/etc/postfix/main.cf` to ensure that only the following `inet_interfaces` line appears:

```
inet_interfaces = localhost
```

This ensures that Postfix only accepts mail messages (such as cron job reports) from the local system and not from the network. This is the default setting and protects Postfix from a network attack.

For removal of the localhost restriction and allowing Postfix to listen on all interfaces the `inet_interfaces = all` setting can be used.
4.3.10.5. Configuring Postfix to Use SASL

The Red Hat Enterprise Linux 7 version of Postfix can use the Dovecot or Cyrus SASL implementations for SMTP Authentication (or SMTP AUTH). SMTP Authentication is an extension of the Simple Mail Transfer Protocol. When enabled, SMTP clients are required to authenticate to the SMTP server using an authentication method supported and accepted by both the server and the client. This section describes how to configure Postfix to make use of the Dovecot SASL implementation.

To install the Dovecot POP/IMAP server, and thus make the Dovecot SASL implementation available on your system, issue the following command as the root user:

```
~]# yum install dovecot
```

The Postfix SMTP server can communicate with the Dovecot SASL implementation using either a UNIX-domain socket or a TCP socket. The latter method is only needed in case the Postfix and Dovecot applications are running on separate machines. This guide gives preference to the UNIX-domain socket method, which affords better privacy.

In order to instruct Postfix to use the Dovecot SASL implementation, a number of configuration changes need to be performed for both applications. Follow the procedures below to effect these changes.

Setting Up Dovecot

1. Modify the main Dovecot configuration file, `/etc/dovecot/conf.d/10-master.conf`, to include the following lines (the default configuration file already includes most of the relevant section, and the lines just need to be uncommented):

   ```
   service auth {
     unix_listener /var/spool/postfix/private/auth {
       mode = 0660
       user = postfix
       group = postfix
     }
   }
   ```

   The above example assumes the use of UNIX-domain sockets for communication between Postfix and Dovecot. It also assumes default settings of the Postfix SMTP server, which include the mail queue located in the `/var/spool/postfix/` directory, and the application running under the postfix user and group. In this way, read and write permissions are limited to the postfix user and group.

   Alternatively, you can use the following configuration to set up Dovecot to listen for Postfix authentication requests through TCP:

   ```
   service auth {
     inet_listener {
       port = 12345
     }
   }
   ```

   In the above example, replace `12345` with the number of the port you want to use.
2. Edit the `/etc/dovecot/conf.d/10-auth.conf` configuration file to instruct Dovecot to provide the Postfix SMTP server with the `plain` and `login` authentication mechanisms:

```
auth_mechanisms = plain login
```

**Setting Up Postfix**

In the case of Postfix, only the main configuration file, `/etc/postfix/main.cf`, needs to be modified. Add or edit the following configuration directives:

1. Enable SMTP Authentication in the Postfix SMTP server:

```
smtpd_sasl_auth_enable = yes
```

2. Instruct Postfix to use the Dovecot SASL implementation for SMTP Authentication:

```
smtpd_sasl_type = dovecot
```

3. Provide the authentication path relative to the Postfix queue directory (note that the use of a relative path ensures that the configuration works regardless of whether the Postfix server runs in a chroot or not):

```
smtpd_sasl_path = private/auth
```

This step assumes that you want to use UNIX-domain sockets for communication between Postfix and Dovecot. To configure Postfix to look for Dovecot on a different machine in case you use TCP sockets for communication, use configuration values similar to the following:

```
smtpd_sasl_path = inet:127.0.0.1:12345
```

In the above example, `127.0.0.1` needs to be substituted by the IP address of the Dovecot machine and `12345` by the port specified in Dovecot’s `/etc/dovecot/conf.d/10-master.conf` configuration file.

4. Specify SASL mechanisms that the Postfix SMTP server makes available to clients. Note that different mechanisms can be specified for encrypted and unencrypted sessions.

```
smtpd_sasl_security_options = noanonymous, noplaintext
smtpd_sasl_tls_security_options = noanonymous
```

The above example specifies that during unencrypted sessions, no anonymous authentication is allowed and no mechanisms that transmit unencrypted user names or passwords are allowed. For encrypted sessions (using TLS), only non-anonymous authentication mechanisms are allowed.

See [http://www.postfix.org/SASL_README.html#smtpd_sasl_security_options](http://www.postfix.org/SASL_README.html#smtpd_sasl_security_options) for a list of all supported policies for limiting allowed SASL mechanisms.

**Additional Resources**

The following online resources provide additional information useful for configuring Postfix SMTP Authentication through SASL.
4.3.11. Securing SSH

Secure Shell (SSH) is a powerful network protocol used to communicate with another system over a secure channel. The transmissions over SSH are encrypted and protected from interception. See the OpenSSH chapter of the Red Hat Enterprise Linux 7 System Administrator's Guide for general information about the SSH protocol and about using the SSH service in Red Hat Enterprise Linux 7.

IMPORTANT

This section draws attention to the most common ways of securing an SSH setup. By no means should this list of suggested measures be considered exhaustive or definitive. See sshd_config(5) for a description of all configuration directives available for modifying the behavior of the sshd daemon and to ssh(1) for an explanation of basic SSH concepts.

4.3.11.1. Cryptographic Login

SSH supports the use of cryptographic keys for logging in to computers. This is much more secure than using only a password. If you combine this method with other authentication methods, it can be considered a multi-factor authentication. See Section 4.3.11.2, “Multiple Authentication Methods” for more information about using multiple authentication methods.

In order to enable the use of cryptographic keys for authentication, the PubkeyAuthentication configuration directive in the /etc/ssh/sshd_config file needs to be set to yes. Note that this is the default setting. Set the PasswordAuthentication directive to no to disable the possibility of using passwords for logging in.

SSH keys can be generated using the ssh-keygen command. If invoked without additional arguments, it creates a 2048-bit RSA key set. The keys are stored, by default, in the ~/.ssh/ directory. You can utilize the -b switch to modify the bit-strength of the key. Using 2048-bit keys is normally sufficient. The Configuring OpenSSH chapter in the Red Hat Enterprise Linux 7 System Administrator's Guide includes detailed information about generating key pairs.

You should see the two keys in your ~/.ssh/ directory. If you accepted the defaults when running the ssh-keygen command, then the generated files are named id_rsa and id_rsa.pub and contain the private and public key respectively. You should always protect the private key from exposure by making it unreadable by anyone else but the file's owner. The public key, however, needs to be transferred to the system you are going to log in to. You can use the ssh-copy-id command to transfer the key to the server:

```
~$ ssh-copy-id -i [user@]server
```

This command will also automatically append the public key to the ~/.ssh/authorized_keys file on the server. The sshd daemon will check this file when you attempt to log in to the server.
Similarly to passwords and any other authentication mechanism, you should change your SSH keys regularly. When you do, make sure you remove any unused keys from the authorized_keys file.

4.3.11.2. Multiple Authentication Methods

Using multiple authentication methods, or multi-factor authentication, increases the level of protection against unauthorized access, and as such should be considered when hardening a system to prevent it from being compromised. Users attempting to log in to a system that uses multi-factor authentication must successfully complete all specified authentication methods in order to be granted access.

Use the AuthenticationMethods configuration directive in the /etc/ssh/sshd_config file to specify which authentication methods are to be utilized. Note that it is possible to define more than one list of required authentication methods using this directive. If that is the case, the user must complete every method in at least one of the lists. The lists need to be separated by blank spaces, and the individual authentication-method names within the lists must be comma-separated. For example:

```plaintext
AuthenticationMethods publickey,gssapi-with-mic publickey,keyboard-interactive
```

An ssdh daemon configured using the above AuthenticationMethods directive only grants access if the user attempting to log in successfully completes either publickey authentication followed by gssapi-with-mic or by keyboard-interactive authentication. Note that each of the requested authentication methods needs to be explicitly enabled using a corresponding configuration directive (such as PubkeyAuthentication) in the /etc/ssh/sshd_config file. See the AUTHENTICATION section of ssh(1) for a general list of available authentication methods.

4.3.11.3. Other Ways of Securing SSH

Protocol Version

Even though the implementation of the SSH protocol supplied with Red Hat Enterprise Linux 7 still supports both the SSH-1 and SSH-2 versions of the protocol for SSH clients, only the latter should be used whenever possible. The SSH-2 version contains a number of improvements over the older SSH-1, and the majority of advanced configuration options is only available when using SSH-2.

Red Hat recommends using SSH-2 to maximize the extent to which the SSH protocol protects the authentication and communication for which it is used. The version or versions of the protocol supported by the ssdh daemon can be specified using the Protocol configuration directive in the /etc/ssh/sshd_config file. The default setting is 2. Note that the SSH-2 version is the only version supported by the Red Hat Enterprise Linux 7 SSH server.

Key Types

While the ssh-keygen command generates a pair of SSH-2 RSA keys by default, using the -t option, it can be instructed to generate DSA or ECDSA keys as well. The ECDSA (Elliptic Curve Digital Signature Algorithm) offers better performance at the same equivalent symmetric key length. It also generates shorter keys.

Non-Default Port

By default, the ssdh daemon listens on TCP port 22. Changing the port reduces the exposure of the system to attacks based on automated network scanning, thus increasing security through obscurity. The port can be specified using the Port directive in the
/etc/ssh/sshd_config configuration file. Note also that the default SELinux policy must be changed to allow for the use of a non-default port. You can do this by modifying the ssh_port_t SELinux type by typing the following command as root:

```bash
~$# semanage -a -t ssh_port_t -p tcp port_number
```

In the above command, replace `port_number` with the new port number specified using the Port directive.

**No Root Login**

Provided that your particular use case does not require the possibility of logging in as the root user, you should consider setting the `PermitRootLogin` configuration directive to `no` in the /etc/ssh/sshd_config file. By disabling the possibility of logging in as the root user, the administrator can audit which user runs what privileged command after they log in as regular users and then gain root rights.

**Using the X Security extension**

The X server in Red Hat Enterprise Linux 7 clients does not provide the X Security extension. Therefore clients cannot request another security layer when connecting to untrusted SSH servers with X11 forwarding. The most applications were not able to run with this extension enabled anyway. By default, the `ForwardX11Trusted` option in the /etc/ssh/sshd_config file is set to `yes`, and there is no difference between the `ssh -X remote_machine` (untrusted host) and `ssh -Y remote_machine` (trusted host) command.

![WARNING]

Red Hat recommends not using X11 forwarding while connecting to untrusted hosts.

### 4.3.12. Securing PostgreSQL

**PostgreSQL** is an Object-Relational database management system (DBMS). In Red Hat Enterprise Linux 7, the `postgresql-server` package provides PostgreSQL. If it is not installed, enter the following command as the root user to install it:

```bash
~$# yum install postgresql-server
```

Before you can start using PostgreSQL, you must initialize a database storage area on disk. This is called a database cluster. To initialize a database cluster, use the command `initdb`, which is installed with PostgreSQL. The desired file system location of your database cluster is indicated by the `-D` option. For example:

```bash
~$ initdb -D /home/postgresql/db1
```

The `initdb` command will attempt to create the directory you specify if it does not already exist. We use the name `/home/postgresql/db1` in this example. The `/home/postgresql/db1` directory contains all the data stored in the database and also the client authentication configuration file:
The following line in the `pg_hba.conf` file allows any authenticated local users to access any databases with their user names:

```
local   all             all                                     trust
```

This can be problematic when you use layered applications that create database users and no local users. If you do not want to explicitly control all user names on the system, remove this line from the `pg_hba.conf` file.

### 4.3.13. Securing Docker

*Docker* is an open source project that automates the deployment of applications inside Linux Containers, and provides the capability to package an application with its runtime dependencies into a container. To make your *Docker* workflow more secure, visit [Red Hat Enterprise Linux Atomic Host 7 Container Security Guide](https://access.redhat.com/articles/31325).

### 4.4. SECURING NETWORK ACCESS

#### 4.4.1. Securing Services With TCP Wrappers and xinetd

TCP Wrappers are capable of much more than denying access to services. This section illustrates how they can be used to send connection banners, warn of attacks from particular hosts, and enhance logging functionality. See the `hosts_options(5)` man page for information about the TCP Wrapper functionality and control language. See the `xinetd.conf(5)` man page for the available flags, which act as options you can apply to a service.

##### 4.4.1.1. TCP Wrappers and Connection Banners

Displaying a suitable banner when users connect to a service is a good way to let potential attackers know that the system administrator is being vigilant. You can also control what information about the system is presented to users. To implement a TCP Wrappers banner for a service, use the `banner` option.

This example implements a banner for `vsftpd`. To begin, create a banner file. It can be anywhere on the system, but it must have same name as the daemon. For this example, the file is called `/etc/banners/vsftpd` and contains the following lines:

```
220-Hello, %c
220-All activity on ftp.example.com is logged.
220-Inappropriate use will result in your access privileges being removed.
```
The \%c token supplies a variety of client information, such as the user name and host name, or the user name and IP address to make the connection even more intimidating.

For this banner to be displayed to incoming connections, add the following line to the /etc/hosts.allow file:

```
vsftpd : ALL : banners /etc/banners/
```

### 4.4.1.2. TCP Wrappers and Attack Warnings

If a particular host or network has been detected attacking the server, TCP Wrappers can be used to warn the administrator of subsequent attacks from that host or network using the `spawn` directive.

In this example, assume that a cracker from the 206.182.68.0/24 network has been detected attempting to attack the server. Place the following line in the /etc/hosts.deny file to deny any connection attempts from that network, and to log the attempts to a special file:

```
ALL : 206.182.68.0 : spawn /bin/echo 'date' \%c \%d >> /var/log/intruder_alert
```

The \%d token supplies the name of the service that the attacker was trying to access.

To allow the connection and log it, place the `spawn` directive in the /etc/hosts.allow file.

**NOTE**

Because the `spawn` directive executes any shell command, it is a good idea to create a special script to notify the administrator or execute a chain of commands in the event that a particular client attempts to connect to the server.

### 4.4.1.3. TCP Wrappers and Enhanced Logging

If certain types of connections are of more concern than others, the log level can be elevated for that service using the `severity` option.

For this example, assume that anyone attempting to connect to port 23 (the Telnet port) on an FTP server is a cracker. To denote this, place an `emerg` flag in the log files instead of the default flag, `info`, and deny the connection.

To do this, place the following line in /etc/hosts.deny:

```
in.telnetd : ALL : severity emerg
```

This uses the default `authpriv` logging facility, but elevates the priority from the default value of `info` to `emerg`, which posts log messages directly to the console.

### 4.4.2. Verifying Which Ports Are Listening

It is important to close unused ports to avoid possible attacks. For unexpected ports in listening state, you should investigate for possible signs of intrusion.
Using netstat for Open Ports Scan

Enter the following command as root to determine which ports are listening for connections from the network:

```bash
~# netstat -pan -A inet,inet6 | grep -v ESTABLISHED
```

Use the -l option of the netstat command to display only listening server sockets:

```bash
~# netstat -tlnw
```
Using ss for Open Ports Scan
Alternatively, use the ss utility to list open ports in the listening state. It can display more TCP and state information than netstat.

```
~]# ss -tlw
etid State Recv-Q Send-Q      Local Address:Port
Peer Address:Port
udp  UNCONN    0      0      ::::ipv6-icmp
       :::*           tcp  LISTEN    0      128            *:sunrpc
       *:*          tcp  LISTEN    0      5            192.168.124.1:domain
       *:*          tcp  LISTEN    0      128            *:ssh
       *:*          tcp  LISTEN    0      128            127.0.0.1:ipp
       *:*          tcp  LISTEN    0      128            ::1:ipp
       *:*          tcp  LISTEN    0      128            ::sunrpc
       *:*          tcp  LISTEN    0      128            ::ssh
       *:*          tcp  LISTEN    0      128            ::1:ipp
       *:*          tcp  LISTEN    0      100            ::1:smtp
       *:*          tcp  LISTEN    0      128            ::ssh
       *:*          tcp  LISTEN    0      128            ::1:smtp
       *:*          tcp  LISTEN    0      5            192.168.124.1:53
       *:*                      *:*                   users:("dnsmasq",pid=1829,fd=5)
       *:*                      *:*                   users:("dnsmasq",pid=1829,fd=3)
       *:*          udph          *:*                   users:("dhclient",pid=977,fd=6)
       *:*          tcp  LISTEN    0      128            192.168.124.1:53
       *:*                      *:*                   users:("dnsmasq",pid=1829,fd=6)
       *:*          tcp  LISTEN    0      128            *:22
       *:*          tcp  LISTEN    0      128            127.0.0.1:631
       *:*                      *:*                   users:("sshd",pid=1176,fd=3)
       *:*          tcp  LISTEN    0      128            *:2500
       *:*                      *:*                   users:("sctp_darn",pid=20985,fd=3)
       *:*          tcp  LISTEN    0      100            127.0.0.1:25
       *:*                      *:*                   users:("master",pid=1664,fd=13)
       *:*          tcp  LISTEN    0      100            127.0.0.1:631
       *:*                      *:*                   users:("master",pid=1664,fd=13)
       *:*          tcp  LISTEN    0      100            127.0.0.1:25
       *:*                      *:*                   users:("master",pid=1664,fd=13)
```

The UNCONN state shows the ports in UDP listening mode.

Make a scan for every IP address shown in the ss output (except for localhost 127.0.0.0 or ::1 range) from an external system. Use the -6 option for scanning an IPv6 address.
Proceed then to make external checks using the **nmap** tool from another remote machine connected through the network to the first system. This can be used to verify rules in **firewalld**. The following is an example to determine which ports are listening for TCP connections:

~# nmap -sT -O 192.168.122.65
Starting Nmap 6.40 ( http://nmap.org ) at 2017-03-27 09:30 CEST
Nmap scan report for 192.168.122.65
Host is up (0.00032s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
111/tcp open  rpcbind
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.7 - 3.9
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at
http://nmap.org/submit/.
Nmap done: 1 IP address (1 host up) scanned in 1.79 seconds

The TCP connect scan (**-sT**), is the default TCP scan type when the TCP SYN scan (**-sS**), is not an option. The **-O** option detects the operating system of the host.

**Using netstat and ss to Scan for Open SCTP Ports**

The **netstat** utility prints information about the Linux networking subsystem. To display protocol statistics for open Stream Control Transmission Protocol (SCTP) ports, enter the following command as **root**:

~# netstat -plnS
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address   Foreign Address  State    PID/Program name
sctp                127.0.0.1:250                    LISTEN
4125/sctp_darn
sctp 0 127.0.0.1:260           127.0.0.1:250    CLOSE
4250/sctp_darn
sctp 0 127.0.0.1:250           127.0.0.1:260    LISTEN
4125/sctp_darn

~# netstat -nl -A inet,inet6 | grep 2500
sctp LISTEN 0.0.0.0:2500
LISTEN

The **ss** utility is also able to show SCTP open ports:

~# ss -an | grep 2500
sctp LISTEN 0 5 *:2500 *:*

See the **ss**(8), **netstat**(8), **nmap**(1), and **services**(5) manual pages for more information.

**4.4.3. Disabling Source Routing**
Source routing is an Internet Protocol mechanism that allows an IP packet to carry information, a list of addresses, that tells a router the path the packet must take. There is also an option to record the hops as the route is traversed. The list of hops taken, the "route record", provides the destination with a return path to the source. This allows the source (the sending host) to specify the route, loosely or strictly, ignoring the routing tables of some or all of the routers. It can allow a user to redirect network traffic for malicious purposes. Therefore, source-based routing should be disabled.

The **accept_source_route** option causes network interfaces to accept packets with the **Strict Source Routing (SSR)** or **Loose Source Routing (LSR)** option set. The acceptance of source routed packets is controlled by **sysctl** settings. Issue the following command as root to drop packets with the SSR or LSR option set:

```
~]# /sbin/sysctl -w net.ipv4.conf.all.accept_source_route=0
```

Disabling the forwarding of packets should also be done in conjunction with the above when possible (disabling forwarding may interfere with virtualization). Issue the commands listed below as root:

These commands disable forwarding of IPv4 and IPv6 packets on all interfaces.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.forwarding=0
~]# /sbin/sysctl -w net.ipv6.conf.all.forwarding=0
```

These commands disable forwarding of all multicast packets on all interfaces.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.mc_forwarding=0
~]# /sbin/sysctl -w net.ipv6.conf.all.mc_forwarding=0
```

Accepting ICMP redirects has few legitimate uses. Disable the acceptance and sending of ICMP redirected packets unless specifically required.

These commands disable acceptance of all ICMP redirected packets on all interfaces.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.accept_redirects=0
~]# /sbin/sysctl -w net.ipv6.conf.all.accept_redirects=0
```

This command disables acceptance of secure ICMP redirected packets on all interfaces.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.secure_redirects=0
```

This command disables acceptance of all IPv4 ICMP redirected packets on all interfaces.

```
~]# /sbin/sysctl -w net.ipv4.conf.all.send_redirects=0
```

There is only a directive to disable sending of IPv4 redirected packets. See **RFC4294** for an explanation of “IPv6 Node Requirements” which resulted in this difference between IPv4 and IPv6.
NOTE

To make these settings persistent across reboots, modify the `/etc/sysctl.conf` file. For example, to disable acceptance of all IPv4 ICMP redirected packets on all interfaces, open the `/etc/sysctl.conf` file with an editor running as the `root` user and add a line as follows:

```
net.ipv4.conf.all.send_redirects=0
```

See the `sysctl` man page, `sysctl(8)`, for more information. See `RFC791` for an explanation of the Internet options related to source based routing and its variants.

WARNING

Ethernet networks provide additional ways to redirect traffic, such as ARP or MAC address spoofing, unauthorized DHCP servers, and IPv6 router or neighbor advertisements. In addition, unicast traffic is occasionally broadcast, causing information leaks. These weaknesses can only be addressed by specific countermeasures implemented by the network operator. Host-based countermeasures are not fully effective.

4.4.3.1. Reverse Path Forwarding

Reverse Path Forwarding is used to prevent packets that arrived through one interface from leaving through a different interface. When outgoing routes and incoming routes are different, it is sometimes referred to as asymmetric routing. Routers often route packets this way, but most hosts should not need to do this. Exceptions are such applications that involve sending traffic out over one link and receiving traffic over another link from a different service provider. For example, using leased lines in combination with xDSL or satellite links with 3G modems. If such a scenario is applicable to you, then turning off reverse path forwarding on the incoming interface is necessary. In short, unless you know that it is required, it is best enabled as it prevents users spoofing IP addresses from local subnets and reduces the opportunity for DDoS attacks.

NOTE

Red Hat Enterprise Linux 7 defaults to using Strict Reverse Path Forwarding following the Strict Reverse Path recommendation from `RFC 3704, Ingress Filtering for Multihomed Networks`.
WARNING

If forwarding is enabled, then Reverse Path Forwarding should only be disabled if there are other means for source-address validation (such as iptables rules for example).

rp_filter

Reverse Path Forwarding is enabled by means of the rp_filter directive. The sysctl utility can be used to make changes to the running system, and permanent changes can be made by adding lines to the /etc/sysctl.conf file. The rp_filter option is used to direct the kernel to select from one of three modes.

To make a temporary global change, enter the following commands as root:

```
sysctl -w net.ipv4.conf.default.rp_filter=integer
sysctl -w net.ipv4.conf.all.rp_filter=integer
```

where integer is one of the following:

- 0 — No source validation.
- 1 — Strict mode as defined in RFC 3704.
- 2 — Loose mode as defined in RFC 3704.

The setting can be overridden per network interface using the net.ipv4.conf.interface.rp_filter command as follows:

```
sysctl -w net.ipv4.conf.interface.rp_filter=integer
```

NOTE

To make these settings persistent across reboots, modify the /etc/sysctl.conf file. For example, to change the mode for all interfaces, open the /etc/sysctl.conf file with an editor running as the root user and add a line as follows:

```
net.ipv4.conf.all.rp_filter=2
```

IPv6_rpfilter

In case of the IPv6 protocol the firewalld daemon applies to Reverse Path Forwarding by default. The setting can be checked in the /etc/firewalld/firewalld.conf file. You can change the firewalld behavior by setting the IPv6_rpfilter option.

If you need a custom configuration of Reverse Path Forwarding, you can perform it without the firewalld daemon by using the ip6tables command as follows:

```
ip6tables -t raw -I PREROUTING -m rpfilter --invert -j DROP
```
This rule should be inserted near the beginning of the raw/PREROUTING chain, so that it applies to all traffic, in particular before the stateful matching rules. For more information about the `iptables` and `ip6tables` services, see Section 5.4, “Using the iptables Service”.

### Enabling Packet Forwarding

To enable packets arriving from outside of a system to be forwarded to another external host, IP forwarding must be enabled in the kernel. Log in as `root` and change the line which reads `net.ipv4.ip_forward = 0` in the `/etc/sysctl.conf` file to the following:

```
net.ipv4.ip_forward = 1
```

To load the changes from the `/etc/sysctl.conf` file, enter the following command:

```
/sbin/sysctl -p
```

To check if IP forwarding is turned on, issue the following command as `root`:

```
/sbin/sysctl net.ipv4.ip_forward
```

If the above command returns a `1`, then IP forwarding is enabled. If it returns a `0`, then you can turn it on manually using the following command:

```
/sbin/sysctl -w net.ipv4.ip_forward=1
```

### 4.4.3.2. Additional Resources

The following are resources which explain more about Reverse Path Forwarding.

- **Installed Documentation**

  `/usr/share/doc/kernel-doc-version/Documentation/networking/ip-sysctl.txt` - This file contains a complete list of files and options available in the directory. Before accessing the kernel documentation for the first time, enter the following command as `root`:

  ```
  ~]# yum install kernel-doc
  ```

- **Online Documentation**

  See [RFC 3704](https://tools.ietf.org/html/rfc3704) for an explanation of Ingress Filtering for Multihomed Networks.

### 4.5. SECURING DNS TRAFFIC WITH DNSSEC

#### 4.5.1. Introduction to DNSSEC

DNSSEC is a set of *Domain Name System Security Extensions* (DNSSEC) that enables a DNS client to authenticate and check the integrity of responses from a DNS nameserver in order to verify their origin and to determine if they have been tampered with in transit.

#### 4.5.2. Understanding DNSSEC
For connecting over the Internet, a growing number of websites now offer the ability to connect securely using HTTPS. However, before connecting to an HTTPS webserver, a DNS lookup must be performed, unless you enter the IP address directly. These DNS lookups are done insecurely and are subject to man-in-the-middle attacks due to lack of authentication. In other words, a DNS client cannot have confidence that the replies that appear to come from a given DNS nameserver are authentic and have not been tampered with. More importantly, a recursive nameserver cannot be sure that the records it obtains from other nameservers are genuine. The DNS protocol did not provide a mechanism for the client to ensure it was not subject to a man-in-the-middle attack. DNSSEC was introduced to address the lack of authentication and integrity checks when resolving domain names using DNS. It does not address the problem of confidentiality.

Publishing DNSSEC information involves digitally signing DNS resource records as well as distributing public keys in such a way as to enable DNS resolvers to build a hierarchical chain of trust. Digital signatures for all DNS resource records are generated and added to the zone as digital signature resource records (RRSIG). The public key of a zone is added as a DNSKEY resource record. To build the hierarchical chain, hashes of the DNSKEY are published in the parent zone as Delegation of Signing (DS) resource records. To facilitate proof of non-existence, the NextSECure (NSEC) and NSEC3 resource records are used. In a DNSSEC signed zone, each resource record set (RRset) has a corresponding RRSIG resource record. Note that records used for delegation to a child zone (NS and glue records) are not signed; these records appear in the child zone and are signed there.

Processing DNSSEC information is done by resolvers that are configured with the root zone public key. Using this key, resolvers can verify the signatures used in the root zone. For example, the root zone has signed the DS record for .com. The root zone also serves NS and glue records for the .com name servers. The resolver follows this delegation and queries for the DNSKEY record of .com using these delegated name servers. The hash of the DNSKEY record obtained should match the DS record in the root zone. If so, the resolver will trust the obtained DNSKEY for .com. In the .com zone, the RRSIG records are created by the .com DNSKEY. This process is repeated similarly for delegations within .com, such as redhat.com. Using this method, a validating DNS resolver only needs to be configured with one root key while it collects many DNSKEYs from around the world during its normal operation. If a cryptographic check fails, the resolver will return SERVFAIL to the application.

DNSSEC has been designed in such a way that it will be completely invisible to applications not supporting DNSSEC. If a non-DNSSEC application queries a DNSSEC capable resolver, it will receive the answer without any of these new resource record types such as RRSIG. However, the DNSSEC capable resolver will still perform all cryptographic checks, and will still return a SERVFAIL error to the application if it detects malicious DNS answers. DNSSEC protects the integrity of the data between DNS servers (authoritative and recursive), it does not provide security between the application and the resolver. Therefore, it is important that the applications are given a secure transport to their resolver. The easiest way to accomplish that is to run a DNSSEC capable resolver on localhost and use 127.0.0.1 in /etc/resolv.conf. Alternatively a VPN connection to a remote DNS server could be used.

Understanding the Hotspot Problem
Wi-Fi Hotspots or VPNs rely on “DNS lies”: Captive portals tend to hijack DNS in order to redirect users to a page where they are required to authenticate (or pay) for the Wi-Fi service. Users connecting to a VPN often need to use an “internal only” DNS server in order to locate resources that do not exist outside the corporate network. This requires additional handling by software. For example, dnssec-trigger can be used to detect if a Hotspot is hijacking the DNS queries and unbound can act as a proxy nameserver to handle the DNSSEC queries.

Choosing a DNSSEC Capable Recursive Resolver
To deploy a DNSSEC capable recursive resolver, either **BIND** or **unbound** can be used. Both enable DNSSEC by default and are configured with the DNSSEC root key. To enable DNSSEC on a server, either will work however the use of **unbound** is preferred on mobile devices, such as notebooks, as it allows the local user to dynamically reconfigure the DNSSEC overrides required for Hotspots when using **dnssec-trigger**, and for VPNs when using **Libreswan**. The **unbound** daemon further supports the deployment of DNSSEC exceptions listed in the **etc/unbound/*d/** directories which can be useful to both servers and mobile devices.

### 4.5.3. Understanding Dnssec-trigger

Once **unbound** is installed and configured in **/etc/resolv.conf**, all **DNS** queries from applications are processed by **unbound**. **dnssec-trigger** only reconfigures the **unbound** resolver when triggered to do so. This mostly applies to roaming client machines, such as laptops, that connect to different Wi-Fi networks. The process is as follows:

- **NetworkManager** “triggers” **dnssec-trigger** when a new **DNS** server is obtained through **DHCP**.

- **Dnssec-trigger** then performs a number of tests against the server and decides whether or not it properly supports DNSSEC.

- If it does, then **dnssec-trigger** reconfigures **unbound** to use that **DNS** server as a forwarder for all queries.

- If the tests fail, **dnssec-trigger** will ignore the new **DNS** server and try a few available fall-back methods.

- If it determines that an unrestricted port 53 (**UDP** and **TCP**) is available, it will tell **unbound** to become a full recursive **DNS** server without using any forwarder.

- If this is not possible, for example because port 53 is blocked by a firewall for everything except reaching the network’s **DNS** server itself, it will try to use **DNS** to port 80, or **TLS** encapsulated **DNS** to port 443. Servers running **DNS** on port 80 and 443 can be configured in **/etc/dnssec-trigger/dnssec-trigger.conf**. Commented out examples should be available in the default configuration file.

- If these fall-back methods also fail, **dnssec-trigger** offers to either operate insecurely, which would bypass DNSSEC completely, or run in “cache only” mode where it will not attempt new **DNS** queries but will answer for everything it already has in the cache.

Wi-Fi Hotspots increasingly redirect users to a sign-on page before granting access to the Internet. During the probing sequence outlined above, if a redirection is detected, the user is prompted to ask if a login is required to gain Internet access. The **dnssec-trigger** daemon continues to probe for DNSSEC resolvers every ten seconds. See Section 4.5.8, “Using Dnssec-trigger” for information on using the **dnssec-trigger** graphical utility.

### 4.5.4. VPN Supplied Domains and Name Servers

Some types of VPN connections can convey a domain and a list of nameservers to use for that domain as part of the VPN tunnel setup. On **Red Hat Enterprise Linux**, this is supported by **NetworkManager**. This means that the combination of **unbound**, **dnssec-trigger**, and **NetworkManager** can properly support domains and name servers provided by VPN software. Once the VPN tunnel comes up, the local **unbound** cache is flushed for all entries of the domain name received, so that queries for names within the domain name
are fetched fresh from the internal name servers reached using the VPN. When the VPN tunnel is terminated, the unbound cache is flushed again to ensure any queries for the domain will return the public IP addresses, and not the previously obtained private IP addresses. See Section 4.5.11, “Configuring DNSSEC Validation for Connection Supplied Domains”.

4.5.5. Recommended Naming Practices

Red Hat recommends that both static and transient names match the fully-qualified domain name (FQDN) used for the machine in DNS, such as host.example.com.

The Internet Corporation for Assigned Names and Numbers (ICANN) sometimes adds previously unregistered Top-Level Domains (such as .yourcompany) to the public register. Therefore, Red Hat strongly recommends that you do not use a domain name that is not delegated to you, even on a private network, as this can result in a domain name that resolves differently depending on network configuration. As a result, network resources can become unavailable. Using domain names that are not delegated to you also makes DNSSEC more difficult to deploy and maintain, as domain name collisions require manual configuration to enable DNSSEC validation. See the ICANN FAQ on domain name collision for more information on this issue.

4.5.6. Understanding Trust Anchors

In a hierarchical cryptographic system, a trust anchor is an authoritative entity which is assumed to be trustworthy. For example, in X.509 architecture, a root certificate is a trust anchor from which a chain of trust is derived. The trust anchor must be put in the possession of the trusting party beforehand to make path validation possible.

In the context of DNSSEC, a trust anchor consists of a DNS name and public key (or hash of the public key) associated with that name. It is expressed as a base 64 encoded key. It is similar to a certificate in that it is a means of exchanging information, including a public key, which can be used to verify and authenticate DNS records. RFC 4033 defines a trust anchor as a configured DNSKEY RR or DS RR hash of a DNSKEY RR. A validating security-aware resolver uses this public key or hash as a starting point for building the authentication chain to a signed DNS response. In general, a validating resolver will have to obtain the initial values of its trust anchors through some secure or trusted means outside the DNS protocol. Presence of a trust anchor also implies that the resolver should expect the zone to which the trust anchor points to be signed.

4.5.7. Installing DNSSEC

4.5.7.1. Installing unbound

In order to validate DNS using DNSSEC locally on a machine, it is necessary to install the DNS resolver unbound (or bind). It is only necessary to install dnssec-trigger on mobile devices. For servers, unbound should be sufficient although a forwarding configuration for the local domain might be required depending on where the server is located (LAN or Internet). dnssec-trigger will currently only help with the global public DNS zone. NetworkManager, dhclient, and VPN applications can often gather the domain list (and nameserver list as well) automatically, but not dnssec-trigger nor unbound.

To install unbound enter the following command as the root user:

```
~]$ yum install unbound
```
4.5.7.2. Checking if unbound is Running

To determine whether the **unbound** daemon is running, enter the following command:

```bash
$ systemctl status unbound
```

The `systemctl status` command will report **unbound** as **Active: inactive (dead)** if the **unbound** service is not running.

4.5.7.3. Starting unbound

To start the **unbound** daemon for the current session, enter the following command as the **root** user:

```bash
# systemctl start unbound
```

Run the `systemctl enable` command to ensure that **unbound** starts up every time the system boots:

```bash
# systemctl enable unbound
```

The **unbound** daemon allows configuration of local data or overrides using the following directories:

- The `/etc/unbound/conf.d` directory is used to add configurations for a specific domain name. This is used to redirect queries for a domain name to a specific DNS server. This is often used for sub-domains that only exist within a corporate WAN.

- The `/etc/unbound/keys.d` directory is used to add trust anchors for a specific domain name. This is required when an internal-only name is DNSSEC signed, but there is no publicly existing DS record to build a path of trust. Another use case is when an internal version of a domain is signed using a different DNSKEY than the publicly available name outside the corporate WAN.

- The `/etc/unbound/local.d` directory is used to add specific DNS data as a local override. This can be used to build blacklists or create manual overrides. This data will be returned to clients by **unbound**, but it will not be marked as DNSSEC signed.

**NetworkManager**, as well as some VPN software, may change the configuration dynamically. These configuration directories contain commented out example entries. For further information see the **unbound.conf**(5) man page.

4.5.7.4. Installing Dnssec-trigger

The **dnssec-trigger** application runs as a daemon, **dnssec-triggerd**. To install **dnssec-trigger** enter the following command as the **root** user:

```bash
# yum install dnssec-trigger
```

4.5.7.5. Checking if the Dnssec-trigger Daemon is Running
To determine whether **dnssec-triggerd** is running, enter the following command:

```
~]$ systemctl status dnssec-triggerd
systemctl status dnssec-triggerd.service
dnssec-triggerd.service - Reconfigure local DNS(SEC) resolver on network change
   Loaded: loaded (/usr/lib/systemd/system/dnssec-triggerd.service; enabled)
   Active: active (running) since Wed 2013-03-13 06:10:44 CET; 1h 41min ago
```

The `systemctl status` command will report **dnssec-triggerd** as **Active: inactive (dead)** if the **dnssec-triggerd** daemon is not running. To start it for the current session enter the following command as the **root** user:

```
~]$ systemctl start dnssec-triggerd
```

Run the `systemctl enable` command to ensure that **dnssec-triggerd** starts up every time the system boots:

```
~]$ systemctl enable dnssec-triggerd
```

### 4.5.8. Using Dnssec-trigger

The **dnssec-trigger** application has a GNOME panel utility for displaying DNSSEC probe results and for performing DNSSEC probe requests on demand. To start the utility, press the Super key to enter the Activities Overview, type DNSSEC and then press Enter. An icon resembling a ships anchor is added to the message tray at the bottom of the screen. Press the round blue notification icon in the bottom right of the screen to reveal it. Right click the anchor icon to display a pop-up menu.

In normal operations **unbound** is used locally as the name server, and **resolv.conf** points to **127.0.0.1**. When you click OK on the Hotspot Sign-On panel this is changed. The DNS servers are queried from **NetworkManager** and put in **resolv.conf**. Now you can authenticate on the Hotspot's sign-on page. The anchor icon shows a big red exclamation mark to warn you that DNS queries are being made insecurely. When authenticated, **dnssec-trigger** should automatically detect this and switch back to secure mode, although in some cases it cannot and the user has to do this manually by selecting **Reprobe**.

**Dnssec-trigger** does not normally require any user interaction. Once started, it works in the background and if a problem is encountered it notifies the user by means of a pop-up text box. It also informs **unbound** about changes to the **resolv.conf** file.

### 4.5.9. Using dig With DNSSEC

To see whether DNSSEC is working, one can use various command line tools. The best tool to use is the **dig** command from the bind-utils package. Other tools that are useful are **drill** from the Idns package and **unbound-host** from the unbound package. The old DNS utilities **nslookup** and **host** are obsolete and should not be used.

To send a query requesting DNSSEC data using **dig**, the option **+dnssec** is added to the command, for example:

```
~]$ dig +dnssec whitehouse.gov
```
In addition to the A record, an RRSIG record is returned which contains the DNSSEC signature, as well as the inception time and expiration time of the signature. The unbound server indicated that the data was DNSSEC authenticated by returning the **ad** bit in the **flags** section at the top.

If DNSSEC validation fails, the **dig** command would return a SERVFAIL error:

```bash
~]$ dig badsign-a.test.dnssec-tools.org
; <<>> DiG 9.9.3-rl.156.01-P1-RedHat-9.9.3-3.P1.el7 <<>> badsign-a.test.dnssec-tools.org
; global options: +cmd
; Got answer:
; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 1010
; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; QUESTION SECTION:
;badsign-a.test.dnssec-tools.org. IN A

;; Query time: 1284 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Aug 22 22:04:52 EDT 2013
;; MSG SIZE  rcvd: 60]
```

To request more information about the failure, DNSSEC checking can be disabled by specifying the **+cd** option to the **dig** command:

```bash
~]$ dig +cd +dnssec badsign-a.test.dnssec-tools.org
; <<>> DiG 9.9.3-rl.156.01-P1-RedHat-9.9.3-3.P1.el7 <<>> +cd +dnssec badsign-a.test.dnssec-tools.org
; global options: +cmd
```
Often, DNSSEC mistakes manifest themselves by bad inception or expiration time, although in this example, the people at www.dnssec-tools.org have mangled this RRSIG signature on purpose, which we would not be able to detect by looking at this output manually. The error will show in the output of systemctl status unbound and the unbound daemon logs these errors to syslog as follows:


An example using unbound-host:

```bash
~]$ unbound-host -C /etc/unbound/unbound.conf -v whitehouse.gov
whitehouse.gov has address 184.25.196.110 (secure)
whitehouse.gov has IPv6 address 2600:1417:11:2:8000::fc4 (secure)
whitehouse.gov has IPv6 address 2600:1417:11:2:8000::fc4 (secure)
whitehouse.gov mail is handled by 105 mail1.eop.gov. (secure)
whitehouse.gov mail is handled by 110 mail5.eop.gov. (secure)
whitehouse.gov mail is handled by 105 mail4.eop.gov. (secure)
whitehouse.gov mail is handled by 110 mail6.eop.gov. (secure)
whitehouse.gov mail is handled by 105 mail2.eop.gov. (secure)
whitehouse.gov mail is handled by 105 mail3.eop.gov. (secure)
```

4.5.10. Setting up Hotspot Detection Infrastructure for Dnssec-trigger

When connecting to a network, dnssec-trigger attempts to detect a Hotspot. A Hotspot is generally a device that forces user interaction with a web page before they can use the network resources. The detection is done by attempting to download a specific fixed web page with known content. If there is a Hotspot, then the content received will not be as expected.

To set up a fixed web page with known content that can be used by dnssec-trigger to detect a Hotspot, proceed as follows:
1. Set up a web server on some machine that is publicly reachable on the Internet. See the Web Servers chapter in the Red Hat Enterprise Linux 7 System Administrator's Guide.

2. Once you have the server running, publish a static page with known content on it. The page does not need to be a valid HTML page. For example, you could use a plain-text file named `hotspot.txt` that contains only the string `OK`. Assuming your server is located at `example.com` and you published your `hotspot.txt` file in the web server `document_root/static/` sub-directory, then the address to your static web page would be `example.com/static/hotspot.txt`. See the DocumentRoot directive in the Web Servers chapter in the Red Hat Enterprise Linux 7 System Administrator's Guide.

3. Add the following line to the `/etc/dnssec-trigger/dnssec-trigger.conf` file:

   ```
   url: "http://example.com/static/hotspot.txt OK"
   ```

   This command adds a URL that is probed using HTTP (port 80). The first part is the URL that will be resolved and the page that will be downloaded. The second part of the command is the text string that the downloaded webpage is expected to contain.

   For more information on the configuration options see the man page `dnssec-trigger.conf(8)`.

### 4.5.11. Configuring DNSSEC Validation for Connection Supplied Domains

By default, forward zones with proper nameservers are automatically added into unbound by `dnssec-trigger` for every domain provided by any connection, except Wi-Fi connections through NetworkManager. By default, all forward zones added into `unbound` are DNSSEC validated.

The default behavior for validating forward zones can be altered, so that all forward zones will **not** be DNSSEC validated by default. To do this, change the `validate_connection_provided_zones` variable in the `dnssec-trigger` configuration file `/etc/dnssec.conf`. As root user, open and edit the line as follows:

```
validate_connection_provided_zones=no
```

The change is not done for any existing forward zones, but only for future forward zones. Therefore if you want to disable DNSSEC for the current provided domain, you need to reconnect.

### 4.5.11.1. Configuring DNSSEC Validation for Wi-Fi Supplied Domains

Adding forward zones for Wi-Fi provided zones can be enabled. To do this, change the `add_wifi_provided_zones` variable in the `dnssec-trigger` configuration file, `/etc/dnssec.conf`. As root user, open and edit the line as follows:

```
add_wifi_provided_zones=yes
```

The change is not done for any existing forward zones, but only for future forward zones. Therefore, if you want to enable DNSSEC for the current Wi-Fi provided domain, you need to reconnect (restart) the Wi-Fi connection.
WARNING

Turning on the addition of Wi-Fi provided domains as forward zones into unbound may have security implications such as:

1. A Wi-Fi access point can intentionally provide you a domain through DHCP for which it does not have authority and route all your DNS queries to its DNS servers.

2. If you have the DNSSEC validation of forward zones turned off, the Wi-Fi provided DNS servers can spoof the IP address for domain names from the provided domain without you knowing it.

4.5.12. Additional Resources

The following are resources which explain more about DNSSEC.

4.5.12.1. Installed Documentation

- **dnssec-trigger(8)** man page — Describes command options for *dnssec-triggerd*, *dnssec-trigger-control* and *dnssec-trigger-panel*.

- **dnssec-trigger.conf(8)** man page — Describes the configuration options for *dnssec-triggerd*.

- **unbound(8)** man page — Describes the command options for *unbound*, the DNS validating resolver.

- **unbound.conf(5)** man page — Contains information on how to configure *unbound*.

- **resolv.conf(5)** man page — Contains information that is read by the resolver routines.

4.5.12.2. Online Documentation


RFC 4033 DNS Security Introduction and Requirements.

**http://www.dnssec.net/**

A website with links to many DNSSEC resources.

**http://www.dnssec-deployment.org/**

The DNSSEC Deployment Initiative, sponsored by the Department for Homeland Security, contains a lot of DNSSEC information and has a mailing list to discuss DNSSEC deployment issues.

**http://www.internetsociety.org/deploy360/dnssec/community/**
The Internet Society's “Deploy 360” initiative to stimulate and coordinate DNSSEC deployment is a good resource for finding communities and DNSSEC activities worldwide.

http://www.unbound.net/
This document contains general information about the unbound DNS service.

http://www.nlnetlabs.nl/projects/dnssec-trigger/
This document contains general information about dnssec-trigger.

### 4.6. SECURING VIRTUAL PRIVATE NETWORKS (VPNS)

In Red Hat Enterprise Linux 7, a Virtual Private Network (VPN) can be configured using the **IPsec** tunneling protocol which is supported by the **Libreswan** application. **Libreswan** is a fork of the **Openswan** application and examples in documentation should be interchangeable. The **NetworkManager IPsec** plug-in is called NetworkManager-libreswan. Users of GNOME Shell should install the NetworkManager-libreswan-gnome package, which has NetworkManager-libreswan as a dependency. Note that the NetworkManager-libreswan-gnome package is only available from the Optional channel. See **Enabling Supplementary and Optional Repositories**.

**Libreswan** is an open-source, user-space **IPsec** implementation available in Red Hat Enterprise Linux 7. It uses the **Internet Key Exchange** (IKE) protocol. **IKE** version 1 and 2 are implemented as a user-level daemon. Manual key establishment is also possible through **ip xfrm** commands, however this is not recommended. **Libreswan** interfaces with the Linux kernel using netlink to transfer the encryption keys. Packet encryption and decryption happen in the Linux kernel.

**Libreswan** uses the **Network Security Services** (NSS) cryptographic library, which is required for **Federal Information Processing Standard** (FIPS) security compliance.

**IMPORTANT**

**IKE/IPsec**, implemented by **Libreswan**, is the only VPN technology recommended for use in Red Hat Enterprise Linux 7. Do not use any other VPN technology without understanding the risks of doing so.

#### 4.6.1. IPsec VPN Using Libreswan

To install **Libreswan**, issue the following command as **root**:

```
~]$# yum install libreswan
```

To check that **Libreswan** is installed, issue the following command:

```
~]$$ yum info libreswan
```

After a new installation of **Libreswan** the NSS database should be initialized as part of the install process. However, should you need to start a new database, first remove the old database as follows:

```
~]$# rm /etc/ipsec.d/*db
```
Then, to initialize a new NSS database, issue the following command as root:

```
~]# ipsec initnss
Initializing NSS database
See 'man pluto' if you want to protect the NSS database with a password
```

If you do not want to use a password for NSS, just press Enter twice when prompted for the password. If you do enter a password then you will have to re-enter it every time Libreswan is started, such as every time the system is booted.

To start the ipsec daemon provided by Libreswan, issue the following command as root:

```
~]# systemctl start ipsec
```

To confirm that the daemon is now running:

```
~]$ systemctl status ipsec
ipsec.service - Internet Key Exchange (IKE) Protocol Daemon for IPsec
   Loaded: loaded (/usr/lib/systemd/system/ipsec.service; disabled)
   Active: active (running) since Wed 2013-08-21 12:14:12 CEST; 18s ago
```

To ensure that Libreswan will start when the system starts, issue the following command as root:

```
~]# systemctl enable ipsec
```

Configure any intermediate as well as host-based firewalls to permit the ipsec service. See Chapter 5, Using Firewalls for information on firewalls and allowing specific services to pass through. Libreswan requires the firewall to allow the following packets:

- **UDP** port 500 for the Internet Key Exchange (IKE) protocol
- **UDP** port 4500 for IKE NAT-Traversal
- Protocol 50 for Encapsulated Security Payload (ESP) IPsec packets
- Protocol 51 for Authenticated Header (AH) IPsec packets (uncommon)

We present three examples of using Libreswan to set up an IPsec VPN. The first example is for connecting two hosts together so that they may communicate securely. The second example is connecting two sites together to form one network. The third example is supporting remote users, known as road warriors in this context.

### 4.6.2. VPN Configurations Using Libreswan

Libreswan does not use the terms “source” or “destination”. Instead, it uses the terms “left” and “right” to refer to end points (the hosts). This allows the same configuration to be used on both end points in most cases, although most administrators use “left” for the local host and “right” for the remote host.

There are three commonly used methods for authentication of endpoints:

- **Pre-Shared Keys** (PSK) is the simplest authentication method. PSK's should consist of random characters and have a length of at least 20 characters.
Raw RSA keys are commonly used for static host-to-host or subnet-to-subnet **IPsec** configurations. The hosts are manually configured with each other’s public RSA key. This method does not scale well when dozens or more hosts all need to setup **IPsec** tunnels to each other.

X.509 certificates are commonly used for large scale deployments where there are many hosts that need to connect to a common **IPsec** gateway. A central **certificate authority** (CA) is used to sign RSA certificates for hosts or users. This central CA is responsible for relaying trust, including the revocations of individual hosts or users.

### 4.6.3. Host-To-Host VPN Using Libreswan

To configure **Libreswan** to create a host-to-host **IPsec** VPN, between two hosts referred to as “left” and “right”, enter the following commands as **root** on both of the hosts (“left” and “right”) to create new raw RSA key pairs:

```bash
~# ipsec newhostkey --configdir /etc/ipsec.d \\   --output /etc/ipsec.d/www.example.com.secrets
```

Generated RSA key pair using the NSS database

This generates an RSA key pair for the host. The process of generating RSA keys can take many minutes, especially on virtual machines with low entropy.

To view the public key, issue the following command as **root** on either of the hosts. For example, to view the public key on the “left” host, run:

```bash
~# ipsec showhostkey --left
```

```
ipsec showhostkey loading secrets from "/etc/ipsec.secrets"

ipsec showhostkey loading secrets from "/etc/ipsec.d/www.example.com.secrets"

ipsec showhostkey loaded private key for keyid: PPK_RSA:AQOjAKLlL
# rsakey AQOjAKLlL
leftrsasigkey=0sAQOjAKLlL4a7YBv [...] W2n417C/4urYHQkCvuIQ==
```

You will need this key to add to the configuration file as explained below.

The secret part is stored in `/etc/ipsec.d/*.db` files, also called the “NSS database”.

To make a configuration file for this host-to-host tunnel, the lines `leftrsasigkey=` and `rightrsasigkey=` from above, are added to a custom configuration file placed in the `/etc/ipsec.d/` directory.

Using an editor running as **root**, create a file with a suitable name in the following format:

```bash
/etc/ipsec.d/my_host-to-host.conf
```

Edit the file as follows:

```bash
conn mytunnel
  leftid=@west.example.com
  left=192.1.2.23
  leftrsasigkey=0sAQOOrlo+h0afUZDlCQmXFrje/oZm [...] W2n417C/4urYHQkCvuIQ==
  rightid=@east.example.com
  right=192.1.2.45
```

You can use the identical configuration file on both left and right hosts. They will auto-detect if they are "left" or "right". If one of the hosts is a mobile host, which implies the IP address is not known in advance, then on the mobile host use %defaultroute as its IP address. This will pick up the dynamic IP address automatically. On the static host that accepts connections from incoming mobile hosts, specify the mobile host using %any for its IP address.

Ensure the leftrsasigkey value is obtained from the "left" host and the rightrsasigkey value is obtained from the "right" host.

Restart `ipsec` to ensure it reads the new configuration:

```bash
~]# systemctl restart ipsec
```

When using the `auto=start` option, the IPsec tunnel should be established within a few seconds. You can manually load and start the tunnel by entering the following commands as `root`:

```bash
~]# ipsec auto --add mytunnel
~]# ipsec auto --up mytunnel
```

### 4.6.3.1. Verify Host-To-Host VPN Using Libreswan

The IKE negotiation takes place on UDP port 500 and when a NAT is detected on UDP port 4500. IPsec packets show up as Encapsulated Security Payload (ESP) packets. The ESP protocol has no ports. When the VPN connection needs to pass through a NAT router, the ESP packets are encapsulated in UDP packets on port 4500.

To verify that packets are being sent through the VPN tunnel, issue a command as `root` in the following format:

```bash
~]# tcpdump -n -i interface esp or udp port 500 or udp port 4500
```

00:32:32.632165 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1a), length 132
00:32:32.632592 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1a), length 132
00:32:32.632592 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 7, length 64
00:32:33.632221 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1b), length 132
00:32:33.632731 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1b), length 132
00:32:33.632731 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 8, length 64
00:32:34.632183 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1c), length 132
00:32:34.632607 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1c), length 132
00:32:34.632607 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 9, length 64
00:32:35.632233 IP 192.1.2.45 > 192.1.2.23: ESP(spi=0x63ad7e17,seq=0x1d), length 132
00:32:35.632685 IP 192.1.2.23 > 192.1.2.45: ESP(spi=0x4841b647,seq=0x1d), length 132
00:32:35.632685 IP 192.0.2.254 > 192.0.1.254: ICMP echo reply, id 2489, seq 10, length 64

Where interface is the interface known to carry the traffic. To end the capture with tcpdump, press Ctrl+C.

**NOTE**

The tcpdump commands interacts a little unexpectedly with IPsec. It only sees the outgoing encrypted packet, not the outgoing plaintext packet. It does see the encrypted incoming packet, as well as the decrypted incoming packet. If possible, run tcpdump on a router between the two machines and not on one of the endpoints itself. When using the Virtual Tunnel Interface (VTI), the physical interface shows ESP packets, while the VTI interface shows the cleartext traffic.

### 4.6.4. Site-to-Site VPN Using Libreswan

In order for Libreswan to create a site-to-site IPsec VPN, joining together two networks, an IPsec tunnel is created between two hosts, endpoints, which are configured to permit traffic from one or more subnets to pass through. They can therefore be thought of as gateways to the remote portion of the network. The configuration of the site-to-site VPN only differs from the host-to-host VPN in that one or more networks or subnets must be specified in the configuration file.

To configure Libreswan to create a site-to-site IPsec VPN, first configure a host-to-host IPsec VPN as described in Section 4.6.3, “Host-To-Host VPN Using Libreswan” and then copy or move the file to a file with a suitable name, such as `/etc/ipsec.d/my_site-to-site.conf`. Using an editor running as root, edit the custom configuration file `/etc/ipsec.d/my_site-to-site.conf` as follows:

```plaintext
conn mysubnet
  also=mytunnel
  leftsubnet=192.0.1.0/24
  rightsubnet=192.0.2.0/24

conn mysubnet6
  also=mytunnel
  connaddrfamily=ipv6
  leftsubnet=2001:db8:0:1::/64
  rightsubnet=2001:db8:0:2::/64

conn mytunnel
  auto=start
  leftid=@west.example.com
  left=192.1.2.23
  leftrsasigkey=0sAQOrlo+hOafUZDlCQmXFrje/oZm ... W2n417C/4urYHQkCvuIQ==
  rightid=@east.example.com
```
To bring the tunnels up, restart Libreswan or manually load and initiate all the connections using the following commands as root:

~]# ipsec auto --add mysubnet

~]# ipsec auto --add mysubnet6

~]# ipsec auto --add mytunnel

~]# ipsec auto --up mysubnet
104 "mysubnet" #1: STATE_MAIN_I1: initiate
003 "mysubnet" #1: received Vendor ID payload [Dead Peer Detection]
003 "mytunnel" #1: received Vendor ID payload [FRAGMENTATION]
106 "mysubnet" #1: STATE_MAIN_I2: sent MI2, expecting MR2
108 "mysubnet" #1: STATE_MAIN_I3: sent MI3, expecting MR3
003 "mytunnel" #1: received Vendor ID payload [CAN-IKEv2]
004 "mysubnet" #1: STATE_MAIN_I4: ISAKMP SA established
{auth=OAKLEY_RSA_SIG cipher=aes_128 prf=oakley_sha group=modp2048}
117 "mysubnet" #2: STATE_QUICK_I1: initiate
004 "mysubnet" #2: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel
mode {ESP=>0x9414a615 <0x1a8eb4ef xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none DPD=none}

~]# ipsec auto --up mysubnet6
003 "mytunnel" #1: received Vendor ID payload [FRAGMENTATION]
117 "mysubnet" #2: STATE_QUICK_I1: initiate
004 "mysubnet" #2: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel
mode {ESP=>0x06fe2099 <0x75eaa862 xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none DPD=none}

~]# ipsec auto --up mytunnel
104 "mytunnel" #1: STATE_MAIN_I1: initiate
003 "mytunnel" #1: received Vendor ID payload [Dead Peer Detection]
003 "mytunnel" #1: received Vendor ID payload [FRAGMENTATION]
106 "mytunnel" #1: STATE_MAIN_I2: sent MI2, expecting MR2
108 "mytunnel" #1: STATE_MAIN_I3: sent MI3, expecting MR3
003 "mytunnel" #1: received Vendor ID payload [CAN-IKEv2]
004 "mytunnel" #1: STATE_MAIN_I4: ISAKMP SA established
{auth=OAKLEY_RSA_SIG cipher=aes_128 prf=oakley_sha group=modp2048}
117 "mytunnel" #2: STATE_QUICK_I1: initiate
004 "mytunnel" #2: STATE_QUICK_I2: sent QI2, IPsec SA established tunnel
mode {ESP=>0x9414a615 >0x1a8eb4ef xfrm=AES_128-HMAC_SHA1 NATOA=none NATD=none DPD=none}

4.6.4.1. Verify Site-to-Site VPN Using Libreswan

Verifying that packets are being sent through the VPN tunnel is the same procedure as explained in Section 4.6.3.1, “Verify Host-To-Host VPN Using Libreswan”.

88
4.6.5. Site-to-Site Single Tunnel VPN Using Libreswan

Often, when a site-to-site tunnel is built, the gateways need to communicate with each other using their internal IP addresses instead of their public IP addresses. This can be accomplished using a single tunnel. If the left host, with host name west, has internal IP address 192.0.1.254 and the right host, with host name east, has internal IP address 192.0.2.254, the following configuration using a single tunnel can be used:

```plaintext
conn mysubnet
  leftid=@west.example.com
  leftrsasigkey=0sAQOrlo+hOafUZDLcQmXFrje/oZm [...] W2n417C/4urYH6q4CuV1IQ==
  left=192.1.2.23
  leftsourceip=192.0.1.254
  leftsubnet=192.0.1.0/24
  rightid=@east.example.com
  rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
  right=192.1.2.45
  rightsourceip=192.0.2.254
  rightsubnet=192.0.2.0/24
  auto=start
  authby=rsasig
```

4.6.6. Subnet Extrusion Using Libreswan

IPsec is often deployed in a hub-and-spoke architecture. Each leaf node has an IP range that is part of a larger range. Leaves communicate with each other through the hub. This is called subnet extrusion. In the example below, we configure the head office with 10.0.0.0/8 and two branches that use a smaller/24 subnet.

At the head office:

```plaintext
conn branch1
  left=1.2.3.4
  leftid=@headoffice
  leftsubnet=0.0.0.0/0
  leftrsasigkey=0sA [...] #
  right=5.6.7.8
  rightid=@branch1
  rightsubnet=10.0.1.0/24
  rightssigkey=0sAXXXX [...] #
  auto=start
  authby=rsasig

conn branch2
  left=1.2.3.4
  leftid=@headoffice
  leftsubnet=0.0.0.0/0
  leftrsasigkey=0sA [...] #
  right=10.11.12.13
  rightid=@branch2
  rightsubnet=10.0.2.0/24
```
At the “branch1” office, we use the same connection. Additionally, we use a pass-through connection to exclude our local LAN traffic from being sent through the tunnel:

```
conn branch1
 left=1.2.3.4
leftid=@headoffice
leftsubnet=0.0.0.0/0
leftrsasigkey=0sA[...]

right=10.11.12.13
rightid=@branch2
rightsubnet=10.0.1.0/24
rightrsasigkey=0sAYYY[...]

auto=start
authby=rsasig
```

```
conn passthrough
left=1.2.3.4
right=0.0.0.0
leftsubnet=10.0.1.0/24
rightsubnet=10.0.1.0/24
authby=never
type=passthrough
auto=route
```

4.6.7. Road Warrior Application Using Libreswan

Road Warriors are traveling users with mobile clients with a dynamically assigned IP address, such as laptops. These are authenticated using certificates. To avoid needing to use the old IKEv1 XAUTH protocol, IKEv2 is used in the following example:

On the server:

```
conn roadwarriors
ikev2=insist
fragmentation=yes
left=1.2.3.4

# if access to the LAN is given, enable this, otherwise use 0.0.0.0/0
# leftsubnet=10.10.0.0/16
leftsubnet=0.0.0.0/0
leftcert=gw.example.com
leftid=%fromcert
leftxauthserver=yes
leftmodecfgserver=yes
right=%any

# trust our own Certificate Agency
rightca=%same

# pick an IP address pool to assign to remote users
# 100.64.0.0/16 prevents RFC1918 clashes when remote users are behind
```
On the mobile client, the Road Warrior’s device, we need to use a slight variation of the above configuration:

```
conn to-vpn-server
  ikev2=insist
  # pick up our dynamic IP
  left=%defaultroute
  leftsubnet=0.0.0.0/0
  leftcert=myname.example.com
  leftid=%fromcert
  leftmodecfgclient=yes
  # right can also be a DNS hostname
  right=1.2.3.4
  # if access to the remote LAN is required, enable this, otherwise use
  rightsubnet=0.0.0.0/0
  # rightsubnet=10.10.0.0/16
  rightsubnet=0.0.0.0/0
  # trust our own Certificate Agency
  rightca=%same
  authby=rsasig
  # allow narrowing to the server’s suggested assigned IP and remote subnet
  narrowing=yes
  # Initiate connection
  auto=start
```

4.6.8. Road Warrior Application Using Libreswan and XAUTH with X.509

**Libreswan** offers a method to natively assign IP address and DNS information to roaming VPN clients as the connection is established by using the XAUTH IPSec extension. XAUTH can be deployed using PSK or X.509 certificates. Deploying using X.509 is more secure. Client certificates can be revoked by a certificate revocation list or by Online Certificate Status Protocol (OCSP). With X.509 certificates, individual clients cannot impersonate the server. With a PSK, also called Group Password, this is theoretically possible.

XAUTH requires the VPN client to additionally identify itself with a user name and password. For One time Passwords (OTP), such as Google Authenticator or RSA SecureID tokens, the one-time token is appended to the user password.
There are three possible back ends for XAUTH:

xauthby=pam

This uses the configuration in /etc/pam.d/pluto to authenticate the user. Pam can be configured to use various back ends by itself. It can use the system account user-password scheme, an LDAP directory, a RADIUS server or a custom password authentication module.

xauthby=file

This uses the configuration file /etc/ipsec.d/passwd (not to be confused with /etc/ipsec.d/nsspassword). The format of this file is similar to the Apache .htpasswd file and the Apache htpasswd command can be used to create entries in this file. However, after the user name and password, a third column is required with the connection name of the IPsec connection used, for example when using a conn remoteusers to offer VPN to remove users, a password file entry should look as follows:

```
user1:$apr1$MIwQ3DHb$1I69LzTnZhnCT2DQmAOK.:remoteusers
```

NOTE: when using the htpasswd command, the connection name has to be manually added after the user:password part on each line.

xauthby=alwaysok

The server will always pretend the XAUTH user and password combination was correct. The client still has to specify a user name and a password, although the server ignores these. This should only be used when users are already identified by X.509 certificates, or when testing the VPN without needing an XAUTH back end.

An example configuration with X.509 certificates:

```
conn xauth-rsa
  ikev2=never
  auto=add
  authby=rsasig
  pfs=no
  rekey=no
  left=ServerIP
  leftcert=vpn.example.com
  #leftid=%fromcert
  leftid=vpn.example.com
  leftsendcert=always
  leftsubnet=0.0.0.0/0
  rightaddresspool=10.234.123.2-10.234.123.254
  right=%any
  rightrsasigkey=%cert
  modecfgdns1=1.2.3.4
  modecfgdns2=8.8.8.8
  modecfgdomain=example.com
  modecfgbanner="Authorized Access is allowed"
  leftxauthserver=yes
  rightxauthclient=yes
  leftmodecfgserver=yes
  rightmodecfgclient=yes
```
modecfgpull=yes
xauthtby=pam
dpddelay=30
dpdttimeout=120
dpdaction=clear
ike_frag=yes
# for walled-garden on xauth failure
# xauthfail=soft
# leftupdown=/custom/_updown

When `xauthfail` is set to soft, instead of hard, authentication failures are ignored, and the VPN is setup as if the user authenticated properly. A custom updown script can be used to check for the environment variable `XAUTH_FAILED`. Such users can then be redirected, for example, using `iptables` DNAT, to a “walled garden” where they can contact the administrator or renew a paid subscription to the service.

VPN clients use the `modecfgdomain` value and the DNS entries to redirect queries for the specified domain to these specified nameservers. This allows roaming users to access internal-only resources using the internal DNS names.

If `leftsubnet` is not `0.0.0.0/0`, split tunneling configuration requests are sent automatically to the client. For example, when using `leftsubnet=10.0.0.0/8`, the VPN client would only send traffic for `10.0.0.0/8` through the VPN.

### 4.6.9. Additional Resources

The following sources of information provide additional resources regarding Libreswan and the `ipsec` daemon.

#### 4.6.9.1. Installed Documentation

- `ipsec(8)` man page — Describes command options for `ipsec`.
- `ipsec.conf(5)` man page — Contains information on configuring `ipsec`.
- `ipsec.secrets(5)` man page — Describes the format of the `ipsec.secrets` file.
- `ipsec_auto(8)` man page — Describes the use of the `auto` command line client for manipulating Libreswan IPsec connections established using automatic exchanges of keys.
- `ipsec_rsasigkey(8)` man page — Describes the tool used to generate RSA signature keys.
- `/usr/share/doc/libreswan-version/README.nss` — Describes the commands for using raw RSA keys and certificates with the NSS crypto library used with the Libreswan `pluto` daemon.

#### 4.6.9.2. Online Documentation

[https://libreswan.org](https://libreswan.org)

The website of the upstream project.


Network Security Services (NSS) project.
4.7. USING OPENSSL

OpenSSL is a library that provides cryptographic protocols to applications. The `openssl` command line utility enables using the cryptographic functions from the shell. It includes an interactive mode.

The `openssl` command line utility has a number of pseudo-commands to provide information on the commands that the version of `openssl` installed on the system supports. The pseudo-commands `list-standard-commands`, `list-message-digest-commands`, and `list-cipher-commands` output a list of all standard commands, message digest commands, or cipher commands, respectively, that are available in the present `openssl` utility.

The pseudo-commands `list-cipher-algorithms` and `list-message-digest-algorithms` list all cipher and message digest names. The pseudo-command `list-public-key-algorithms` lists all supported public key algorithms. For example, to list the supported public key algorithms, issue the following command:

```
~]$ openssl list-public-key-algorithms
```

The pseudo-command `no-command-name` tests whether a `command-name` of the specified name is available. Intended for use in shell scripts. See man `openssl(1)` for more information.

4.7.1. Creating and Managing Encryption Keys

With OpenSSL, public keys are derived from the corresponding private key. Therefore the first step, once having decided on the algorithm, is to generate the private key. In these examples the private key is referred to as `privkey.pem`. For example, to create an RSA private key using default parameters, issue the following command:

```
~]$ openssl genpkey -algorithm RSA -out privkey.pem
```

The RSA algorithm supports the following options:

- `rsa_keygen_bits:numbits` — The number of bits in the generated key. If not specified 1024 is used.

- `rsa_keygen_pubexp:value` — The RSA public exponent value. This can be a large decimal value, or a hexadecimal value if preceded by 0x. The default value is 65537.

For example, to create a 2048 bit RSA private key using 3 as the public exponent, issue the following command:

```
~]$ openssl genpkey -algorithm RSA -out privkey.pem -rsa_keygen_bits:2048 -pkeyopt rsa_keygen_pubexp:3
```

To encrypt the private key as it is output using 128 bit AES and the passphrase “hello”, issue the following command:

```
~]$ openssl genpkey -algorithm RSA -out privkey.pem -aes-128-cbc -pass pass:hello
```

See man `genpkey(1)` for more information on generating private keys.
4.7.2. Generating Certificates

To generate a certificate using **OpenSSL**, it is necessary to have a private key available. In these examples the private key is referred to as `privkey.pem`. If you have not yet generated a private key, see Section 4.7.1, “Creating and Managing Encryption Keys”

To have a certificate signed by a certificate authority (CA), it is necessary to generate a certificate and then send it to a CA for signing. This is referred to as a certificate signing request. See Section 4.7.2.1, “Creating a Certificate Signing Request” for more information. The alternative is to create a self-signed certificate. See Section 4.7.2.2, “Creating a Self-signed Certificate” for more information.

4.7.2.1. Creating a Certificate Signing Request

To create a certificate for submission to a CA, issue a command in the following format:

```bash
~]$ openssl req -new -key privkey.pem -out cert.csr
```

This will create an X.509 certificate called `cert.csr` encoded in the default privacy-enhanced electronic mail (PEM) format. The name PEM is derived from “Privacy Enhancement for Internet Electronic Mail” described in RFC 1424. To generate a certificate file in the alternative DER format, use the `-outform DER` command option.

After issuing the above command, you will be prompted for information about you and the organization in order to create a distinguished name (DN) for the certificate. You will need the following information:

- The two letter country code for your country
- The full name of your state or province
- City or Town
- The name of your organization
- The name of the unit within your organization
- Your name or the host name of the system
- Your email address

The `req(1)` man page describes the PKCS# 10 certificate request and generating utility. Default settings used in the certificate creating process are contained within the `/etc/pki/tls/openssl.cnf` file. See man `openssl.cnf(5)` for more information.

4.7.2.2. Creating a Self-signed Certificate

To generate a self-signed certificate, valid for 366 days, issue a command in the following format:

```bash
~]$ openssl req -new -x509 -key privkey.pem -out selfcert.pem -days 366
```

4.7.2.3. Creating a Certificate Using a Makefile
The /etc/pki/tls/certs/ directory contains a Makefile which can be used to create certificates using the make command. To view the usage instructions, issue a command as follows:

```
~]$ make -f /etc/pki/tls/certs/Makefile
```

Alternatively, change to the directory and issue the make command as follows:

```
~]$ cd /etc/pki/tls/certs/
~]$ make
```

See the make(1) man page for more information.

### 4.7.3. Verifying Certificates

A certificate signed by a CA is referred to as a trusted certificate. A self-signed certificate is therefore an untrusted certificate. The verify utility uses the same SSL and S/MIME functions to verify a certificate as is used by OpenSSL in normal operation. If an error is found it is reported and then an attempt is made to continue testing in order to report any other errors.

To verify multiple individual X.509 certificates in PEM format, issue a command in the following format:

```
~]$ openssl verify cert1.pem cert2.pem
```

To verify a certificate chain the leaf certificate must be in cert.pem and the intermediate certificates which you do not trust must be directly concatenated in untrusted.pem. The trusted root CA certificate must be either among the default CA listed in /etc/pki/tls/certs/ca-bundle.crt or in a cacert.pem file. Then, to verify the chain, issue a command in the following format:

```
~]$ openssl verify -untrusted untrusted.pem -CAfile cacert.pem cert.pem
```

See man verify(1) for more information.

**IMPORTANT**

Verification of signatures using the MD5 hash algorithm is disabled in Red Hat Enterprise Linux 7 due to insufficient strength of this algorithm. Always use strong algorithms such as SHA256.

### 4.7.4. Encrypting and Decrypting a File

For encrypting (and decrypting) files with OpenSSL, either the pkeyutl or enc built-in commands can be used. With pkeyutl, RSA keys are used to perform the encrypting and decrypting, whereas with enc, symmetric algorithms are used.

**Using RSA Keys**

To encrypt a file called plaintext, issue a command as follows:

```
~]$ openssl pkeyutl -in plaintext -out cyphertext -inkey privkey.pem
```
The default format for keys and certificates is PEM. If required, use the `-keyform DER` option to specify the DER key format.

To specify a cryptographic engine, use the `-engine` option as follows:

```
~$ openssl pkeyutl -in plaintext -out cyphertext -inkey privkey.pem -engine id
```

Where `id` is the ID of the cryptographic engine. To check the availability of an engine, issue the following command:

```
~$ openssl engine -t
```

To sign a data file called `plaintext`, issue a command as follows:

```
~$ openssl pkeyutl -sign -in plaintext -out sigtext -inkey privkey.pem
```

To verify a signed data file and to extract the data, issue a command as follows:

```
~$ openssl pkeyutl -verifyrecover -in sig -inkey key.pem
```

To verify the signature, for example using a DSA key, issue a command as follows:

```
~$ openssl pkeyutl -verify -in file -sigfile sig -inkey key.pem
```

The `pkeyutl(1)` manual page describes the public key algorithm utility.

**Using Symmetric Algorithms**

To list available symmetric encryption algorithms, execute the `enc` command with an unsupported option, such as `-l`:

```
~$ openssl enc -l
```

To specify an algorithm, use its name as an option. For example, to use the `aes-128-cbc` algorithm, use the following syntax:

```
openssl enc -aes-128-cbc
```

To encrypt a file called `plaintext` using the `aes-128-cbc` algorithm, enter the following command:

```
~$ openssl enc -aes-128-cbc -in plaintext -out plaintext.aes-128-cbc
```

To decrypt the file obtained in the previous example, use the `-d` option as in the following example:

```
~$ openssl enc -aes-128-cbc -d -in plaintext.aes-128-cbc -out plaintext
```
IMPORTANT

The enc command does not properly support AEAD ciphers, and the ecb mode is not considered secure. For best results, do not use other modes than cbc, cfb, ofb, or ctr.

4.7.5. Generating Message Digests

The dgst command produces the message digest of a supplied file or files in hexadecimal form. The command can also be used for digital signing and verification. The message digest command takes the following form:

```
openssl dgst algorithm -out filename -sign private-key
```

Where algorithm is one of md5|md4|md2|sha1|sha|mdc2|ripemd160|dss1. At time of writing, the SHA1 algorithm is preferred. If you need to sign or verify using DSA, then the dss1 option must be used together with a file containing random data specified by the -rand option.

To produce a message digest in the default Hex format using the sha1 algorithm, issue the following command:

```
~$ openssl dgst sha1 -out digest-file
```

To digitally sign the digest, using a private key privkey.pem, issue the following command:

```
~$ openssl dgst sha1 -out digest-file -sign privkey.pem
```

See man dgst(1) for more information.

4.7.6. Generating Password Hashes

The passwd command computes the hash of a password. To compute the hash of a password on the command line, issue a command as follows:

```
~$ openssl passwd password
```

The -crypt algorithm is used by default.

To compute the hash of a password from standard input, using the MD5 based BSD algorithm 1, issue a command as follows:

```
~$ openssl passwd -1 password
```

The -apr1 option specifies the Apache variant of the BSD algorithm.

To compute the hash of a password stored in a file, and using a salt xx, issue a command as follows:

```
~$ openssl passwd -salt xx -in password-file
```
The password is sent to standard output and there is no -out option to specify an output file. The -table will generate a table of password hashes with their corresponding clear text password.

See man sslpasswd(1) for more information and examples.

4.7.7. Generating Random Data

To generate a file containing random data, using a seed file, issue the following command:

```
~]$ openssl rand -out rand-file -rand seed-file
```

Multiple files for seeding the random data process can be specified using the colon, : , as a list separator.

See man rand(1) for more information.

4.7.8. Benchmarking Your System

To test the computational speed of a system for a given algorithm, issue a command in the following format:

```
~]$ openssl speed algorithm
```

where algorithm is one of the supported algorithms you intended to use. To list the available algorithms, type openssl speed and then press tab.

4.7.9. Configuring OpenSSL

OpenSSL has a configuration file /etc/pki/tls/openssl.cnf, referred to as the master configuration file, which is read by the OpenSSL library. It is also possible to have individual configuration files for each application. The configuration file contains a number of sections with section names as follows: [ section_name ]. Note the first part of the file, up until the first [ section_name ], is referred to as the default section. When OpenSSL is searching for names in the configuration file the named sections are searched first. All OpenSSL commands use the master OpenSSL configuration file unless an option is used in the command to specify an alternative configuration file. The configuration file is explained in detail in the config(5) man page.

Two RFCs explain the contents of a certificate file. They are:

- Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile
- Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile

4.8. USING STUNNEL

The stunnel program is an encryption wrapper between a client and a server. It listens on the port specified in its configuration file, encrypts the communication with the client, and forwards the data to the original daemon listening on its usual port. This way, you can secure any service that itself does not support any type of encryption, or improve the security of a service that uses a type of encryption that you want to avoid for security
reasons, such as SSL versions 2 and 3, affected by the POODLE SSL vulnerability (CVE-2014-3566). See https://access.redhat.com/solutions/1234773 for details. CUPS is an example of a component that does not provide a way to disable SSL in its own configuration.

4.8.1. Installing stunnel

Install the stunnel package by entering the following command as root:

```bash
~$ yum install stunnel
```

4.8.2. Configuring stunnel as a TLS Wrapper

To configure stunnel, follow these steps:

1. You need a valid certificate for stunnel regardless of what service you use it with. If you do not have a suitable certificate, you can apply to a Certificate Authority to obtain one, or you can create a self-signed certificate.

   **WARNING**
   
   Always use certificates signed by a Certificate Authority for servers running in a production environment. Self-signed certificates are only appropriate for testing purposes or private networks.

See Section 4.7.2.1, “Creating a Certificate Signing Request” for more information about certificates granted by a Certificate Authority. On the other hand, to create a self-signed certificate for stunnel, enter the `/etc/pki/tls/certs/` directory and type the following command as root:

```bash
certs]$ make stunnel.pem
```

Answer all of the questions to complete the process.

2. When you have a certificate, create a configuration file for stunnel. It is a text file in which every line specifies an option or the beginning of a service definition. You can also keep comments and empty lines in the file to improve its legibility, where comments start with a semicolon.

   The stunnel RPM package contains the `/etc/stunnel/` directory, in which you can store the configuration file. Although stunnel does not require any special format of the file name or its extension, use `/etc/stunnel/stunnel.conf`. The following content configures stunnel as a TLS wrapper:

```bash
cert = /etc/pki/tls/certs/stunnel.pem
; Allow only TLS, thus avoiding SSL
sslVersion = TLSv1
chroot = /var/run/stunnel
```
setuid = nobody
setgid = nobody
pid = /stunnel.pid
socket = l:TCP_NODELAY=1
socket = r:TCP_NODELAY=1

[service_name]
accept = port
cconnect = port
TIMEOUTclose = 0

Alternatively, you can avoid SSL by replacing the line containing `sslVersion = TLSv1` with the following lines:

options = NO_SSLv2
options = NO_SSLv3

The purpose of the options is as follows:

- `cert` — the path to your certificate
- `sslVersion` — the version of SSL; note that you can use `TLS` here even though SSL and TLS are two independent cryptographic protocols
- `chroot` — the changed root directory in which the stunnel process runs, for greater security
- `setuid, setgid` — the user and group that the `stunnel` process runs as; `nobody` is a restricted system account
- `pid` — the file in which `stunnel` saves its process ID, relative to `chroot`
- `socket` — local and remote socket options; in this case, disable Nagle's algorithm to improve network latency
- `[service_name]` — the beginning of the service definition; the options used below this line apply to the given service only, whereas the options above affect `stunnel` globally
- `accept` — the port to listen on
- `connect` — the port to connect to; this must be the port that the service you are securing uses
- `TIMEOUTclose` — how many seconds to wait for the `close_notify` alert from the client; `0` instructs `stunnel` not to wait at all
- `options` — OpenSSL library options

Example 4.2. Securing CUPS

To configure stunnel as a TLS wrapper for `CUPS`, use the following values:

[cups]
accept = 632
cconnect = 631
Instead of 632, you can use any free port that you prefer. 631 is the port that CUPS normally uses.

3. Create the chroot directory and give the user specified by the setuid option write access to it. To do so, enter the following commands as root:

```
~]# mkdir /var/run/stunnel
~]# chown nobody:nobody /var/run/stunnel
```

This allows stunnel to create the PID file.

4. If your system is using firewall settings that disallow access to the new port, change them accordingly. See Section 5.3.1.6, “Opening Ports in the Firewall” for details.

5. When you have created the configuration file and the chroot directory, and when you are sure that the specified port is accessible, you are ready to start using stunnel.

### 4.8.3. Starting, Stopping, and Restarting stunnel

To start stunnel, enter the following command as root:

```
~]# stunnel /etc/stunnel/stunnel.conf
```

By default, stunnel uses /var/log/secure to log its output.

To terminate stunnel, kill the process by running the following command as root:

```
~]# kill `cat /var/run/stunnel/stunnel.pid`
```

If you edit the configuration file while stunnel is running, terminate stunnel and start it again for your changes to take effect.

### 4.9. ENCRYPTION

#### 4.9.1. Using LUKS Disk Encryption

Linux Unified Key Setup-on-disk-format (or LUKS) allows you to encrypt partitions on your Linux computer. This is particularly important when it comes to mobile computers and removable media. LUKS allows multiple user keys to decrypt a master key, which is used for the bulk encryption of the partition.

**Overview of LUKS**

**What LUKS does**

- LUKS encrypts entire block devices and is therefore well-suited for protecting the contents of mobile devices such as removable storage media or laptop disk drives.

- The underlying contents of the encrypted block device are arbitrary. This makes it useful for encrypting swap devices. This can also be useful with certain databases that use specially formatted block devices for data storage.
LUKS uses the existing device mapper kernel subsystem.

LUKS provides passphrase strengthening which protects against dictionary attacks.

LUKS devices contain multiple key slots, allowing users to add backup keys or passphrases.

What LUKS does not do:

- LUKS is not well-suited for applications requiring many (more than eight) users to have distinct access keys to the same device.

- LUKS is not well-suited for applications requiring file-level encryption.

4.9.1.1. LUKS Implementation in Red Hat Enterprise Linux

Red Hat Enterprise Linux 7 utilizes LUKS to perform file system encryption. By default, the option to encrypt the file system is unchecked during the installation. If you select the option to encrypt your hard drive, you will be prompted for a passphrase that will be asked every time you boot the computer. This passphrase "unlocks" the bulk encryption key that is used to decrypt your partition. If you choose to modify the default partition table you can choose which partitions you want to encrypt. This is set in the partition table settings.

The default cipher used for LUKS (see `cryptsetup --help`) is aes-cbc-essiv:sha256 (ESSIV - Encrypted Salt-Sector Initialization Vector). Note that the installation program, Anaconda, uses by default XTS mode (aes-xts-plain64). The default key size for LUKS is 256 bits. The default key size for LUKS with Anaconda (XTS mode) is 512 bits. Ciphers that are available are:

- AES - Advanced Encryption Standard - FIPS PUB 197
- Twofish (A 128-bit Block Cipher)
- Serpent
- cast5 - RFC 2144
- cast6 - RFC 2612

4.9.1.2. Manually Encrypting Directories

**WARNING**

Following this procedure will remove all data on the partition that you are encrypting. You WILL lose all your information! Make sure you backup your data to an external source before beginning this procedure!

1. Enter runlevel 1 by typing the following at a shell prompt as root:

   ```
   ```
1. Unmount your existing `/home`

```
umount /home
```

2. Unmount your existing `/home`:

```
2. Unmount your existing `/home`
```

3. If the command in the previous step fails, use `fuser` to find processes hogging `/home` and kill them:

```
fuser -mvk /home
```

4. Verify `/home` is no longer mounted:

```
grep home /proc/mounts
```

5. Fill your partition with random data:

```
shred -v --iterations=1 /dev/VG00/LV_home
```

This command proceeds at the sequential write speed of your device and may take some time to complete. It is an important step to ensure no unencrypted data is left on a used device, and to obfuscate the parts of the device that contain encrypted data as opposed to just random data.

6. Initialize your partition:

```
cryptsetup --verbose --verify-passphrase luksFormat /dev/VG00/LV_home
```

7. Open the newly encrypted device:

```
cryptsetup luksOpen /dev/VG00/LV_home home
```

8. Make sure the device is present:

```
ls -l /dev/mapper | grep home
```

9. Create a file system:

```
mkfs.ext3 /dev/mapper/home
```

10. Mount the file system:

```
mount /dev/mapper/home /home
```

11. Make sure the file system is visible:

```
df -h | grep home
```

12. Add the following to the `/etc/crypttab` file:
13. Edit the `/etc/fstab` file, removing the old entry for `/home` and adding the following line:

```
/home /dev/VG00/LV_home none
```

14. Restore default SELinux security contexts:

```
/sbin/restorecon -v -R /home
```

15. Reboot the machine:

```
shutdown -r now
```

16. The entry in the `/etc/crypttab` makes your computer ask your `luks` passphrase on boot.

17. Log in as root and restore your backup.

You now have an encrypted partition for all of your data to safely rest while the computer is off.

### 4.9.1.3. Add a New Passphrase to an Existing Device

Use the following command to add a new passphrase to an existing device:

```
cryptsetup luksAddKey device
```

After being prompted for any one of the existing passphrases for authentication, you will be prompted to enter the new passphrase.

### 4.9.1.4. Remove a Passphrase from an Existing Device

Use the following command to remove a passphrase from an existing device:

```
cryptsetup luksRemoveKey device
```

You will be prompted for the passphrase you want to remove and then for any one of the remaining passphrases for authentication.

### 4.9.1.5. Creating Encrypted Block Devices in Anaconda

You can create encrypted devices during system installation. This allows you to easily configure a system with encrypted partitions.

To enable block device encryption, check the **Encrypt System** check box when selecting automatic partitioning or the **Encrypt** check box when creating an individual partition, software RAID array, or logical volume. After you finish partitioning, you will be prompted for an encryption passphrase. This passphrase will be required to access the encrypted devices. If you have pre-existing LUKS devices and provided correct passphrases for them...
earlier in the install process the passphrase entry dialog will also contain a check box. Checking this check box indicates that you would like the new passphrase to be added to an available slot in each of the pre-existing encrypted block devices.

NOTE
Checking the Encrypt System check box on the Automatic Partitioning screen and then choosing Create custom layout does not cause any block devices to be encrypted automatically.

NOTE
You can use kickstart to set a separate passphrase for each new encrypted block device.

4.9.1.6. Additional Resources
For additional information on LUKS or encrypting hard drives under Red Hat Enterprise Linux 7 visit one of the following links:

- LUKS home page
- LUKS/cryptsetup FAQ
- LUKS - Linux Unified Key Setup Wikipedia article
- HOWTO: Creating an encrypted Physical Volume (PV) using a second hard drive and pvmove

4.9.2. Creating GPG Keys
GPG is used to identify yourself and authenticate your communications, including those with people you do not know. GPG allows anyone reading a GPG-signed email to verify its authenticity. In other words, GPG allows someone to be reasonably certain that communications signed by you actually are from you. GPG is useful because it helps prevent third parties from altering code or intercepting conversations and altering the message.

4.9.2.1. Creating GPG Keys in GNOME
To create a GPG Key in GNOME, follow these steps:

1. Install the Seahorse utility, which makes GPG key management easier:

   ~]# yum install seahorse

2. To create a key, from the Applications → Accessories menu select Passwords and Encryption Keys, which starts the application Seahorse.

3. From the File menu select New and then PGP Key. Then click Continue.

4. Type your full name, email address, and an optional comment describing who you are (for example: John C. Smith, jsmith@example.com, Software Engineer). Click Create. A dialog is displayed asking for a passphrase for the key. Choose a strong
passphrase but also easy to remember. Click **OK** and the key is created.

**WARNING**

If you forget your passphrase, you will not be able to decrypt the data.

To find your GPG key ID, look in the **Key ID** column next to the newly created key. In most cases, if you are asked for the key ID, prepend **0x** to the key ID, as in `0x6789ABCD`. You should make a backup of your private key and store it somewhere secure.

### 4.9.2.2. Creating GPG Keys in KDE

To create a GPG Key in **KDE**, follow these steps:

1. Start the **KPGp** program from the main menu by selecting **Applications → Utilities → Encryption Tool**. If you have never used **KPGp** before, the program walks you through the process of creating your own GPG keypair.

2. A dialog box appears prompting you to create a new key pair. Enter your name, email address, and an optional comment. You can also choose an expiration time for your key, as well as the key strength (number of bits) and algorithms.

3. Enter your passphrase in the next dialog box. At this point, your key appears in the main **KPGp** window.

**WARNING**

If you forget your passphrase, you will not be able to decrypt the data.

To find your GPG key ID, look in the **Key ID** column next to the newly created key. In most cases, if you are asked for the key ID, prepend **0x** to the key ID, as in `0x6789ABCD`. You should make a backup of your private key and store it somewhere secure.

### 4.9.2.3. Creating GPG Keys Using the Command Line

1. Use the following shell command:

   ```
   $ gpg2 --gen-key
   ```

   This command generates a key pair that consists of a public and a private key. Other people use your public key to authenticate and decrypt your communications. Distribute your public key as widely as possible, especially to people who you know will want to receive authentic communications from you, such as a mailing list.

2. A series of prompts directs you through the process. Press the **Enter** key to assign a
default value if desired. The first prompt asks you to select what kind of key you prefer:

Please select what kind of key you want:
(1) RSA and RSA (default)
(2) DSA and Elgamal
(3) DSA (sign only)
(4) RSA (sign only)
Your selection?

In almost all cases, the default is the correct choice. An RSA/RSA key allows you not only to sign communications, but also to encrypt files.

3. Choose the key size:

RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)

Again, the default, 2048, is sufficient for almost all users, and represents an extremely strong level of security.

4. Choose when the key will expire. It is a good idea to choose an expiration date instead of using the default, which is none. If, for example, the email address on the key becomes invalid, an expiration date will remind others to stop using that public key.

Please specify how long the key should be valid.
0 = key does not expire
d = key expires in n days
w = key expires in n weeks
m = key expires in n months
y = key expires in n years
key is valid for? (0)

Entering a value of 1y, for example, makes the key valid for one year. (You may change this expiration date after the key is generated, if you change your mind.)

5. Before the gpg2 application asks for signature information, the following prompt appears:

Is this correct (y/N)?

Enter y to finish the process.

6. Enter your name and email address for your GPG key. Remember this process is about authenticating you as a real individual. For this reason, include your real name. If you choose a bogus email address, it will be more difficult for others to find your public key. This makes authenticating your communications difficult. If you are using this GPG key for self-introduction on a mailing list, for example, enter the email address you use on that list.

Use the comment field to include aliases or other information. (Some people use different keys for different purposes and identify each key with a comment, such as "Office" or "Open Source Projects.")
7. At the confirmation prompt, enter the letter 0 to continue if all entries are correct, or use the other options to fix any problems. Finally, enter a passphrase for your secret key. The `gpg2` program asks you to enter your passphrase twice to ensure you made no typing errors.

8. Finally, `gpg2` generates random data to make your key as unique as possible. Move your mouse, type random keys, or perform other tasks on the system during this step to speed up the process. Once this step is finished, your keys are complete and ready to use:

    pub 1024D/1B2AFA1C 2005-03-31 John Q. Doe <jqdoe@example.com>
    Key fingerprint = 117C FE83 22EA B843 3E86 6486 4320 545E 1B2A FA1C
    sub 1024g/CEA4B22E 2005-03-31 [expires: 2006-03-31]

9. The key fingerprint is a shorthand "signature" for your key. It allows you to confirm to others that they have received your actual public key without any tampering. You do not need to write this fingerprint down. To display the fingerprint at any time, use this command, substituting your email address:

    ~]$ gpg2 --fingerprint jqdoe@example.com

Your "GPG key ID" consists of 8 hex digits identifying the public key. In the example above, the GPG key ID is 1B2AFA1C. In most cases, if you are asked for the key ID, prepend 0x to the key ID, as in 0x6789ABCD.

### WARNING
If you forget your passphrase, the key cannot be used and any data encrypted using that key will be lost.

#### 4.9.2.4. About Public Key Encryption

1. [Wikipedia - Public Key Cryptography](#)

2. [HowStuffWorks - Encryption](#)

#### 4.9.3. Using openCryptoki for Public-Key Cryptography

`openCryptoki` is a Linux implementation of PKCS#11, which is a *Public-Key Cryptography Standard* that defines an application programming interface (API) to cryptographic devices called tokens. Tokens may be implemented in hardware or software. This chapter provides an overview of the way the `openCryptoki` system is installed, configured, and used in Red Hat Enterprise Linux 7.

##### 4.9.3.1. Installing openCryptoki and Starting the Service

To install the basic `openCryptoki` packages on your system, including a software implementation of a token for testing purposes, enter the following command as root:
~# yum install opencryptoki

Depending on the type of hardware tokens you intend to use, you may need to install additional packages that provide support for your specific use case. For example, to obtain support for Trusted Platform Module (TPM) devices, you need to install the opencryptoki-tpmtok package.

See the Installing Packages section of the Red Hat Enterprise Linux 7 System Administrator's Guide for general information on how to install packages using the Yum package manager.

To enable the openCryptoki service, you need to run the pkcsslotd daemon. Start the daemon for the current session by executing the following command as root:

~# systemctl start pkcsslotd

To ensure that the service is automatically started at boot time, enter the following command:

~# systemctl enable pkcsslotd

See the Managing Services with systemd chapter of the Red Hat Enterprise Linux 7 System Administrator's Guide for more information on how to use systemd targets to manage services.

4.9.3.2. Configuring and Using openCryptoki

When started, the pkcsslotd daemon reads the /etc/opencryptoki/opencryptoki.conf configuration file, which it uses to collect information about the tokens configured to work with the system and about their slots.

The file defines the individual slots using key-value pairs. Each slot definition can contain a description, a specification of the token library to be used, and an ID of the slot's manufacturer. Optionally, the version of the slot's hardware and firmware may be defined. See the opencryptoki.conf(5) manual page for a description of the file's format and for a more detailed description of the individual keys and the values that can be assigned to them.

To modify the behavior of the pkcsslotd daemon at run time, use the pkcsconf utility. This tool allows you to show and configure the state of the daemon, as well as to list and modify the currently configured slots and tokens. For example, to display information about tokens, issue the following command (note that all non-root users that need to communicate with the pkcsslotd daemon must be a part of the pkcs11 system group):

~$ pkcsconf -t

See the pkcsconf(1) manual page for a list of arguments available with the pkcsconf tool.
4.9.4. Using Smart Cards to Supply Credentials to OpenSSH

The smart card is a lightweight hardware security module in a USB stick, MicroSD, or SmartCard form factor. It provides a remotely manageable secure key store. In Red Hat Enterprise Linux 7, OpenSSH supports authentication using smart cards.

To use your smart card with OpenSSH, store the public key from the card to the
```
./.ssh/authorized_keys
```
file. Install the PKCS#11 library provided by the opensc package on the client. PKCS#11 is a Public-Key Cryptography Standard that defines an application programming interface (API) to cryptographic devices called tokens. Enter the following command as root:

```
~$ yum install opensc
```

4.9.4.1. Retrieving a Public Key from a Card

To list the keys on your card, use the ssh-keygen command. Specify the shared library (OpenSC in the following example) with the `-D` directive.

```
~$ ssh-keygen -D /usr/lib64/pkcs11/opensc-pkcs11.so
ssh-rsa AAAAB3NzaC1yc...+[...]+g4Mb9
```

4.9.4.2. Storing a Public Key on a Server

To enable authentication using a smart card on a remote server, transfer the public key to the remote server. Do it by copying the retrieved string (key) and pasting it to the remote shell, or by storing your key to a file (`smartcard.pub` in the following example) and using the ssh-copy-id command:

```
~$ ssh-copy-id -f -i smartcard.pub user@hostname
user@hostname's password:
Number of key(s) added: 1
Now try logging into the machine, with: "ssh user@hostname"
and check to make sure that only the key(s) you wanted were added.
```

Storing a public key without a private key file requires to use the `SSH_COPY_ID_LEGACY=1` environment variable or the `-f` option.

4.9.4.3. Authenticating to a Server with a Key on a Smart Card
OpenSSH can read your public key from a smart card and perform operations with your 
private key without exposing the key itself. This means that the private key does not leave 
the card. To connect to a remote server using your smart card for authentication, enter the 
following command and enter the PIN protecting your card:

```bash
[localhost ~]$ ssh -I /usr/lib64/pkcs11/opensc-pkcs11.so hostname
Enter PIN for 'Test (UserPIN)'
[hostname ~]$
```

Replace the `hostname` with the actual host name to which you want to connect.

To save unnecessary typing next time you connect to the remote server, store the path to 
the PKCS#11 library in your `~/.ssh/config` file:

```bash
Host hostname
    PKCS11Provider /usr/lib64/pkcs11/opensc-pkcs11.so
```

Connect by running the `ssh` command without any additional options:

```bash
[localhost ~]$ ssh hostname
Enter PIN for 'Test (UserPIN)'
[hostname ~]$
```

### 4.9.4.4. Using ssh-agent to Automate PIN Logging In

Set up environmental variables to start using `ssh-agent`. You can skip this step in most 
cases because `ssh-agent` is already running in a typical session. Use the following 
command to check whether you can connect to your authentication agent:

```bash
~]$ ssh-add -l
Could not open a connection to your authentication agent.
~]$ eval `ssh-agent`
```

To avoid writing your PIN every time you connect using this key, add the card to the agent 
by running the following command:

```bash
~]$ ssh-add -s /usr/lib64/pkcs11/opensc-pkcs11.so
Enter PIN for 'Test (UserPIN)'
Card added: /usr/lib64/pkcs11/opensc-pkcs11.so
```

To remove the card from `ssh-agent`, use the following command:

```bash
~]$ ssh-add -e /usr/lib64/pkcs11/opensc-pkcs11.so
Card removed: /usr/lib64/pkcs11/opensc-pkcs11.so
```
NOTE

FIPS 201-2 requires explicit user action by the Personal Identity Verification (PIV) cardholder as a condition for use of the digital signature key stored on the card. OpenSC correctly enforces this requirement.

However, for some applications it is impractical to require the cardholder to enter the PIN for each signature. To cache the smart card PIN, remove the # character before the `pin_cache_ignore_user_consent = true;` option in the `/etc/opensc-x86_64.conf`.

See the Cardholder Authentication for the PIV Digital Signature Key (NISTIR 7863) report for more information.

4.9.4.5. Additional Resources

Setting up your hardware or software token is described in the Smart Card support in Red Hat Enterprise Linux 7 article.

For more information about the `pkcs11-tool` utility for managing and using smart cards and similar PKCS#11 security tokens, see the `pkcs11-tool(1)` man page.

4.9.5. Trusted and Encrypted Keys

Trusted and encrypted keys are variable-length symmetric keys generated by the kernel that utilize the kernel keyring service. The fact that the keys never appear in user space in an unencrypted form means that their integrity can be verified, which in turn means that they can be used, for example, by the extended verification module (EVM) to verify and confirm the integrity of a running system. User-level programs can only ever access the keys in the form of encrypted blobs.

Trusted keys need a hardware component: the Trusted Platform Module (TPM) chip, which is used to both create and encrypt (seal) the keys. The TPM seals the keys using a 2048-bit RSA key called the storage root key (SRK).

In addition to that, trusted keys may also be sealed using a specific set of the TPM’s platform configuration register (PCR) values. The PCR contains a set of integrity-management values that reflect the BIOS, boot loader, and operating system. This means that PCR-sealed keys can only be decrypted by the TPM on the exact same system on which they were encrypted. However, once a PCR-sealed trusted key is loaded (added to a keyring), and thus its associated PCR values are verified, it can be updated with new (or future) PCR values, so that a new kernel, for example, can be booted. A single key can also be saved as multiple blobs, each with different PCR values.

Encrypted keys do not require a TPM, as they use the kernel AES encryption, which makes them faster than trusted keys. Encrypted keys are created using kernel-generated random numbers and encrypted by a master key when they are exported into user-space blobs. This master key can be either a trusted key or a user key, which is their main disadvantage — if the master key is not a trusted key, the encrypted key is only as secure as the user key used to encrypt it.

4.9.5.1. Working with Keys

Prior to any operations with keys, relevant kernel modules need to be loaded. For trusted keys, it is the trusted module, and for encrypted keys, it is the encrypted-keys module. Use the following command as the root user to load both of these modules at once:
~]# modprobe trusted encrypted-keys

Trusted and encrypted keys can be created, loaded, exported, and updated using the keyctl utility. For detailed information about using keyctl, see keyctl(1).

NOTE

In order to use a TPM (such as for creating and sealing trusted keys), it needs to be enabled and active. This can be usually achieved through a setting in the machine's BIOS or using the tpm_setactive command from the tpm-tools package of utilities. Also, the TrouSers application needs to be installed (the trousers package), and the tcsd daemon, which is a part of the TrouSers suite, running to communicate with the TPM.

To create a trusted key using a TPM, execute the keyctl command with the following syntax:

```
keyctl add trusted name "new keylength [options]" keyring
```

Using the above syntax, an example command can be constructed as follows:

```bash
~]$ keyctl add trusted kmk "new 32" @u 642500861
```

The above example creates a trusted key called kmk with the length of 32 bytes (256 bits) and places it in the user keyring (@u). The keys may have a length of 32 to 128 bytes (256 to 1024 bits). Use the `show` subcommand to list the current structure of the kernel keyrings:

```bash
~]$ keyctl show
```

The `print` subcommand outputs the encrypted key to the standard output. To export the key to a user-space blob, use the `pipe` subcommand as follows:

```bash
~]$ keyctl pipe 642500861 > kmk.blob
```

To load the trusted key from the user-space blob, use the `add` command again with the blob as an argument:

```bash
~]$ keyctl add trusted kmk "load `cat kmk.blob`" @u 268728824
```

The TPM-sealed trusted key can then be employed to create secure encrypted keys. The following command syntax is used for generating encrypted keys:

```
~]$ keyctl add encrypted name "new [format] key-type:master-key-name keylength" keyring
```
Based on the above syntax, a command for generating an encrypted key using the already created trusted key can be constructed as follows:

```bash
~$ keyctl add encrypted encr-key "new trusted:kmk 32" @u 159771175
```

To create an encrypted key on systems where a TPM is not available, use a random sequence of numbers to generate a user key, which is then used to seal the actual encrypted keys.

```bash
~$ keyctl add user kmk-user "$\`dd if=/dev/urandom bs=1 count=32 2>/dev/null\`" @u 427069434
```

Then generate the encrypted key using the random-number user key:

```bash
~$ keyctl add encrypted encr-key "new user:kmk-user 32" @u 1012412758
```

The `list` subcommand can be used to list all keys in the specified kernel keyring:

```bash
~$ keyctl list @u
2 keys in keyring:
427069434: --alswrv 1000 1000 user: kmk-user
1012412758: --alswrv 1000 1000 encrypted: encr-key
```

**IMPORTANT**

Keep in mind that encrypted keys that are not sealed by a master trusted key are only as secure as the user master key (random-number key) used to encrypt them. Therefore, the master user key should be loaded as securely as possible and preferably early during the boot process.

### 4.9.5.2. Additional Resources

The following offline and online resources can be used to acquire additional information pertaining to the use of trusted and encrypted keys.

**Installed Documentation**

- `keyctl(1)` — Describes the use of the `keyctl` utility and its subcommands.

**Online Documentation**

- [Red Hat Enterprise Linux 7 SELinux User's and Administrator's Guide](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/selinux_user_and_administrator_guide) — The SELinux User's and Administrator's Guide for Red Hat Enterprise Linux 7 describes the basic principles of SELinux and documents in detail how to configure and use SELinux with various services, such as the Apache HTTP Server.


**See Also**

Section A.2, “Public-key Encryption” describes the public-key cryptographic approach and the various cryptographic protocols it uses.

4.9.6. Using the Random Number Generator

In order to be able to generate secure cryptographic keys that cannot be easily broken, a source of random numbers is required. Generally, the more random the numbers are, the better the chance of obtaining unique keys. Entropy for generating random numbers is usually obtained from computing environmental “noise” or using a hardware random number generator.

The rngd daemon, which is a part of the rng-tools package, is capable of using both environmental noise and hardware random number generators for extracting entropy. The daemon checks whether the data supplied by the source of randomness is sufficiently random and then stores it in the random-number entropy pool of the kernel. The random numbers it generates are made available through the /dev/random and /dev/urandom character devices.

The difference between /dev/random and /dev/urandom is that the former is a blocking device, which means it stops supplying numbers when it determines that the amount of entropy is insufficient for generating a properly random output. Conversely, /dev/urandom is a non-blocking source, which reuses the entropy pool of the kernel and is thus able to provide an unlimited supply of pseudo-random numbers, albeit with less entropy. As such, /dev/urandom should not be used for creating long-term cryptographic keys.

To install the rng-tools package, issue the following command as the root user:

~]# yum install rng-tools

To start the rngd daemon, execute the following command as root:

~]# systemctl start rngd

To query the status of the daemon, use the following command:

~]# systemctl status rngd

To start the rngd daemon with optional parameters, execute it directly. For example, to specify an alternative source of random-number input (other than /dev/hwrng), use the following command:

~]# rngd --rng-device=/dev/hwrng

The above command starts the rngd daemon with /dev/hwrng as the device from which random numbers are read. Similarly, you can use the -o (or --random-device) option to choose the kernel device for random-number output (other than the default /dev/random). See the rngd(8) manual page for a list of all available options.

To check which sources of entropy are available in a given system, execute the following command as root:

~]# systemctl status rngd
The following will be changed in a future release. https://bugzilla.redhat.com/show_bug.cgi?id=1437044

~$ rngd -vf
Unable to open file: /dev/tpm0
Available entropy sources:
  DRNG

NOTE

After entering the `rngd -v` command, the according process continues running in background. The `-b`, `--background` option (become a daemon) is applied by default.

If there is not any TPM device present, you will see only the Intel Digital Random Number Generator (DRNG) as a source of entropy. To check if your CPU supports the RDRAND processor instruction, enter the following command:

~$ cat /proc/cpuinfo | grep rdrand

NOTE

For more information and software code examples, see Intel Digital Random Number Generator (DRNG) Software Implementation Guide.

The rng-tools package also contains the `rngtest` utility, which can be used to check the randomness of data. To test the level of randomness of the output of `/dev/random`, use the `rngtest` tool as follows:

~$ cat /dev/random | rngtest -c 1000
rngtest 5
Copyright (c) 2004 by Henrique de Moraes Holschuh
This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

rngtest: starting FIPS tests...
rngtest: bits received from input: 20000032
rngtest: FIPS 140-2 successes: 998
rngtest: FIPS 140-2 failures: 2
rngtest: FIPS 140-2(2001-10-10) Monobit: 0
rngtest: FIPS 140-2(2001-10-10) Poker: 0
rngtest: FIPS 140-2(2001-10-10) Runs: 0
rngtest: FIPS 140-2(2001-10-10) Long run: 2
rngtest: FIPS 140-2(2001-10-10) Continuous run: 0
rngtest: input channel speed: (min=1.171; avg=8.453; max=11.374)Mibits/s
rngtest: FIPS tests speed: (min=15.545; avg=143.126; max=157.632)Mibits/s
rngtest: Program run time: 2390520 microseconds

A high number of failures shown in the output of the `rngtest` tool indicates that the randomness of the tested data is insufficient and should not be relied upon. See the `rngtest(1)` manual page for a list of options available for the `rngtest` utility.
Red Hat Enterprise Linux 7 introduced the **virtio RNG** (Random Number Generator) device that provides **KVM** virtual machines with access to entropy from the host machine. With the recommended setup, hwrng feeds into the entropy pool of the host Linux kernel (through `/dev/random`), and **QEMU** will use `/dev/random` as the source for entropy requested by guests.

![Diagram of virtio RNG device](image)

**Figure 4.1. The virtio RNG device**

Previously, Red Hat Enterprise Linux 7.0 and Red Hat Enterprise Linux 6 guests could make use of the entropy from hosts through the rngd user space daemon. Setting up the daemon was a manual step for each Red Hat Enterprise Linux installation. With Red Hat Enterprise Linux 7.1, the manual step has been eliminated, making the entire process seamless and automatic. The use of rngd is now not required and the guest kernel itself fetches entropy from the host when the available entropy falls below a specific threshold. The guest kernel is then in a position to make random numbers available to applications as soon as they request them.

The Red Hat Enterprise Linux installer, **Anaconda**, now provides the **virtio-rng** module in its installer image, making available host entropy during the Red Hat Enterprise Linux installation.

### 4.10. USING NETWORK-BOUND DISK ENCRYPTION
The Network-Bound Disk Encryption (NBDE) allows the user to encrypt root volumes of hard drives on physical and virtual machines without requiring to manually enter a password when systems are restarted.

In Red Hat Enterprise Linux 7, NBDE is implemented through the following components and technologies:

![Diagram of NBDE components]

**Figure 4.2. The Network-Bound Disk Encryption using Clevis and Tang**

Tang is a server for binding data to network presence. It makes a system containing your data available when the system is bound to a certain secure network. Tang is stateless and does not require TLS or authentication. Unlike escrow-based solutions, where the server stores all encryption keys and has knowledge of every key ever used, Tang never sees a single client key. Tang never gains any identifying information from the client.

The Clevis is a pluggable framework for automated decryption. In NBDE, Clevis provides automated unlocking of LUKS volumes. The clevis package provides the client side of the feature.

A Clevis Pin is a plug-in into the Clevis framework. One of such PINs is a plug-in that implements interactions with the NBDE server — Tang.

Clevis and Tang are generic client and server components to provide network-bound encryption. In Red Hat Enterprise Linux 7, they are used in conjunction to encrypt and decrypt root volumes of hard drives to accomplish the Network-Bound Disk Encryption.

Both client- and server-side components use the José library to perform encryption and decryption operations.

The Clevis Pin for Tang server gets a list of the Tang server's advertised asymmetric keys. Alternatively, since the keys are asymmetric, a list of Tang’s public keys can be distributed out of the band so that clients can operate without access to the Tang server. This mode is called offline provisioning.

The Clevis Pin for Tang uses one of the public keys to generate a unique, cryptographically-strong encryption key. Once the data are encrypted using this key, the key is discarded. The client should store the state produced by this provisioning operation in a convenient location. This process of encrypting data is the provisioning step. The provisioning state for NBDE is stored in the LUKS header leveraging the luksmeta package.
When the client is ready to access its data, it loads the metadata produced in the provisioning step and it responds to recover the encryption key. This process is the recovery step.

In NBDE, Clevis binds a LUKS volume using a PIN so that it can be automatically unlocked. After successful completion of the binding process, the disk can be unlocked using the provided Dracut unlocker.

4.10.1. Deploying a Tang server

To install the tang package and its dependencies, enter the following command as root:

```bash
~# yum install tang
```

Enable and start the tangd service using systemd:

```bash
~# systemctl enable tangd.socket --now
Created symlink from /etc/systemd/system/multi-user.target.wants/tangd.socket to /usr/lib/systemd/system/tangd.socket.
```

Since tangd uses the systemd socket activation mechanism, the server starts as soon as the first connection comes in. A new set of cryptographic keys is automatically generated at the first start.

To perform cryptographic operations such as manual key generation, use the jose utility. Enter the jose -h command or see the jose(1) man pages for more information.

Example 4.3. Rotating Tang Keys

It is important to periodically rotate your keys. The precise interval at which you should rotate them depends upon your application, key sizes, and institutional policy. For some common recommendations, see the Cryptographic Key Length Recommendation page.

To rotate keys, start with the generation of new keys in the key database directory, typically /var/db/tang. For example, you can create new signature and exchange keys with the following commands:

```bash
~# DB=/var/db/tang
~# jose jwk gen -i '{"alg":"ES512"}' -o $DB/new_sig.jwk
~# jose jwk gen -i '{"alg":"ECMR"}' -o $DB/new_exc.jwk
```

Rename the old keys to have a leading . to hide them from advertisement:

```bash
~# mv $DB/old_sig.jwk $DB/.old_sig.jwk
~# mv $DB/old_exc.jwk $DB/.old_exc.jwk
```

Tang immediately picks up all changes. No restart is required.

At this point, new client bindings pick up the new keys and old clients can continue to utilize the old keys. When you are sure that all old clients use the new keys, you can remove the old keys.
WARNING

Be aware that removing the old keys while clients are still using them can result in data loss.

Tang uses port 80 for communication. This port is also widely-used for web servers. To change Tang’s port number, override the `tangd.socket` unit file using the standard `systemd` mechanisms. See Red Hat Enterprise Linux 7 System Administrator’s Guide: Creating and Modifying systemd Unit Files for more information.

4.10.1.1. Deploying High-Availability Systems

Tang provides two methods for building a high-availability deployment:

1. **Client Redundancy (Recommended)**

   Clients should be configured with the ability to bind to multiple Tang servers. In this setup, each Tang server has its own keys and clients are able to decrypt by contacting a subset of these servers. Clevis already supports this workflow through its `sss` plug-in.

   For more information about this setup, see the following man pages:

   - `tang(8)`, section High Availability
   - `clevis(1)`, section Shamir’s Secret Sharing
   - `clevis-encrypt-sss(1)`

   Red Hat recommends this method for a high-availability deployment.

2. **Key Sharing**

   For redundancy purposes, more than one instance of Tang can be deployed. To set up a second or any subsequent instance, install the `tang` packages and copy the key directory to the new host using `rsync` over SSH. Note that Red Hat does not recommend this method because sharing keys increases the risk of key compromise and requires additional automation infrastructure.

4.10.2. Deploying an Encryption Client

To install the Clevis pluggable framework and its PINs on the machine with an encrypted volume (client), enter the following command as `root`:

```bash
~]# yum install clevis
```

To encrypt plaintext data into a JSON Web Encryption (JWE) ciphertext object using a Tang PIN, use the `clevis encrypt tang` command, for example:

```bash
~]$ clevis encrypt tang '{"url":"http://tang.srv"}' < PLAINTEXT > JWE
```

The advertisement contains the following signing keys:
Do you wish to trust these keys? [ynYN] y

Change the http://tang.srv URL in the previous example to match the URL of the server where tang is installed. The JWE output file contains your encrypted cipher text. This cipher text is read from the PLAINTEXT input file.

To decrypt data, use the **clevis decrypt** command and provide the cipher text (JWE):

```
~$ clevis decrypt < JWE > PLAINTEXT
```

For more information, see the **clevis-encrypt-tang(1)** man page or use the built-in CLI help:

```
$ clevis
Usage: clevis COMMAND [OPTIONS]

  clevis bind luks    Binds a LUKSv1 device using the specified policy
  clevis decrypt     Decrypts using the policy defined at encryption time
  clevis encrypt http Encrypts using a REST HTTP escrow server policy
  clevis encrypt sss  Encrypts using a Shamir's Secret Sharing policy
  clevis encrypt tang Encrypts using a Tang binding server policy

~$ clevis decrypt
Usage: clevis decrypt < JWE > PLAINTEXT

Decrypts using the policy defined at encryption time

~$ clevis encrypt tang
Usage: clevis encrypt tang CONFIG < PLAINTEXT > JWE

Encrypts using a Tang binding server policy

This command uses the following configuration properties:

- **url**: <string>   The base URL of the Tang server (REQUIRED)
- **thp**: <string>   The thumbprint of a trusted signing key
- **adv**: <string>   A filename containing a trusted advertisement
- **adv**: <object>   A trusted advertisement (raw JSON)

Obtaining the thumbprint of a trusted signing key is easy. If you have access to the Tang server's database directory, simply do:

```
$ jose jwk thp -i $DBDIR/$SIG.jwk
```

Alternatively, if you have certainty that your network connection is not compromised (not likely), you can download the advertisement yourself using:

```
$ curl -f $URL/adv > adv.jws
```
4.10.3. Configuring Manual Enrollment

To bind an existing LUKS volume to its automation policy, install the clevis-luks subpackage:

```bash
~# yum install clevis-luks
```

Then use the `clevis bind luks` command, for example:

```bash
~# clevis bind luks -d /dev/sda tang '{"url":"http://tang.srv"}'
```

The advertisement contains the following signing keys:

160571020-E216qjfdDiwVmidOZjA

Do you wish to trust these keys? [ynYN] y

You are about to initialize a LUKS device for metadata storage. Attempting to initialize it may result in data loss if data was already written into the LUKS header gap in a different format. A backup is advised before initialization is performed.

Do you wish to initialize /dev/sda? [yn] y

Enter existing LUKS password:

This command performs four steps:

1. Creates a new key with the same entropy as the LUKS master key.
2. Encrypts the new key with Clevis.
3. Stores the Clevis JWE object in the LUKS header with LUKSMeta.
4. Enables the new key for use with LUKS.

This disk can now be unlocked with your existing password as well as with the Clevis policy. For more information, see the `clevis-bind-luks(1)` man page.

**NOTE**

The binding procedure assumes that there is at least one free LUKS password slot. The `clevis bind luks` command takes one of the slots.

To verify that the Clevis JWE object is successfully placed in a LUKS header, use the `luksmeta show` command:

```bash
~# luksmeta show -d /dev/sda
0   active empty
1   active cb6e8904-81ff-40da-a84a-07ab9ab5715e
2   inactive empty
3   inactive empty
4   inactive empty
5   inactive empty
6   inactive empty
7   inactive empty
```

CHAPTER 4. HARDENING YOUR SYSTEM WITH TOOLS AND SERVICES
To enable the early boot system to process the disk binding, enter the following commands on an already-installed system:

```bash
~# yum install clevis-dracut
~# dracut -f
```

4.10.4. Configuring Automated Enrollment Using Kickstart

Clevis can integrate with Kickstart to provide a fully automated enrollment process.

1. Instruct Kickstart to partition the disk such that the root partition has enabled LUKS encryption with a temporary password. The password is temporary for the enrollment process.

   ```bash
   part /boot --fstype="xfs" --ondisk=vda --size=256
   part / --fstype="xfs" --ondisk=vda --grow --encrypted --passphrase=temppass
   ```

2. Install the related Clevis packages by listing them in the `%packages` section:

   ```bash
   %packages
   clevis-dracut
   %end
   ```

3. Call `clevis bind luks` to perform binding in the `%post` section. Afterward, remove the temporary password:

   ```bash
   %post
   clevis bind luks -f -k- -d /dev/vda2 \
   tang '{"url":"http://tang.srv","thp":"_OsIkOT-E216qjfdDiwVmidoZjA"}' \
   <<< "temppass"
   cryptsetup luksRemoveKey /dev/vda2 - <<< "temppass"
   %end
   ```

   In the above example, note that we specify the thumbprint that we trust on the Tang server as part of our binding configuration, enabling binding to be completely non-interactive.

For more information on Kickstart installations, see the Red Hat Enterprise Linux 7 Installation Guide. For information on Linux Unified Key Setup-on-disk-format (LUKS), see Section 4.9.1, “Using LUKS Disk Encryption”.

4.10.5. Deploying Virtual Machines in a NBDE Network

The `clevis bind luks` command does not change the LUKS master key. This implies that if you create a LUKS-encrypted image for use in a virtual machine or cloud environment, all the instances that run this image will share a master key. This is extremely insecure and should be avoided at all times.

This is not a limitation of Clevis but a design principle of LUKS. If you wish to have encrypted root volumes in a cloud, you need to make sure that you perform the installation process (usually using Kickstart) for each instance of Red Hat Enterprise Linux in a cloud as well. The images cannot be shared without also sharing a LUKS master key.
If you intend to deploy automated unlocking in a virtualized environment, Red Hat strongly recommends that you use Kickstart (see Section 4.10.4, “Configuring Automated Enrollment Using Kickstart”) or another automated provisioning tool to ensure that each encrypted VM has a unique master key.

### 4.10.6. Building Automatically-enrollable VM Images for Cloud Environments

Deploying automated decryption images in a cloud environment can provide a unique set of challenges. Like other virtualization environments detailed above, images should be instantiated at most once to avoid sharing the LUKS master key. Therefore, automated deployment systems such as Kickstart should be used to ensure master key uniqueness during the image building process.

Cloud environments enable two Tang server deployment options which we consider here. First, the Tang server can be deployed within the cloud environment itself. Second, the Tang server can be deployed outside of the cloud on independent infrastructure with a VPN link between the two infrastructures.

Deploying Tang natively in the cloud does allow for easy deployment. However, given that it shares infrastructure with the data persistence layer of ciphertext of other systems, it may be possible for both the Tang server’s private key and the Clevis metadata to be stored on the same physical disk. Access to this physical disk permits a full compromise of the ciphertext data.

**IMPORTANT**

For this reason, Red Hat strongly recommends that the Tang server is deployed off premises with a VPN link between the cloud and the Tang server. This ensures that the Tang server’s private key cannot be accidentally combined with the Clevis metadata. It also provides local control of the Tang server if the cloud infrastructure is in risk.

### 4.10.7. Additional Resources

For more information, see the following man pages:

- `tang(8)`
- `clevis(1)`
- `jose(1)`

### 4.11. CHECKING INTEGRITY WITH AIDE

Advanced Intrusion Detection Environment (AIDE) is a utility that creates a database of files on the system, and then uses that database to ensure file integrity and detect system intrusions.

#### 4.11.1. Installing AIDE

To install the aide package, enter the following command as root:

```
~]# yum install aide
```
To generate an initial database, enter the following command as root:

```bash
~$ aide --init
AIDE, version 0.15.1
### AIDE database at /var/lib/aide/aide.db.new.gz initialized.
```

**NOTE**

In the default configuration, the `aide --init` command checks just a set of directories and files defined in the `/etc/aide.conf` file. To include additional directories or files in the AIDE database, and to change their watched parameters, edit `/etc/aide.conf` accordingly.

To start using the database, remove the `.new` substring from the initial database file name:

```bash
~$ mv /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz
```

To change the location of the AIDE database, edit the `/etc/aide.conf` file and modify the `DBDIR` value. For additional security, store the database, configuration, and the `/usr/sbin/aide` binary file in a secure location such as a read-only media.

**IMPORTANT**

To avoid SELinux denials after the AIDE database location change, update your SELinux policy accordingly. See the SELinux User's and Administrator's Guide for more information.

### 4.11.2. Performing Integrity Checks

To initiate a manual check, enter the following command as root:

```bash
~$ aide --check
AIDE 0.15.1 found differences between database and filesystem!!
Start timestamp: 2017-03-30 14:12:56

Summary:
  Total number of files: 147173
  Added files: 1
  Removed files: 0
  Changed files: 2
...
```

At a minimum, AIDE should be configured to run a weekly scan. At most AIDE should be run daily. For example, to schedule a daily execution of AIDE at 4:05 am using cron (see the Automating System Tasks chapter in the System Administrator’s Guide), add the following line to `/etc/crontab`:

```
05 4 * * * root /usr/sbin/aide --check
```

### 4.11.3. Updating an AIDE Database
After the changes of your system such as package updates or configuration files adjustments are verified, update your baseline AIDE database:

```bash
~]# aide --update
```

The `aide --update` command creates the `/var/lib/aide/aide.db.new.gz` database file. To start using it for integrity checks, remove the `.new` substring from the file name.

### 4.11.4. Additional Resources

For additional information on AIDE, see the following documentation:

- `aide(1)` man page
- `aide.conf(5)` man page
- The AIDE manual

### 4.12. USING USBGUARD

The USBGuard software framework provides system protection against intrusive USB devices by implementing basic whitelisting and blacklisting capabilities based on device attributes. To enforce a user-defined policy, USBGuard uses the Linux kernel USB device authorization feature. The USBGuard framework provides the following components:

- The daemon component with an inter-process communication (IPC) interface for dynamic interaction and policy enforcement.
- The command-line interface to interact with a running USBGuard instance.
- The rule language for writing USB device authorization policies.
- The C++ API for interacting with the daemon component implemented in a shared library.

#### 4.12.1. Installing USBGuard

To install the usbguard package, enter the following command as `root`:

```bash
~]# yum install usbguard
```

To create the initial rule set, enter the following command as `root`:

```bash
~]# usbguard generate-policy > /etc/usbguard/rules.conf
```
NOTE

To customize the **USBGuard** rule set, edit the `/etc/usbguard/rules.conf` file. See the `usbguard-rules.conf(5)` man page for more information. Additionally, see Section 4.12.3, “Using the Rule Language to Create Your Own Policy” for examples.

To start the **USBGuard** daemon, enter the following command as `root`:

```
~# systemctl start usbguard.service
```

To ensure **USBGuard** starts automatically at system start, use the following command as `root`:

```
~# systemctl enable usbguard.service
```

To list all USB devices recognized by **USBGuard**, enter the following command as `root`:

```
~# usbguard list-devices
```

To authorize a device to interact with the system, use the **allow-device** option:

```
~# usbguard allow-device 6
```

To deauthorize and remove a device from the system, use the **reject-device** option. To just deauthorize a device, use the `usbguard` command with the **block-device** option:

```
~# usbguard block-device 6
```

**USBGuard** uses the `block` and `reject` terms with the following meaning:

- block - do not talk to this device for now
• reject - ignore this device as if did not exist

To see all options of the `usbguard` command, enter it with the `--help` directive:

```
~]$ usbguard --help
```

### 4.12.2. Creating a White List and a Black List

The `usbguard-daemon.conf` file is loaded by the `usbguard` daemon after it parses its command-line options and is used to configure runtime parameters of the daemon. To override the default configuration file (`/etc/usbguard/usbguard-daemon.conf`), use the `-c` command-line option. See the `usbguard-daemon(8)` man page for further details.

To create a white list or a black list, edit the `usbguard-daemon.conf` file and use the following options:

**USBGuard configuration file**

**RuleFile=<path>**

The `usbguard` daemon use this file to load the policy rule set from it and to write new rules received through the IPC interface.

**IPCAllowedUsers=<username> [<username> ...]**

A space-delimited list of user names that the daemon will accept IPC connections from.

**IPCAllowedGroups=<groupname> [<groupname> ...]**

A space-delimited list of group names that the daemon will accept IPC connections from.

**IPCAccessControlFiles=<path>**

Path to a directory holding the IPC access control files.

**ImplicitPolicyTarget=<target>**

How to treat devices that do not match any rule in the policy. Accepted values: allow, block, reject.

**PresentDevicePolicy=<policy>**

How to treat devices that are already connected when the daemon starts:

- allow - authorize every present device
- block - deauthorize every present device
- reject - remove every present device
- keep - just sync the internal state and leave it
- apply-policy - evaluate the ruleset for every present device

**PresentControllerPolicy=<policy>**

How to treat USB controllers that are already connected when the daemon starts:

- allow - authorize every present device
- block - deauthorize every present device
- reject - remove every present device
- keep - just sync the internal state and leave it
- apply-policy - evaluate the ruleset for every present device

Example 4.4. USBGuard configuration

The following configuration file orders the `usbguard` daemon to load rules from the `/etc/usbguard/rules.conf` file and it allows only users from the `usbguard` group to use the IPC interface:

```bash
RuleFile=/etc/usbguard/rules.conf
IPCAccessControlFiles=/etc/usbguard/IPCAccessControl.d/
```

To specify the IPC Access Control List (ACL), use the `usbguard add-user` or `usbguard remove-user` commands. See the `usbguard(1)` for more details. In this example, to allow users from the `usbguard` group to modify USB device authorization state, list USB devices, listen to exception events, and list USB authorization policy, enter the following command as `root`:

```
~$]# usbguard add-user -g usbguard --devices=modify,list,listen --policy=list --exceptions=listen
```

IMPORTANT

The daemon provides the `USBGuard` public IPC interface. In Red Hat Enterprise Linux, the access to this interface is by default limited to the `root` user only. Consider setting either the the `IPCAccessControlFiles` option (recommended) or the `IPCAccessAllowedUsers` and `IPCAccessAllowedGroups` options to limit access to the IPC interface. Do not leave the ACL unconfigured as this exposes the IPC interface to all local users and it allows them to manipulate the authorization state of USB devices and modify the `USBGuard` policy.

For more information, see the IPC Access Control section in the `usbguard-daemon.conf(5)` man page.

4.12.3. Using the Rule Language to Create Your Own Policy

The `usbguard` daemon decides whether to authorize a USB device based on a policy defined by a set of rules. When a USB device is inserted into the system, the daemon scans the existing rules sequentially and when a matching rule is found, it either authorizes (allows), deauthorizes (blocks) or removes (rejects) the device, based on the rule target. If no matching rule is found, the decision is based on an implicit default target. This implicit default is to block the device until a decision is made by the user.

The rule language grammar is the following:

```
rule ::= target device_id device_attributes conditions.
```
target ::= "allow" | "block" | "reject".
device_id ::= "*:*" | vendor_id ":*" | vendor_id ":" product_id.
device_attributes ::= device_attributes | attribute.
device_attributes ::= .
conditions ::= conditions | condition.
conditions ::= .

For more details about the rule language such as targets, device specification, or device attributes, see the `usbguard-rules.conf(5)` man page.

### Example 4.5. USBguard example policies

#### Allow USB mass storage devices and block everything else

This policy blocks any device that is not just a mass storage device. Devices with a hidden keyboard interface in a USB flash disk are blocked. Only devices with a single mass storage interface are allowed to interact with the operating system. The policy consists of a single rule:

```
allow with-interface equals { 08:*:* }
```

The blocking is implicit because there is no block rule. Implicit blocking is useful to desktop users because a desktop applet listening to USBGuard events can ask the user for a decision if an implicit target was selected for a device.

#### Allow a specific Yubikey device to be connected through a specific port

Reject everything else on that port.

```
allow 1050:0011 name "Yubico Yubikey II" serial "0001234567" via-port "1-2" hash "044b5e168d40ee0245478416caf3d998"
reject via-port "1-2"
```

#### Reject devices with suspicious combination of interfaces

A USB flash disk which implements a keyboard or a network interface is very suspicious. The following set of rules forms a policy which allows USB flash disks and explicitly rejects devices with an additional and suspicious interface.

```
allow with-interface equals { 08:*:* }
reject with-interface all-of { 08:*:* 03:00:* }
reject with-interface all-of { 08:*:* 03:01:* }
reject with-interface all-of { 08:*:* e0:*:* }
reject with-interface all-of { 08:*:* 02:*:* }
```

### NOTE

Blacklisting is the wrong approach and you should not just blacklist a set of devices and allow the rest. The policy above assumes that blocking is the implicit default. Rejecting a set of devices considered as "bad" is a good approach how to limit the exposure of the system to such devices as much as possible.
Allow a keyboard-only USB device

The following rule allows a keyboard-only USB device only if there is not a USB device with a keyboard interface already allowed.

```
allow with-interface one-of { 03:00:01 03:01:01 } if !allowed-matches(with-interface one-of { 03:00:01 03:01:01 })
```

After an initial policy generation using the `usbguard generate-policy` command, edit the `/etc/usbguard/rules.conf` to customize the USBGuard policy rules.

```
~$ usbguard generate-policy > rules.conf
~$ vim rules.conf
```

To install the updated policy and make your changes effective, use the following commands:

```
~# install -m 0600 -o root -g root rules.conf /etc/usbguard/rules.conf
```

### 4.12.4. Additional Resources

For additional information on USBGuard, see the following documentation:

- `usbguard(1)` man page
- `usbguard-rules.conf(5)` man page
- `usbguard-daemon(8)` man page
- `usbguard-daemon.conf(5)` man page
- The USBGuard homepage

### 4.13. HARDENING TLS CONFIGURATION

**TLS (Transport Layer Security)** is a cryptographic protocol used to secure network communications. When hardening system security settings by configuring preferred key-exchange protocols, authentication methods, and encryption algorithms, it is necessary to bear in mind that the broader the range of supported clients, the lower the resulting security. Conversely, strict security settings lead to a limited compatibility with clients, which can result in some users being locked out of the system. Be sure to target the strictest available configuration and only relax it when it is required for compatibility reasons.

Note that the default settings provided by libraries included in Red Hat Enterprise Linux 7 are secure enough for most deployments. The TLS implementations use secure algorithms where possible while not preventing connections from or to legacy clients or servers. Apply the hardened settings described in this section in environments with strict security requirements where legacy clients or servers that do not support secure algorithms or protocols are not expected or allowed to connect.

#### 4.13.1. Choosing Algorithms to Enable
There are several components that need to be selected and configured. Each of the following directly influences the robustness of the resulting configuration (and, consequently, the level of support in clients) or the computational demands that the solution has on the system.

**Protocol Versions**
The latest version of **TLS** provides the best security mechanism. Unless you have a compelling reason to include support for older versions of **TLS** (or even **SSL**), allow your systems to negotiate connections using only the latest version of **TLS**.

Do not allow negotiation using **SSL** version 2 or 3. Both of those versions have serious security vulnerabilities. Only allow negotiation using **TLS** version 1.0 or higher. The current version of **TLS**, 1.2, should always be preferred.

**NOTE**
Please note that currently, the security of all versions of **TLS** depends on the use of **TLS** extensions, specific ciphers (see below), and other workarounds. All **TLS** connection peers need to implement secure renegotiation indication (**RFC 5746**), must not support compression, and must implement mitigating measures for timing attacks against **CBC**-mode ciphers (the Lucky Thirteen attack). **TLS 1.0** clients need to additionally implement record splitting (a workaround against the BEAST attack). **TLS 1.2** supports **Authenticated Encryption with Associated Data** (**AEAD**) mode ciphers like **AES-GCM**, **AES-CCM**, or **Camellia-GCM**, which have no known issues. All the mentioned mitigations are implemented in cryptographic libraries included in Red Hat Enterprise Linux.

See Table 4.6, “Protocol Versions” for a quick overview of protocol versions and recommended usage.

**Table 4.6. Protocol Versions**

<table>
<thead>
<tr>
<th>Protocol Version</th>
<th>Usage Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SSL v2</strong></td>
<td>Do not use. Has serious security vulnerabilities.</td>
</tr>
<tr>
<td><strong>SSL v3</strong></td>
<td>Do not use. Has serious security vulnerabilities.</td>
</tr>
<tr>
<td><strong>TLS 1.0</strong></td>
<td>Use for interoperability purposes where needed. Has known issues that cannot be mitigated in a way that guarantees interoperability, and thus mitigations are not enabled by default. Does not support modern cipher suites.</td>
</tr>
<tr>
<td><strong>TLS 1.1</strong></td>
<td>Use for interoperability purposes where needed. Has no known issues but relies on protocol fixes that are included in all the <strong>TLS</strong> implementations in Red Hat Enterprise Linux. Does not support modern cipher suites.</td>
</tr>
<tr>
<td><strong>TLS 1.2</strong></td>
<td>Recommended version. Supports the modern <strong>AEAD</strong> cipher suites.</td>
</tr>
</tbody>
</table>

Some components in Red Hat Enterprise Linux are configured to use **TLS 1.0** even though they provide support for **TLS 1.1** or even **1.2**. This is motivated by an attempt to achieve the highest level of interoperability with external services that may not support the latest TLS version.
versions of TLS. Depending on your interoperability requirements, enable the highest available version of TLS.

**IMPORTANT**

SSL v3 is not recommended for use. However, if, despite the fact that it is considered insecure and unsuitable for general use, you absolutely must leave SSL v3 enabled, see Section 4.8, “Using stunnel” for instructions on how to use stunnel to securely encrypt communications even when using services that do not support encryption or are only capable of using obsolete and insecure modes of encryption.

**Cipher Suites**

Modern, more secure cipher suites should be preferred to old, insecure ones. Always disable the use of eNULL and aNULL cipher suites, which do not offer any encryption or authentication at all. If at all possible, cipher suites based on RC4 or HMAC-MD5, which have serious shortcomings, should also be disabled. The same applies to the so-called export cipher suites, which have been intentionally made weaker, and thus are easy to break.

While not immediately insecure, cipher suites that offer less than 128 bits of security should not be considered for their short useful life. Algorithms that use 128 bit of security or more can be expected to be unbreakable for at least several years, and are thus strongly recommended. Note that while 3DES ciphers advertise the use of 168 bits, they actually offer 112 bits of security.

Always give preference to cipher suites that support (perfect) forward secrecy (PFS), which ensures the confidentiality of encrypted data even in case the server key is compromised. This rules out the fast RSA key exchange, but allows for the use of ECDHE and DHE. Of the two, ECDHE is the faster and therefore the preferred choice.

You should also give preference to AEAD ciphers, such as AES-GCM, before CBC-mode ciphers as they are not vulnerable to padding oracle attacks. Additionally, in many cases, AES-GCM is faster than AES in CBC mode, especially when the hardware has cryptographic accelerators for AES.

Note also that when using the ECDHE key exchange with ECDSA certificates, the transaction is even faster than pure RSA key exchange. To provide support for legacy clients, you can install two pairs of certificates and keys on a server: one with ECDSA keys (for new clients) and one with RSA keys (for legacy ones).

**Public Key Length**

When using RSA keys, always prefer key lengths of at least 3072 bits signed by at least SHA-256, which is sufficiently large for true 128 bits of security.
WARNING

Keep in mind that the security of your system is only as strong as the weakest link in the chain. For example, a strong cipher alone does not guarantee good security. The keys and the certificates are just as important, as well as the hash functions and keys used by the Certification Authority (CA) to sign your keys.

4.13.2. Using Implementations of TLS

Red Hat Enterprise Linux 7 is distributed with several full-featured implementations of TLS. In this section, the configuration of OpenSSL and GnuTLS is described. See Section 4.13.3, “Configuring Specific Applications” for instructions on how to configure TLS support in individual applications.

The available TLS implementations offer support for various cipher suites that define all the elements that come together when establishing and using TLS-secured communications.

Use the tools included with the different implementations to list and specify cipher suites that provide the best possible security for your use case while considering the recommendations outlined in Section 4.13.1, “Choosing Algorithms to Enable”. The resulting cipher suites can then be used to configure the way individual applications negotiate and secure connections.

IMPORTANT

Be sure to check your settings following every update or upgrade of the TLS implementation you use or the applications that utilize that implementation. New versions may introduce new cipher suites that you do not want to have enabled and that your current configuration does not disable.

4.13.2.1. Working with Cipher Suites in OpenSSL

OpenSSL is a toolkit and a cryptography library that support the SSL and TLS protocols. On Red Hat Enterprise Linux 7, a configuration file is provided at /etc/pki/tls/openssl.cnf. The format of this configuration file is described in config(1). See also Section 4.7.9, “Configuring OpenSSL”.

To get a list of all cipher suites supported by your installation of OpenSSL, use the openssl command with the ciphers subcommand as follows:

```bash
~]$ openssl ciphers -v 'ALL:COMPLEMENTOFALL'
```

Pass other parameters (referred to as cipher strings and keywords in OpenSSL documentation) to the ciphers subcommand to narrow the output. Special keywords can be used to only list suites that satisfy a certain condition. For example, to only list suites that are defined as belonging to the HIGH group, use the following command:

```bash
~]$ openssl ciphers -v 'HIGH'
```
See the ciphers(1) manual page for a list of available keywords and cipher strings.

To obtain a list of cipher suites that satisfy the recommendations outlined in Section 4.13.1, “Choosing Algorithms to Enable”, use a command similar to the following:

```bash
~]$ openssl ciphers -v 'kEECDH+aECDSA+AES:kEECDH+AES+aRSA:kEDH+aRSA+AES' | column -t
```

<table>
<thead>
<tr>
<th>Cipher Suite</th>
<th>Protocol</th>
<th>Key Exchange</th>
<th>Authentication</th>
<th>Encryption</th>
<th>MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECDHE-EDSA-AES256-GCM-SHA384</td>
<td>TLSv1.2</td>
<td>Kx=ECDH</td>
<td>Au=ECDSA</td>
<td>Enc=AESGCM(256)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-EDSA-AES256-SHA384</td>
<td>TLSv1.2</td>
<td>Kx=ECDH</td>
<td>Au=ECDSA</td>
<td>Enc=AES(256)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-EDSA-AES256-SHA</td>
<td>SSLv3</td>
<td>Kx=ECDH</td>
<td>Au=ECDSA</td>
<td>Enc=AES(256)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-EDSA-AES128-GCM-SHA256</td>
<td>TLSv1.2</td>
<td>Kx=ECDH</td>
<td>Au=ECDSA</td>
<td>Enc=AESGCM(128)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-EDSA-AES128-SHA256</td>
<td>TLSv1.2</td>
<td>Kx=ECDH</td>
<td>Au=ECDSA</td>
<td>Enc=AES(128)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-EDSA-AES128-SHA</td>
<td>SSLv3</td>
<td>Kx=ECDH</td>
<td>Au=ECDSA</td>
<td>Enc=AES(128)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-RSA-AES256-GCM-SHA384</td>
<td>TLSv1.2</td>
<td>Kx=ECDH</td>
<td>Au=RSA</td>
<td>Enc=AESGCM(256)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-RSA-AES256-SHA384</td>
<td>TLSv1.2</td>
<td>Kx=ECDH</td>
<td>Au=RSA</td>
<td>Enc=AES(256)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-RSA-AES256-SHA</td>
<td>SSLv3</td>
<td>Kx=ECDH</td>
<td>Au=RSA</td>
<td>Enc=AES(256)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-RSA-AES128-GCM-SHA256</td>
<td>TLSv1.2</td>
<td>Kx=ECDH</td>
<td>Au=RSA</td>
<td>Enc=AESGCM(128)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-RSA-AES128-SHA256</td>
<td>TLSv1.2</td>
<td>Kx=ECDH</td>
<td>Au=RSA</td>
<td>Enc=AES(128)</td>
<td></td>
</tr>
<tr>
<td>ECDHE-RSA-AES128-SHA</td>
<td>SSLv3</td>
<td>Kx=ECDH</td>
<td>Au=RSA</td>
<td>Enc=AES(128)</td>
<td></td>
</tr>
<tr>
<td>DHE-RSA-AES256-GCM-SHA384</td>
<td>TLSv1.2</td>
<td>Kx=DH</td>
<td>Au=RSA</td>
<td>Enc=AESGCM(256)</td>
<td></td>
</tr>
<tr>
<td>DHE-RSA-AES256-SHA256</td>
<td>TLSv1.2</td>
<td>Kx=DH</td>
<td>Au=RSA</td>
<td>Enc=AES(256)</td>
<td></td>
</tr>
<tr>
<td>DHE-RSA-AES256-SHA</td>
<td>SSLv3</td>
<td>Kx=DH</td>
<td>Au=RSA</td>
<td>Enc=AES(256)</td>
<td></td>
</tr>
<tr>
<td>DHE-RSA-AES128-GCM-SHA256</td>
<td>TLSv1.2</td>
<td>Kx=DH</td>
<td>Au=RSA</td>
<td>Enc=AESGCM(128)</td>
<td></td>
</tr>
<tr>
<td>DHE-RSA-AES128-SHA256</td>
<td>TLSv1.2</td>
<td>Kx=DH</td>
<td>Au=RSA</td>
<td>Enc=AES(128)</td>
<td></td>
</tr>
<tr>
<td>DHE-RSA-AES128-SHA</td>
<td>SSLv3</td>
<td>Kx=DH</td>
<td>Au=RSA</td>
<td>Enc=AES(128)</td>
<td></td>
</tr>
</tbody>
</table>

The above command omits all insecure ciphers, gives preference to **ephemeral elliptic curve Diffie-Hellman** key exchange and **ECDSA** ciphers, and omits **RSA** key exchange (thus ensuring **perfect forward secrecy**).

Note that this is a rather strict configuration, and it might be necessary to relax the conditions in real-world scenarios to allow for a compatibility with a broader range of clients.

### 4.13.2.2. Working with Cipher Suites in GnuTLS

**GnuTLS** is a communications library that implements the **SSL** and **TLS** protocols and related technologies.
NOTE
The GnuTLS installation on Red Hat Enterprise Linux 7 offers optimal default configuration values that provide sufficient security for the majority of use cases. Unless you need to satisfy special security requirements, it is recommended to use the supplied defaults.

Use the `gnutls-cli` command with the `-l` (or `--list`) option to list all supported cipher suites:

```
~$ gnutls-cli -l
```

To narrow the list of cipher suites displayed by the `-l` option, pass one or more parameters (referred to as priority strings and keywords in GnuTLS documentation) to the `--priority` option. See the GnuTLS documentation at http://www.gnutls.org/manual/gnutls.html#Priority-Strings for a list of all available priority strings. For example, issue the following command to get a list of cipher suites that offer at least 128 bits of security:

```
~$ gnutls-cli --priority SECURE128 -l
```

To obtain a list of cipher suites that satisfy the recommendations outlined in Section 4.13.1, “Choosing Algorithms to Enable”, use a command similar to the following:

```
```

Cipher suites for SECURE256:+SECURE128:-VERS-TLS-ALL:+VERS-TLS1.2:-RSA:-DHE-DSS:-CAMELLIA-128-CBC:-CAMELLIA-256-CBC

```
TLS_ECDHE_ECDSA_AES_256_GCM_SHA384                      0xc0, 0x2c
TLS1.2
TLS_ECDHE_ECDSA_AES_256_CBC_SHA384                      0xc0, 0x24
TLS1.2
TLS_ECDHE_ECDSA_AES_256_CBC_SHA1                        0xc0, 0x0a
SSL3.0
TLS_ECDHE_ECDSA_AES_128_GCM_SHA256                      0xc0, 0x2b
TLS1.2
TLS_ECDHE_ECDSA_AES_128_CBC_SHA256                      0xc0, 0x23
TLS1.2
TLS_ECDHE_ECDSA_AES_128_CBC_SHA1                        0xc0, 0x09
SSL3.0
TLS_ECDHE_RSA_AES_256_GCM_SHA384                        0xc0, 0x30
TLS1.2
TLS_ECDHE_RSA_AES_256_CBC_SHA1                          0xc0, 0x14
SSL3.0
TLS_ECDHE_RSA_AES_128_GCM_SHA256                        0xc0, 0x2f
TLS1.2
TLS_ECDHE_RSA_AES_128_CBC_SHA256                        0xc0, 0x27
TLS1.2
TLS_ECDHE_RSA_AES_128_CBC_SHA1                          0xc0, 0x13
SSL3.0
TLS_DHE_RSA_AES_256_CBC_SHA256                          0x00, 0x6b
TLS1.2
TLS_DHE_RSA_AES_128_CBC_SHA1                            0x00, 0x39
SSL3.0
TLS_DHE_RSA_AES_128_CBC_SHA256                          0x00, 0x9e
```
The above command limits the output to ciphers with at least 128 bits of security while giving preference to the stronger ones. It also forbids RSA key exchange and DSS authentication.

Note that this is a rather strict configuration, and it might be necessary to relax the conditions in real-world scenarios to allow for a compatibility with a broader range of clients.

### 4.13.3. Configuring Specific Applications

Different applications provide their own configuration mechanisms for TLS. This section describes the TLS-related configuration files employed by the most commonly used server applications and offers examples of typical configurations.

Regardless of the configuration you choose to use, always make sure to mandate that your server application enforces server-side cipher order, so that the cipher suite to be used is determined by the order you configure.

#### 4.13.3.1. Configuring the Apache HTTP Server

The Apache HTTP Server can use both OpenSSL and NSS libraries for its TLS needs. Depending on your choice of the TLS library, you need to install either the mod_ssl or the mod_nss module (provided by eponymous packages). For example, to install the package that provides the OpenSSL mod_ssl module, issue the following command as root:

```bash
~]# yum install mod_ssl
```

The mod_ssl package installs the `/etc/httpd/conf.d/ssl.conf` configuration file, which can be used to modify the TLS-related settings of the Apache HTTP Server. Similarly, the mod_nss package installs the `/etc/httpd/conf.d/nss.conf` configuration file.

Install the httpd-manual package to obtain complete documentation for the Apache HTTP Server, including TLS configuration. The directives available in the `/etc/httpd/conf.d/ssl.conf` configuration file are described in detail in `/usr/share/httpd/manual/mod/mod_ssl.html`. Examples of various settings are in `/usr/share/httpd/manual/ssl/ssl_howto.html`.

When modifying the settings in the `/etc/httpd/conf.d/ssl.conf` configuration file, be sure to consider the following three directives at the minimum:

**SSLProtocol**

Use this directive to specify the version of TLS (or SSL) you want to allow.
SSLCipherSuite
Use this directive to specify your preferred cipher suite or disable the ones you want to disallow.

SSLHonorCipherOrder
Uncomment and set this directive to on to ensure that the connecting clients adhere to the order of ciphers you specified.

For example:

```plaintext
SSLProtocol all -SSLv2 -SSLv3
SSLCipherSuite HIGH:!aNULL:!MD5
SSLHonorCipherOrder on
```

Note that the above configuration is the bare minimum, and it can be hardened significantly by following the recommendations outlined in Section 4.13.1, “Choosing Algorithms to Enable”.

To configure and use the `mod_nss` module, modify the `/etc/httpd/conf.d/nss.conf` configuration file. The `mod_nss` module is derived from `mod_ssl`, and as such it shares many features with it, not least the structure of the configuration file, and the directives that are available. Note that the `mod_nss` directives have a prefix of `NSS` instead of `SSL`. See https://git.fedorahosted.org/cgit/mod_nss.git/plain/docs/mod_nss.html for an overview of information about `mod_nss`, including a list of `mod_ssl` configuration directives that are not applicable to `mod_nss`.

### 4.13.3.2. Configuring the Dovecot Mail Server

To configure your installation of the Dovecot mail server to use TLS, modify the `/etc/dovecot/conf.d/10-ssl.conf` configuration file. You can find an explanation of some of the basic configuration directives available in that file in `/usr/share/doc/dovecot-2.2.10/wiki/SSL.DovecotConfiguration.txt` (this help file is installed along with the standard installation of Dovecot).

When modifying the settings in the `/etc/dovecot/conf.d/10-ssl.conf` configuration file, be sure to consider the following three directives at the minimum:

**ssl_protocols**
Use this directive to specify the version of TLS (or SSL) you want to allow.

**ssl_cipher_list**
Use this directive to specify your preferred cipher suites or disable the ones you want to disallow.

**ssl_prefer_server_ciphers**
Uncomment and set this directive to yes to ensure that the connecting clients adhere to the order of ciphers you specified.

For example:
ssl_protocols = !SSLv2 !SSLv3
ssl_cipher_list = HIGH:aNULL:MD5
ssl_prefer_server_ciphers = yes

Note that the above configuration is the bare minimum, and it can be hardened significantly by following the recommendations outlined in Section 4.13.1, “Choosing Algorithms to Enable”.

4.13.4. Additional Information

For more information about TLS configuration and related topics, see the resources listed below.

Installed Documentation

- config(1) — Describes the format of the /etc/ssl/openssl.conf configuration file.
- ciphers(1) — Includes a list of available OpenSSL keywords and cipher strings.
- /usr/share/httpd/manual/mod/mod_ssl.html — Contains detailed descriptions of the directives available in the /etc/httpd/conf.d/ssl.conf configuration file used by the mod_ssl module for the Apache HTTP Server.
- /usr/share/httpd/manual/ssl/ssl_howto.html — Contains practical examples of real-world settings in the /etc/httpd/conf.d/ssl.conf configuration file used by the mod_ssl module for the Apache HTTP Server.
- /usr/share/doc/dovecot-2.2.10/wiki/SSL.DovecotConfiguration.txt — Explains some of the basic configuration directives available in the /etc/dovecot/conf.d/10-ssl.conf configuration file used by the Dovecot mail server.

Online Documentation

- Red Hat Enterprise Linux 7 SELinux User’s and Administrator’s Guide — The SELinux User’s and Administrator’s Guide for Red Hat Enterprise Linux 7 describes the basic principles of SELinux and documents in detail how to configure and use SELinux with various services, such as the Apache HTTP Server.

See Also

- Section A.2.4, “SSL/TLS” provides a concise description of the SSL and TLS protocols.
- Section 4.7, “Using OpenSSL” describes, among other things, how to use OpenSSL to create and manage keys, generate certificates, and encrypt and decrypt files.

4.14. USING SHARED SYSTEM CERTIFICATES

The Shared System Certificates storage allows NSS, GnuTLS, OpenSSL, and Java to share a default source for retrieving system certificate anchors and black list information. By default, the trust store contains the Mozilla CA list, including positive and negative trust. The system allows updating of the core Mozilla CA list or choosing another certificate list.

As a user I use the same trust store for all applications.

In Red Hat Enterprise Linux 7, the consolidated system-wide trust store is located in the /etc/pki/ca-trust/ and /usr/share/pki/ca-trust-source/ directories. The trust settings in /usr/share/pki/ca-trust-source/ are processed with lower priority than settings in /etc/pki/ca-trust/.

Certificate files are treated depending on the subdirectory they are installed to:

- /usr/share/pki/ca-trust-source/anchors/ or /etc/pki/ca-trust/source/anchors/ - for trust anchors. See Section 4.5.6, “Understanding Trust Anchors”.

- /usr/share/pki/ca-trust-source/blacklist/ or /etc/pki/ca-trust/source/blacklist/ - for distrusted certificates.

- /usr/share/pki/ca-trust-source/ or /etc/pki/ca-trust/source/ - for certificates in the extended BEGIN TRUSTED file format.

4.14.2. Adding New Certificates

As an administrator I am able to add new certificates to the trust store in a consistent way.

To add a certificate in the simple PEM or DER file formats to the list of CAs trusted on the system, copy the certificate file to the /usr/share/pki/ca-trust-source/anchors/ or /etc/pki/ca-trust/source/anchors/ directory. To update system-wide trust store configuration, use the update-ca-trust command, for example:

```
# cp ~/certificate-trust-examples/Cert-trust-test-ca.pem /usr/share/pki/ca-trust-source/anchors/
# update-ca-trust
```

NOTE

While the Firefox browser is able to use an added certificate without executing update-ca-trust, it is recommended to run update-ca-trust after a CA change. Also note that browsers, such as Firefox, Epiphany, or Chromium, cache files, and you might need to clear browser’s cache or restart your browser to load current system certificates configuration.

4.14.3. Managing Trusted System Certificates

As a user/administrator I can list used (dis)trusted system certificates.

To list, extract, add, remove, or change trust anchors, use the trust command. To see the built-in help for this command, enter it without any argument or with the --help directive:

```
$ trust
usage: trust command <args>...
```

Common trust commands are:

- list List trust or certificates
- extract Extract certificates and trust
extract-compat  Extract trust compatibility bundles
anchor          Add, remove, change trust anchors
dump            Dump trust objects in internal format

See 'trust <command> --help' for more information

To list all system trust anchors and certificates, use the trust list command:

```
$ trust list
pkcs11:id=%d2%87%b4%e3%df%37%27%93%55%f6%56%ea%81%e5%36%cc%8c%1e%3f%bd;type=cert
  type: certificate
  label: ACCVRAIZ1
  trust: anchor
  category: authority

pkcs11:id=%a6%b3%e1%2b%2b%49%b6%d7%73%a1%aa%94%f5%01%e7%73%65%4c%ac%50;type=cert
  type: certificate
  label: ACEDICOM Root
  trust: anchor
  category: authority
...
```

All sub-commands of the trust commands offer a detailed built-in help, for example:

```
$ trust list --help
usage: trust list --filter=<what>

--filter=<what> filter of what to export
  ca-anchors certificate anchors
  blacklist blacklisted certificates
  trust-policy anchors and blacklist (default)
  certificates all certificates
  pkcs11:object=xx a PKCS#11 URI

--purpose=<usage> limit to certificates usable for the purpose
  server-auth for authenticating servers
  client-auth for authenticating clients
  email for email protection
  code-signing for authenticating signed code
  1.2.3.4.5... an arbitrary object id

-v, --verbose show verbose debug output
-q, --quiet suppress command output
```

To store a trust anchor into the system-wide trust store, use the trust anchor sub-command and specify a path.to a certificate, for example:

```
# trust anchor path.to/certificate.crt
```

To remove a certificate, use either a path.to a certificate or an ID of a certificate:

```
# trust anchor --remove path.to/certificate.crt
# trust anchor --remove "pkcs11:id=%AA%BB%CC%DD%EE;type=cert"
```
4.14.4. Additional Resources

For more information, see the following man pages:

- `update-ca-trust(8)`
- `trust(1)`

4.15. USING MACSEC

Media Access Control Security (MACsec, IEEE 802.1AE) encrypts and authenticates all traffic in LANs with the GCM-AES-128 algorithm. **MACsec** can protect not only IP but also Address Resolution Protocol (ARP), Neighbor Discovery (ND), or DHCP. While **IPsec** operates on the network layer (layer 3) and **SSL** or **TLS** on the transport layer (layer 4), **MACsec** operates in the data link layer (layer 2). Combine **MACsec** with security protocols for other networking layers to take advantage of different security features that these standards provide.

See the **MACsec: a different solution to encrypt network traffic** article for more information about the architecture of a **MACsec** network, use case scenarios, and configuration examples.

For examples how to configure MACsec using **wpa_supplicant** and **NetworkManager**, see the **Red Hat Enterprise Linux 7 Networking Guide**.

4.16. REMOVING DATA SECURELY USING SCRUB

The **scrub** utility sets patterns on special files or disk devices to make retrieving data more difficult. Using **scrub** is faster than writing random data on a disk. This process provides high availability, reliability, and data protection.

To start using the **scrub** command, install the **scrub** package:

```
~$ yum install scrub
```

The **scrub** utility operates in one of the following basic modes:

**Character or Block Device**

The **special file** corresponding to a whole disk is scrubbed and all data on it, is destroyed. This is the most effective method.

```
scrub [OPTIONS] special file
```

**File**

A regular file is scrubbed and only the data in the file is destroyed.

```
scrub [OPTIONS] file
```

**Directory**

With the `-X` option, a directory is created and filled with files until the file system is full. Then, the files are scrubbed as in file mode.
Example 4.6. Scrubbing a Raw Device

To **scrub** a raw device /dev/sdf1 with default National Nuclear Security Administration (NNSA) patterns, enter the following command:

```
~]# scrub /dev/sdf1
scrub: using NNSA NAP-14.1-C patterns
scrub: please verify that device size below is correct!
scrub: scrubbing /dev/sdf1 1995650048 bytes (~1GB)
scrub: random |................................................|
scrub: random |................................................|
scrub: 0x00   |................................................|
scrub: verify |................................................|
```

Example 4.7. Scrubbing a File

1. Create a 1MB file:
   ```
   ~]$ base64 /dev/urandom | head -c $[ 1024*1024 ] > file.txt
   ```

2. Show the file size:
   ```
   ~]$ ls -lh
   total 1.0M
   -rw-rw-r--. 1 username username 1.0M Sep  8 15:23 file.txt
   ```

3. Show the contents of the file:
   ```
   ~]$ head -1 file.txt
   JnNpaTEveB/IYsbM9lhuJdw+0jKhwCIBUsxLXLAYb8uItotUlNHKKeS/7bCRKdogE
   P+yJm8VQkL
   ```

4. Scrub the file:
   ```
   ~]$ scrub file.txt
   scrub: using NNSA NAP-14.1-C patterns
   scrub: scrubbing file.txt 1048576 bytes (~1024KB)
   scrub: random |................................................|
   scrub: random |................................................|
   scrub: 0x00   |................................................|
   scrub: verify |................................................|
   ```

5. Verify that the file contents have been scrubbed:
   ```
   ~]$ cat file.txt
   SCRUBBED!
   ```

6. Verify that the file size remains the same:
For more information on **scrub** modes, options, methods, and caveats, see the `scrub(1)` man page.
CHAPTER 5. USING FIREWALLS

The `firewalld` daemon provides a dynamically managed firewall with support for network “zones” to assign a level of trust to a network and its associated connections and interfaces. It has support for IPv4 and IPv6 firewall settings. It supports Ethernet bridges and IP set and has a separation of runtime and permanent configuration options. It also has an interface for services or applications to add firewall rules directly. The complete communication with `firewalld` is done using D-Bus.

**NOTE**

To expand your expertise, you might also be interested in the Red Hat Server Hardening (RH413) training course.

5.1. INTRODUCTION TO FIREWALLD

The firewall daemon uses the restore commands of `iptables`, `ip6tables`, and `ebtables` by default to speed up all firewall actions that are changing the rule set. The normal commands are used if the configuration setting `IndividualCalls` is set to `yes` in the `firewalld.conf` file or if the rules cannot be applied with the restore commands as a fallback solution. Using the normal commands results in significant slow down.

To use the graphical `firewall-config` tool, press the `Super` key to enter the Activities Overview, type `firewall`, and press `Enter`. The `firewall-config` tool appears. You will be prompted for an administrator password.

The sidebar on the left shows the Active Bindings of the active zones. These are grouped by Connections, which are handled by NetworkManager, Interfaces, and Sources.

The `firewall-config` tool has a drop-down selection menu labeled Configuration. This enables selecting between Runtime and Permanent mode. Notice that if you select Permanent, an additional row of icons appears in the left-hand corner. These icons only appear in permanent configuration mode because a service’s parameters cannot be changed in Runtime mode. This setting does not affect the Active Bindings sidebar.

The firewall service provided by `firewalld` is dynamic rather than static because changes to the configuration can be made anytime and are immediately set live. There is no need to save or apply the changes. No unintended disruption of existing network connections occurs as no part of the firewall has to be reloaded.

A command-line client, `firewall-cmd`, is provided. It can be used to make permanent and non-permanent runtime changes as explained in `man firewall-cmd(1)`. Permanent changes need to be made as explained in the `firewalld(1)` man page. Note that the `firewall-cmd` command can be run by the root user and also by an administrative user, in other words, a member of the wheel group. In the latter case, the command will be authorized through the `polkit` mechanism.

The command-line client `firewall-offline-cmd` can only be used by the root user to alter the permanent environment. It is not talking to `firewalld`, but it is using a part of the `firewalld` core and the I/O backends to alter the configuration. It is not recommended to use this tool while `firewalld` is active. It could be used, but changes done with the `firewall-offline-cmd` are not applied immediately to `firewalld`. The changes are applied to the permanent environment after `firewalld` was able to detect file changes in
the file system. For example, the `firewall-offline-cmd` command is used while installing to set up the firewall. It can also be used in the post-installation stage to alter the firewall configuration before the freshly installed system has been booted.

The `firewall-applet` application is able to quickly launch the `NetworkManager` configuration tab for the network connection in use. You can make changes to the assigned firewall zone using the `General` tab. This applet is not installed by default in Red Hat Enterprise Linux.

The configuration for `firewalld` is stored in various XML files in `/usr/lib/firewalld/` and `/etc/firewalld/`. This allows a great deal of flexibility as the files can be edited, written to, backed up, used as templates for other installations, and so on. The configuration in `/usr/lib/firewalld/` is the default and also the fallback configuration, while the configuration in `/etc/firewalld/` is the system specific configuration.

All applications communicate with `firewalld` using the D-Bus interface.

5.1.1. Comparison of firewalld to system-config-firewall and iptables

The essential differences between `firewalld` and the `iptables` (and `ip6tables`) services are:

- The `iptables` service stores configuration in `/etc/sysconfig/iptables` and `/etc/sysconfig/ip6tables`, while `firewalld` stores it in various XML files in `/usr/lib/firewalld/` and `/etc/firewalld/`. Note that the `/etc/sysconfig/iptables` file does not exist as `firewalld` is installed by default on Red Hat Enterprise Linux.

- With the `iptables` service, every single change means flushing all the old rules and reading all the new rules from `/etc/sysconfig/iptables`, while with `firewalld` there is no recreating of all the rules. Only the differences are applied. Consequently, `firewalld` can change the settings during runtime without existing connections being lost.

Both use `iptables tool` to talk to the kernel packet filter.

**NOTE**

`firewalld` is not able to import firewall settings from the `/etc/sysconfig/ip*tables` files. To import `lokkit` or `system-config-firewall` settings, use the `firewall-offline-cmd` and the `/etc/sysconfig/system-config-firewall` file. Custom rules files cannot be imported to `firewalld`. The imported settings are applied to the default zone.
5.1.2. Understanding Network Zones

`firewalld` can be used to separate networks into different zones based on the level of trust the user has decided to place on the interfaces and traffic within that network. `NetworkManager` informs `firewalld` to which zone an interface belongs. An interface's assigned zone can be changed by `NetworkManager` or via the `firewall-config` tool, which can open the relevant `NetworkManager` window for you. You can also use the `firewall-cmd` command-line tool. If an interface is controlled by `NetworkManager` and the user changes the zone of the interface using `firewall-cmd`, `firewall-offline-cmd`, or `firewall-config`, then this request is forwarded to `NetworkManager` and is not handled by `firewalld`.

The zone settings in `/etc/firewalld/` are a range of preset settings, which can be quickly applied to a network interface. They are listed below with a brief explanation.

**drop**
- Any incoming network packets are dropped; there is no reply. Only outgoing network connections are possible.

**block**
- Any incoming network connections are rejected with an `icmp-host-prohibited` message for `IPv4` and `icmp6-adm-prohibited` for `IPv6`. Only network connections initiated from within the system are possible.

**public**
- For use in public areas. You do not trust the other computers on the network to not harm your computer. Only selected incoming connections are accepted.

**external**
- For use on external networks with masquerading enabled, especially for routers. You do not trust the other computers on the network to not harm your computer. Only selected incoming connections are accepted.
dmz
For computers in your demilitarized zone that are publicly-accessible with limited access to your internal network. Only selected incoming connections are accepted.

work
For use in work areas. You mostly trust the other computers on networks to not harm your computer. Only selected incoming connections are accepted.

home
For use in home areas. You mostly trust the other computers on networks to not harm your computer. Only selected incoming connections are accepted.

internal
For use on internal networks. You mostly trust the other computers on the networks to not harm your computer. Only selected incoming connections are accepted.

trusted
All network connections are accepted.

It is possible to designate one of these zones to be the default zone. When interface connections are added to NetworkManager, they are assigned to the default zone. On installation, the default zone in firewalld is set to be the public zone.

Choosing a Network Zone
The network zone names have been chosen to be self-explanatory and to allow users to quickly make a reasonable decision. A review of the default configuration settings should be made and unnecessary services disabled according to your needs and risk assessments.

The zone names and settings are proposals and can be changed according to the needs. A built-in zone cannot be removed, but it is possible to revert the zone configuration back to the initial defaults by loading the zone defaults either in the permanent configuration of firewall-config or firewall-cmd.

5.1.3. Understanding Predefined Services
A service can be a list of local ports, protocols, source ports, and destinations as well as a list of firewall helper modules automatically loaded if a service is enabled. The use of predefined services makes it easier for the user to enable and disable access to a service. Using the predefined services or custom-defined services, as opposed to opening ports or ranges of ports, may make administration easier. Service configuration options and generic file information are described in the firewalld.service(5) man page. The services are specified by means of individual XML configuration files, which are named in the following format: service-name.xml. Protocol names are preferred over service or application names in firewalld.

To view the list of services using the graphical firewall-config tool, press the Super key to enter the Activities Overview, type firewall, and press Enter. The firewall-config tool appears. You will be prompted for an administrator password. You can now view the list of services under the Services tab.

To list all services available on the system, enter the following command:
To get the settings of a service, use the following command:

```bash
$ firewall-cmd --info-service=service-name
```

To list only the default predefined services available using the command-line, enter the following command:

```bash
$ ls /usr/lib/firewalld/services/
```

**NOTE**

The root user is not needed to list files in `/usr/lib/firewalld`. Make sure to change the attributes accordingly after an addition of custom private files.

Files in `/usr/lib/firewalld/services/` must not be edited. Only the files in `/etc/firewalld/services/` should be edited.

To list the system or user-created services, enter the following command as `root`:

```bash
$ ls /etc/firewalld/services/
```

Services can be added and removed using the graphical `firewall-config` tool, `firewall-cmd`, and `firewall-offline-cmd`. Alternatively, you can edit the XML files in `/etc/firewalld/services/`. If a service has not been added or changed by the user, then no corresponding XML file will be found in `/etc/firewalld/services/`. The files `/usr/lib/firewalld/services/` can be used as templates if you want to add or change a service.

To add a new service in a terminal, use `firewall-cmd`, or `firewall-offline-cmd` in case of not active `firewalld`, enter the following command to add a new and empty service:

```bash
$ firewall-cmd --permanent --new-service=service-name
```

To add a new service using a local file, use the following command:

```bash
$ firewall-cmd --permanent --new-service-from-file=service-name.xml
```

You can change the service name with the additional `--name=service-name` option.

As soon as service settings are changed, an updated copy of the service is placed into `/etc/firewalld/services/`.

As `root`, you can enter the following command to copy a service manually:

```bash
$ cp /usr/lib/firewalld/services/service-name.xml /etc/firewalld/services/service-name.xml
```

`firewalld` loads files from `/usr/lib/firewalld/services` in the first place. If files are placed in `/etc/firewalld/services` and they are valid, then these will override the matching files from `/usr/lib/firewalld/services`. The overriden files in
/usr/lib/firewalld/services will be used as soon as the matching files in
/etc/firewalld/services have been removed or if firewalld has been asked to load the
defaults of the services. This applies to the permanent environment only. A reload is
needed to get these fallbacks also in the runtime environment.

5.1.4. Understanding the Direct Interface

firewalld has direct interface, which enables directly passing rules to iptables, ip6tables
and ebtables. It is primarily intended for use by applications. It is not recommended and it
is dangerous to use the direct interface if you are not very familiar with iptables, as you
could inadvertently cause a breach in the firewall. As long as the tracked interface parts
are used, it is still possible to query firewalld and see the changes made by an application
using this mode. The untracked passthrough mode is only intended for services that
completely take care of the own rule set, such as libvirt and docker. The direct interface is
used by adding the --direct option to the firewall-cmd command.

The direct interface mode is intended for services or applications to add specific firewall
rules during runtime. The rules can be made permanent by adding the --permanent option
using the firewall-cmd --permanent --direct command or by modifying
/etc/firewalld/direct.xml. If the rules are not made permanent, then they need to be
applied every time after receiving the start, restart, or reload message from firewalld
using D-Bus. With the direct interface, it is possible to add chains, rules, and tracked and
untracked passthrough rules. You can also use direct rules in zone-specific chains.

5.2. INSTALLING FIREWALLD

In Red Hat Enterprise Linux 7, firewalld is installed by default. If required, to ensure that it
is, enter the following command as root:

    ~> # yum install firewalld

The graphical user interface configuration tool firewall-config is installed by default in
some versions of Red Hat Enterprise Linux 7. If required, enter the following command as
root to ensure firewall-config is installed:

    ~> # yum install firewall-config

To install the optional firewall-applet, enter the following command as root:

    ~> # yum install firewall-applet

The applet is visible in the panel of the desktop with the next login. The use of the applet in
GNOME is limited according to the GNOME rules for applets.

5.2.1. Stopping firewalld

To stop firewalld, enter the following command as root:

    ~> # systemctl stop firewalld

To prevent firewalld from starting automatically at system start, enter the following
command as root:
~# systemctl disable firewalld

To make sure firewalld is not started by accessing the **firewalld** D-Bus interface and also if other services require **firewalld**, enter the following command as **root**:

~# systemctl mask firewalld
~# systemctl start firewalld

**5.2.2. Starting firewalld**

To start **firewalld**, enter the following command as **root**:

~# systemctl unmask firewalld
~# systemctl start firewalld

To ensure **firewalld** starts automatically at system start, enter the following command as **root**:

~# systemctl enable firewalld

**5.2.3. Checking If firewalld Is Running**

To check if **firewalld** is running, enter the following command:

~$ 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 Tue 2016-10-11 09:15:58 CEST; 2 days ago
    Docs: man:firewalld(1)
  Main PID: 721 (firewalld)
  CGroup: /system.slice/firewalld.service
        └─721 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid

Oct 11 09:15:57 localhost.localdomain systemd[1]: Starting firewalld -
dynami...
Oct 11 09:15:58 localhost.localdomain systemd[1]: Started firewalld -
dynamic...

Hint: Some lines were ellipsized, use -l to show in full.

In addition, check if **firewall-cmd** can connect to the daemon by entering the following command:

~$ firewall-cmd --state
running

**5.3. CONFIGURING FIREWALLD**

The firewall service, implemented by the **firewalld** daemon, can be configured using the **firewall-config** graphical user interface tool, using the **firewall-cmd** and **firewall-offline-cmd** command-line interface tools, and by editing XML configuration files. These methods will be described in order.
5.3.1. Configuring firewalld Using The Graphical User Interface

5.3.1.1. Starting the Graphical Firewall Configuration Tool

To start the graphical `firewall-config` tool, press the **Super** key to enter the Activities Overview, type `firewall`, and press **Enter**. The `firewall-config` tool appears. You will be prompted for an administrator password.

To start the graphical firewall configuration tool using the command-line, enter the following command:

```
~$ firewall-config
```

The **Firewall Configuration** window opens. Note that this command can be run as a normal user, but you will be prompted occasionally for an administrator password.

![Figure 5.2. The Firewall Configuration Tool](image)

Look for the “Connection to firewalld established” message in the lower-left corner. This indicates that the `firewall-config` tool is connected to `firewalld`. Note that the **ICMP**
Types, IPSets, Direct Configuration, and Lockdown Whitelist tabs are only visible after being selected from the View drop-down menu. The Active Bindings sidebar on the left is visible by default.

### 5.3.1.2. Changing the Firewall Settings

To immediately change the current firewall settings, ensure the current view is set to Runtime. Alternatively, to edit the settings to be applied at the next system start or firewall reload, select Permanent from the drop-down list.

**NOTE**

When making changes to the firewall settings in Runtime mode, your selection takes immediate effect when you set or clear the check box associated with the service. You should keep this in mind when working on a system that may be in use by other users.

When making changes to the firewall settings in Permanent mode, your selection will only take effect when you reload the firewall or the system restarts. Click the Options menu and select Reload Firewall.

You can select zones in the left-hand side column. You will notice the zones have some services enabled; you may need to resize the window or scroll to see the full list. You can customize the settings by selecting and deselecting a service.

### 5.3.1.3. Adding an Interface to a Zone

To add a connection (the interfaces used by a connection) to a zone, start firewall-config. Click on the zone in the zone list on the left and select the Interfaces tab on the right. Click on the Add button to rise a new dialog to add the interface.

To change the zone setting for an interface, double-click the proper connection or interface in the Active Bindings sidebar. Select the new firewall zone from the drop-down menu in the following dialog and confirm by clicking OK.

Alternatively, to add or reassign an interface of a connection to a zone, start firewall-config, select Options from the menu bar, and select Change Zones of Connections from the drop-down menu. The Connections, Interface, and Source list displays. Select the connection to be reassigned. The Select Zone for Connection window appears. Select the new firewall zone from the drop-down menu and click OK.

For connections handled by NetworkManager, the request to change the zone is forwarded to NetworkManager. The zone interface setting will not be saved in firewalld.

You can also use the firewall-cmd command-line tool or the firewall-applet applet to change the zone for a connection, interface, and source.

The connections without specific zone settings are automatically bound to the default zone. A change of the default zone consequently applies to the zone bindings of all such connections.

### 5.3.1.4. Setting the Default Zone

To set the default zone that new interfaces will be assigned to, start firewall-config, select Options from the menu bar, and select Change Default Zone from the drop-down menu.
The Default Zone window appears. Select the zone from the list that you want to be used as the default zone and click OK. Alternatively, enter the following command:

```
~]$ firewall-cmd --set-default-zone=zone-name
```

### 5.3.1.5. Configuring Services

To enable or disable a predefined or custom service, start the firewall-config tool and select the network zone whose services are to be configured. Select the Services tab and select the check box for each type of service you want to trust. Clear the check box to block a service.

To edit a service, start the firewall-config tool and select Permanent mode from the drop-down selection menu labeled Configuration. Additional icons and menu buttons appear at the bottom of the Services window. Select the service you want to configure.

The Ports, Protocols, and Source Port tabs enables adding, changing, and removing of ports, protocols, and source port for the selected service. The modules tab is for configuring Netfilter helper modules. The Destination tab enables limiting traffic to a particular destination address and Internet Protocol (IPv4 or IPv6).

**NOTE**

It is not possible to alter service settings in Runtime mode.

### 5.3.1.6. Opening Ports in the Firewall

To permit traffic through the firewall to a certain port, start the firewall-config tool and select the network zone whose settings you want to change. Select the Ports tab and click the Add button on the right-hand side. The Port and Protocol window opens.

Enter the port number or range of ports to permit. Select tcp or udp from the drop-down list.

### 5.3.1.7. Opening Protocols in the Firewall

To permit traffic through the firewall using a certain protocol, start the firewall-config tool and select the network zone whose settings you want to change. Select the Protocols tab and click the Add button on the right-hand side. The Protocol window opens.

Either select a protocol from the drop-down list or select the Other Protocol check box and enter the protocol in the field.

### 5.3.1.8. Opening Source Ports in the Firewall

To permit traffic through the firewall from a certain port, start the firewall-config tool and select the network zone whose settings you want to change. Select the Source Port tab and click the Add button on the right-hand side. The Source Port window opens.

Enter the port number or range of ports to permit. Select tcp or udp from the drop-down list.

### 5.3.1.9. Enabling IPv4 Address Masquerading
To translate **IPv4** addresses to a single external address, start the `firewall-config` tool and select the network zone whose addresses are to be translated. Select the **Masquerading** tab and select the check box to enable the translation of **IPv4** addresses to a single address.

**NOTE**

To enable masquerading for **IPv6**, use a rich rule.

### 5.3.1.10. Configuring Port Forwarding

To forward inbound network traffic, or “packets”, for a specific port to an internal address or alternative port, first enable IP address masquerading, then select the **Port Forwarding** tab.

Select the protocol of the incoming traffic and the port or range of ports on the upper section of the window. The lower section is for setting details about the destination.

To forward traffic to a local port (a port on the same system), select the **Local forwarding** check box. Enter the local port or range of ports for the traffic to be sent to.

To forward traffic to another **IPv4** address, select the **Forward to another port** check box. Enter the destination IP address and port or port range. The default is to send to the same port if the port field is left empty. Click **OK** to apply the changes.

### 5.3.1.11. Configuring the ICMP Filter

To enable or disable an **ICMP** filter, start the `firewall-config` tool and select the network zone whose messages are to be filtered. Select the **ICMP Filter** tab and select the check box for each type of **ICMP** message you want to filter. Clear the check box to disable a filter. This setting is per direction and the default allows everything.

To edit an **ICMP** type, start the `firewall-config` tool and select **Permanent** mode from the drop-down selection menu labeled **Configuration**. Additional icons appear at the bottom of the **Services** window. Select **Yes** in the following dialog to enable masquerading and to make forwarding to another machine working.

To enable inverting the **ICMP Filter**, click the **Invert Filter** check box on the right. Only marked **ICMP** types are now accepted, all other are rejected. In a zone using the DROP target, they are dropped.

### 5.3.1.12. Configuring Rich Rules

To enable or disable a rich rule, start the `firewall-config` tool and select the network zone whose services are to be configured. Select the **Rich Rules** tab and click the **Add** button on the right-hand side. The **Rich Rule** window appears.

Select the **Family** the rule should be added to, leave it at **ipv4** and **ipv6** to add a rule for **IPv4** and **IPv6**. Enable the **Element** check box if you want to select **aservice**, **port**, **protocol**, **icmp-block**, **forward-port**, **source port** or if you want to enable **masquerade** in the rule. For all elements except **masquerade**, it is needed to click on the button on the right. The **Service** window appears to select the setting of the element.

With the **Action** check box, you can enable a custom action for the rule like **accept**, **reject**, **drop**, or **mark**. If the rule **Family** is set to either **ipv4** or **ipv6**, you can enable the...
with the **Type** check box to select an alternative reject type from the drop-down menu that matches the rule **Family**. Additionally, you can set a limit for this action by enabling the **With limit** check box.

In the **Source** section, you can select a source match for this rule. This can be an IP address or range, a MAC address or an IP set. With the **inverted** check box, you can negate this match. The IP address is only selectable if the **Family** is either **ipv4** or **ipv6**.

With the **Destination** address, you can select an IP address matching the selected **Family**. It is selectable only if the **Family** is either **ipv4** or **ipv6**.

To enable logging to the system log with the rule, use the **Log** check box. It is necessary to select a logging prefix in the **Prefix** text field. Please select the log **Level**. It can be **emergency**, **alert**, **critical**, **error**, **warning**, **notice**, **info** or **debug**. With the optional limit, the amount of log messages in the system log can be selected. If logging is enabled the rule will be duplicated to be able to log.

To enable logging using the Linux **Audit** system, use the **Audit** check box. See the **System Auditing** chapter for more information.

If the rule is complete and the **OK** button is active, you can add the rule. If the button is not active, there is a tooltip that shows what is missing or not appropriate.

### 5.3.1.13. Configuring Sources

To add a source to a zone, start **firewall-config**. Click on a zone in the zone list on the left and select the **Sources** tab on the right. With clicking the Add button, there will be a new dialog to add the source. A source can either be an **IP** address or range, a **MAC** address or an **ipset**. Select the type in the drop-down menu on the left and click the button on the right to select or enter the setting.

### 5.3.2. Configuring IP Sets Using firewall-config

To configure IP sets, start the **firewall-config** tool and select the **IPSets** tab. Select an IP set from the list on the left to change the runtime settings of an IP set that has been created with **firewalld** already.

To add new IP sets or to change base IP set settings, switch to **Permanent** mode. Additional icons and menu buttons appear at the bottom of the **IPSets** window. Select the IP set you want to configure. The entries tab on the right shows the entries that are part of the IP set. There are no entries listed for IP sets that use a timeout, as the entries are kept and handled in kernel space.

With the **Add** button, you can add single entries, but also entries from a file. With **Remove** you can remove the selected entry, all entries and also entries from a file. The file should contain an entry per line. Lines starting with a hash or semicolon are ignored. Also empty lines.

After clicking on the + button to add a new IP set, a new window appears to configure the base IP set settings. There are three settings that need to be configured for an IP set: **Name**, **Type**, and **Family**. **Name** can contain all alphanumeric characters and additionally ‘-’, ‘:’, ‘.’, and ‘‘’. The maximum name length is 32 characters. **Type** can be: **hash:ip**, **hash:net**, and **hash:mac**. Bitmap types are not supported by **firewalld** as they can be only used with **IPv4**. Combined types are not supported, too.
To have a simple and fast IP address or network set, use the `hash:net` type. The `hash:ip` type expands all ranges and network segments internally and reaches the hash limit soon.

For these types, it is also necessary to define `Family`. This can be either `inet` for IPv4 or `inet6` for IPv6.

To store MAC addresses in an IP set use `hash:mac -Family` is not selectable in this case. To define a lifetime of the added entries for use with external services like fail2ban, use the `Timeout` setting. Note that `firewalld` is not able to show the temporarily stored entries with a timeout. Use the `ipset` command for such entries.

To define the initial hash size for an IP set, use the `Hashsize` setting. Limit the maximum number of elements that can be stored in an IP set by using the `Maxelem` field.

You can use the created IP set as a source in a zone, in a rich rule, and also in a direct rule. For more information on IP sets and the settings, see Section 5.4, “Using the `iptables` Service”. To use IP sets that are not supported by `firewalld`, see Section 5.4.1.1, “Using IP Sets with `firewalld`”.

### 5.3.3. Configuring the Firewall Using the `firewall-cmd` Command-Line Tool

The `firewall-cmd` command-line tool is part of the `firewalld` application that is installed by default. You can verify that it is installed by checking the version or displaying the help output. Enter the following command to check the version:

```bash
~ $ firewall-cmd --version
```

Enter the following command to view the help output:

```bash
~ $ firewall-cmd --help
```

We list a selection of commands below; for a full list see the `firewall-cmd(1)` man page.

**NOTE**

To make a command permanent or persistent, add the `--permanent` option to all commands apart from the `--direct` commands (which are by their nature temporary). Note that this not only means the change will be permanent, but that the change will only take effect after `firewalld` reload, service restart, or after system reboot. Settings made with `firewall-cmd` without the `--permanent` option take effect immediately but are only valid till next firewall reload, system boot, or `firewalld` service restart. Reloading the `firewalld` does not in itself break connections, but be aware you are discarding temporary changes by doing so.

To make a command both persistent and take effect immediately, enter the command twice: once with the `--permanent` and once without. This is because a `firewalld` reload takes more time than just repeating a command because it has to reload all configuration files and recreate the whole firewall configuration. While reloading, the policy for built-in chains is set to DROP for security reasons and is then reset to ACCEPT at the end. Service disruption is possible during the reload.
Figure 5.3. The firewalld Architecture

IMPORTANT

All options to change the zone binding for interfaces that are under control of NetworkManager are forwarded to NetworkManager. These changes are not applied to the firewalld configuration if the request for NetworkManager succeeds. This is also the case with the --permanent option.

For interfaces that are not under control of NetworkManager, the change applies to the firewalld configuration. If there is an ifcfg file that uses this interface, then the ZONE= setting in this ifcfg file is adapted to make sure that the configuration in firewalld and the ifcfg file is consistent. If there is more than one ifcfg file using this interface then the first one is used.

See the Red Hat Enterprise Linux 7 Networking Guide for information on NetworkManager and working with ifcfg files.

For configuration settings such as the default zone, there is no difference between the runtime and permanent environment when using the command-line and GUI tools.

5.3.4. Viewing the Firewall Settings Using the Command-Line Interface (CLI)

To get a text display of the state of firewalld, enter the following command:

```
~$ firewall-cmd --state
```

To view the list of active zones with a list of the interfaces currently assigned to them, enter the following command:

```
~$ firewall-cmd --get-active-zones
public
  interfaces: em1
```
To find out the zone that an interface, for example, em1, is currently assigned to, enter the following command:

```
~]$ firewall-cmd --get-zone-of-interface=em1
public
```

To find out all the interfaces assigned to a zone, for example, the public zone, enter the following command as **root**:

```
~]$ firewall-cmd --zone=public --list-interfaces
em1 wlan0
```

This information is obtained from **NetworkManager** and only shows interfaces, not connections.

To find out all the settings of a zone, for example, the public zone, enter the following command as **root**:

```
~]$ firewall-cmd --zone=public --list-all
public
  interfaces:
    services: mdns dhcpv6-client ssh
  ports:
    forward-ports:
    icmp-blocks: source-quench
```

To view the zone information, use the **--info-zone** option. To get the verbose output with the description and short description, use the additional **-v** option.

```
~]$ firewall-cmd --info-zone=public
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: em1
  sources:
    services: dhcpv6-client mdns ssh
    ports:
    protocols:
    masquerade: no
    forward-ports:
    source-ports:
    icmp-blocks:
    rich rules:
```

To view the list of services currently loaded, enter the following command as **root**:

```
~]$ firewall-cmd --get-services
cluster-suite pop3s bacula-client smtp ipp radius bacula ftp mdns samba
dhcpv6-client dns openvpn imaps samba-client http https ntp vnc-server
telnet libvirt ssh ipsec ipp-client amanda-client tftp-client nfs tftp
libvirt-tls
```

This lists the names of the predefined services loaded from **/usr/lib/firewalld/services/** as well as any custom services that are currently loaded. Note that the configuration files themselves are named **service-name.xml**.
To list the custom services that have been created but not loaded, use the following command as **root**:

```
~]# firewall-cmd --permanent --get-services
```

This lists all services, including custom services configured in `/etc/firewalld/services/`, even if they are not yet loaded.

To show the settings of the **ftp** service, use the following command as **root**:

```
~]# firewall-cmd --info-service=ftp
ftp
  ports: 21/tcp
  protocols:
  source-ports:
  modules: nf_conntrack_ftp
  destination:
```

To view the settings in permanent configuration mode, use the **--permanent** option.

### 5.3.5. Changing the Firewall Settings Using the Command-Line Interface (CLI)

#### 5.3.5.1. Dropping All Packets (Panic Mode)

To start dropping all incoming and outgoing packets, enter the following command as **root**:

```
~]# firewall-cmd --panic-on
```

All incoming and outgoing packets will be dropped. Active connections will be terminated after a period of inactivity; the time taken depends on the individual session timeout values.

To start passing incoming and outgoing packets again, enter the following command as **root**:

```
~]# firewall-cmd --panic-off
```

After disabling panic mode, established connections might work again if panic mode was enabled for a short period of time.

To find out if panic mode is enabled or disabled, enter the following command:

```
~]$ firewall-cmd --query-panic
```

The command prints **yes** with exit status 0 if enabled. It prints **no** with exit status 1 otherwise.

#### 5.3.5.2. Reloading the Firewall Using the Command-Line Interface (CLI)

To reload the firewall without interrupting user connections (without losing state information), enter the following command:
A firewall reload involves reloading all configuration files and recreating the whole firewall configuration. While reloading, the policy for built-in chains is set to DROP for security reasons and is then reset to ACCEPT at the end. Service disruption is therefore possible during the reload. Alternatively as the root user, send the SIGHUP signal to reload the firewall.

To reload the firewall and interrupt user connections, discarding state information, enter the following command as root:

```
~$ firewall-cmd --reload
```

This command should normally only be used in case of severe firewall problems. For example, use this command if there are state information problems and no connection can be established but the firewall rules are correct.

### 5.3.5.3. Add an Interface to a Zone Using the Command-Line Interface (CLI)

To add an interface to a zone (for example, to add em1 to the public zone), enter the following command as root:

```
~# firewall-cmd --zone=public --add-interface=em1
```

To make this setting persistent, repeat the commands adding the --permanent option.

### 5.3.5.4. Add an Interface to a Zone by Editing the Interface Configuration File

To add an interface to a zone by editing the ifcfg-em1 configuration file (for example, to add em1 to the work zone), add the following line to ifcfg-em1 as root:

```
ZONE=work
```

Note that if you omit the ZONE option, or use ZONE=, or ZONE='', then the default zone will be used.

NetworkManager will automatically reconnect and the zone will be set accordingly.

### 5.3.5.5. Configuring the Default Zone by Editing the firewalld Configuration File

As root, open /etc/firewalld/firewalld.conf and edit the file as follows:

```
# default zone
# The default zone used if an empty zone string is used.
# Default: public
DefaultZone=home
```

Reload the firewall by entering the following command as root:

```
~# firewall-cmd --reload
```
This will reload the firewall without losing state information (that is, TCP sessions will not be terminated), but service disruption is possible during the reload.

5.3.5.6. Setting the Default Zone by Using the Command-Line Interface (CLI)

To set the default zone (for example, to public), enter the following command as root:

```bash
~$ firewall-cmd --set-default-zone=public
```

This change will take effect immediately; in this case, it is not necessary to reload the firewall.

5.3.5.7. Opening Ports in the Firewall Using the Command-Line Interface (CLI)

To list all open ports for a zone (for example, dmz), enter the following command as root:

```bash
~$ firewall-cmd --zone=dmz --list-ports
```

Note that this will not show ports opened as a result of the `--add-services` command.

To add a port to a zone (for example, to allow TCP traffic to port 8080 to the dmz zone), enter the following command as root:

```bash
~$ firewall-cmd --zone=dmz --add-port=8080/tcp
```

To make this setting persistent, repeat the command adding the `--permanent` option.

To add a range of ports to a zone (for example, to allow the ports from 5060 to 5061 to the public zone, enter the following command as root:

```bash
~$ firewall-cmd --zone=public --add-port=5060-5061/udp
```

To make this setting persistent, repeat the command adding the `--permanent` option.

5.3.5.8. Opening Protocols Using the Command Line Interface (CLI)

To list all open ports for a zone (dmz, for example), enter the following command as root:

```bash
~$ firewall-cmd --zone=dmz --list-protocols
```

Note that this command does not show protocols opened as a result of the `firewall-cmd --add-services` command.

To add a protocol to a zone (for example, to allow ESP traffic to the dmz zone), enter the following command as root:

```bash
~$ firewall-cmd --zone=dmz --add-protocol=esp
```

To make this setting persistent, add the `--permanent` option.
5.3.5.9. Opening Source Ports Using the Command Line Interface (CLI)

To list all open source ports for a zone (for example, the dmz zone), enter the following command as **root**:

```
~# firewall-cmd --zone=dmz --list-source-ports
```

Note that this command does not show source ports opened as a result of the `firewall-cmd --add-services` command.

To add a source port to a zone (for example, to allow TCP traffic from port 8080 to the dmz zone), use the following command as **root**:

```
~# firewall-cmd --zone=dmz --add-source-port=8080/tcp
```

To add a range of source ports to a zone (for example, to allow the ports from 5060 to 5061 to the public zone), enter the following command as **root**:

```
~# firewall-cmd --zone=public --add-source-port=5060-5061/udp
```

To make the settings persistent, add the **--permanent** option.

5.3.5.10. Adding a Service to a Zone Using the Command-Line Interface (CLI)

To add a service to a zone (for example, to allow SMTP to the work zone), enter the following command as **root**:

```
~# firewall-cmd --zone=work --add-service=smtp
```

To make this setting persistent, repeat the command adding the **--permanent** option.

5.3.5.11. Removing a Service from a Zone Using the Command-Line Interface (CLI)

To remove a service from a zone (for example, to remove SMTP from the work zone), enter the following command as **root**:

```
~# firewall-cmd --zone=work --remove-service=smtp
```

To make this change persistent, repeat the command adding the **--permanent** option. This change will not break established connections. If that is your intention, you can use the **--complete-reload** option, but this will break all established connections — not just for the service you have removed.

5.3.5.12. Adding a Service to a Zone by Editing XML Files

To view the default zone files, enter the following command as **root**:

```
~# ls /usr/lib/firewalld/zones/
block.xml  drop.xml  home.xml  public.xml  work.xml
dmz.xml  external.xml  internal.xml  trusted.xml
```
These files must not be edited. They are used by default if no equivalent file exists in the 
/etc/firewalld/zones/ directory.

To view the zone files that have been changed from the default, enter the following command as root:

```
~# ls /etc/firewalld/zones/
    external.xml  public.xml  public.xml.old
```

In the example shown above, the work zone file does not exist. To add the work zone file, enter the following command as root:

```
~# cp /usr/lib/firewalld/zones/work.xml /etc/firewalld/zones/
```

You can now edit the file in the /etc/firewalld/zones/ directory. If you delete the file, firewalld will fall back to using the default file in /usr/lib/firewalld/zones/.

To add a service to a zone (for example, to allow SMTP to the work zone), add the following line to the /etc/firewalld/zones/work.xml file as root:

```
<service name="smtp"/>
```

### 5.3.5.13. Removing a Service from a Zone by Editing XML files

An editor running with root privileges is required to edit the XML zone files. To view the files for previously configured zones, enter the following command as root:

```
~# ls /etc/firewalld/zones/
    external.xml  public.xml  work.xml
```

To remove a service from a zone (for example, to remove SMTP from the work zone), use an editor with root privileges to edit the /etc/firewalld/zones/work.xml file to remove the following line:

```
<service name="smtp"/>
```

If no other changes have been made to the work.xml file, it can be removed and firewalld will use the default /usr/lib/firewalld/zones/work.xml configuration file after the next reload or system boot.

### 5.3.5.14. Configuring IP Address Masquerading

To check if IP masquerading is enabled (for example, for the external zone), enter the following command as root:

```
~# firewall-cmd --zone=external --query-masquerade
```

The command prints yes with exit status 0 if enabled. It prints no with exit status 1 otherwise. If zone is omitted, the default zone will be used.

To enable IP masquerading, enter the following command as root:

```
~# firewall-cmd --zone=external --add-masquerade
```
To make this setting persistent, repeat the command adding the `--permanent` option.

To disable IP masquerading, enter the following command as `root`:

```
~# firewall-cmd --zone=external --remove-masquerade
```

To make this setting persistent, repeat the command adding the `--permanent` option.

### 5.3.5.15. Configuring Port Forwarding Using the Command-Line Interface (CLI)

To forward inbound network packets from one port to an alternative port or address, first enable IP address masquerading for a zone (for example, `external`), by entering the following command as `root`:

```
~# firewall-cmd --zone=external --add-masquerade
```

To forward packets to a local port (a port on the same system), enter the following command as `root`:

```
~# firewall-cmd --zone=external --add-forward-port=port=22:proto=tcp:toport=3753
```

In this example, the packets intended for port 22 are now forwarded to port 3753. The original destination port is specified with the `port` option. This option can be a port or port range, together with a protocol. The protocol, if specified, must be one of either `tcp` or `udp`. The new local port (the port or range of ports to which the traffic is being forwarded to) is specified with the `toport` option. To make this setting persistent, repeat the commands adding the `--permanent` option.

To forward packets to another IPv4 address, usually an internal address, without changing the destination port, enter the following command as `root`:

```
~# firewall-cmd --zone=external --add-forward-port=port=22:proto=tcp:toaddr=192.0.2.55
```

In this example, the packets intended for port 22 are now forwarded to the same port at the address given with the `toaddr`. The original destination port is specified with the `port` option. This option can be a port or port range, together with a protocol. The protocol, if specified, must be one of either `tcp` or `udp`. The new destination port (the port or range of ports to which the traffic is being forwarded to) is specified with the `toport` option. To make this setting persistent, repeat the command adding the `--permanent` option.

To forward packets to another port at another IPv4 address, usually an internal address, enter the following command as `root`:

```
~# firewall-cmd --zone=external --add-forward-port=port=22:proto=tcp:toport=2055:toaddr=192.0.2.55
```

In this example, the packets intended for port 22 are now forwarded to port 2055 at the address given with the `toaddr`. The original destination port is specified with the `port` option. This option can be a port or port range, together with a protocol. The protocol,
if specified, must be one of either tcp or udp. The new destination port, the port or range of ports to which the traffic is being forwarded to, is specified with the toport option. To make this setting persistent, repeat the command adding the --permanent option.

5.3.6. Configuring the Firewall Using XML Files

The configuration settings for firewalld are stored in XML files in the /etc/firewalld/ directory. Do not edit the files in the /usr/lib/firewalld/ directory (the files define the default settings). You will need root user permissions to view and edit the XML files. The XML files are explained in three man pages:

- firewalld.icmptype(5) man page — Describes XML configuration files for ICMP filtering.
- firewalld.service(5) man page — Describes XML configuration files for firewalld service.
- firewalld.zone(5) man page — Describes XML configuration files for firewalld zone configuration.

The XML files can be created and edited directly or created indirectly using the graphical and command-line tools. Organizations can distribute them in RPM files, which can make management and version control easier. Tools like Puppet can distribute such configuration files.

5.3.7. Using the Direct Interface

It is possible to add and remove chains during runtime by using the --direct option with the firewall-cmd tool. A few examples are presented here. See the firewall-cmd(1) man page for more information.

It is dangerous to use the direct interface if you are not very familiar with iptables as you could inadvertently cause a breach in the firewall.

The direct interface mode is intended for services or applications to add specific firewall rules during runtime. The rules can be made permanent by adding the --permanent option using the firewall-cmd --permanent --direct command or by modifying /etc/firewalld/direct.xml. See man firewalld.direct(5) for information on the /etc/firewalld/direct.xml file.

5.3.7.1. Adding a Rule Using the Direct Interface

To add a rule to the “IN_public_allow” chain, enter the following command as root:

```bash
~$ firewall-cmd --direct --add-rule ipv4 filter IN_public_allow \ 0 -m tcp -p tcp --dport 666 -j ACCEPT
```

Add the --permanent option to make the setting persistent.

5.3.7.2. Removing a Rule Using the Direct Interface

To remove a rule from the “IN_public_allow” chain, enter the following command as root:

```bash
~$ firewall-cmd --direct --remove-rule ipv4 filter IN_public_allow \ 0 -m tcp -p tcp --dport 666 -j ACCEPT
```
Add the --permanent option to make the setting persistent.

### 5.3.7.3. Listing Rules Using the Direct Interface

To list the rules in the “IN_public_allow” chain, enter the following command as root:

```
~]# firewall-cmd --direct --get-rules ipv4 filter IN_public_allow
```

Note that this command (the --get-rules option) only lists rules previously added using the --add-rule option. It does not list existing iptables rules added by other means.

### 5.3.8. Configuring Complex Firewall Rules with the "Rich Language" Syntax

With the “rich language” syntax, complex firewall rules can be created in a way that is easier to understand than the direct-interface method. In addition, the settings can be made permanent. The language uses keywords with values and is an abstract representation of iptables rules. Zones can be configured using this language; the current configuration method will still be supported.

#### 5.3.8.1. Formatting of the Rich Language Commands

All the commands in this section need to be run as root. The format of the command to add a rule is as follows:

```
firewall-cmd [--zone=zone] --add-rich-rule='rule' [--timeout=timeval]
```

This will add a rich language rule rule for zone zone. This option can be specified multiple times. If the zone is omitted, the default zone is used. If a timeout is supplied, the rule or rules only stay active for the amount of time specified and will be removed automatically afterwards. The time value can be followed by s (seconds), m (minutes), or h (hours) to specify the unit of time. The default is seconds.

To remove a rule:

```
firewall-cmd [--zone=zone] --remove-rich-rule='rule'
```

This will remove a rich language rule rule for zone zone. This option can be specified multiple times. If the zone is omitted, the default zone is used.

To check if a rule is present:

```
firewall-cmd [--zone=zone] --query-rich-rule='rule'
```

This will return whether a rich language rule rule has been added for the zone zone. The command prints yes with exit status 0 if enabled. It prints no with exit status 1 otherwise. If the zone is omitted, the default zone is used.

For information about the rich language representation used in the zone configuration files, see the firewalld.zone(5) man page.

#### 5.3.8.2. Understanding the Rich Rule Structure
The format or structure of the rich rule commands is as follows:

```plaintext
rule [family="rule family"]
    [ source [NOT] [address="address"] [mac="mac-address"] [ipset="ipset"]
    ]
    [ destination [NOT] address="address" ]
    [ element ]
    [ log [prefix="prefix text"] [level="log level"] [limit value="rate/duration"] ]
    [ audit ]
    [ action ]
```

**NOTE**

The structure of the rich rule in the file uses the NOT keyword to invert the sense of the source and destination address commands, but the command line uses the invert="true" option.

A rule is associated with a particular zone. A zone can have several rules. If some rules interact or contradict, the first rule that matches the packet applies.

### 5.3.8.3. Understanding the Rich Rule Command Options

**family**

If the rule family is provided, either ipv4 or ipv6, it limits the rule to IPv4 or IPv6, respectively. If the rule family is not provided, the rule is added for both IPv4 and IPv6. If source or destination addresses are used in a rule, then the rule family needs to be provided. This is also the case for port forwarding.

**Source and Destination Addresses**

**source**

By specifying the source address, the origin of a connection attempt can be limited to the source address. A source address or address range is either an IP address or a network IP address with a mask for IPv4 or IPv6. For IPv4, the mask can be a network mask or a plain number. For IPv6, the mask is a plain number. The use of host names is not supported. It is possible to invert the sense of the source address command by adding the NOT keyword; all but the supplied address matches.

A MAC address and also an IP set with type hash:mac can be added for IPv4 and IPv6 if no family is specified for the rule. Other IP sets need to match the family setting of the rule.

**destination**

By specifying the destination address, the target can be limited to the destination address. The destination address uses the same syntax as the source address for IP address or address ranges. The use of source and destination addresses is optional, and the use of a destination addresses is not possible with all elements. This depends on the use of destination addresses, for example, in service entries. You can combine destination and action.

**Elements**
The element can be **only one** of the following element types: **service**, **port**, **protocol**, **masquerade**, **icmp-block**, **forward-port**, and **source-port**.

**service**

The **service** element is one of the **firewalld** provided services. To get a list of the predefined services, enter the following command:

```
~]$ firewall-cmd --get-services
```

If a service provides a destination address, it will conflict with a destination address in the rule and will result in an error. The services using destination addresses internally are mostly services using multicast. The command takes the following form:

```
service name=service_name
```

**port**

The **port** element can either be a single port number or a port range, for example **5060-5062**, followed by the protocol, either as **tcp** or **udp**. The command takes the following form:

```
port port=number_or_range protocol=protocol
```

**protocol**

The **protocol** value can be either a protocol ID number or a protocol name. For allowed **protocol** entries, see `/etc/protocols`. The command takes the following form:

```
protocol value=protocol_name_or_ID
```

**icmp-block**

Use this command to block one or more **ICMP** types. The **ICMP** type is one of the **ICMP** types **firewalld** supports. To get a listing of supported **ICMP** types, enter the following command:

```
~]$ firewall-cmd --get-icmptypes
```

Specifying an action is not allowed here. **icmp-block** uses the action **reject** internally. The command takes the following form:

```
icmp-block name=icmptype_name
```

**masquerade**

Turns on IP masquerading in the rule. A source address can be provided to limit masquerading to this area, but not a destination address. Specifying an action is not allowed here.

**forward-port**

Forward packets from a local port with protocol specified as **tcp** or **udp** to either another port locally, to another machine, or to another port on another machine. The **port** and **to-port** can either be a single port number or a port range. The destination address is a
simple IP address. Specifying an action is not allowed here. The `forward-port` command uses the action `accept` internally. The command takes the following form:

```
forward-port port=number_or_range protocol=protocol /
    to-port=number_or_range to-addr=address
```

**source-port**

Matches the source port of the packet - the port that is used on the origin of a connection attempt. To match a port on current machine, use the `port` element. The `source-port` element can either be a single port number or a port range (for example, 5060-5062) followed by the protocol as `tcp` or `udp`. The command takes the following form:

```
source-port port=number_or_range protocol=protocol
```

**Logging**

Log

Log new connection attempts to the rule with kernel logging, for example, in syslog. You can define a prefix text that will be added to the log message as a prefix. Log level can be one of `emerg`, `alert`, `crit`, `error`, `warning`, `notice`, `info`, or `debug`. The use of log is optional. It is possible to limit logging as follows:

```
log [prefix=prefix text] [level=log level] limit value=rate/duration
```

The rate is a natural positive number [1, ..], with the duration of `s` means seconds, `m` means minutes, `h` means hours, and `d` means days. The maximum limit value is `1/d`, which means at maximum one log entry per day.

Audit

Audit provides an alternative way for logging using audit records sent to the service `auditd`. The audit type can be one of `ACCEPT`, `REJECT`, or `DROP`, but it is not specified after the command `audit` as the audit type will be automatically gathered from the rule action. Audit does not have its own parameters, but limit can be added optionally. The use of audit is optional.

**Action**

accept|reject|drop|mark

An action can be one of `accept`, `reject`, `drop`, or `mark`. The rule can only contain an element or a source. If the rule contains an element, then new connections matching the element will be handled with the action. If the rule contains a source, then everything from the source address will be handled with the action specified.

```
accept | reject [type=reject type] | drop | mark set="mark[/mask]"
```

With `accept`, all new connection attempts will be granted. With `reject`, they will be rejected and their source will get a reject message. The reject type can be set to use another value. With `drop`, all packets will be dropped immediately and no information is sent to the source. With `mark` all packets will be marked with the given `mark` and the optional `mask`. 
5.3.8.4. Using the Rich Rule Log Command

Logging can be done with the **Netfilter** log target and also with the audit target. A new chain is added to all zones with a name in the format “zone\_log”, where `zone` is the zone name. This is processed before the **deny** chain to have the proper ordering. The rules or parts of them are placed in separate chains, according to the action of the rule, as follows:

```
zone_log
  zone_deny
  zone_allow
```

All logging rules will be placed in the “zone\_log” chain, which will be parsed first. All **reject** and **drop** rules will be placed in the “zone\_deny” chain, which will be parsed after the log chain. All **accept** rules will be placed in the “zone\_allow” chain, which will be parsed after the **deny** chain. If a rule contains **log** and also **deny** or **allow** actions, the parts of the rule that specify these actions are placed in the matching chains.

5.3.8.4.1. Using the Rich Rule Log Command Example 1

Enable new **IPv4** and **IPv6** connections for authentication header protocol **AH**:

```
rule protocol value="ah" accept
```

5.3.8.4.2. Using the Rich Rule Log Command Example 2

Allow new **IPv4** and **IPv6** connections for protocol **FTP** and log 1 per minute using audit:

```
rule service name="ftp" log limit value="1/m" audit accept
```

5.3.8.4.3. Using the Rich Rule Log Command Example 3

Allow new **IPv4** connections from address **192.168.0.0/24** for protocol **TFTP** and log 1 per minute using syslog:

```
rule family="ipv4" source address="192.168.0.0/24" service name="tftp" log prefix="tftp" level="info" limit value="1/m" accept
```

5.3.8.4.4. Using the Rich Rule Log Command Example 4

New **IPv6** connections from **1:2:3:4:6::** for protocol **RADIUS** are all rejected and logged at a rate of 3 per minute. New **IPv6** connections from other sources are accepted:

```
rule family="ipv6" source address="1:2:3:4:6::" service name="radius" log prefix="dns" level="info" limit value="3/m" reject
rule family="ipv6" service name="radius" accept
```

5.3.8.4.5. Using the Rich Rule Log Command Example 5

Forward **IPv6** packets received from **1:2:3:4:6::** on port 4011 with protocol **TCP** to **1::2:3:4:7** on port 4012:

```
rule family="ipv6" source address="1:2:3:4:6::" forward-port to-addr="1::2:3:4:7" to-port="4012" protocol="tcp" port="4011"
```
5.3.8.4.6. Using the Rich Rule Log Command Example 6

Whitelist a source address to allow all connections from this source.

```plaintext
rule family="ipv4" source address="192.168.2.2" accept
```

See the `firewalled.richlanguage(5)` man page for more examples.

5.3.9. Firewall Lockdown

Local applications or services are able to change the firewall configuration if they are running as `root` (for example, `libvirt`). With this feature, the administrator can lock the firewall configuration so that either no applications or only applications that are added to the lockdown whitelist are able to request firewall changes. The lockdown settings default to disabled. If enabled, the user can be sure that there are no unwanted configuration changes made to the firewall by local applications or services.

5.3.9.1. Configuring Firewall Lockdown

Using an editor running as `root`, add the following line to the `/etc/firewalld/firewalld.conf` file as follows:

```plaintext
Lockdown=yes
```

Reload the firewall using the following command as `root`:

```plaintext
~$ firewall-cmd --reload
```

Try to enable the `imaps` service in the default zone using the following command as an administrative user (a user in the `wheel` group; usually the first user on the system). You will be prompted for the user password:

```plaintext
~$ firewall-cmd --add-service=imaps
Error: ACCESS_DENIED: lockdown is enabled
```

To enable the use of `firewall-cmd`, enter the following command as `root`:

```plaintext
~$ firewall-cmd --add-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/firewall-cmd*'
```

Add the `--permanent` option if you want to make it persistent.

Reload the firewall as `root`:

```plaintext
~$ firewall-cmd --reload
```

Try to enable the `imaps` service again in the default zone by entering the following command as an administrative user. You will be prompted for the user password:

```plaintext
~$ firewall-cmd --add-service=imaps
```

This time the command succeeds.
5.3.9.2. Configuring IP Set options with the Command-Line Client

IP sets can be used in `firewalld` zones as sources and also as sources in rich rules. It is also possible to use the IP sets created with `firewalld` in a direct rule.

To list the IP sets known to `firewalld` in the permanent environment, use the following command as `root`:

```bash
~# firewall-cmd --permanent --get-ipsets
```

To add a new IP set, use the following command using the permanent environment as `root`:

```bash
~# firewall-cmd --permanent --new-ipset=test --type=hash:net
success
```

The previous command creates a new IP set with the name `test` and the `hash:net` type for IPv4. To create an IP set for use with IPv6, add the `--option=family=inet6` option. To make the new setting effective in the runtime environment, reload `firewalld`. List the new IP set with the following command as `root`:

```bash
~# firewall-cmd --permanent --get-ipsets
test
```

To get more information about the IP set, use the following command as `root`:

```bash
~# firewall-cmd --permanent --info-ipset=test
  test
type: hash:net
  options:
  entries:
```

Note that the IP set does not have any entries at the moment. To add an entry to the `test` IP set, use the following command as `root`:

```bash
~# firewall-cmd --permanent --ipset=test --add-entry=192.168.0.1
success
```

The previous command adds the IP address `192.168.0.1` to the IP set. To get the list of current entries in the IP set, use the following command as `root`:

```bash
~# firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
```

Generate a file containing a list of IP addresses, for example:

```bash
~# cat > iplist.txt
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
EOL
```
The file with the list of IP addresses for an IP set should contain an entry per line. Lines starting with a hash, a semi-colon, or empty lines are ignored.

To add the addresses from the \textit{iplist.txt} file, use the following command as \texttt{root}:

\begin{lstlisting}[language=bash]
~\$ firewall-cmd --permanent --ipset=test --add-entries-from-file=iplist.txt
success
\end{lstlisting}

To see the extended entries list of the IP set, use the following command as \texttt{root}:

\begin{lstlisting}[language=bash]
~\$ firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
\end{lstlisting}

To remove the addresses from the IP set and to check the updated entries list, use the following commands as \texttt{root}:

\begin{lstlisting}[language=bash]
~\$ firewall-cmd --permanent --ipset=test --remove-entries-from-file=iplist.txt
success
~\$ firewall-cmd --permanent --ipset=test --get-entries 192.168.0.1
\end{lstlisting}

You can add the IP set as a source to a zone to handle all traffic coming in from any of the addresses listed in the IP set with a zone. For example, to add the \texttt{test} IP set as a source to the \texttt{drop} zone to drop all packets coming from all entries listed in the \texttt{test} IP set, use the following command as \texttt{root}:

\begin{lstlisting}[language=bash]
~\$ firewall-cmd --permanent --zone=drop --add-source=ipset:test
success
\end{lstlisting}

The \texttt{ipset:} prefix in the source shows \texttt{firewalld} that the source is an IP set and not an IP address or an address range.

Only the creation and removal of IP sets is limited to the permanent environment, all other IP set options can be used also in the runtime environment without the \texttt{--permanent} option.

\section*{5.3.9.3. Configuring Lockdown with the Command-Line Client}

To query whether lockdown is enabled, use the following command as \texttt{root}:

\begin{lstlisting}[language=bash]
~\$ firewall-cmd --query-lockdown
\end{lstlisting}

The command prints \texttt{yes} with exit status 0 if lockdown is enabled. It prints \texttt{no} with exit status 1 otherwise.

To enable lockdown, enter the following command as \texttt{root}:

\begin{lstlisting}[language=bash]
~\$ firewall-cmd --lockdown-on
\end{lstlisting}
To disable lockdown, use the following command as root:

```
~# firewall-cmd --lockdown-off
```

### 5.3.9.4. Configuring Lockdown Whitelist Options with the Command Line

The lockdown whitelist can contain commands, security contexts, users and user IDs. If a command entry on the whitelist ends with an asterisk 
```
* 
```
then all command lines starting with that command will match. If the 
```
* 
```
is not there then the absolute command including arguments must match.

The context is the security (SELinux) context of a running application or service. To get the context of a running application use the following command:

```
~$ ps -e --context
```

That command returns all running applications. Pipe the output through the `grep` tool to get the application of interest. For example:

```
~$ ps -e --context | grep example_program
```

To list all command lines that are on the whitelist, enter the following command as root:

```
~# firewall-cmd --list-lockdown-whitelist-commands
```

To add a command `command` to the whitelist, enter the following command as root:

```
~# firewall-cmd --add-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/command'
```

To remove a command `command` from the whitelist, enter the following command as root:

```
~# firewall-cmd --remove-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/command'
```

To query whether the command `command` is on the whitelist, enter the following command as root:

```
~# firewall-cmd --query-lockdown-whitelist-command='/usr/bin/python -Es /usr/bin/command'
```

The command prints `yes` with exit status `0` if true. It prints `no` with exit status `1` otherwise.

To list all security contexts that are on the whitelist, enter the following command as root:

```
~# firewall-cmd --list-lockdown-whitelist-contexts
```

To add a context `context` to the whitelist, enter the following command as root:

```
~# firewall-cmd --add-lockdown-whitelist-context=context
```

Add the `--permanent` option to make it persistent.
To remove a context context from the whitelist, enter the following command as \texttt{root}:

\begin{verbatim}
  ~\$ firewall-cmd --remove-lockdown-whitelist-context=context
\end{verbatim}

Add the \texttt{--permanent} option to make it persistent.

To query whether the context context is on the whitelist, enter the following command as \texttt{root}:

\begin{verbatim}
  ~\$ firewall-cmd --query-lockdown-whitelist-context=context
\end{verbatim}

Prints \texttt{yes} with exit status \texttt{0}, if true, prints \texttt{no} with exit status \texttt{1} otherwise.

To list all user IDs that are on the whitelist, enter the following command as \texttt{root}:

\begin{verbatim}
  ~\$ firewall-cmd --list-lockdown-whitelist-uids
\end{verbatim}

To add a user ID \texttt{uid} to the whitelist, enter the following command as \texttt{root}:

\begin{verbatim}
  ~\$ firewall-cmd --add-lockdown-whitelist-uid=uid
\end{verbatim}

Add the \texttt{--permanent} option to make it persistent.

To remove a user ID \texttt{uid} from the whitelist, enter the following command as \texttt{root}:

\begin{verbatim}
  ~\$ firewall-cmd --remove-lockdown-whitelist-uid=uid
\end{verbatim}

Add the \texttt{--permanent} option to make it persistent.

To query whether the user ID \texttt{uid} is on the whitelist, enter the following command:

\begin{verbatim}
  ~\$ firewall-cmd --query-lockdown-whitelist-uid=uid
\end{verbatim}

Prints \texttt{yes} with exit status \texttt{0}, if true, prints \texttt{no} with exit status \texttt{1} otherwise.

To list all user names that are on the whitelist, enter the following command as \texttt{root}:

\begin{verbatim}
  ~\$ firewall-cmd --list-lockdown-whitelist-users
\end{verbatim}

To add a user name \texttt{user} to the whitelist, enter the following command as \texttt{root}:

\begin{verbatim}
  ~\$ firewall-cmd --add-lockdown-whitelist-user=user
\end{verbatim}

Add the \texttt{--permanent} option to make it persistent.

To remove a user name \texttt{user} from the whitelist, enter the following command as \texttt{root}:

\begin{verbatim}
  ~\$ firewall-cmd --remove-lockdown-whitelist-user=user
\end{verbatim}

Add the \texttt{--permanent} option to make it persistent.

To query whether the user name \texttt{user} is on the whitelist, enter the following command:

\begin{verbatim}
  ~\$ firewall-cmd --query-lockdown-whitelist-user=user
\end{verbatim}
Prints **yes** with exit status **0**, if true, prints **no** with exit status **1** otherwise.

### 5.3.9.5. Configuring Lockdown Whitelist Options with Configuration Files

The default whitelist configuration file contains the **NetworkManager** context and the default context of **libvirt**. The user ID **0** is also in the list.

```xml
<?xml version="1.0" encoding="utf-8"?>
<whitelist>
    <selinux context="system_u:system_r:NetworkManager_t:s0"/>
    <selinux context="system_u:system_r:virtd_t:s0-s0:c0.c1023"/>
    <user id="0"/>
</whitelist>
```

Following is an example whitelist configuration file enabling all commands for the **firewall-cmd** utility, for a user called **user** whose user ID is **815**:

```xml
<?xml version="1.0" encoding="utf-8"?>
<whitelist>
    <command name="/usr/bin/python -Es /bin/firewall-cmd*"/>
    <selinux context="system_u:system_r:NetworkManager_t:s0"/>
    <user id="815"/>
    <user name="user"/>
</whitelist>
```

This example shows both **user id** and **user name**, but only one option is required. Python is the interpreter and is prepended to the command line. You can also use a specific command, for example:

```
/usr/bin/python /bin/firewall-cmd --lockdown-on
```

In that example only the **--lockdown-on** command will be allowed.

**NOTE**

In Red Hat Enterprise Linux 7, all utilities are placed in the **/usr/bin/** directory and the **/bin/** directory is sym-linked to the **/usr/bin/** directory. In other words, although the path for **firewall-cmd** when run as **root** might resolve to **/bin/firewall-cmd**, **/usr/bin/firewall-cmd** can now be used. All new scripts should use the new location. But be aware that if scripts that run as **root** have been written to use the **/bin/firewall-cmd** path, then that command path must be whitelisted in addition to the **/usr/bin/firewall-cmd** path traditionally used only for non-**root** users.

The "*" at the end of the name attribute of a command means that all commands that start with this string will match. If the "*" is not there then the absolute command including arguments must match.

### 5.3.10. Configuring Logging for Denied Packets

With the **LogDenied** option in the **firewalld**, it is possible to add a simple logging mechanism for denied packets. These are the packets that are rejected or dropped. To change the setting of the logging, edit the **/etc/firewalld/firewalld.conf** file or use
the command-line or GUI configuration tool.

If LogDenied is enabled, logging rules are added right before the reject and drop rules in the INPUT, FORWARD and OUTPUT chains for the default rules and also the final reject and drop rules in zones. The possible values for this setting are: all, unicast, broadcast, multicast, and off. The default setting is off. With the unicast, broadcast, and multicast setting, the pkttype match is used to match the link-layer packet type. With all, all packets are logged.

To list the actual LogDenied setting with firewall-cmd, use the following command as root:

```
~# firewall-cmd --get-log-denied
off
```

To change the LogDenied setting, use the following command as root:

```
~# firewall-cmd --set-log-denied=all
success
```

To change the LogDenied setting with the firewall-config GUI configuration tool, start firewall-config, click the Options menu and select Change Log Denied menuitem. The LogDenied window appears. Select the new LogDenied setting from the drop-down menu and click OK.

5.4. USING THE IPTABLES SERVICE

To use the iptables and ip6tables services instead of firewalld, first disable firewalld by running the following command as root:

```
~# systemctl disable firewalld
~# systemctl stop firewalld
```

Then install the iptables-services package by entering the following command as root:

```
~# yum install iptables-services
```

The iptables-services package contains the iptables service and the ip6tables service.

Then, to start the iptables and ip6tables services, enter the following commands as root:

```
~# systemctl start iptables
~# systemctl start ip6tables
```

To enable the services to start on every system start, enter the following commands:

```
~# systemctl enable iptables
~# systemctl enable ip6tables
```

5.4.1. iptables and IP Sets

The ipset utility is used to administer IP sets in the Linux kernel. An IP set is a framework for storing IP addresses, port numbers, IP and MAC address pairs, or IP address and port number pairs. The sets are indexed in such a way that very fast matching can be made
against a set even when the sets are very large. IP sets enable simpler and more manageable configurations as well as providing performance advantages when using **iptables**. The **iptables** matches and targets referring to sets create references which protect the given sets in the kernel. A set cannot be destroyed while there is a single reference pointing to it.

The use of **ipset** enables **iptables** commands, such as those below, to be replaced by a set:

```bash
~]# iptables -A INPUT -s 10.0.0.0/8 -j DROP
~]# iptables -A INPUT -s 172.16.0.0/12 -j DROP
~]# iptables -A INPUT -s 192.168.0.0/16 -j DROP
```

The set is created as follows:

```bash
~]# ipset create my-block-set hash:net
~]# ipset add my-block-set 10.0.0.0/8
~]# ipset add my-block-set 172.16.0.0/12
~]# ipset add my-block-set 192.168.0.0/16
```

The set is then referenced in an **iptables** command as follows:

```bash
~]# iptables -A INPUT -m set --set my-block-set src -j DROP
```

If the set is used more than once a saving in configuration time is made. If the set contains many entries a saving in processing time is made.

### 5.4.1.1. Using IP Sets with **firewalld**

To use IP sets that are not supported by **firewalld**, a permanent direct rule is required to reference the set, and a custom service must be added to create these IP sets. This service needs to be started before **firewalld** starts, otherwise **firewalld** is not able to add the direct rules using these sets. You can add permanent direct rules with the `/etc/firewalld/direct.xml` file.

To see the list of IP set types supported by **firewalld**, enter the following command as `root`.

```bash
~]# firewall-cmd --get-ipset-types
hash:mac hash:net hash:net,iface hash:net,net hash:net,port
hash:net,port,net
```

**Procedure 5.1. Configuring a Custom Service for an IP Set**

Configure a custom service to create and load the IP set structure before **firewalld** starts.

1. Using an editor running as `root`, create a file as follows:

```bash
~]# vi /etc/systemd/system/ipset_name.service
[Unit]
Description=ipset_name
Before=firewalld.service

[Service]
```
2. Use the IP set permanently in **firewalld**:

```bash
~]# vi /etc/firewalld/direct.xml
<?xml version="1.0" encoding="utf-8"?>
<direct>
  <rule ipv="ipv4" table="filter" chain="INPUT" priority="0">-m set
  --match-set <replaceable>ipset_name</replaceable> src -j DROP</rule>
</direct>
```

3. A **firewalld** reload is required to activate the changes:

```bash
~]# firewall-cmd --reload
```

This will reload the firewall without losing state information (TCP sessions will not be terminated), but service disruption is possible during the reload.

### 5.4.1.2. Installing ipset

To install the **ipset** utility, enter the following command as **root**:

```bash
~]# yum install ipset
```

To see the usage message:

```bash
~]$ ipset --help
ipset v6.11
Usage: ipset [options] COMMAND
output truncated
```

### 5.4.1.3. ipset Commands

The format of the **ipset** command is as follows:

```bash
ipset [options] command [command-options]
```

Where **command** is one of:

- create | add | del | test | destroy | list | save | restore | flush | rename | swap | help | version | -

Allowed **options** are:

- -exist | -output [ plain | save | xml ] | -quiet | -resolve | -sorted | -name | -terse
The **create** command is used to create a new data structure to store a set of IP data. The **add** command adds new data to the set, the data added is referred to as an element of the set.

The **-exist** option suppresses error message if the element already exists, and it has a special role in updating a time out value. To change a time out, use the **ipset add** command and specify all the data for the element again, changing only the time out value as required, and using the **-exist** option.

The **test** option is for testing if the element already exists within a set.

The format of the **create** command is as follows:

```
ipset create set-name type-name [create-options]
```

The **set-name** is a suitable name chosen by the user, the **type-name** is the name of the data structure used to store the data comprising the set. The format of the **type-name** is as follows:

```
method:datatype[,datatype[,datatype]]
```

The allowed methods for storing data are:

```
bitmap | hash | list
```

The allowed data types are:

```
ip | net | mac | port | iface
```

When adding, deleting, or testing entries in a set, the same comma separated data syntax must be used for the data that makes up one entry, or element, in the set. For example:

```
ipset add set-name ipaddr,portnum,ipaddr
```

**NOTE**

A set cannot contain **IPv4** and **IPv6** addresses at the same time. When a set is created it is bound to a family, **inet** for **IPv4** or **inet6** for **IPv6**, and the default is **inet**.

**Example 5.1. Create an IP Set**

To create an IP set consisting of a source IP address, a port, and destination IP address, run a command as follows:

```
~]# ipset create my-set hash:ip,port,ip
```

Once the set is created, entries can be added as follows:

```
~]# ipset add my-set 192.168.1.2,80,192.168.2.2
~]# ipset add my-set 192.168.1.2,443,192.168.2.2
```
The set types have the following optional parameters in common. They must be specified when the set is created in order for them to be used:

- **timeout** — The value given with the `create` command will be the default value for the set created. If a value is given with the `add` command, it will be the initial non-default value for the element.

- **counters** — If the option is given with the `create` command then packet and byte counters are created for every element in the set. If no value is given with the `add` command then the counters start from zero.

- **comment** — If the option is given with the `create` command then a quoted string of text can be passed with the `add` command to document the purpose of the element being added. Note that quotation marks are not allowed within the string, and escape characters will have no effect within IP set.

### Example 5.2. List an IP Set

To list the contents of a specific IP Set, **my-set**, run a command as follows:

```
~]> ipset list my-set  
Name: my-set  
Type: hash:ip,port,ip  
Header: family inet hashsize 1024 maxelem 65536  
Size in memory: 8360  
References: 0  
Members:  
192.168.1.2,tcp:80,192.168.2.2  
192.168.1.2,tcp:443,192.168.2.2  
```

Omit the set name to list all sets.

### Example 5.3. Test the Elements of an IP Set

Listing the contents of large sets is time consuming. You can test for the existence of an element as follows:

```
~]> ipset test my-set 192.168.1.2,80,192.168.2.2  
192.168.1.2,tcp:80,192.168.2.2 is in set my-set.  
```

### 5.4.1.4. IP Set Types

**bitmap:ip**

Stores an IPv4 host address, a network range, or an IPv4 network addresses with the prefix-length in CIDR notation if the `netmask` option is used when the set is created. It can optionally store a timeout value, a counter value, and a comment. It can store up to 65536 entries. The command to create the `bitmap:ip` set has the following format:

```
ipset create set-name range start_ipaddr-end_ipaddr |ipaddr/prefix-length [netmask prefix-length] [timeout value] [counters] [comment]  
```
Example 5.4. Create an IP Set for a Range of Addresses Using a Prefix Length

To create an IP set for a range of addresses using a prefix length, make use of the `bitmap:ip` set type as follows:

```
~]# ipset create my-range bitmap:ip range 192.168.33.0/28
```

Once the set is created, entries can be added as follows:

```
~]# ipset add my-range 192.168.33.1
```

Review the members of the list:

```
~]# ipset list my-range
Name: my-range
Type: bitmap:ip
Header: range 192.168.33.0-192.168.33.15
Size in memory: 84
References: 0
Members:
192.168.33.1
```

To add a range of addresses:

```
~]# ipset add my-range 192.168.33.2-192.168.33.4
```

Review the members of the list:

```
~]# ipset list my-range
Name: my-range
Type: bitmap:ip
Header: range 192.168.33.0-192.168.33.15
Size in memory: 84
References: 0
Members:
192.168.33.1
192.168.33.2
192.168.33.3
192.168.33.4
```

Example 5.5. Create an IP Set for a Range of Addresses Using a Netmask

To create an IP set for a range of address using a netmask, make use of the `bitmap:ip` set type as follows:

```
~]# ipset create my-big-range bitmap:ip range 192.168.124.0-192.168.126.0 netmask 24
```

Once the set is created, entries can be added as follows:

```
~]# ipset add my-big-range 192.168.124.0
```
If you attempt to add an address, the range containing that address will be added:

```bash
~# ipset add my-big-range 192.168.125.150
~# ipset list my-big-range
Name: my-big-range
Type: bitmap:ip
Header: range 192.168.124.0-192.168.126.255 netmask 24
Size in memory: 84
References: 0
Members:
192.168.124.0
192.168.125.0
```

**bitmap:ip,mac**

Stores an IPv4 address and a MAC address as a pair. It can store up to 65536 entries.

```bash
ipset create my-range bitmap:ip,mac range 192.168.1.0/24
```

Once the set is created, entries can be added as follows:

```bash
~# ipset add my-range 192.168.1.1,12:34:56:78:9A:BC
```

**bitmap:port**

Stores a range of ports. It can store up to 65536 entries.

```bash
ipset create my-port-range bitmap:port range 1024-49151
```

The set match and SET target netfilter kernel modules interpret the stored numbers as TCP or UDP port numbers. The protocol can optionally be specified together with the port. The `proto` only needs to be specified if a service name is used, and that name does not exist as a TCP service.

**Example 5.7. Create an IP Set for a Range of Ports**

To create an IP set for a range of ports, make use of the `bitmap:port` set type as follows:

```bash
~# ipset create my-permitted-port-range bitmap:port range 1024-49151
```
Once the set is created, entries can be added as follows:

```
~$ ipset add my-permitted-port-range 5060-5061
```

### hash:ip

Stores a host or network address in the form of a hash. By default, an address specified without a network prefix length is a host address. The all-zero IP address cannot be stored.

```bash
ipset create my-addresses hash:ip
```

The `inet` family is the default, if `family` is omitted addresses will be interpreted as IPv4 addresses. The `hashsize` value is the initial hash size to use and defaults to 1024. The `maxelem` value is the maximum number of elements which can be stored in the set, it defaults to 65536.

The `netfilter` tool searches for a network prefix which is the most specific, it tries to find the smallest block of addresses that match.

### Example 5.8. Create an IP Set for IP Addresses

To create an IP set for IP addresses, make use of the `hash:ip` set type as follows:

```
~$ ipset create my-addresses hash:ip
```

Once the set is created, entries can be added as follows:

```
~$ ipset add my-addresses 10.10.10.0
```

If additional options such as netmask and timeout are required, they must be specified when the set is created. For example:

```
~$ ipset create my-busy-addresses hash:ip maxelem 24 netmask 28 timeout 100
```

The `maxelem` option restricts to total number of elements in the set, thus conserving memory space.

The timeout option means that elements will only exist in the set for the number of seconds specified. For example:

```
~$ ipset add my-busy-addresses 192.168.60.0 timeout 100
```

The following output shows the time counting down:

```
~$ ipset list my-busy-addresses
Name: my-busy-addresses
Type: hash:ip
Header: family inet hashsize 1024 maxelem 24 netmask 28 timeout 100
Size in memory: 8300
References: 0
```
Members:
192.168.60.0 timeout 90
~]# ipset list my-busy-addresses
Name: my-busy-addresses
Type: hash:ip
Header: family inet hashsize 1024 maxelem 24 netmask 20 timeout 100
Size in memory: 8300
References: 0
Members:
192.168.60.0 timeout 83

The element will be removed from the set when the timeout period ends.

See the ipset(8) manual page for more examples.

5.5. ADDITIONAL RESOURCES

The following sources of information provide additional resources regarding firewalld.

5.5.1. Installed Documentation

- firewalld(1) man page — Describes command options for firewalld.
- firewalld.conf(5) man page — Contains information to configure firewalld.
- firewall-cmd(1) man page — Describes command options for the firewalld command-line client.
- firewall-config(1) man page — Describes settings for the firewall-config tool.
- firewall-offline-cmd(1) man page — Describes command options for the firewalld offline command-line client.
- firewalld.icmptype(5) man page — Describes XML configuration files for ICMP filtering.
- firewalld.ipset(5) man page — Describes XML configuration files for the firewalld IP sets.
- firewalld.service(5) man page — Describes XML configuration files for firewalld service.
- firewalld.zone(5) man page — Describes XML configuration files for firewalld zone configuration.
- firewalld.direct(5) man page — Describes the firewalld direct interface configuration file.
- firewalld.lockdown-whitelist(5) man page — Describes the firewalld lockdown whitelist configuration file.
- firewall.richlanguage(5) man page — Describes the firewalld rich language rule syntax.
- `firewalld.zones(5)` man page — General description of what zones are and how to configure them.
- `firewalld.dbus(5)` man page — Describes the D-Bus interface of `firewalld`.

5.5.2. Online Documentation

The Linux Audit system provides a way to track security-relevant information on your system. Based on pre-configured rules, Audit generates log entries to record as much information about the events that are happening on your system as possible. This information is crucial for mission-critical environments to determine the violator of the security policy and the actions they performed. Audit does not provide additional security to your system; rather, it can be used to discover violations of security policies used on your system. These violations can further be prevented by additional security measures such as SELinux.

The following list summarizes some of the information that Audit is capable of recording in its log files:

- Date and time, type, and outcome of an event.
- Sensitivity labels of subjects and objects.
- Association of an event with the identity of the user who triggered the event.
- All modifications to Audit configuration and attempts to access Audit log files.
- All uses of authentication mechanisms, such as SSH, Kerberos, and others.
- Changes to any trusted database, such as /etc/passwd.
- Attempts to import or export information into or from the system.
- Include or exclude events based on user identity, subject and object labels, and other attributes.

The use of the Audit system is also a requirement for a number of security-related certifications. Audit is designed to meet or exceed the requirements of the following certifications or compliance guides:

- Controlled Access Protection Profile (CAPP)
- Labeled Security Protection Profile (LSPP)
- Rule Set Base Access Control (RSBAC)
- National Industrial Security Program Operating Manual (NISPOM)
- Federal Information Security Management Act (FISMA)
- Payment Card Industry — Data Security Standard (PCI-DSS)
- Security Technical Implementation Guides (STIG)

Audit has also been:

- Evaluated by National Information Assurance Partnership (NIAP) and Best Security Industries (BSI).
- Certified to LSPP/CAPP/RSBAC/EAL4+ on Red Hat Enterprise Linux 5.
Use Cases

Watching file access
Audit can track whether a file or a directory has been accessed, modified, executed, or the file’s attributes have been changed. This is useful, for example, to detect access to important files and have an Audit trail available in case one of these files is corrupted.

Monitoring system calls
Audit can be configured to generate a log entry every time a particular system call is used. This can be used, for example, to track changes to the system time by monitoring the `settimeofday`, `clock_adjtime`, and other time-related system calls.

Recording commands run by a user
Audit can track whether a file has been executed, so rules can be defined to record every execution of a particular command. For example, a rule can be defined for every executable in the `/bin` directory. The resulting log entries can then be searched by user ID to generate an audit trail of executed commands per user.

Recording execution of system pathnames
Aside from watching file access which translates a path to an inode at rule invocation, Audit can now watch the execution of a path even if it does not exist at rule invocation, or if the file is replaced after rule invocation. This allows rules to continue to work after upgrading a program executable or before it is even installed.

Recording security events
The `pam_faillock` authentication module is capable of recording failed login attempts. Audit can be set up to record failed login attempts as well, and provides additional information about the user who attempted to log in.

Searching for events
Audit provides the `ausearch` utility, which can be used to filter the log entries and provide a complete audit trail based on a number of conditions.

Running summary reports
The `aureport` utility can be used to generate, among other things, daily reports of recorded events. A system administrator can then analyze these reports and investigate suspicious activity further.

Monitoring network access
The `iptables` and `ebtables` utilities can be configured to trigger Audit events, allowing system administrators to monitor network access.

NOTE
System performance may be affected depending on the amount of information that is collected by Audit.
6.1. AUDIT SYSTEM ARCHITECTURE

The Audit system consists of two main parts: the user-space applications and utilities, and the kernel-side system call processing. The kernel component receives system calls from user-space applications and filters them through one of the three filters: *user*, *task*, or *exit*. Once a system call passes the *exclude* filter, it is sent through one of the aforementioned filters, which, based on the Audit rule configuration, sends it to the Audit daemon for further processing. Figure 6.1, “Audit System Architecture” illustrates this process.

![Figure 6.1. Audit System Architecture](image)

The user-space Audit daemon collects the information from the kernel and creates entries in a log file. Other Audit user-space utilities interact with the Audit daemon, the kernel Audit component, or the Audit log files:

- **audisp** — the Audit dispatcher daemon interacts with the Audit daemon and sends events to other applications for further processing. The purpose of this daemon is to provide a plug-in mechanism so that real-time analytical programs can interact with Audit events.

- **auditctl** — the Audit control utility interacts with the kernel Audit component to manage rules and to control a number of settings and parameters of the event generation process.

- The remaining Audit utilities take the contents of the Audit log files as input and generate output based on user's requirements. For example, the **aureport** utility generates a report of all recorded events.

6.2. INSTALLING THE AUDIT PACKAGES

In order to use the Audit system, you must have the audit packages installed on your system. The audit packages (`audit` and `audit-libs`) are installed by default on Red Hat Enterprise Linux 7. If you do not have these packages installed, execute the following command as the root user to install Audit and the dependencies:

```
~]# yum install audit
```

6.3. CONFIGURING THE AUDIT SERVICE
The Audit daemon can be configured in the `/etc/audit/auditd.conf` file. This file consists of configuration parameters that modify the behavior of the Audit daemon. Empty lines and text following a hash sign (`#`) are ignored. For further details, see the `auditd.conf(5)` man page.

6.3.1. Configuring `auditd` for a Secure Environment

The default `auditd` configuration should be suitable for most environments. However, if your environment has to meet strict security policies, the following settings are suggested for the Audit daemon configuration in the `/etc/audit/auditd.conf` file:

**log_file**

The directory that holds the Audit log files (usually `/var/log/audit/`) should reside on a separate mount point. This prevents other processes from consuming space in this directory, and provides accurate detection of the remaining space for the Audit daemon.

**max_log_file**

Specifies the maximum size of a single Audit log file, must be set to make full use of the available space on the partition that holds the Audit log files.

**max_log_file_action**

Decides what action is taken once the limit set in `max_log_file` is reached, should be set to `keep_logs` to prevent Audit log files from being overwritten.

**space_left**

Specifies the amount of free space left on the disk for which an action that is set in the `space_left_action` parameter is triggered. Must be set to a number that gives the administrator enough time to respond and free up disk space. The `space_left` value depends on the rate at which the Audit log files are generated.

**space_left_action**

It is recommended to set the `space_left_action` parameter to `email` or `exec` with an appropriate notification method.

**admin_space_left**

Specifies the absolute minimum amount of free space for which an action that is set in the `admin_space_left_action` parameter is triggered, must be set to a value that leaves enough space to log actions performed by the administrator.

**admin_space_left_action**

Should be set to `single` to put the system into single-user mode and allow the administrator to free up some disk space.

**disk_full_action**

Specifies an action that is triggered when no free space is available on the partition that holds the Audit log files, must be set to `halt` or `single`. This ensures that the system is either shut down or operating in single-user mode when Audit can no longer log events.

**disk_error_action**
Specifies an action that is triggered in case an error is detected on the partition that holds the Audit log files, must be set to `syslog`, `single`, or `halt`, depending on your local security policies regarding the handling of hardware malfunctions.

**flush**

Should be set to `incremental_async`. It works in combination with the `freq` parameter, which determines how many records can be sent to the disk before forcing a hard synchronization with the hard drive. The `freq` parameter should be set to `100`. These parameters assure that Audit event data is synchronized with the log files on the disk while keeping good performance for bursts of activity.

The remaining configuration options should be set according to your local security policy.

### 6.4. STARTING THE AUDIT SERVICE

Once `auditd` is configured, start the service to collect Audit information and store it in the log files. Execute the following command as the root user to start `auditd`:

```
~ # service auditd start
```

**NOTE**

The `service` command is the only way to correctly interact with the `auditd` daemon. You need to use the `service` command so that the `auid` value is properly recorded. You can use the `systemctl` command only for two actions: `enable` and `status`.

To configure `auditd` to start at boot time using the following command as the root user:

```
~ # systemctl enable auditd
```

A number of other actions can be performed on `auditd` using the `service auditd action` command, where `action` can be one of the following:

- **stop**
  
  Stops `auditd`.

- **restart**
  
  Restarts `auditd`.

- **reload** or **force-reload**
  
  Reloads the configuration of `auditd` from the `/etc/audit/auditd.conf` file.

- **rotate**
  
  Rotates the log files in the `/var/log/audit/` directory.

- **resume**
  
  Resumes logging of Audit events after it has been previously suspended, for example, when there is not enough free space on the disk partition that holds the Audit log files.
condrestart or try-restart
  Restarts auditd only if it is already running.

status
  Displays the running status of auditd.

6.5. DEFINING AUDIT RULES

The Audit system operates on a set of rules that define what is to be captured in the log files. The following types of Audit rules can be specified:

Control rules
  Allow the Audit system's behavior and some of its configuration to be modified.

File system rules
  Also known as file watches, allow the auditing of access to a particular file or a directory.

System call rules
  Allow logging of system calls that any specified program makes.

Audit rules can be set:

- on the command line using the auditctl utility. Note that these rules are not persistent across reboots. For details, see Section 6.5.1, “Defining Audit Rules with auditctl”
- in the /etc/audit/audit.rules file. For details, see Section 6.5.3, “Defining Persistent Audit Rules and Controls in the /etc/audit/audit.rules File”

6.5.1. Defining Audit Rules with auditctl

The auditctl command allows you to control the basic functionality of the Audit system and to define rules that decide which Audit events are logged.

NOTE

All commands which interact with the Audit service and the Audit log files require root privileges. Ensure you execute these commands as the root user. Additionally, CAP_AUDIT_CONTROL is required to set up audit services and CAP_AUDIT_WRITE is required to log user messages.

Defining Control Rules

The following are some of the control rules that allow you to modify the behavior of the Audit system:

-b
  sets the maximum amount of existing Audit buffers in the kernel, for example:

  ~]# auditctl -b 8192
-f
sets the action that is performed when a critical error is detected, for example:

```bash
~$ auditctl -f 2
```

The above configuration triggers a kernel panic in case of a critical error.

-e
enables and disables the Audit system or locks its configuration, for example:

```bash
~$ auditctl -e 2
```

The above command locks the Audit configuration.

-r
sets the rate of generated messages per second, for example:

```bash
~$ auditctl -r 0
```

The above configuration sets no rate limit on generated messages.

-s
reports the status of the Audit system, for example:

```bash
~$ auditctl -s
AUDIT_STATUS: enabled=1 flag=2 pid=0 rate_limit=0 backlog_limit=8192 lost=259 backlog=0
```

-l
lists all currently loaded Audit rules, for example:

```bash
~$ auditctl -l
-w /etc/passwd -p wa -k passwd_changes
-w /etc/selinux -p wa -k selinux_changes
-w /sbin/insmod -p x -k module_insertion
```

-D
deletes all currently loaded Audit rules, for example:

```bash
~$ auditctl -D
No rules
```

### Defining File System Rules
To define a file system rule, use the following syntax:

```bash
auditctl -w path_to_file -p permissions -k key_name
```

where:
- `path_to_file` is the file or directory that is audited.
- `permissions` are the permissions that are logged:
  - `r` — read access to a file or a directory.
  - `w` — write access to a file or a directory.
  - `x` — execute access to a file or a directory.
  - `a` — change in the file's or directory's attribute.
- `key_name` is an optional string that helps you identify which rule or a set of rules generated a particular log entry.

### Example 6.1. File System Rules

To define a rule that logs all write access to, and every attribute change of, the `/etc/passwd` file, execute the following command:

```
~]$ auditctl -w /etc/passwd -p wa -k passwd_changes
```

Note that the string following the `-k` option is arbitrary.

To define a rule that logs all write access to, and every attribute change of, all the files in the `/etc/selinux/` directory, execute the following command:

```
~]$ auditctl -w /etc/selinux/ -p wa -k selinux_changes
```

To define a rule that logs the execution of the `/sbin/insmod` command, which inserts a module into the Linux kernel, execute the following command:

```
~]$ auditctl -w /sbin/insmod -p x -k module_insertion
```

### Defining System Call Rules

To define a system call rule, use the following syntax:

```
auditctl -a action,filter -S system_call -F field=value -k key_name
```

where:

- `action` and `filter` specify when a certain event is logged. `action` can be either `always` or `never`. `filter` specifies which kernel rule-matching filter is applied to the event. The rule-matching filter can be one of the following: `task`, `exit`, `user`, and `exclude`. For more information about these filters, see the beginning of Section 6.1, “Audit System Architecture”.

- `system_call` specifies the system call by its name. A list of all system calls can be found in the `/usr/include/asm/unstd_64.h` file. Several system calls can be grouped into one rule, each specified after its own `-S` option.
- **field=value** specifies additional options that further modify the rule to match events based on a specified architecture, group ID, process ID, and others. For a full listing of all available field types and their values, see the `auditctl`(8) man page.

- **key_name** is an optional string that helps you identify which rule or a set of rules generated a particular log entry.

**Example 6.2. System Call Rules**

To define a rule that creates a log entry every time the **adjtimex** or **settimeofday** system calls are used by a program, and the system uses the 64-bit architecture, execute the following command:

```
~# auditctl -a always,exit -F arch=b64 -S adjtimex -S settimeofday -k time_change
```

To define a rule that creates a log entry every time a file is deleted or renamed by a system user whose ID is 1000 or larger, execute the following command:

```
~# auditctl -a always,exit -S unlink -S unlinkat -S rename -S renameat -F auid>=1000 -F auid!=4294967295 -k delete
```

Note that the **-F auid!=4294967295** option is used to exclude users whose login UID is not set.

It is also possible to define a file system rule using the system call rule syntax. The following command creates a rule for system calls that is analogous to the **-w /etc/shadow -p wa** file system rule:

```
~# auditctl -a always,exit -F path=/etc/shadow -F perm=wa
```

### 6.5.2. Defining Executable File Rules

To define an executable file rule, use the following syntax:

```
auditctl  -a action,filter [ -F arch=cpu -S system_call] -F exe=path_to_executable_file -F key_name
```

where:

- **action** and **filter** specify when a certain event is logged. **action** can be either **always** or **never**. **filter** specifies which kernel rule-matching filter is applied to the event. The rule-matching filter can be one of the following: **task**, **exit**, **user**, and **exclude**. For more information about these filters, see the beginning of **Section 6.1, “Audit System Architecture”**.

- **system_call** specifies the system call by its name. A list of all system calls can be found in the `/usr/include/asm/unistd_64.h` file. Several system calls can be grouped into one rule, each specified after its own **-S** option.

- **path_to_executable_file** is the absolute path to the executable file that is audited.
key_name is an optional string that helps you identify which rule or a set of rules generated a particular log entry.

**Example 6.3. Executable File Rules**

To define a rule that logs all execution of the `/bin/id` program, execute the following command:

```
~]# auditctl -F exe=/bin/id -S execve -k execution_bin_id
```

**6.5.3. Defining Persistent Audit Rules and Controls in the `/etc/audit/audit.rules` File**

To define Audit rules that are persistent across reboots, you must either directly include them in the `/etc/audit/audit.rules` file or use the `augenrules` program that reads rules located in the `/etc/audit/rules.d/` directory. The `/etc/audit/audit.rules` file uses the same `auditctl` command line syntax to specify the rules. Empty lines and text following a hash sign (`#`) are ignored.

The `auditctl` command can also be used to read rules from a specified file using the `-R` option, for example:

```
~]# auditctl -R /usr/share/doc/audit/rules/30-stig.rules
```

**Defining Control Rules**

A file can contain only the following control rules that modify the behavior of the Audit system: `-b`, `-D`, `-e`, `-f`, `-r`, `--loginuid-immutable`, and `--backlog_wait_time`. For more information on these options, see the section called “Defining Control Rules”.

**Example 6.4. Control Rules in audit.rules**

```
# Delete all previous rules
-D

# Set buffer size
-b 8192

# Make the configuration immutable -- reboot is required to change audit rules
-e 2

# Panic when a failure occurs
-f 2

# Generate at most 100 audit messages per second
-r 100

# Make login UID immutable once it is set (may break containers)
--loginuid-immutable 1
```

**Defining File System and System Call Rules**
File system and system call rules are defined using the `auditctl` syntax. The examples in Section 6.5.1, “Defining Audit Rules with `auditctl`” can be represented with the following rules file:

```
-w /etc/passwd -p wa -k passwd_changes
-w /etc/selinux/ -p wa -k selinux_changes
-w /sbin/insmod -p x -k module_insertion

-a always,exit -F arch=b64 -S adjtimex -S settimeofday -k time_change
-a always,exit -S unlink -S unlinkat -S rename -S renameat -F auid>=1000
-F auid!=4294967295 -k delete
```

### Preconfigured Rules Files

In the `/usr/share/doc/audit/rules/` directory, the audit package provides a set of pre-configured rules files according to various certification standards:

- **30-nispom.rules** — Audit rule configuration that meets the requirements specified in the Information System Security chapter of the National Industrial Security Program Operating Manual.

- **30-pci-dss-v31.rules** — Audit rule configuration that meets the requirements set by Payment Card Industry Data Security Standard (PCI DSS) v3.1.

- **30-stig.rules** — Audit rule configuration that meets the requirements set by Security Technical Implementation Guides (STIG).

To use these configuration files, create a backup of your original `/etc/audit/audit.rules` file and copy the configuration file of your choice over the `/etc/audit/audit.rules` file:

```
~ # cp /etc/audit/audit.rules /etc/audit/audit.rules_backup
~ # cp /usr/share/doc/audit/rules/30-stig.rules /etc/audit/audit.rules
```

**NOTE**

The Audit rules have a numbering scheme that allows them to be ordered. To learn more about the naming scheme, see the `/usr/share/doc/audit/rules/README-rules` file.

### Using `augenrules` to Define Persistent Rules

The `augenrules` script reads rules located in the `/etc/audit/rules.d/` directory and compiles them into an `audit.rules` file. This script processes all files that end in `.rules` in a specific order based on their natural sort order. The files in this directory are organized into groups with following meanings:

- **10** - Kernel and auditctl configuration
- **20** - Rules that could match general rules but you want a different match
- **30** - Main rules
- **40** - Optional rules
50 - Server-specific rules
70 - System local rules
90 - Finalize (immutable)

The rules are not meant to be used all at once. They are pieces of a policy that should be thought out and individual files copied to `/etc/audit/rules.d/`. For example, to set a system up in the STIG configuration, copy rules 10-base-config, 30-stig, 31-privileged, and 99-finalize.

Once you have the rules in the `/etc/audit/rules.d/` directory, load them by running the `augenrules` script with the `--load` directive:

```
~# augenrules --load
augenrules --load No rules
enabled 1
failure 1
pid 634
rate_limit 0
backlog_limit 8192
lost 0
backlog 0
enabled 1
failure 1
pid 634
rate_limit 0
backlog_limit 8192
lost 0
backlog 1
```

For more information on the Audit rules and the `augenrules` script, see the `audit.rules(8)` and `augenrules(8)` man pages.

### 6.6. UNDERSTANDING AUDIT LOG FILES

By default, the Audit system stores log entries in the `/var/log/audit/audit.log` file; if log rotation is enabled, rotated `audit.log` files are stored in the same directory.

The following Audit rule logs every attempt to read or modify the `/etc/ssh/sshd_config` file:

```
-w /etc/ssh/sshd_config -p warx -k sshd_config
```

If the `auditd` daemon is running, for example, using the following command creates a new event in the Audit log file:

```
~$ cat /etc/ssh/sshd_config
```

This event in the `audit.log` file looks as follows:

```
type=SYSCALL msg=audit(1364481363.243:24287): arch=c000003e syscall=2 success=no exit=-13 a0=7fffd19c5592 a1=0 a2=7fffd19c4b50 a3=a items=1 ppid=2686 pid=3538 auid=1000 uid=1000 gid=1000 euid=1000 suid=1000
```
The above event consists of four records, which share the same time stamp and serial number. Records always start with the `type=` keyword. Each record consists of several `name=value` pairs separated by a white space or a comma. A detailed analysis of the above event follows:

**First Record**

`type=SYSCALL`

The `type` field contains the type of the record. In this example, the `SYSCALL` value specifies that this record was triggered by a system call to the kernel.

For a list of all possible type values and their explanations, see Section B.2, “Audit Record Types”.

`msg=audit(1364481363.243:24287):`

The `msg` field records:

- a time stamp and a unique ID of the record in the form `audit(time_stamp:ID)`. Multiple records can share the same time stamp and ID if they were generated as part of the same Audit event. The time stamp is using the Unix time format - seconds since 00:00:00 UTC on 1 January 1970.

- various event-specific `name=value` pairs provided by the kernel or user space applications.

`arch=c000003e`

The `arch` field contains information about the CPU architecture of the system. The value, `c000003e`, is encoded in hexadecimal notation. When searching Audit records with the `ausearch` command, use the `-i` or `--interpret` option to automatically convert hexadecimal values into their human-readable equivalents. The `c000003e` value is interpreted as x86_64.

`syscall=2`

The `syscall` field records the type of the system call that was sent to the kernel. The value, 2, can be matched with its human-readable equivalent in the `/usr/include/asm/unistd_64.h` file. In this case, 2 is the open system call. Note that the `ausyscall` utility allows you to convert system call numbers to their human-readable equivalents. Use the `ausyscall --dump` command to display a listing of all system calls along with their numbers. For more information, see the `ausyscall(8)` man page.

`success=no`

The `success` field records whether the system call recorded in that particular event succeeded or failed. In this case, the call did not succeed.
exit=-13

The `exit` field contains a value that specifies the exit code returned by the system call. This value varies for different system call. You can interpret the value to its human-readable equivalent with the following command:

```
~]# ausearch --interpret --exit -13
```

Note that the previous example assumes that your Audit log contains an event that failed with exit code `-13`.

`a0=7fffd19c5592, a1=0, a2=7fffd19c5592, a3=a`

The `a0` to `a3` fields record the first four arguments, encoded in hexadecimal notation, of the system call in this event. These arguments depend on the system call that is used; they can be interpreted by the `ausearch` utility.

`items=1`

The `items` field contains the number of auxiliary records that follow the syscall record.

`ppid=2686`

The `ppid` field records the Parent Process ID (PPID). In this case, `2686` was the PPID of the parent process such as `bash`.

`pid=3538`

The `pid` field records the Process ID (PID). In this case, `3538` was the PID of the `cat` process.

`auid=1000`

The `auid` field records the Audit user ID, that is the loginuid. This ID is assigned to a user upon login and is inherited by every process even when the user's identity changes, for example, by switching user accounts with the `su - john` command.

`uid=1000`

The `uid` field records the user ID of the user who started the analyzed process. The user ID can be interpreted into user names with the following command: `ausearch -i --uid UID`.

`gid=1000`

The `gid` field records the group ID of the user who started the analyzed process.

`euid=1000`

The `euid` field records the effective user ID of the user who started the analyzed process.

`suid=1000`

The `suid` field records the set user ID of the user who started the analyzed process.

`fsuid=1000`

The `fsuid` field records the file system user ID of the user who started the analyzed process.
**egid=1000**
The **egid** field records the effective group ID of the user who started the analyzed process.

**sgid=1000**
The **sgid** field records the set group ID of the user who started the analyzed process.

**fsgid=1000**
The **fsgid** field records the file system group ID of the user who started the analyzed process.

**tty=pts0**
The **tty** field records the terminal from which the analyzed process was invoked.

**ses=1**
The **ses** field records the session ID of the session from which the analyzed process was invoked.

**comm="cat"**
The **comm** field records the command-line name of the command that was used to invoke the analyzed process. In this case, the **cat** command was used to trigger this Audit event.

**exe="/bin/cat"**
The **exe** field records the path to the executable that was used to invoke the analyzed process.

**subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023**
The **subj** field records the SELinux context with which the analyzed process was labeled at the time of execution.

**key="sshd_config"**
The **key** field records the administrator-defined string associated with the rule that generated this event in the Audit log.

---

**Second Record**

**type=CWD**
In the second record, the **type** field value is **CWD** — current working directory. This type is used to record the working directory from which the process that invoked the system call specified in the first record was executed.

The purpose of this record is to record the current process's location in case a relative path winds up being captured in the associated PATH record. This way the absolute path can be reconstructed.

**msg=audit(1364481363.243:24287)**
The `msg` field holds the same time stamp and ID value as the value in the first record. The time stamp is using the Unix time format - seconds since 00:00:00 UTC on 1 January 1970.

cwd="/home/user_name"

The `cwd` field contains the path to the directory in which the system call was invoked.

**Third Record**

type=PATH

In the third record, the `type` field value is `PATH`. An Audit event contains a `PATH`-type record for every path that is passed to the system call as an argument. In this Audit event, only one path (`/etc/ssh/sshd_config`) was used as an argument.

`msg=audit(1364481363.243:24287):`

The `msg` field holds the same time stamp and ID value as the value in the first and second record.

`item=0`

The `item` field indicates which item, of the total number of items referenced in the `SYSCALL` type record, the current record is. This number is zero-based; a value of 0 means it is the first item.

`name="/etc/ssh/sshd_config"

The `name` field records the full path of the file or directory that was passed to the system call as an argument. In this case, it was the `/etc/ssh/sshd_config` file.

`inode=409248`

The `inode` field contains the inode number associated with the file or directory recorded in this event. The following command displays the file or directory that is associated with the 409248 inode number:

```
~]# find / -inum 409248 -print
/etc/ssh/sshd_config
```

dev=fd:00

The `dev` field specifies the minor and major ID of the device that contains the file or directory recorded in this event. In this case, the value represents the `/dev/fd/0` device.

`mode=0100600`

The `mode` field records the file or directory permissions, encoded in numerical notation as returned by the `stat` command in the `st_mode` field. See the `stat(2)` man page for more information. In this case, 0100600 can be interpreted as `-rw-------`, meaning that only the root user has read and write permissions to the `/etc/ssh/sshd_config` file.

`ouid=0`

The `ouid` field records the object owner's user ID.
The `ogid` field records the object owner's group ID.

The `rdev` field contains a recorded device identifier for special files only. In this case, it is not used as the recorded file is a regular file.

The `obj` field records the SELinux context with which the recorded file or directory was labeled at the time of execution.

### Fourth Record

**type=PROCTITLE**

The `type` field contains the type of the record. In this example, the `PROCTITLE` value specifies that this record gives the full command-line that triggered this Audit event, triggered by a system call to the kernel.

The `proctitle` field records the full command-line of the command that was used to invoke the analyzed process. The field is encoded in hexadecimal notation to not allow the user to influence the Audit log parser. The text decodes to the command that triggered this Audit event. When searching Audit records with the `ausearch` command, use the `-i` or `--interpret` option to automatically convert hexadecimal values into their human-readable equivalents. The `636174002F6574632F7373682F737368645F636F6E666967` value is interpreted as `x86_64`.

The Audit event analyzed above contains only a subset of all possible fields that an event can contain. For a list of all event fields and their explanation, see Section B.1, “Audit Event Fields”. For a list of all event types and their explanation, see Section B.2, “Audit Record Types”.

### Example 6.6. Additional audit.log Events

The following Audit event records a successful start of the `auditd` daemon. The `ver` field shows the version of the Audit daemon that was started.

```
    type=DAEMON_START msg=audit(1363713609.192:5426): auditd start, ver=2.2
    format=raw kernel=2.6.32-358.2.1.el6.x86_64 auid=1000 pid=4979
    subj=unconfined_u:system_r:auditd_t:s0 res=success
```

The following Audit event records a failed attempt of user with UID of 1000 to log in as the root user.

```
    type=USER_AUTH msg=audit(1364475353.159:24270): user pid=3280 uid=1000
    auid=1000 ses=1 subj=unconfined_u:unconfined_r:unconfined_t:s0-
    s0:c0.c1023 msg='op=PAM:authentication acct="root" exe="/bin/su"
    hostname=? addr=? terminal=pts/0 res=failed'
```
6.7. SEARCHING THE AUDIT LOG FILES

The `ausearch` utility allows you to search Audit log files for specific events. By default, `ausearch` searches the `/var/log/audit/audit.log` file. You can specify a different file using the `ausearch` options `-if file_name` command. Supplying multiple options in one `ausearch` command is equivalent to using the `AND` operator between field types and the `OR` operator between multiple instances of the same field type.

**Example 6.7. Using `ausearch` to Search Audit Log Files**

To search the `/var/log/audit/audit.log` file for failed login attempts, use the following command:

```
~]# ausearch --message USER_LOGIN --success no --interpret
```

To search for all account, group, and role changes, use the following command:

```
~]# ausearch -m ADD_USER -m DEL_USER -m ADD_GROUP -m USER_CHAUGHTOK -m DEL_GROUP -m CHGRP_ID -m ROLE_ASSIGN -m ROLE_REMOVE -i
```

To search for all logged actions performed by a certain user, using the user's login ID (`auid`), use the following command:

```
~]# ausearch -ua 1000 -i
```

To search for all failed system calls from yesterday up until now, use the following command:

```
~]# ausearch --start yesterday --end now -m SYSCALL -sv no -i
```

For a full listing of all `ausearch` options, see the `ausearch(8)` man page.

6.8. CREATING AUDIT REPORTS

The `aureport` utility allows you to generate summary and columnar reports on the events recorded in Audit log files. By default, all `audit.log` files in the `/var/log/audit/` directory are queried to create the report. You can specify a different file to run the report against using the `aureport` options `-if file_name` command.

**Example 6.8. Using `aureport` to Generate Audit Reports**

To generate a report for logged events in the past three days excluding the current example day, use the following command:

```
~]# aureport --start 04/08/2013 00:00:00 --end 04/11/2013 00:00:00
```

To generate a report of all executable file events, use the following command:

```
~]# aureport -x
```
To generate a summary of the executable file event report above, use the following command:

```
~\# aureport -x --summary
```

To generate a summary report of failed events for all users, use the following command:

```
~\# aureport -u --failed --summary -i
```

To generate a summary report of all failed login attempts per each system user, use the following command:

```
~\# aureport --login --summary -i
```

To generate a report from an `ausearch` query that searches all file access events for user ID **1000**, use the following command:

```
~\# ausearch --start today --loginuid 1000 --raw | aureport -f --summary
```

To generate a report of all Audit files that are queried and the time range of events they include, use the following command:

```
~\# aureport -t
```

For a full listing of all `aureport` options, see the `aureport(8)` man page.

### 6.9. ADDITIONAL RESOURCES

For more information about the Audit system, see the following sources.

**Online Sources**

- The Linux Audit Documentation Project page: [https://github.com/linux-audit/audit-documentation/wiki](https://github.com/linux-audit/audit-documentation/wiki).

**Installed Documentation**

Documentation provided by the audit package can be found in the `/usr/share/doc/audit/` directory.

**Manual Pages**

- `audispd.conf(5)`
- `auditd.conf(5)`
- `ausearch-expression(5)`
- `audit.rules(7)`
- `audispd(8)`
- `auditctl(8)`
- auditd(8)
- aulast(8)
- aulastlog(8)
- aureport(8)
- ausearch(8)
- ausyscall(8)
- autrace(8)
- auvirt(8)
7.1. SECURITY COMPLIANCE IN RED HAT ENTERPRISE LINUX

A **compliance audit** is a process of figuring out whether a given object follows all the rules written out in a compliance policy. The **compliance policy** is defined by security professionals who specify desired settings, often in the form of a checklist, that are to be used in the computing environment.

The compliance policy can vary substantially across organizations and even across different systems within the same organization. Differences among these policies are based on the purpose of these systems and its importance for the organization. The custom software settings and deployment characteristics also raise a need for custom policy checklists.

Red Hat Enterprise Linux provides tools that allow for fully automated compliance audit. These tools are based on the Security Content Automation Protocol (SCAP) standard and are designed for automated tailoring of compliance policies.

**Security Compliance Tools Supported on Red Hat Enterprise Linux 7**

- **SCAP Workbench** — The `scap-workbench` graphical utility is designed to perform configuration and vulnerability scans on a single local or remote system. It can be also used to generate security reports based on these scans and evaluations.

- **OpenSCAP** — The `oscap` command-line utility is designed to perform configuration and vulnerability scans on a local system, to validate security compliance content, and to generate reports and guides based on these scans and evaluations.

- **Script Check Engine (SCE)** — **SCE** is an extension to the SCAP protocol that allows administrators to write their security content using a scripting language, such as Bash, Python, or Ruby. The **SCE** extension is provided in the `openscap-engine-sce` package.

- **SCAP Security Guide (SSG)** — The `scap-security-guide` package provides the latest collection of security policies for Linux systems. The guidance consists of a catalog of practical hardening advice, linked to government requirements where applicable. The project bridges the gap between generalized policy requirements and specific implementation guidelines.

If you require performing automated compliance audits on multiple systems remotely, you can utilize OpenSCAP solution for Red Hat Satellite. For more information see Section 7.7, “Using OpenSCAP with Red Hat Satellite” and Section 7.9, “Additional Resources”.

7.2. DEFINING COMPLIANCE POLICY

The security or compliance policy is rarely written from scratch. **ISO 27000** standard series, derivative works, and other sources provide security policy templates and practice recommendations that should be helpful to start with. However, organizations building theirs information security program need to amend the policy templates to align with their needs. The policy template should be chosen on the basis of its relevancy to the company.
environment and then the template has to be adjusted because either the template contains build-in assumptions which cannot be applied to the organization, or the template explicitly requires that certain decisions have to be made.

Red Hat Enterprise Linux auditing capabilities are based on the Security Content Automation Protocol (SCAP) standard. SCAP is a synthesis of interoperable specifications that standardize the format and nomenclature by which software flaw and security configuration information is communicated, both to machines and humans. SCAP is a multi-purpose framework of specifications that supports automated configuration, vulnerability and patch checking, technical control compliance activities, and security measurement.

In other words, SCAP is a vendor-neutral way of expressing security policy, and as such it is widely used in modern enterprises. SCAP specifications create an ecosystem where the format of security content is well known and standardized while the implementation of the scanner or policy editor is not mandated. Such a status enables organizations to build their security policy (SCAP content) once, no matter how many security vendors do they employ.

The latest version of SCAP includes several underlying standards. These components are organized into groups according to their function within SCAP as follows:

**SCAP Components**

- **Languages** — This group consists of SCAP languages that define standard vocabularies and conventions for expressing compliance policy.
  
  - The eXtensible Configuration Checklist Description Format (XCCDF) — A language designed to express, organize, and manage security guidance.
  
  - Open Vulnerability and Assessment Language (OVAL) — A language developed to perform logical assertion about the state of the scanned system.
  
  - Open Checklist Interactive Language (OCIL) — A language designed to provide a standard way to query users and interpret user responses to the given questions.
  
  - Asset Identification (AI) — A language developed to provide a data model, methods, and guidance for identifying security assets.
  
  - Asset Reporting Format (ARF) — A language designed to express the transport format of information about collected security assets and the relationship between assets and security reports.

- **Enumerations** — This group includes SCAP standards that define naming format and an official list or dictionary of items from certain security-related areas of interest.
  
  - Common Configuration Enumeration (CCE) — An enumeration of security-relevant configuration elements for applications and operating systems.
  
  - Common Platform Enumeration (CPE) — A structured naming scheme used to identify information technology (IT) systems, platforms, and software packages.
  
  - Common Vulnerabilities and Exposures (CVE) — A reference method to a collection of publicly known software vulnerabilities and exposures.

- **Metrics** — This group comprises of frameworks to identify and evaluate security risks.
Common Configuration Scoring System (CCSS) — A metric system to evaluate security-relevant configuration elements and assign them scores in order to help users to prioritize appropriate response steps.

Common Vulnerability Scoring System (CVSS) — A metric system to evaluate software vulnerabilities and assign them scores in order to help users prioritize their security risks.

Integrity — An SCAP specification to maintain integrity of SCAP content and scan results.

Trust Model for Security Automation Data (TMSAD) — A set of recommendations explaining usage of existing specification to represent signatures, hashes, key information, and identity information in context of an XML file within a security automation domain.

Each of the SCAP components has its own XML-based document format and its XML name space. A compliance policy expressed in SCAP can either take a form of a single OVAL definition XML file, data stream file, single zip archive, or a set of separate XML files containing an XCCDF file that represents a policy checklist.

### 7.2.1. The XCCDF File Format

The XCCDF language is designed to support information interchange, document generation, organizational and situational tailoring, automated compliance testing, and compliance scoring. The language is mostly descriptive and does not contain any commands to perform security scans. However, an XCCDF document can refer to other SCAP components, and as such it can be used to craft a compliance policy that is portable among all the target platforms with the exception of the related assessment documents (OVAL, OCIL).

The common way to represent a compliance policy is a set of XML files where one of the files is an XCCDF checklist. This XCCDF file usually points to the assessment resources, multiple OVAL, OCIL and the Script Check Engine (SCE) files. Furthermore, the file set can contain a CPE dictionary file and an OVAL file defining objects for this dictionary.

Being an XML-based language, the XCCDF defines and uses a vast selection of XML elements and attributes. The following list briefly introduces the main XCCDF elements; for more details about XCCDF, consult the NIST Interagency Report 7275 Revision 4

**Main XML Elements of the XCCDF Document**

- `<xccdf:Benchmark>` — This is a root element that encloses the whole XCCDF document. It may also contain checklist metadata, such as a title, description, list of authors, date of the latest modification, and status of the checklist acceptance.

- `<xccdf:Rule>` — This is a key element that represents a checklist requirement and holds its description. It may contain child elements that define actions verifying or enforcing compliance with the given rule or modify the rule itself.

- `<xccdf:Value>` — This key element is used for expressing properties of other XCCDF elements within the benchmark.

- `<xccdf:Group>` — This element is used to organize an XCCDF document to structures with the same context or requirement domains by gathering the `<xccdf:Rule>`, `<xccdf:Value>`, and `<xccdf:Group>` elements.
<xccdf:Profile> — This element serves for a named tailoring of the XCCDF benchmark. It allows the benchmark to hold several different tailorings. <xccdf:Profile> utilizes several selector elements, such as <xccdf:select> or <xccdf:refine-rule>, to determine which elements are going to be modified and processed while it is in effect.

<xccdf:Tailoring> — This element allows defining the benchmark profiles outside the benchmark, which is sometimes desirable for manual tailoring of the compliance policy.

<xccdf:TestResult> — This element serves for keeping the scan results for the given benchmark on the target system. Each <xccdf:TestResult> should refer to the profile that was used to define the compliance policy for the particular scan and it should also contain important information about the target system that is relevant for the scan.

<xccdf:rule-result> — This is a child element of <xccdf:TestResult> that is used to hold the result of applying a specific rule from the benchmark to the target system.

<xccdf:fix> — This is a child element of <xccdf:Rule> that serves for remediation of the target system that is not compliant with the given rule. It can contain a command or script that is run on the target system in order to bring the system into compliance the rule.

<xccdf:check> — This is a child element of <xccdf:Rule> that refers to an external source which defines how to evaluate the given rule.

<xccdf:select> — This is a selector element that is used for including or excluding the chosen rules or groups of rules from the policy.

<xccdf:set-value> — This is a selector element that is used for overwriting the current value of the specified <xccdf:Value> element without modifying any of its other properties.

<xccdf:refine-value> — This is a selector element that is used for specifying constraints of the particular <xccdf:Value> element during policy tailoring.

<xccdf:refine-rule> — This selector element allows overwriting properties of the selected rules.

Example 7.1. An Example of an XCCDF Document

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Benchmark xmlns="http://checklists.nist.gov/xccdf/1.2"
   id="xccdf_com.example.www_benchmark_test">
  <status>incomplete</status>
  <version>0.1</version>
  <Profile id="xccdf_com.example.www_profile_1">
    <title>Profile title is compulsory</title>
    <select idref="xccdf_com.example.www_group_1"
      selected="true"/>
    <select idref="xccdf_com.example.www_rule_1"
      selected="true"/>
    <refine-value idref="xccdf_com.example.www_value_1"
      selector="telnet service"/>
  </Profile>
</Benchmark>
```
7.2.2. The OVAL File Format

The Open Vulnerability Assessment Language (OVAL) is the essential and oldest component of SCAP. The main goal of the OVAL standard is to enable interoperability among security products. That is achieved by standardization of the following three domains:

1. Representation of the target system configuration.
2. Analysis of the target system for the presence of a particular machine state.
3. Reporting the results of the comparison between the specified machine state and the observed machine state.

Unlike other tools or custom scripts, the OVAL language describes a desired state of resources in a declarative manner. The OVAL language code is never executed directly, but by means of an OVAL interpreter tool called scanner. The declarative nature of OVAL ensures that the state of the assessed system is not accidentally modified, which is important because security scanners are often run with the highest possible privileges.
OVAL specification is open for public comments and contribution and various IT companies collaborate with the MITRE Corporation, federally funded not-for-profit organization. The OVAL specification is continuously evolving and different editions are distinguished by a version number. The current version 5.11.1 was released in April 2015.

Like all other SCAP components, OVAL is based on XML. The OVAL standard defines several document formats. Each of them includes different kind of information and serves a different purpose.

**The OVAL Document Formats**

- The *OVAL Definitions* format is the most common OVAL file format that is used directly for system scans. The OVAL Definitions document describes the desired state of the target system.

- The *OVAL Variables* format defines variables used to amend the OVAL Definitions document. The OVAL Variables document is typically used in conjunction with the OVAL Definitions document to tailor the security content for the target system at runtime.

- The *OVAL System Characteristics* format holds information about the assessed system. The OVAL System Characteristics document is typically used to compare the actual state of the system against the expected state defined by an OVAL Definitions document.

- The *OVAL Results* is the most comprehensive OVAL format that is used to report results of the system evaluation. The OVAL Results document typically contains copy of the evaluated OVAL definitions, bound OVAL variables, OVAL system characteristics, and results of tests that are computed based on comparison of the system characteristics and the definitions.

- The *OVAL Directives* format is used to tailor verbosity of an OVAL Result document by either including or excluding certain details.

- The *OVAL Common Model* format contains definitions of constructs and enumerations used in several other OVAL schemes. It is used to reuse OVAL definitions in order to avoid duplications across multiple documents.

The OVAL Definitions document consists of a set of configuration requirements where each requirement is defined in the following five basic sections: *definitions*, *tests*, *objects*, *states*, and *variables*. The elements within the definitions section describe which of the tests shall be fulfilled to satisfy the given definition. The test elements link objects and states together. During the system evaluation, a test is considered passed when a resource of the assessed system that is denoted by the given object element corresponds with the given state element. The variables section defines external variables which may be used to adjust elements from the states section. Besides these sections, the OVAL Definitions document typically contains also the *generator* and *signature* sections. The *generator* section holds information about the document origin and various additional information related to its content.

Each element from the OVAL document basic sections is unambiguously identified by an identifier in the following form:

```
oval:namespace:type:ID
```
where namespace is a name space defining the identifier, type is either def for definitions elements, tst for tests elements, obj for objects element, ste for states elements, and var for variables elements, and ID is an integer value of the identifier.

Example 7.2. An Example of an OVAL Definitions Document

```xml
<?xml version="1.0" encoding="utf-8"?>
<oval_definitions
   xmlns:lin-def="http://oval.mitre.org/XMLSchema/oval-definitions-5#linux"
   xmlns:oval="http://oval.mitre.org/XMLSchema/oval-common-5"
   xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <generator>
      <oval:product_name>vim</oval:product_name>
      <oval:schema_version>5.10.1</oval:schema_version>
      <oval:timestamp>2012-11-22T15:00:00+01:00</oval:timestamp>
   </generator>
   <definitions>
      <definition class="inventory"
         id="oval:org.open-scap.cpe.rhel:def:7"
         version="1">
         <metadata>
            <title>Red Hat Enterprise Linux 7</title>
            <affected family="unix">
               <platform>Red Hat Enterprise Linux 7</platform>
            </affected>
            <reference ref_id="cpe:/o:redhat:enterprise_linux:7"
               source="CPE"/>
            <description>
               The operating system installed on the system is Red Hat Enterprise Linux 7
            </description>
         </metadata>
         <criteria>
            <criterion comment="Red Hat Enterprise Linux 7 is installed"
               test_ref="oval:org.open-scap.cpe.rhel:tst:7"/>
         </criteria>
      </definition>
   </definitions>
   <tests>
      <lin-def:rpminfo_test check_existence="at_least_one_exists"
         id="oval:org.open-scap.cpe.rhel:tst:7"
         version="1"
         check="at least one"
         comment="redhat-release is version 7">
      </lin-def:rpminfo_test>
   </tests>
   <objects>
      <lin-def:rpmverifyfile_object id="oval:org.open-scap.cpe.redhat-release:obj:1"
         version="1"/>
   </objects>
</oval_definitions>
```

CHAPTER 7. COMPLIANCE AND VULNERABILITY SCANNING WITH OPENSCAF

215
7.2.3. The Data Stream Format

SCAP data stream is a file format used since SCAP version 1.2 and it represents a bundle of XCCDF, OVAL, and other component files which can be used to define a compliance policy expressed by an XCCDF checklist. It also contains an index and catalog that allow splitting the given data stream into files according to the SCAP components.

The data stream uses XML format that consists of a header formed by a table of contents and a list of the `<ds:component>` elements. Each of these elements encompasses an SCAP component such as XCCDF, OVAL, CPE, and other. The data stream file may contain multiple components of the same type, and thus covering all security policies needed by your organization.

Example 7.3. An Example of a Data Stream Header

```xml
<ds:data-stream-collection
 xmlns:ds="http://scap.nist.gov/schema/scap/source/1.2"
 xmlns:xlink="http://www.w3.org/1999/xlink"
 xmlns:cat="urn:oasis:names:tc:entity:xmlns:xml:catalog"
 id="scap_org.open-scap_collection_from_xccdf_ssg-rhel7-xccdf-1.2.xml"
```
<ds:data-stream id="scap_org.open-scap_datastream_from_xccdf_ssg-rhel7-xccdf-1.2.xml"
  scap-version="1.2" use-case="OTHER">
  <ds:dictionaries>
    <ds:component-ref id="scap_org.open-scap_cref_output--ssg-rhel7-cpe-dictionary.xml"
      xlink:href="#scap_org.open-scap_comp_output--ssg-rhel7-cpe-dictionary.xml">
      <cat:catalog>
        <cat:uri name="ssg-rhel7-cpe-oval.xml"
          uri="#scap_org.open-scap_cref_output--ssg-rhel7-cpe-oval.xml"/>
      </cat:catalog>
    </ds:component-ref>
  </ds:dictionaries>
  <ds:checklists>
    <ds:component-ref id="scap_org.open-scap_cref_ssg-rhel7-xccdf-1.2.xml"
      xlink:href="#scap_org.open-scap_comp_ssg-rhel7-xccdf-1.2.xml">
      <cat:catalog>
        <cat:uri name="ssg-rhel7-oval.xml"
          uri="#scap_org.open-scap_cref_ssg-rhel7-oval.xml"/>
      </cat:catalog>
    </ds:component-ref>
  </ds:checklists>
  <ds:checks>
    <ds:component-ref id="scap_org.open-scap_cref_ssg-rhel7-oval.xml"
      xlink:href="#scap_org.open-scap_comp_ssg-rhel7-oval.xml"/>
    <ds:component-ref id="scap_org.open-scap_cref_output--ssg-rhel7-cpe-oval.xml"
      xlink:href="#scap_org.open-scap_comp_output--ssg-rhel7-cpe-oval.xml"/>
    <ds:component-ref id="scap_org.open-scap_cref_output--ssg-rhel7-oval.xml"
      xlink:href="#scap_org.open-scap_comp_output--ssg-rhel7-oval.xml"/>
  </ds:checks>
</ds:data-stream>
<ds:component id="scap_org.open-scap_comp_ssg-rhel7-oval.xml"
  timestamp="2014-03-14T16:21:59">
  <oval_definitions xmlns="http://oval.mitre.org/XMLSchema/oval-definitions-5"
    xmlns:oval="http://oval.mitre.org/XMLSchema/oval-common-5"
    xmlns:ind="http://oval.mitre.org/XMLSchema/oval-definitions-5#independent"
    xmlns:unix="http://oval.mitre.org/XMLSchema/oval-definitions-5#unix"
    xmlns:linux="http://oval.mitre.org/XMLSchema/oval-definitions-5#linux"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      oval-common-schema.xsd
      http://oval.mitre.org/XMLSchema/oval-definitions-5
      oval-definitions-schema.xsd
      http://oval.mitre.org/XMLSchema/oval-definitions-
7.3. USING SCAP WORKBENCH

SCAP Workbench (scap-workbench) is a graphical utility that enables users to perform configuration and vulnerability scans on a single local or a remote system, perform remediation of the system, and generate reports based on scan evaluations. Note that compared with the oscap command-line utility, SCAP Workbench has only limited functionality. SCAP Workbench can also process only security content in the form of XCCDF and data-stream files.

The following sections explain how to install, start, and utilize SCAP Workbench in order to perform system scans, remediation, scan customization, and display relevant examples for these tasks.

7.3.1. Installing SCAP Workbench

To install SCAP Workbench on your system, enter the following command as `root`:

```
~# yum install scap-workbench
```

This command installs all packages required by SCAP Workbench to function properly, including the scap-workbench package that provides the utility itself. Note that required dependencies, such as the qt and openssh packages, will be automatically updated to the newest available version if the packages are already installed on your system.

Before you can start using SCAP Workbench effectively, you also need to install or import some security content on your system. For example, you can install the SCAP Security Guide (SSG) package, scap-security-guide, which contains the currently most evolved and elaborate set of security policies for Linux systems. To install the SCAP Security Guide package on your system, enter the following command as `root`:

```
~# yum install scap-security-guide
```

After you install scap-security-guide on your system, unless specified otherwise, the SSG security content is available under the `/usr/share/xml/scap/ssg/content/` directory, and you can proceed with other security compliance operations.

To find other possible sources of existing SCAP content that might suit your needs, see Section 7.9, “Additional Resources”.

7.3.2. Running SCAP Workbench

After a successful installation of both, the SCAP Workbench utility and SCAP content, you can start using SCAP Workbench on your systems. For running SCAP Workbench from the GNOME Classic desktop environment, press the Super key to enter the Activities
Overview, type `scap-workbench`, and then press `Enter`. The **Super** key appears in a variety of guises, depending on the keyboard and other hardware, but often as either the Windows or Command key, and typically to the left of the Spacebar key.

![Open SCAP Security Guide Window](image)

**Figure 7.1. Open SCAP Security Guide Window**

As soon as you start the utility, the **Open SCAP Security Guide** window appears. After a selection one of the guides, the **SCAP Workbench** window appears. This window consists of several interactive components, which you should become familiar with before you start scanning your system:

**File**

This menu list offers several options to load or save a SCAP-related content. To show the initial **Open SCAP Security Guide** window, click the menu item with the same name. Alternatively, load another customization file in the XCCDF format by clicking **Open Other Content**. To save your customization as an XCCDF XML file, use the **Save Customization Only** item. The **Save All** allows you to save SCAP files either to the selected directory or as an RPM package.

**Customization**

This combo box informs you about the customization used for the given security policy. You can select custom rules that will be applied for the system evaluation by clicking
this combo box. The default value is (no customization), which means that there will be no changes to the used security policy. If you made any changes to the selected security profile, you can save those changes as an XML file by clicking the Save Customization Only item in the File menu.

Profile
This combo box contains the name of the selected security profile. You can select the security profile from a given XCCDF or data-stream file by clicking this combo box. To create a new profile that inherits properties of the selected security profile, click the Customize button.

Target
The two radio buttons enable you to select whether the system to be evaluated is a local or remote machine.

Selected Rules
This field displays a list of security rules that are subject of the security policy. Expanding a particular security rule provides detailed information about that rule.

Status bar
This is a graphical bar that indicates status of an operation that is being performed.

Fetch remote resources
This check box allows to instruct the scanner to download a remote OVAL content defined in an XML file.

Dry run
Use this check box to get command line arguments to the diagnostics window instead of running the scan.

Remediate
This check box enables the remediation feature during the system evaluation. If you check this box, SCAP Workbench will attempt to correct system settings that would fail to match the state defined by the policy.

Scan
This button allows you to start the evaluation of the specified system.
The main functionality of SCAP Workbench is to perform security scans on a selected system in accordance with the given XCCDF or data stream file. To evaluate your system against the selected security policy, follow these steps:

1. Select a security policy by using either the Open SCAP Security Guide window, or Open Other Content in the File menu and search the respective XCCDF, SCAP RPM or data stream file.

**WARNING**

Selecting a security policy results in the loss of any previous customization changes that were not saved. To re-apply the lost options, you have to choose the available profile and customization content again. Note that your previous customizations may not be applicable with the new security policy.
2. To use a pre-arranged a file with customized security content specific to your use case, you can load this file by clicking on the **Customization** combo box. You can also create a custom tailoring file by altering an available security profile. For more information, see Section 7.3.4, “Customizing Security Profiles”.

   a. Select the *(no customization)* option if you do not want to use any customization for the current system evaluation. This is the default option if no previous customization was selected.

   b. Select the *(open customization file...)* option to search for the particular tailoring file to be used for the current system evaluation.

   c. If you have previously used some customization file, **SCAP Workbench** remembers this file and adds it to the list. This simplifies repetitive application of the same scan.

3. Select a suitable security profile by clicking the **Profile** combo box.

   a. To modify the selected profile, click the **Customize** button. For more information about profile customization, see Section 7.3.4, “Customizing Security Profiles”.

4. Select either of two Target radio buttons to scan either a local or a remote machine.

   a. If you have selected a remote system, specify it by entering the user name, host name, and the port information as shown in the following example. If you have previously used the remote scan, you can also select a remote system from a list of recently scanned machines.

   ![Figure 7.3. Specifying a Remote System](image)

   **Figure 7.3. Specifying a Remote System**

5. You can allow automatic correction of the system configuration by selecting the **Remediate** check box. With this option enabled, **SCAP Workbench** attempts to change the system configuration in accordance with the security rules applied by the policy, should the related checks fail during the system scan.

   ![WARNING]

   If not used carefully, running the system evaluation with the remediation option enabled could render the system non-functional.

6. Click the **Scan** button to initiate the system scan.

### 7.3.4. Customizing Security Profiles
After selecting the security profile that suits your security policy, you can further adjust it by clicking the **Customize** button. This will open the new Customization window that allows you to modify the currently selected XCCDF profile without actually changing the respective XCCDF file.

![Customization Window](image)

**Figure 7.4. Customizing the Selected Security Profile**

The Customization window contains a complete set of XCCDF elements relevant to the selected security profile with detailed information about each element and its functionality. You can enable or disable these elements by selecting or de-selecting the respective check boxes in the main field of this window. The Customization window also supports **undo** and **redo** functionality; you can undo or redo your selections by clicking the respective arrow icon in the top left corner of the window.

You can also change variables that will later be used for evaluation. Find the desired item in the Customization window, navigate to the right part and use the **Modify value** field.
After you have finished your profile customizations, confirm the changes by clicking the **Confirm Customization** button. Your changes are now in the memory and do not persist if **SCAP Workbench** is closed or certain changes, such as selecting a new SCAP content or choosing another customization option, are made. To store your changes, click the **Save Customization** button in the **SCAP Workbench** window. This action allows you to save your changes to the security profile as an XCCDF customization file in the chosen directory. Note that this customization file can be further selected with other profiles.

### 7.3.5. Saving SCAP Content

**SCAP Workbench** also allows you to save SCAP content that is used with your system evaluations. You can either save a customization file separately (see Section 7.3.4, “Customizing Security Profiles”) or you can save all security content at once by clicking the **Save content** combo box and selecting either the **Save into a directory** or **Save as RPM** options.

By selecting the **Save into a directory** option, **SCAP Workbench** saves both the XCCDF or data-stream file and the customization file to the specified location. This can be useful as a backup solution.

By selecting the **Save as RPM** option, you can instruct **SCAP Workbench** to create an RPM package containing the XCCDF or data stream file and customization file. This is useful for distributing the desired security content to systems that cannot be scanned remotely, or just for delivering the content for further processing.
7.3.6. Viewing Scan Results and Generating Scan Reports

After the system scan is finished, three new buttons, **Clear**, **Save Results**, and **Show Report**, will appear instead of the **Scan** button.

![Figure 7.6. Saving the Current SCAP Content as an RPM Package](image)

**WARNING**

Clicking the **Clear** button permanently removes the scan results.

To store the scan results in the form of an XCCDF, ARF, or HTML file, click the **Save Results** combo box. Choose the **HTML Report** option to generate the scan report in human-readable form. The XCCDF and ARF (data stream) formats are suitable for further automatic processing. You can repeatedly choose all three options.

If you prefer to view the scan results immediately without saving them, you can click the **Show Report** button, which opens the scan results in the form of a temporary HTML file in your default web browser.
7.4. USING OSCAP

The oscap command-line utility enables users to scan local systems, validate security compliance content, and generate reports and guides based on these scans and evaluations. This utility serves as a front end to the OpenSCAP library and groups its functionalities to modules (sub-commands) based on the type of SCAP content it processes.

The following sections explain how to install oscap and perform the most common operations. Examples are provided to illustrate these tasks. To learn more about specific sub-commands, use the --help option with an oscap command:

    oscap [options] module module_operation [module_operation_options_and_arguments] --help

where module represents the type of SCAP content that is being processed, and module_operation is a sub-command for the specific operation on the SCAP content.

Example 7.4. Getting Help on a Specific oscap Operation

```
~]$ oscap ds sds-split --help
oscap -> ds -> sds-split

Split given SourceDataStream into separate files
Usage: oscap [options] ds sds-split [options] SDS TARGET_DIRECTORY

SDS - Source data stream that will be split into multiple files.
TARGET_DIRECTORY - Directory of the resulting files.

Options:
    --datastream-id <id>          - ID of the datastream in the
collection to use.
    --xccdf-id <id>               - ID of XCCDF in the datastream that
should be evaluated.
```

To learn about all oscap features and the complete list of its options, see the oscap(8) manual page.

7.4.1. Installing oscap

To install oscap to your system, enter the following command as root:

```
~]$ yum install openscap-scanner
```

This command allows you to install all packages required by oscap to function properly, including the openscap package. To be able to write your own security content, you should also install the openscap-engine-sce package, which provides the Script Check Engine (SCE). The SCE is an extension of the SCAP protocol that allows content authors to write their security content using a scripting language, such as Bash, Python, or Ruby. Note that the openscap-engine-sce package is only available from the Optional channel. See Enabling Supplementary and Optional Repositories.

Optionally, after installing oscap, you can check the capabilities of your version of oscap,
what specifications it supports, where the certain oscap files are stored, what kinds of SCAP objects you can use, and other useful information. To display this information, type the following command:

```
~$ oscap -V
OpenSCAP command line tool (oscap) 1.0.4

==== Supported specifications ====
XCCDF Version: 1.2
OVAL Version: 5.10.1
CPE Version: 2.3
CVSS Version: 2.0
CVE Version: 2.0
Asset Identification Version: 1.1
Asset Reporting Format Version: 1.1

==== Capabilities added by auto-loaded plugins ====
SCE Version: 1.0 (from libopenscap_sce.so.8)

==== Paths ====
Schema files: /usr/share/openscap/schemas
Schematron files: /usr/share/openscap/xsl
Default CPE files: /usr/share/openscap/cpe
Probes: /usr/libexec/openscap

==== Inbuilt CPE names ====
Red Hat Enterprise Linux - cpe:/o:redhat:enterprise_linux
Red Hat Enterprise Linux 5 - cpe:/o:redhat:enterprise_linux:5
Red Hat Enterprise Linux 6 - cpe:/o:redhat:enterprise_linux:6
Red Hat Enterprise Linux 7 - cpe:/o:redhat:enterprise_linux:7
Fedora 16 - cpe:/o:fedoraproject:fedora:16
Fedora 17 - cpe:/o:fedoraproject:fedora:17
Fedora 18 - cpe:/o:fedoraproject:fedora:18
Fedora 19 - cpe:/o:fedoraproject:fedora:19
Fedora 20 - cpe:/o:fedoraproject:fedora:20
Fedora 21 - cpe:/o:fedoraproject:fedora:21
Red Hat Enterprise Linux Optional Productivity Applications -
cpe:/a:redhat:rhel_productivity
Red Hat Enterprise Linux Optional Productivity Applications 5 -
cpe:/a:redhat:rhel_productivity:5

==== Supported OVAL objects and associated OpenSCAP probes ====
  system_info                        probe_system_info
  family                            probe_family
  filehash                          probe_filehash
  environmentvariable               probe_environmentvariable
  textfilecontent54                 probe_textfilecontent54
  textfilecontent                   probe_textfilecontent
  variable                          probe_variable
  xmlfilecontent                    probe_xmlfilecontent
  environmentvariable58             probe_environmentvariable58
  filehash58                        probe_filehash58
  inetlisteningservers              probe_inetlisteningservers
  rpminfo                           probe_rpminfo
  partition                         probe_partition
Before you can start using oscap effectively, you also need to install or import some security content on your system. For example, you can install the SCAP Security Guide (SSG) package, scap-security-guide, which contains the currently most evolved and elaborate set of security polices for Linux systems. To install the SCAP Security Guide package on your system, enter the following command as root:

```
~]# yum install scap-security-guide
```

After you install scap-security-guide on your system, unless specified otherwise, the SSG security content is available under the `/usr/share/xml/scap/ssg/content/` directory, and you can proceed with other security compliance operations.

To find other possible sources of existing SCAP content that might suit your needs, see Section 7.9, “Additional Resources”.

After installing the SCAP content on your system, oscap can process the content when supplied with the file path to the content. The oscap utility supports SCAP version 1.2 and is backward-compatible with SCAP versions 1.1 and 1.0, so it can process earlier versions of SCAP content without any special requirements.

### 7.4.2. Displaying SCAP Content

SCAP standard defines numerous file formats. The oscap utility can process or create files conforming to many of the formats. In order to further process the given file with SCAP content, you need to understand how to use oscap with the given file type. If you are unsure how to use a particular file, you can either open and read the file, or you can use the info module of oscap which parses the file and extracts relevant information in human-readable format.

enter the following command to examine the internal structure of a SCAP document and display useful information such as the document type, specification version, a status of the document, the date the document was published, and the date the document was copied to a file system:

```
oscap info file
```
where file is the full path to the security content file being examined. The following example better illustrates the usage of the oscap info command:

**Example 7.5. Displaying Information About SCAP Content**

```
~]$ oscap info /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
Document type: Source Data Stream
Imported: 2014-03-14T12:22:01

Stream: scap_org.open-scap_datastream_from_xccdf_ssg-rhel7-xccdf-1.2.xml
Generated: (null)
Version: 1.2
Checklists:
  Ref-Id: scap_org.open-scap_cref_ssg-rhel7-xccdf-1.2.xml
Profiles:
  xccdf_org.ssgproject.content_profile_test
  xccdf_org.ssgproject.content_profile_rht-ccp
  xccdf_org.ssgproject.content_profile_common
  xccdf_org.ssgproject.content_profile_stig-rhel7-server-upstream
  Referenced check files:
    ssg-rhel7-oval.xml
    http://oval.mitre.org/XMLSchema/oval-definitions-5
Checks:
  Ref-Id: scap_org.open-scap_cref_ssg-rhel7-oval.xml
  Ref-Id: scap_org.open-scap_cref_output--ssg-rhel7-cpe-oval.xml
  Ref-Id: scap_org.open-scap_cref_output--ssg-rhel7-oval.xml
Dictionaries:
  Ref-Id: scap_org.open-scap_cref_output--ssg-rhel7-cpe-dictionary.xml
```

### 7.4.3. Scanning the System

The most important functionality of oscap is to perform configuration and vulnerability scans of a local system. The following is a general syntax of the respective command:

```
oscap [options] module eval [module_operation_options_and_arguments]
```

The oscap utility can scan systems against the SCAP content represented by both a**XCCDF** (The eXtensible Configuration Checklist Description Format) benchmark and **OVAL** (Open Vulnerability and Assessment Language) definitions. The security policy can be in the form of a single OVAL or XCCDF file or multiple separate XML files where each file represents a different component (XCCDF, OVAL, CPE, CVE, and others). The result of a scan can be printed to both standard output and an XML file. The result file can then be further processed by oscap in order to generate a report in a human-readable format. The following examples illustrate the most common usage of the command.

**Example 7.6. Scanning the System Using the SSG OVAL definitions**

To scan your system against the SSG OVAL definition file while evaluating all definitions, enter the following command:
The results of the scan are stored as the `scan-oval-results.xml` file in the current directory.

### Example 7.7. Scanning the System Using the SSG OVAL definitions

To evaluate a particular OVAL definition from the security policy represented by the SSG data stream file, enter the following command:

```bash
$ oscap oval eval --id oval:ssg:def:100 --results scan-oval-results.xml /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

The results of the scan are stored as the `scan-oval-results.xml` file in the current directory.

### Example 7.8. Scanning the System Using the SSG XCCDF benchmark

To perform the SSG XCCDF benchmark for the `xccdf_org.ssgproject.content_profile_rht-ccp` profile on your system, enter the following command:

```bash
$ oscap xccdf eval --profile xccdf_org.ssgproject.content_profile_rht-ccp --results scan-xccdf-results.xml /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

The results of the scan are stored as the `scan-xccdf-results.xml` file in the current directory.

**NOTE**

The `--profile` command-line argument selects the security profile from the given XCCDF or data stream file. The list of available profiles can be obtained by running the `oscap info` command. If the `--profile` command-line argument is omitted the default XCCDF profile is used as required by SCAP standard. Note that the default XCCDF profile may or may not be an appropriate security policy.

### 7.4.4. Generating Reports and Guides

Another useful features of `oscap` is the ability to generate SCAP content in a human-readable format. The `oscap` utility allows you to transform an XML file into the HTML or plain-text format. This feature is used to generate security guides and checklists, which serve as a source of information, as well as guidance for secure system configuration. The results of system scans can also be transformed to well-readable result reports. The general command syntax is the following:
oscap module generate sub-module [specific_module/sub-module_options_and_arguments] file

where module is either xccdf or oval, sub-module is a type of the generated document, and file represents an XCCDF or OVAL file.

The following are the most common examples of the command usage:

**Example 7.9. Generating a Guide with a Checklist**

To produce an SSG guide with a checklist for the xccdf_org.ssgproject.content_profile_rht-ccp profile, enter the following command:

```bash
~]$ oscap xccdf generate guide --profile xccdf_org.ssgproject.content_profile_rht-ccp /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml > ssg-guide-checklist.html
```

The guide is stored as the **ssg-guide-checklist.html** file in the current directory.

**Example 7.10. Transforming an SSG OVAL Scan Result into a Report**

To transform a result of an SSG OVAL scan into an HTML file, enter the following command:

```bash
~]$ oscap oval generate report scan-oval-results.xml > ssg-scan-oval-report.html
```

The result report is stored as the **ssg-scan-oval-report.html** file in the current directory. This example assumes that you run the command from the same location where the **scan-oval-results.xml** file is stored. Otherwise you need to specify the fully-qualified path of the file that contains the scan results.

**Example 7.11. Transforming an SSG XCCDF Scan Result into a Report**

To transform a result of an SSG XCCDF scan into an HTML file, enter the following command:

```bash
~]$ oscap xccdf generate report scan-xccdf-results.xml > scan-xccdf-report.html
```

The result report is stored as the **ssg-scan-xccdf-report.html** file in the current directory. Alternatively, you can generate this report in the time of the scan using the --report command-line argument:

```bash
~]$ oscap xccdf eval --profile xccdf_org.ssgproject.content_profile_rht-ccp --results scan-xccdf-results.xml --report scan-xccdf-report.html /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```
7.4.5. Validating SCAP Content

Before you start using a security policy on your systems, you should first verify the policy in order to avoid any possible syntax or semantic errors in the policy. The `oscap` utility can be used to validate the security content against standard SCAP XML schemas. The validation results are printed to the standard error stream (stderr). The general syntax of such a validation command is the following:

```
oscapp module validate [module_options_and_arguments] file
```

where `file` is the full path to the file being validated. The only exception is the data stream module (ds), which uses the `sds-validate` operation instead of `validate`. Note that all SCAP components within the given data stream are validated automatically and none of the components is specified separately, as can be seen in the following example:

```
~]$ oscap ds sds-validate /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

With certain SCAP content, such as OVAL specification, you can also perform a Schematron validation. The Schematron validation is slower than the standard validation but provides deeper analysis, and is thus able to detect more errors. The following SSG example shows typical usage of the command:

```
~]$ oscap oval validate --schematron /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

7.4.6. Using OpenSCAP to Remediate the System

OpenSCAP allows to automatically remediate systems that have been found in a non-compliant state. For system remediation, an XCCDF file with instructions is required. The `scap-security-guide` package contains certain remediation instructions.

System remediation consists of the following steps:

1. **OpenSCAP** performs a regular XCCDF evaluation.

2. An assessment of the results is performed by evaluating the OVAL definitions. Each rule that has failed is marked as a candidate for remediation.

3. **OpenSCAP** searches for an appropriate fix element, resolves it, prepares the environment, and executes the fix script.

4. Any output of the fix script is captured by **OpenSCAP** and stored within the `rule-result` element. The return value of the fix script is stored as well.

5. Whenever **OpenSCAP** executes a fix script, it immediately evaluates the OVAL definition again (to verify that the fix script has been applied correctly). During this second run, if the OVAL evaluation returns success, the result of the rule is fixed, otherwise it is an error.

6. Detailed results of the remediation are stored in an output XCCDF file. It contains two `TestResult` elements. The first `TestResult` element represents the scan prior to the remediation. The second `TestResult` is derived from the first one and contains remediation results.
There are three modes of operation of OpenSCAP with regard to remediation: online, offline, and review.

### 7.4.6.1. OpenSCAP Online Remediation

Online remediation executes fix elements at the time of scanning. Evaluation and remediation are performed as a part of a single command.

To enable online remediation, use the `--remediate` command-line option. For example, to execute online remediation using the scap-security-guide package, run:

```bash
~]$ oscap xccdf eval --remediate --profile xccdf_org.ssgproject.content_profile_rht-ccp --results scan-xccdf-results.xml /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

The output of this command consists of two sections. The first section shows the result of the scan prior to the remediation, and the second section shows the result of the scan after applying the remediation. The second part can contain only fixed and error results. The fixed result indicates that the scan performed after the remediation passed. The error result indicates that even after applying the remediation, the evaluation still does not pass.

### 7.4.6.2. OpenSCAP Offline Remediation

Offline remediation allows you to postpone fix execution. In the first step, the system is only evaluated, and the results are stored in a TestResult element in an XCCDF file.

In the second step, oscap executes the fix scripts and verifies the result. It is safe to store the results into the input file, no data will be lost. During offline remediation, OpenSCAP creates a new TestResult element that is based on the input one and inherits all the data. The newly created TestResult differs only in the rule-result elements that have failed. For those, remediation is executed.

To perform offline remediation using the scap-security-guide package, run:

```bash
~]$ oscap xccdf eval --profile xccdf_org.ssgproject.content_profile_rht-ccp --results scan-xccdf-results.xml /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml

~]$ oscap xccdf remediate --results scan-xccdf-results.xml scan-xccdf-results.xml
```

### 7.4.6.3. OpenSCAP Remediation Review

The review mode allows users to store remediation instructions to a file for further review. The remediation content is not executed during this operation.

To generate remediation instructions in the form of a shell script, run:

```bash
~]$ oscap xccdf generate fix --template urn:xccdf:fix:script:sh --profile xccdf_org.ssgproject.content_profile_rht-ccp --output my-remediation-script.sh /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

### 7.5. USING OPENS SCP WITH DOCKER
The oscap-docker command-line utility allows users to use the oscap program to scan their docker-formatted container images and containers.

The following section:

- explains the installation of oscap-docker
- offers basic examples of usage

To learn more about sub-commands, use the --help option with the oscap-docker or oscap commands.

NOTE

Because a container file system is influenced by mount points, results of scanning an image might differ from results of scanning a container created from it. For example, a mount point that provides only data for processes inside the container in a custom path does not affect results of a scan. However, a mount point in a container done to paths, such as /bin, /etc, or /var, providing new binaries or configuration files can cause differences in results of scanning.

To enable the scanning of images and containers, install the docker package. See the Getting Docker in Red Hat Enterprise Linux 7 chapter of the Getting Started with Containers guide for instructions on installing Docker.

Enter the following command to install oscap-docker:

```
# yum install openscap-utils
```

The following examples use the Red Hat Enterprise Linux 7 image.

```
# docker pull registry.access.redhat.com/rhel7
# docker images
```

<table>
<thead>
<tr>
<th>REPOSITORY</th>
<th>TAG</th>
<th>IMAGE ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>registry.access.redhat.com/rhel7</td>
<td>latest</td>
<td>c453594215e4</td>
</tr>
</tbody>
</table>

7.5.1. Scanning Docker Images and Containers for Vulnerabilities

The oscap-docker command provides two ways to scan images and containers for vulnerabilities:

- The image-cve or container-cve sub-commands determine the version of the operating system, download the CVE stream applicable to the given system, and run a vulnerability scan:
  
  ```
  # oscap-docker image-cve registry.access.redhat.com/rhel7
  ```

- Another, more flexible way, enables you to specify OVAL definitions when checking for vulnerabilities. Use the image or container sub-command together with oscap arguments for an OVAL evaluation. For example, to scan an image using a
definitions file downloaded from Red Hat OVAL repository, use the following command:

```
# oscap-docker image registry.access.redhat.com/rhel7 oval eval com.redhat.rhsa-all.xml
```

### 7.5.2. Scanning Configuration Compliance of Docker Images and Containers

To scan for configuration compliance, use the SCAP content provided by the SCAP Security Guide (SSG). Ensure the `scap-security-guide` package is installed:

```
# yum install scap-security-guide
```

To verify compliance of Red Hat Enterprise Linux 7 image with the Defense Information Systems Agency Security Technical Implementation Guide (DISA STIG) policy, enter the following command:

```
# oscap-docker image registry.access.redhat.com/rhel7 xccdf eval --profile xccdf_org.ssgproject.content_profile_stig-rhel7-disa /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

The following command scans the running environment of the container with ID `5ef05eef4a01`:

```
# oscap-docker container 5ef05eef4a01 xccdf eval --profile xccdf_org.ssgproject.content_profile_stig-rhel7-disa /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

To get an XML file with results, use the `--results` argument and for generating an HTML report of an evaluation, add the `--report` argument. See the `oscap(8)` man page for more information.

### 7.6. USING OPENSCAP WITH ATOMIC

To verify all the container images and containers present on the system are free of known CVE vulnerabilities or common misconfigurations, use the OpenSCAP scanning capabilities through the `atomic scan` command.

**Atomic Scan**

To install the `atomic` tool for container management, enter the following command:

```
# yum install atomic
```

After the `atomic` tool is installed, you also need a scanner. Red Hat recommends choosing the OpenSCAP-based `rhel7/openscap` docker image:

```
# atomic install rhel7/openscap
```

Scan the containers and container images using the `atomic scan` command:

```
# atomic scan $ID
```
Where $ID$ is the ID of the container. If you want to scan all container images or containers, use the `--images` or `--containers` directive, respectively. To scan both types, use the `--all` directive.

**The OpenSCAP Scanner**

The `rhel7/openscap` container image as the default scanner of the `atomic scan` currently supports two scan types targeting Red Hat Enterprise Linux systems only. To list supported scan types, enter the following command:

```
# atomic scan --scanner openscap --list
```

The default scan type is `CVE scan`. Use it for checking the target for known security vulnerabilities as defined in the CVE OVAL definitions released by Red Hat.

![WARNING]

The OVAL definitions used by the `CVE scan` type are bundled in the container image during the build process. Red Hat provides weekly updates of the container image; ensure the definitions are up to date.

The second supported scan type is `standards_compliance`, where Standard System Security Profile of the SCAP Security Guide is used for evaluation. This is security baseline profile of Red Hat Enterprise Linux.

**Example 7.12. Scanning the Container Image with Atomic Scan**

The following example of the `atomic scan` usage shows how to scan a Red Hat Enterprise Linux image and then list of all found vulnerabilities with `--verbose` directive.

```
# docker pull rhel7
Using default tag: latest
98a88a8b722a: Download complete
# atomic scan 98a88a8b722a
Container/Image   Cri   Imp   Med   Low
---------------   ---   ---   ---   ---
98a88a8b722a       0     0     0     0
# atomic scan --verbose 98a88a8b722a
docker run -t --rm -v /etc/localtime:/etc/localtime -v /run/atomic/2016-10-14-06-42-55-991951:/scanin -v /var/lib/atomic/openscap/2016-10-14-06-42-55-991951:/scanout:rw,Z -v /etc/oscapd:/etc/oscapd:ro
rhel7/openscap oscapd-evaluate scan --no-standard-compliance --targets chroots-in-dir:///scanin --output /scanout
INFO:OpenSCAP Daemon one-off evaluator 0.1.6
WARNING:Can't import the 'docker' package. Container scanning functionality will be disabled.
INFO:Creating tasks directory at '/var/lib/oscapd/tasks' because it didn't exist.
INFO:Creating results directory at '/var/lib/oscapd/results' because it didn't exist.
```

236
INFO: Creating results work in progress directory at '/var/lib/oscapd/work_in_progress' because it didn't exist.
INFO: Evaluated EvaluationSpec, exit_code=0.
INFO: Evaluated EvaluationSpec, exit_code=0.
INFO: [100.00%] Scanned target 'chroot://scanin/98a88a8b722a71835dd761c88451c681a8f1bc6e577f90d4dc8b234100bd4861'

98a88a8b722a (registry.access.redhat.com/rhel7:latest)

98a88a8b722a passed the scan

Files associated with this scan are in /var/lib/atomic/openscap/2016-10-14-06-42-55-991951.

NOTE

For a detailed description of the atomic command usage and containers, see the Product Documentation for Red Hat Enterprise Linux Atomic Host. The Red Hat Customer Portal also provides a guide to the Atomic command line interface (CLI).

7.7. USING OPENS'CAP WITH RED HAT SATELLITE

When running multiple Red Hat Enterprise Linux systems, it is important to keep all your systems compliant with your security policy and perform security scans and evaluations remotely from one location. This can be achieved by using Red Hat Satellite 5.5 or later with the spacewalk-oscap package installed on your Satellite client. The package is available from the Red Hat Network Tools channel. See How to enable/disable a repository using Red Hat Subscription-Manager?

This solution supports two methods of performing security compliance scans, viewing and further processing of the scan results. You can either use the OpenSCAP Satellite Web Interface or run commands and scripts from the Satellite API. For more information about this solution to security compliance, its requirements and capabilities, see the Red Hat Satellite documentation.

7.8. PRACTICAL EXAMPLES

This section demonstrates practical usage of certain security content provided for Red Hat products.

7.8.1. Auditing Security Vulnerabilities of Red Hat Products

Red Hat continuously provides OVAL definitions for their products. These definitions allow for fully automated audit of vulnerabilities in the installed software. To find out more information about this project, see http://www.redhat.com/security/data/metrics/. To download these definitions, enter the following command:

The users of Red Hat Satellite 5 may find useful the XCCDF part of the patch definitions. To download these definitions, enter the following command:

```bash
~$ wget http://www.redhat.com/security/data/metrics/com.redhat.rhsa-all.xccdf.xml
```

To audit security vulnerabilities for the software installed on the system, enter the following command:

```bash
~$ oscap oval eval --results rhsa-results-oval.xml --report oval-report.html com.redhat.rhsa-all.xml
```

The `oscap` utility maps Red Hat Security Advisories to CVE identifiers that are linked to the National Vulnerability Database and reports which security advisories are not applied.

**NOTE**

Note that these OVAL definitions are designed to only cover software and updates released by Red Hat. You need to provide additional definitions in order to detect the patch status of third-party software.

### 7.8.2. Auditing System Settings with SCAP Security Guide

The SCAP Security Guide (SSG) project's package, `scap-security-guide`, contains the latest set of security policies for Linux systems. To install the SCAP Security Guide package on your system, enter the following command as root:

```bash
~# yum install scap-security-guide
```

A part of `scap-security-guide` is also a guidance for Red Hat Enterprise Linux 7 settings. To inspect the security content available with `scap-security-guide`, use the `oscap info` module:

```bash
~$ oscap info /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

The output of this command is an outline of the SSG document and it contains available configuration profiles. To audit your system settings, choose a suitable profile and run the appropriate evaluation command. For example, the following command is used to assess the given system against a draft SCAP profile for Red Hat Certified Cloud Providers:

```bash
~$ oscap xccdf eval --profile xccdf_org.ssgproject.content_profile_rhtccp --results ssg-rhel7-xccdf-result.xml --report ssg-rhel7-report.html /usr/share/xml/scap/ssg/content/ssg-rhel7-ds.xml
```

### 7.9. ADDITIONAL RESOURCES

For more information about various security compliance fields of interest, see the resources below.

**Installed Documentation**

- oscap(8) — The manual page for the `oscap` command-line utility provides a complete list of available options and their usage explanation.
• scap-workbench(8) — The manual page for the **SCAP Workbench** application provides a basic information about the application as well as some links to potential sources of SCAP content.

• scap-security-guide(8) — The manual page for **scap-security-guide** provides further documentation about the various available SCAP security profiles. Examples how to utilize the provided benchmarks using the **OpenSCAP** utility are provided as well.

**Online Documentation**

• **The OpenSCAP project page**— The home page to the OpenSCAP project provides detailed information about the **oscap** utility and other components and projects related to SCAP.

• **The SCAP Workbench project page**— The home page to the SCAP Workbench project provides detailed information about the **scap-workbench** application.

• **The SCAP Security Guide (SSG) project page**— The home page to the SSG project that provides the latest security content for Red Hat Enterprise Linux.

• **National Institute of Standards and Technology (NIST) SCAP page**— This page represents a vast collection of SCAP related materials, including SCAP publications, specifications, and the SCAP Validation Program.

• **National Vulnerability Database (NVD)** — This page represents the largest repository of SCAP content and other SCAP standards based vulnerability management data.

• **Red Hat OVAL content repository** — This is a repository containing OVAL definitions for Red Hat Enterprise Linux systems.

• **MITRE CVE** — This is a database of publicly known security vulnerabilities provided by the MITRE corporation.

• **MITRE OVAL** — This page represents an OVAL related project provided by the MITRE corporation. Amongst other OVAL related information, these pages contain the latest version of the OVAL language and a huge repository of OVAL content, counting over 22 thousands OVAL definitions.

• **Red Hat Satellite documentation** — This set of guides describes, amongst other topics, how to maintain system security on multiple systems by using OpenSCAP.
CHAPTER 8. FEDERAL STANDARDS AND REGULATIONS

In order to maintain security levels, it is possible for your organization to make efforts to comply with federal and industry security specifications, standards and regulations. This chapter describes some of these standards and regulations.

8.1. FEDERAL INFORMATION PROCESSING STANDARD (FIPS)

The Federal Information Processing Standard (FIPS) Publication 140-2 is a computer security standard, developed by the U.S. Government and industry working group to validate the quality of cryptographic modules. See the official FIPS publications here: http://csrc.nist.gov/publications/PubsFIPS.html. At the time of the Red Hat Enterprise Linux 7.3 release, Publication 140-3 is at Draft status, and may not represent the completed standard.

The FIPS 140-2 standard ensures that cryptographic tools implement their algorithms properly. See the full FIPS 140-2 standard at http://dx.doi.org/10.6028/NIST.FIPS.140-2 for further details on these levels and the other specifications of the FIPS standard.

To see the complete list of all FIPS 140-2 certificates, visit http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm. To learn about compliance requirements, see the Red Hat Government: Standards page.

8.1.1. Enabling FIPS Mode

To make Red Hat Enterprise Linux compliant with the Federal Information Processing Standard (FIPS) Publication 140-2, you need to make several changes to ensure that accredited cryptographic modules are used. You can either enable FIPS mode during system installation or after it.

During the System Installation
To fulfill the strict FIPS 140-2 compliance, add the **fips=1** kernel option to the kernel command line during system installation. With this option, all keys' generations are done with FIPS-approved algorithms and continuous monitoring tests in place. After the installation, the system is configured to boot into FIPS mode automatically.

**IMPORTANT**

Ensure that the system has plenty of entropy during the installation process by moving the mouse around or by pressing many keystrokes. The recommended amount of keystrokes is 256 and more. Less than 256 keystrokes could generate a non-unique key.

After the System Installation
To turn your system, kernel and user space, into FIPS mode anytime after the system installation, follow these steps:

1. Make sure prelinking is disabled.

For proper operation of the in-module integrity verification, prelinking of libraries and binaries has to be disabled. Prelinking is done by the prelink package, which is not installed by default. To disable prelinking, set the **PRELINKING=no** option in the...
/etc/sysconfig/prelink configuration file. To disable existing prelinking on all system files, use the `prelink -u -a` command.

2. Install the dracut-fips package:

   ~]# yum install dracut-fips

   For the CPUs with the AES New Instructions (AES-NI) support, install the dracut-fips-aesni package as well:

   ~]# yum install dracut-fips-aesni

3. Regenerate the `initramfs` file.

   To enable the in-module integrity verification and to have all required modules present during the kernel boot, the `initramfs` file has to be regenerated:

   ~]# dracut -v -f

   **WARNING**

   This operation will overwrite the existing `initramfs` file.

4. Modify boot loader configuration.

   To boot into FIPS mode, add the `fips=1` option to the kernel command line of the boot loader. If your `/boot` or `/boot/EFI/` partitions reside on separate partitions, add the `boot=<partition>` (where `<partition>` stands for `/boot` or `/boot/EFI`) parameter to the kernel command line as well.

   To identify the boot partition, enter the following command:

   ~]$, df /boot
   Filesystem   1K-blocks  Used  Available  Use%  Mounted on
   /dev/sda1    495844     53780   416464   12%   /boot

   To ensure that the `boot=` configuration option works even if the device naming changes between boots, identify the universally unique identifier (UUID) of the partition by running the following command:

   ~]$, blkid /dev/sda1
   /dev/sda1: UUID="05c000f1-f899-467b-a4d9-d5ca4424c797" TYPE="ext4"

   Append the UUID to the kernel command line:

   boot=UUID=05c000f1-f899-467b-a4d9-d5ca4424c797

   Depending on your boot loader, make the following changes:
grub2

Add the `fips=1` and `boot=<partition of /boot or /boot/EFI>` options to the `GRUB_CMDLINE_LINUX` key in the `/etc/default/grub` file. To apply the changes to `/etc/default/grub`, rebuild the `grub.cfg` file as follows:

- On BIOS-based machines, enter the following command as `root`:
  ```bash
  ~]# grub2-mkconfig -o /boot/grub2/grub.cfg
  ```
- On UEFI-based machines, enter the following command as `root`:
  ```bash
  ~]# grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
  ```

zipl (on the IBM z Systems architecture only)

Add the `fips=1` and `boot=<partition of /boot>` options to the `/etc/zipl.conf` to the kernel command line and apply the changes by running the following command as `root`:

```bash
~]# zipl
```

5. Reboot your system.

### 8.2. NATIONAL INDUSTRIAL SECURITY PROGRAM OPERATING MANUAL (NISPOM)

The NISPOM (also called DoD 5220.22-M), as a component of the National Industrial Security Program (NISP), establishes a series of procedures and requirements for all government contractors with regard to classified information. The current NISPOM is dated February 28, 2006, with incorporated major changes from March 28, 2013. The NISPOM document can be downloaded from the following URL: [http://www.nispom.org/NISPOM-download.html](http://www.nispom.org/NISPOM-download.html).

### 8.3. PAYMENT CARD INDUSTRY DATA SECURITY STANDARD (PCI DSS)

From [https://www.pcisecuritystandards.org/about/index.shtml](https://www.pcisecuritystandards.org/about/index.shtml): The PCI Security Standards Council is an open global forum, launched in 2006, that is responsible for the development, management, education, and awareness of the PCI Security Standards, including the Data Security Standard (DSS).


### 8.4. SECURITY TECHNICAL IMPLEMENTATION GUIDE

A Security Technical Implementation Guide (STIG) is a methodology for standardized secure installation and maintenance of computer software and hardware.

See the following URL for more information on STIG: [http://iase.disa.mil/stigs/Pages/index.aspx](http://iase.disa.mil/stigs/Pages/index.aspx).
Appendix A. Encryption Standards

A.1. Synchronous Encryption

A.1.1. Advanced Encryption Standard — AES

In cryptography, the Advanced Encryption Standard (AES) is an encryption standard adopted by the U.S. Government. The standard comprises three block ciphers, AES-128, AES-192 and AES-256, adopted from a larger collection originally published as Rijndael. Each AES cipher has a 128-bit block size, with key sizes of 128, 192 and 256 bits, respectively. The AES ciphers have been analyzed extensively and are now used worldwide, as was the case with its predecessor, the Data Encryption Standard (DES).[3]

A.1.1.1. AES History

AES was announced by National Institute of Standards and Technology (NIST) as U.S. FIPS PUB 197 (FIPS 197) on November 26, 2001 after a 5-year standardization process. Fifteen competing designs were presented and evaluated before Rijndael was selected as the most suitable. It became effective as a standard May 26, 2002. It is available in many different encryption packages. AES is the first publicly accessible and open cipher approved by the NSA for top secret information (see the Security section in the Wikipedia article on AES).[4]

The Rijndael cipher was developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen, and submitted by them to the AES selection process. Rijndael is a portmanteau of the names of the two inventors.[5]

A.1.2. Data Encryption Standard — DES

The Data Encryption Standard (DES) is a block cipher (a form of shared secret encryption) that was selected by the National Bureau of Standards as an official Federal Information Processing Standard (FIPS) for the United States in 1976 and which has subsequently enjoyed widespread use internationally. It is based on a symmetric-key algorithm that uses a 56-bit key. The algorithm was initially controversial with classified design elements, a relatively short key length, and suspicions about a National Security Agency (NSA) backdoor. DES consequently came under intense academic scrutiny which motivated the modern understanding of block ciphers and their cryptanalysis.[6]

A.1.2.1. DES History

DES is now considered to be insecure for many applications. This is chiefly due to the 56-bit key size being too small; in January, 1999, distributed.net and the Electronic Frontier Foundation collaborated to publicly break a DES key in 22 hours and 15 minutes. There are also some analytical results which demonstrate theoretical weaknesses in the cipher, although they are unfeasible to mount in practice. The algorithm is believed to be practically secure in the form of Triple DES, although there are theoretical attacks. In recent years, the cipher has been superseded by the Advanced Encryption Standard (AES).[7]

In some documentation, a distinction is made between DES as a standard and DES the algorithm which is referred to as the DEA (the Data Encryption Algorithm).[8]

A.2. Public-Key Encryption
Public-key cryptography is a cryptographic approach, employed by many cryptographic algorithms and cryptosystems, whose distinguishing characteristic is the use of asymmetric key algorithms instead of or in addition to symmetric key algorithms. Using the techniques of public-key-private key cryptography, many methods of protecting communications or authenticating messages formerly unknown have become practical. They do not require a secure initial exchange of one or more secret keys as is required when using symmetric key algorithms. It can also be used to create digital signatures.[9]

Public key cryptography is a fundamental and widely used technology around the world, and is the approach which underlies such Internet standards as Transport Layer Security (TLS) (successor to SSL), PGP and GPG.[10]

The distinguishing technique used in public key cryptography is the use of asymmetric key algorithms, where the key used to encrypt a message is not the same as the key used to decrypt it. Each user has a pair of cryptographic keys — a public key and a private key. The private key is kept secret, whilst the public key may be widely distributed. Messages are encrypted with the recipient’s public key and can only be decrypted with the corresponding private key. The keys are related mathematically, but the private key cannot be feasibly (ie, in actual or projected practice) derived from the public key. It was the discovery of such algorithms which revolutionized the practice of cryptography beginning in the middle 1970s.[11]

In contrast, Symmetric-key algorithms, variations of which have been used for some thousands of years, use a single secret key shared by sender and receiver (which must also be kept private, thus accounting for the ambiguity of the common terminology) for both encryption and decryption. To use a symmetric encryption scheme, the sender and receiver must securely share a key in advance.[12]

Because symmetric key algorithms are nearly always much less computationally intensive, it is common to exchange a key using a key-exchange algorithm and transmit data using that key and a symmetric key algorithm. PGP, and the SSL/TLS family of schemes do this, for instance, and are called hybrid cryptosystems in consequence.[13]

### A.2.1. Diffie-Hellman

Diffie–Hellman key exchange (D–H) is a cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher.[14]

#### A.2.1.1. Diffie-Hellman History

The scheme was first published by Whitfield Diffie and Martin Hellman in 1976, although it later emerged that it had been separately invented a few years earlier within GCHQ, the British signals intelligence agency, by Malcolm J. Williamson but was kept classified. In 2002, Hellman suggested the algorithm be called Diffie-Hellman-Merkle key exchange in recognition of Ralph Merkle’s contribution to the invention of public-key cryptography (Hellman, 2002).[15]

Although Diffie-Hellman key agreement itself is an anonymous (non-authenticated) key-agreement protocol, it provides the basis for a variety of authenticated protocols, and is used to provide perfect forward secrecy in Transport Layer Security’s ephemeral modes (referred to as EDH or DHE depending on the cipher suite).[16]
U.S. Patent 4,200,770, now expired, describes the algorithm and credits Hellman, Diffie, and Merkle as inventors.[17]

**A.2.2. RSA**

In cryptography, RSA (which stands for Rivest, Shamir and Adleman who first publicly described it) is an algorithm for public-key cryptography. It is the first algorithm known to be suitable for signing as well as encryption, and was one of the first great advances in public key cryptography. RSA is widely used in electronic commerce protocols, and is believed to be secure given sufficiently long keys and the use of up-to-date implementations.

**A.2.3. DSA**

DSA (Digital Signature Algorithm) is a standard for digital signatures, a United States federal government standard for digital signatures. DSA is for signatures only and is not an encryption algorithm. [18]

**A.2.4. SSL/TLS**

Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), are cryptographic protocols that provide security for communications over networks such as the Internet. TLS and SSL encrypt the segments of network connections at the Transport Layer end-to-end.

Several versions of the protocols are in widespread use in applications like web browsing, electronic mail, Internet faxing, instant messaging and voice-over-IP (VoIP).[19]

**A.2.5. Cramer-Shoup Cryptosystem**

The Cramer–Shoup system is an asymmetric key encryption algorithm, and was the first efficient scheme proven to be secure against adaptive chosen ciphertext attack using standard cryptographic assumptions. Its security is based on the computational intractability (widely assumed, but not proved) of the decisional Diffie–Hellman assumption. Developed by Ronald Cramer and Victor Shoup in 1998, it is an extension of the ElGamal cryptosystem. In contrast to ElGamal, which is extremely malleable, Cramer–Shoup adds additional elements to ensure non-malleability even against a resourceful attacker. This non-malleability is achieved through the use of a collision-resistant hash function and additional computations, resulting in a ciphertext which is twice as large as in ElGamal.[20]

**A.2.6. ElGamal Encryption**

In cryptography, the ElGamal encryption system is an asymmetric key encryption algorithm for public-key cryptography which is based on the Diffie-Hellman key agreement. It was described by Taher ElGamal in 1985. ElGamal encryption is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems.[21]

---


Security Guide

http://en.wikipedia.org/wiki/Advanced_Encryption_Standard


[21] "ElGamal encryption" Wikipedia. 24 February 2010
## APPENDIX B. AUDIT SYSTEM REFERENCE

### B.1. AUDIT EVENT FIELDS

Table B.1, “Event Fields” lists all currently-supported Audit event fields. An event field is the value preceding the equal sign in the Audit log files.

**Table B.1. Event Fields**

<table>
<thead>
<tr>
<th>Event Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>a0, a1, a2, a3</td>
<td>Records the first four arguments of the system call, encoded in hexadecimal notation.</td>
</tr>
<tr>
<td>acct</td>
<td>Records a user's account name.</td>
</tr>
<tr>
<td>addr</td>
<td>Records the IPv4 or IPv6 address. This field usually follows a <code>hostname</code> field and contains the address the host name resolves to.</td>
</tr>
<tr>
<td>arch</td>
<td>Records information about the CPU architecture of the system, encoded in hexadecimal notation.</td>
</tr>
<tr>
<td>auid</td>
<td>Records the Audit user ID. This ID is assigned to a user upon login and is inherited by every process even when the user's identity changes (for example, by switching user accounts with <code>su - john</code>).</td>
</tr>
<tr>
<td>capability</td>
<td>Records the number of bits that were used to set a particular Linux capability. For more information on Linux capabilities, see the <code>capabilities(7)</code> man page.</td>
</tr>
<tr>
<td>cap_fi</td>
<td>Records data related to the setting of an inherited file system-based capability.</td>
</tr>
<tr>
<td>cap_fp</td>
<td>Records data related to the setting of a permitted file system-based capability.</td>
</tr>
<tr>
<td>cap_pe</td>
<td>Records data related to the setting of an effective process-based capability.</td>
</tr>
<tr>
<td>cap_pi</td>
<td>Records data related to the setting of an inherited process-based capability.</td>
</tr>
<tr>
<td>cap_pp</td>
<td>Records data related to the setting of a permitted process-based capability.</td>
</tr>
<tr>
<td>cgroup</td>
<td>Records the path to the <code>cgroup</code> that contains the process at the time the Audit event was generated.</td>
</tr>
<tr>
<td>Event Field</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>cmd</strong></td>
<td>Records the entire command line that is executed. This is useful in case of shell interpreters where the <strong>exe</strong> field records, for example, <code>/bin/bash</code> as the shell interpreter and the <strong>cmd</strong> field records the rest of the command line that is executed, for example <code>helloworld.sh --help</code>.</td>
</tr>
<tr>
<td><strong>comm</strong></td>
<td>Records the command that is executed. This is useful in case of shell interpreters where the <strong>exe</strong> field records, for example, <code>/bin/bash</code> as the shell interpreter and the <strong>comm</strong> field records the name of the script that is executed, for example <code>helloworld.sh</code>.</td>
</tr>
<tr>
<td><strong>cwd</strong></td>
<td>Records the path to the directory in which a system call was invoked.</td>
</tr>
<tr>
<td><strong>data</strong></td>
<td>Records data associated with TTY records.</td>
</tr>
<tr>
<td><strong>dev</strong></td>
<td>Records the minor and major ID of the device that contains the file or directory recorded in an event.</td>
</tr>
<tr>
<td><strong>devmajor</strong></td>
<td>Records the major device ID.</td>
</tr>
<tr>
<td><strong>devminor</strong></td>
<td>Records the minor device ID.</td>
</tr>
<tr>
<td><strong>egid</strong></td>
<td>Records the effective group ID of the user who started the analyzed process.</td>
</tr>
<tr>
<td><strong>euid</strong></td>
<td>Records the effective user ID of the user who started the analyzed process.</td>
</tr>
<tr>
<td><strong>exe</strong></td>
<td>Records the path to the executable that was used to invoke the analyzed process.</td>
</tr>
<tr>
<td><strong>exit</strong></td>
<td>Records the exit code returned by a system call. This value varies by system call. You can interpret the value to its human-readable equivalent with the following command: <code>ausearch --interpret --exit exit_code</code></td>
</tr>
<tr>
<td><strong>family</strong></td>
<td>Records the type of address protocol that was used, either IPv4 or IPv6.</td>
</tr>
<tr>
<td><strong>filetype</strong></td>
<td>Records the type of the file.</td>
</tr>
<tr>
<td><strong>flags</strong></td>
<td>Records the file system name flags.</td>
</tr>
<tr>
<td><strong>fsgid</strong></td>
<td>Records the file system group ID of the user who started the analyzed process.</td>
</tr>
<tr>
<td>Event Field</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>fsuid</td>
<td>Records the file system user ID of the user who started the analyzed process.</td>
</tr>
<tr>
<td>gid</td>
<td>Records the group ID.</td>
</tr>
<tr>
<td>hostname</td>
<td>Records the host name.</td>
</tr>
<tr>
<td>icmptype</td>
<td>Records the type of a Internet Control Message Protocol (ICMP) package that is received. Audit messages containing this field are usually generated by <code>iptables</code>.</td>
</tr>
<tr>
<td>id</td>
<td>Records the user ID of an account that was changed.</td>
</tr>
<tr>
<td>inode</td>
<td>Records the inode number associated with the file or directory recorded in an Audit event.</td>
</tr>
<tr>
<td>inode_gid</td>
<td>Records the group ID of the inode's owner.</td>
</tr>
<tr>
<td>inode_uid</td>
<td>Records the user ID of the inode's owner.</td>
</tr>
<tr>
<td>items</td>
<td>Records the number of path records that are attached to this record.</td>
</tr>
<tr>
<td>key</td>
<td>Records the user defined string associated with a rule that generated a particular event in the Audit log.</td>
</tr>
<tr>
<td>list</td>
<td>Records the Audit rule list ID. The following is a list of known IDs:</td>
</tr>
<tr>
<td></td>
<td>- 0 — <strong>user</strong></td>
</tr>
<tr>
<td></td>
<td>- 1 — <strong>task</strong></td>
</tr>
<tr>
<td></td>
<td>- 4 — <strong>exit</strong></td>
</tr>
<tr>
<td></td>
<td>- 5 — <strong>exclude</strong></td>
</tr>
<tr>
<td>mode</td>
<td>Records the file or directory permissions, encoded in numerical notation.</td>
</tr>
<tr>
<td>msg</td>
<td>Records a time stamp and a unique ID of a record, or various event-specific <code>&lt;name&gt;=&lt;value&gt;</code> pairs provided by the kernel or user space applications.</td>
</tr>
<tr>
<td>msgtype</td>
<td>Records the message type that is returned in case of a user-based AVC denial. The message type is determined by D-Bus.</td>
</tr>
<tr>
<td>name</td>
<td>Records the full path of the file or directory that was passed to the system call as an argument.</td>
</tr>
<tr>
<td>Event Field</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>new-disk</td>
<td>Records the name of a new disk resource that is assigned to a virtual machine.</td>
</tr>
<tr>
<td>new-mem</td>
<td>Records the amount of a new memory resource that is assigned to a virtual machine.</td>
</tr>
<tr>
<td>new-vcpu</td>
<td>Records the number of a new virtual CPU resource that is assigned to a virtual machine.</td>
</tr>
<tr>
<td>new-net</td>
<td>Records the MAC address of a new network interface resource that is assigned to a virtual machine.</td>
</tr>
<tr>
<td>new_gid</td>
<td>Records a group ID that is assigned to a user.</td>
</tr>
<tr>
<td>oauuid</td>
<td>Records the user ID of the user that has logged in to access the system (as opposed to, for example, using <code>su</code>) and has started the target process. This field is exclusive to the record of type <code>OBJ_PID</code>.</td>
</tr>
<tr>
<td>ocomm</td>
<td>Records the command that was used to start the target process. This field is exclusive to the record of type <code>OBJ_PID</code>.</td>
</tr>
<tr>
<td>opid</td>
<td>Records the process ID of the target process. This field is exclusive to the record of type <code>OBJ_PID</code>.</td>
</tr>
<tr>
<td>oses</td>
<td>Records the session ID of the target process. This field is exclusive to the record of type <code>OBJ_PID</code>.</td>
</tr>
<tr>
<td>ouid</td>
<td>Records the real user ID of the target process</td>
</tr>
<tr>
<td>obj</td>
<td>Records the SELinux context of an object. An object can be a file, a directory, a socket, or anything that is receiving the action of a subject.</td>
</tr>
<tr>
<td>obj_gid</td>
<td>Records the group ID of an object.</td>
</tr>
<tr>
<td>obj_lev_high</td>
<td>Records the high SELinux level of an object.</td>
</tr>
<tr>
<td>obj_lev_low</td>
<td>Records the low SELinux level of an object.</td>
</tr>
<tr>
<td>obj_role</td>
<td>Records the SELinux role of an object.</td>
</tr>
<tr>
<td>obj_uid</td>
<td>Records the UID of an object.</td>
</tr>
<tr>
<td>obj_user</td>
<td>Records the user that is associated with an object.</td>
</tr>
<tr>
<td>Event Field</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>ogid</td>
<td>Records the object owner's group ID.</td>
</tr>
<tr>
<td>old-disk</td>
<td>Records the name of an old disk resource when a new disk resource is assigned to a virtual machine.</td>
</tr>
<tr>
<td>old-mem</td>
<td>Records the amount of an old memory resource when a new amount of memory is assigned to a virtual machine.</td>
</tr>
<tr>
<td>old-vcpu</td>
<td>Records the number of an old virtual CPU resource when a new virtual CPU is assigned to a virtual machine.</td>
</tr>
<tr>
<td>old-net</td>
<td>Records the MAC address of an old network interface resource when a new network interface is assigned to a virtual machine.</td>
</tr>
<tr>
<td>old_prom</td>
<td>Records the previous value of the network promiscuity flag.</td>
</tr>
<tr>
<td>ouid</td>
<td>Records the real user ID of the user who started the target process.</td>
</tr>
<tr>
<td>path</td>
<td>Records the full path of the file or directory that was passed to the system call as an argument in case of AVC-related Audit events.</td>
</tr>
<tr>
<td>perm</td>
<td>Records the file permission that was used to generate an event (that is, read, write, execute, or attribute change).</td>
</tr>
<tr>
<td>pid</td>
<td>The <code>pid</code> field semantics depend on the origin of the value in this field. In fields generated from user-space, this field holds a process ID. In fields generated by the kernel, this field holds a thread ID. The thread ID is equal to process ID for single-threaded processes. Note that the value of this thread ID is different from the values of pthread_t IDs used in user-space. For more information, see the <code>gettid(2)</code> man page.</td>
</tr>
<tr>
<td>ppid</td>
<td>Records the Parent Process ID (PID).</td>
</tr>
<tr>
<td>proctitle</td>
<td>Records the full command-line of the command that was used to invoke the analyzed process. The field is encoded in hexadecimal notation to not allow the user to influence the Audit log parser. The text decodes to the command that triggered this Audit event. When searching Audit records with the <code>ausearch</code> command, use the <code>-i</code> or <code>--interpret</code> option to automatically convert hexadecimal values into their human-readable equivalents.</td>
</tr>
<tr>
<td>prom</td>
<td>Records the network promiscuity flag.</td>
</tr>
<tr>
<td>proto</td>
<td>Records the networking protocol that was used. This field is specific to Audit events generated by <code>iptables</code>.</td>
</tr>
<tr>
<td>Event Field</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>res</td>
<td>Records the result of the operation that triggered the Audit event.</td>
</tr>
<tr>
<td>result</td>
<td>Records the result of the operation that triggered the Audit event.</td>
</tr>
<tr>
<td>saddr</td>
<td>Records the socket address.</td>
</tr>
<tr>
<td>sauid</td>
<td>Records the sender Audit login user ID. This ID is provided by D-Bus as the kernel is unable to see which user is sending the original auid.</td>
</tr>
<tr>
<td>ses</td>
<td>Records the session ID of the session from which the analyzed process was invoked.</td>
</tr>
<tr>
<td>sgid</td>
<td>Records the set group ID of the user who started the analyzed process.</td>
</tr>
<tr>
<td>sig</td>
<td>Records the number of a signal that causes a program to end abnormally. Usually, this is a sign of a system intrusion.</td>
</tr>
<tr>
<td>subj</td>
<td>Records the SELinux context of a subject. A subject can be a process, a user, or anything that is acting upon an object.</td>
</tr>
<tr>
<td>subj_clr</td>
<td>Records the SELinux clearance of a subject.</td>
</tr>
<tr>
<td>subj_role</td>
<td>Records the SELinux role of a subject.</td>
</tr>
<tr>
<td>subj_sen</td>
<td>Records the SELinux sensitivity of a subject.</td>
</tr>
<tr>
<td>subj_user</td>
<td>Records the user that is associated with a subject.</td>
</tr>
<tr>
<td>success</td>
<td>Records whether a system call was successful or failed.</td>
</tr>
<tr>
<td>suid</td>
<td>Records the set user ID of the user who started the analyzed process.</td>
</tr>
<tr>
<td>syscall</td>
<td>Records the type of the system call that was sent to the kernel.</td>
</tr>
<tr>
<td>terminal</td>
<td>Records the terminal name (without /dev/).</td>
</tr>
<tr>
<td>tty</td>
<td>Records the name of the controlling terminal. The value <strong>none</strong> is used if the process has no controlling terminal.</td>
</tr>
<tr>
<td>uid</td>
<td>Records the real user ID of the user who started the analyzed process.</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Event Field</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>vm</td>
<td>Records the name of a virtual machine from which the Audit event originated.</td>
</tr>
</tbody>
</table>

## B.2. AUDIT RECORD TYPES

Table B.2, “Record Types” lists all currently-supported types of Audit records. The event type is specified in the `type=` field at the beginning of every Audit record.

### Table B.2. Record Types

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCT_LOCK</td>
<td>Triggered when a user-space user account is locked by the administrator.</td>
</tr>
<tr>
<td>ACCT_UNLOCK</td>
<td>Triggered when a user-space user account is unlocked by the administrator.</td>
</tr>
<tr>
<td>ADD_GROUP</td>
<td>Triggered when a user-space group is added.</td>
</tr>
<tr>
<td>ADD_USER</td>
<td>Triggered when a user-space user account is added.</td>
</tr>
<tr>
<td>ANOM_ABEND[a]</td>
<td>Triggered when a processes ends abnormally (with a signal that could cause a core dump, if enabled).</td>
</tr>
<tr>
<td>ANOM_ACCESS_FS[a]</td>
<td>Triggered when a file or a directory access ends abnormally.</td>
</tr>
<tr>
<td>ANOM_ADD_ACCT[a]</td>
<td>Triggered when a user-space account addition ends abnormally.</td>
</tr>
<tr>
<td>ANOM_AMTU_FAIL[a]</td>
<td>Triggered when a failure of the Abstract Machine Test Utility (AMTU) is detected.</td>
</tr>
<tr>
<td>ANOM_CRYPTO_FAIL[a]</td>
<td>Triggered when a failure in the cryptographic system is detected.</td>
</tr>
<tr>
<td>ANOM_DEL_ACCT[a]</td>
<td>Triggered when a user-space account deletion ends abnormally.</td>
</tr>
<tr>
<td>ANOM_EXEC[a]</td>
<td>Triggered when an execution of a file ends abnormally.</td>
</tr>
<tr>
<td>ANOM_LINK[a]</td>
<td>Triggered when suspicious use of file links is detected.</td>
</tr>
<tr>
<td>ANOM_LOGIN_ACCT[a]</td>
<td>Triggered when an account login attempt ends abnormally.</td>
</tr>
<tr>
<td>ANOM_LOGIN_FAILURES[a]</td>
<td>Triggered when the limit of failed login attempts is reached.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Explanation</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ANOM_LOGIN_LOCATION[a]</td>
<td>Triggered when a login attempt is made from a forbidden location.</td>
</tr>
<tr>
<td>ANOM_LOGIN_SESSIONS[a]</td>
<td>Triggered when a login attempt reaches the maximum amount of concurrent sessions.</td>
</tr>
<tr>
<td>ANOM_LOGIN_TIME[a]</td>
<td>Triggered when a login attempt is made at a time when it is prevented by, for example, pam_time.</td>
</tr>
<tr>
<td>ANOM_MAX_DAC[a]</td>
<td>Triggered when the maximum amount of Discretionary Access Control (DAC) failures is reached.</td>
</tr>
<tr>
<td>ANOM_MAX_MAC[a]</td>
<td>Triggered when the maximum amount of Mandatory Access Control (MAC) failures is reached.</td>
</tr>
<tr>
<td>ANOM_MK_EXEC[a]</td>
<td>Triggered when a file is made executable.</td>
</tr>
<tr>
<td>ANOM_MOD_ACCT[a]</td>
<td>Triggered when a user-space account modification ends abnormally.</td>
</tr>
<tr>
<td>ANOM_PROMISCUOUS[a]</td>
<td>Triggered when a device enables or disables promiscuous mode.</td>
</tr>
<tr>
<td>ANOM_RBAC_FAIL[a]</td>
<td>Triggered when a Role-Based Access Control (RBAC) self-test failure is detected.</td>
</tr>
<tr>
<td>ANOM_RBAC_INTEGRITY_FAIL[a]</td>
<td>Triggered when a Role-Based Access Control (RBAC) file integrity test failure is detected.</td>
</tr>
<tr>
<td>ANOM_ROOT_TRANS[a]</td>
<td>Triggered when a user becomes root.</td>
</tr>
<tr>
<td>AVC</td>
<td>Triggered to record an SELinux permission check.</td>
</tr>
<tr>
<td>AVC_PATH</td>
<td>Triggered to record the dentry and vfsmount pair when an SELinux permission check occurs.</td>
</tr>
<tr>
<td>BPRM_FCAPS</td>
<td>Triggered when a user executes a program with a file system capability.</td>
</tr>
<tr>
<td>CAPSET</td>
<td>Triggered to record the capabilities being set for process-based capabilities, for example, running as root to drop capabilities.</td>
</tr>
<tr>
<td>CHGRP_ID</td>
<td>Triggered when a user-space group ID is changed.</td>
</tr>
<tr>
<td>CHUSER_ID</td>
<td>Triggered when a user-space user ID is changed.</td>
</tr>
<tr>
<td>CONFIG_CHANGE</td>
<td>Triggered when the Audit system configuration is modified.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Explanation</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CRED_ACQ</td>
<td>Triggered when a user acquires user-space credentials.</td>
</tr>
<tr>
<td>CRED_DISP</td>
<td>Triggered when a user disposes of user-space credentials.</td>
</tr>
<tr>
<td>CRED_REFR</td>
<td>Triggered when a user refreshes their user-space credentials.</td>
</tr>
<tr>
<td>CRYPTO_FAILURE_USER</td>
<td>Triggered when a decrypt, encrypt, or randomize cryptographic operation fails.</td>
</tr>
<tr>
<td>CRYPTO_IKE_SA</td>
<td>Triggered when an Internet Key Exchange Security Association is established.</td>
</tr>
<tr>
<td>CRYPTO_IPSEC_SA</td>
<td>Triggered when an Internet Protocol Security Association is established.</td>
</tr>
<tr>
<td>CRYPTO_KEY_USER</td>
<td>Triggered to record the cryptographic key identifier used for cryptographic purposes.</td>
</tr>
<tr>
<td>CRYPTO_LOGIN</td>
<td>Triggered when a cryptographic officer login attempt is detected.</td>
</tr>
<tr>
<td>CRYPTO_LOGOUT</td>
<td>Triggered when a cryptographic officer logout attempt is detected.</td>
</tr>
<tr>
<td>CRYPTO_PARAM_CHANGE_USER</td>
<td>Triggered when a change in a cryptographic parameter is detected.</td>
</tr>
<tr>
<td>CRYPTO_REPLAY_USER</td>
<td>Triggered when a replay attack is detected.</td>
</tr>
<tr>
<td>CRYPTO_SESSION</td>
<td>Triggered to record parameters set during a TLS session establishment.</td>
</tr>
<tr>
<td>CRYPTO_TEST_USER</td>
<td>Triggered to record cryptographic test results as required by the FIPS-140 standard.</td>
</tr>
<tr>
<td>CWD</td>
<td>Triggered to record the current working directory.</td>
</tr>
<tr>
<td>DAC_CHECK</td>
<td>Triggered to record DAC check results.</td>
</tr>
<tr>
<td>DAEMON_ABORT</td>
<td>Triggered when a daemon is stopped due to an error.</td>
</tr>
<tr>
<td>DAEMON_ACCEPT</td>
<td>Triggered when the <code>auditd</code> daemon accepts a remote connection.</td>
</tr>
<tr>
<td>DAEMON_CLOSE</td>
<td>Triggered when the <code>auditd</code> daemon closes a remote connection.</td>
</tr>
<tr>
<td>DAEMON_CONFIG</td>
<td>Triggered when a daemon configuration change is detected.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Explanation</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DAEMON_END</td>
<td>Triggered when a daemon is successfully stopped.</td>
</tr>
<tr>
<td>DAEMON_ERR</td>
<td>Triggered when an <code>auditd</code> daemon internal error is detected.</td>
</tr>
<tr>
<td>DAEMON_RESUME</td>
<td>Triggered when the <code>auditd</code> daemon resumes logging.</td>
</tr>
<tr>
<td>DAEMON_ROTATE</td>
<td>Triggered when the <code>auditd</code> daemon rotates the Audit log files.</td>
</tr>
<tr>
<td>DAEMON_START</td>
<td>Triggered when the <code>auditd</code> daemon is started.</td>
</tr>
<tr>
<td>DEL_GROUP</td>
<td>Triggered when a user-space group is deleted.</td>
</tr>
<tr>
<td>DEL_USER</td>
<td>Triggered when a user-space user is deleted.</td>
</tr>
<tr>
<td>DEV_ALLOC</td>
<td>Triggered when a device is allocated.</td>
</tr>
<tr>
<td>DEV_DEALLOC</td>
<td>Triggered when a device is deallocated.</td>
</tr>
<tr>
<td>EOE</td>
<td>Triggered to record the end of a multi-record event.</td>
</tr>
<tr>
<td>EXECVE</td>
<td>Triggered to record arguments of the <code>execve(2)</code> system call.</td>
</tr>
<tr>
<td>FD_PAIR</td>
<td>Triggered to record the use of the <code>pipe</code> and <code>socketpair</code> system calls.</td>
</tr>
<tr>
<td>FEATURE_CHANGE</td>
<td>Triggered when an Audit feature changed value.</td>
</tr>
<tr>
<td>FS_RELABEL</td>
<td>Triggered when a file system relabel operation is detected.</td>
</tr>
<tr>
<td>GRP_AUTH</td>
<td>Triggered when a group password is used to authenticate against a user-space group.</td>
</tr>
<tr>
<td>GRP_CHAUTHTOK</td>
<td>Triggered when a group account password or PIN is modified.</td>
</tr>
<tr>
<td>GRP_MGMT</td>
<td>Triggered to record user-space group account attribute modification.</td>
</tr>
<tr>
<td>INTEGRITY_DATA</td>
<td>Triggered to record a data integrity verification event run by the kernel.</td>
</tr>
<tr>
<td>INTEGRITY_HASH</td>
<td>Triggered to record a hash type integrity verification event run by the kernel.</td>
</tr>
<tr>
<td>INTEGRITY_METADATA</td>
<td>Triggered to record a metadata integrity verification event run by the kernel.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Explanation</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>INTEGRITY_PCR[^b]</td>
<td>Triggered to record Platform Configuration Register (PCR) invalidation messages.</td>
</tr>
<tr>
<td>INTEGRITY_RULE[^b]</td>
<td>Triggered to record a policy rule.</td>
</tr>
<tr>
<td>INTEGRITY_STATUS[^b]</td>
<td>Triggered to record the status of integrity verification.</td>
</tr>
<tr>
<td>IPC</td>
<td>Triggered to record information about a Inter-Process Communication object referenced by a system call.</td>
</tr>
<tr>
<td>IPC_SET_PERM</td>
<td>Triggered to record information about new values set by an IPC_SET control operation on an IPC object.</td>
</tr>
<tr>
<td>KERN_MODULE</td>
<td>Triggered to record a kernel module name on load or unload.</td>
</tr>
<tr>
<td>KERNEL</td>
<td>Triggered to record the initialization of the Audit system.</td>
</tr>
<tr>
<td>KERNEL_OTHER</td>
<td>Triggered to record information from third-party kernel modules.</td>
</tr>
<tr>
<td>LABEL_LEVEL_CHANGE</td>
<td>Triggered when an object's level label is modified.</td>
</tr>
<tr>
<td>LABEL_OVERRIDE</td>
<td>Triggered when an administrator overrides an object's level label.</td>
</tr>
<tr>
<td>LOGIN</td>
<td>Triggered to record relevant login information when a user log in to access the system.</td>
</tr>
<tr>
<td>MAC_CHECK</td>
<td>Triggered when a user space MAC (Mandatory Access Control) decision is made.</td>
</tr>
<tr>
<td>MAC_CIPSOV4_ADD</td>
<td>Triggered when a Commercial Internet Protocol Security Option (CIPSO) user adds a new Domain of Interpretation (DOI). Adding DOIs is a part of the packet labeling capabilities of the kernel provided by NetLabel.</td>
</tr>
<tr>
<td>MAC_CIPSOV4_DEL</td>
<td>Triggered when a CIPSO user deletes an existing DOI. Adding DOIs is a part of the packet labeling capabilities of the kernel provided by NetLabel.</td>
</tr>
<tr>
<td>MAC_CONFIG_CHANGE</td>
<td>Triggered when an SELinux Boolean value is changed.</td>
</tr>
<tr>
<td>MAC_IPSEC_EVENT</td>
<td>Triggered to record information about an IPSec event, when one is detected, or when the IPSec configuration changes.</td>
</tr>
<tr>
<td>MAC_MAP_ADD</td>
<td>Triggered when a new Linux Security Module (LSM) domain mapping is added. LSM domain mapping is a part of the packet labeling capabilities of the kernel provided by NetLabel.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Explanation</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MAC_MAP_DEL</td>
<td>Triggered when an existing LSM domain mapping is deleted. LSM domain mapping is a part of the packet labeling capabilities of the kernel provided by NetLabel.</td>
</tr>
<tr>
<td>MAC_POLICY_LOAD</td>
<td>Triggered when a SELinux policy file is loaded.</td>
</tr>
<tr>
<td>MAC_STATUS</td>
<td>Triggered when the SELinux mode (enforcing, permissive, off) is changed.</td>
</tr>
<tr>
<td>MAC_UNLBL_ALLOW</td>
<td>Triggered when unlabeled traffic is allowed when using the packet labeling capabilities of the kernel provided by NetLabel.</td>
</tr>
<tr>
<td>MAC_UNLBL_STCADD</td>
<td>Triggered when a static label is added when using the packet labeling capabilities of the kernel provided by NetLabel.</td>
</tr>
<tr>
<td>MAC_UNLBL_STCDEL</td>
<td>Triggered when a static label is deleted when using the packet labeling capabilities of the kernel provided by NetLabel.</td>
</tr>
<tr>
<td>MMAP</td>
<td>Triggered to record a file descriptor and flags of the <code>mmap(2)</code> system call.</td>
</tr>
<tr>
<td>MQ_GETSETATTR</td>
<td>Triggered to record the <code>mq_getattr(3)</code> and <code>mq_setattr(3)</code> message queue attributes.</td>
</tr>
<tr>
<td>MQ_NOTIFY</td>
<td>Triggered to record arguments of the <code>mq_notify(3)</code> system call.</td>
</tr>
<tr>
<td>MQ_OPEN</td>
<td>Triggered to record arguments of the <code>mq_open(3)</code> system call.</td>
</tr>
<tr>
<td>MQ_SENDRECV</td>
<td>Triggered to record arguments of the <code>mq_send(3)</code> and <code>mq_receive(3)</code> system calls.</td>
</tr>
<tr>
<td>NETFILTER_CFG</td>
<td>Triggered when Netfilter chain modifications are detected.</td>
</tr>
<tr>
<td>NETFILTER_PKT</td>
<td>Triggered to record packets traversing Netfilter chains.</td>
</tr>
<tr>
<td>OBJ_PID</td>
<td>Triggered to record information about a process to which a signal is sent.</td>
</tr>
<tr>
<td>PATH</td>
<td>Triggered to record file name path information.</td>
</tr>
<tr>
<td>PROCTITLE</td>
<td>Gives the full command-line that triggered this Audit event, triggered by a system call to the kernel.</td>
</tr>
<tr>
<td>RESP_ACCT_LOCK[c]</td>
<td>Triggered when a user account is locked.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Explanation</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RESP_ACCT_LOCK_TIME</td>
<td>Triggered when a user account is locked for a specified period of time.</td>
</tr>
<tr>
<td>RESP_ACCT_REMOTE</td>
<td>Triggered when a user account is locked from a remote session.</td>
</tr>
<tr>
<td>RESP_ACCT_UNLOCK_TIME</td>
<td>Triggered when a user account is unlocked after a configured period of time.</td>
</tr>
<tr>
<td>RESP_ALERT</td>
<td>Triggered when an alert email is sent.</td>
</tr>
<tr>
<td>RESP_ANOMALY</td>
<td>Triggered when an anomaly was not acted upon.</td>
</tr>
<tr>
<td>RESP_EXEC</td>
<td>Triggered when an intrusion detection program responds to a threat originating from the execution of a program.</td>
</tr>
<tr>
<td>RESP_HALT</td>
<td>Triggered when the system is shut down.</td>
</tr>
<tr>
<td>RESP_KILL_PROC</td>
<td>Triggered when a process is terminated.</td>
</tr>
<tr>
<td>RESP_SEBOOL</td>
<td>Triggered when an SELinux Boolean value is set.</td>
</tr>
<tr>
<td>RESP_SINGLE</td>
<td>Triggered when the system is put into single-user mode.</td>
</tr>
<tr>
<td>RESP_TERM_ACCESS</td>
<td>Triggered when a session is terminated.</td>
</tr>
<tr>
<td>RESP_TERM_LOCK</td>
<td>Triggered when a terminal is locked.</td>
</tr>
<tr>
<td>ROLE_ASSIGN</td>
<td>Triggered when an administrator assigns a user to an SELinux role.</td>
</tr>
<tr>
<td>ROLE_MODIFY</td>
<td>Triggered when an administrator modifies an SELinux role.</td>
</tr>
<tr>
<td>ROLE_REMOVE</td>
<td>Triggered when an administrator removes a user from an SELinux role.</td>
</tr>
<tr>
<td>SECCOMP</td>
<td>Triggered when a SECure COMPuting event is detected.</td>
</tr>
<tr>
<td>SELINUX_ERR</td>
<td>Triggered when an internal SELinux error is detected.</td>
</tr>
<tr>
<td>SERVICE_START</td>
<td>Triggered when a service is started.</td>
</tr>
<tr>
<td>SERVICE_STOP</td>
<td>Triggered when a service is stopped.</td>
</tr>
<tr>
<td>SOCKADDR</td>
<td>Triggered to record a socket address.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Explanation</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SOCKETCALL</td>
<td>Triggered to record arguments of the <code>sys_socketcall</code> system call (used to multiplex many socket-related system calls).</td>
</tr>
<tr>
<td>SYSCALL</td>
<td>Triggered to record a system call to the kernel.</td>
</tr>
<tr>
<td>SYSTEM_BOOT</td>
<td>Triggered when the system is booted up.</td>
</tr>
<tr>
<td>SYSTEM_RUNLEVEL</td>
<td>Triggered when the system's run level is changed.</td>
</tr>
<tr>
<td>SYSTEM_SHUTDOWN</td>
<td>Triggered when the system is shut down.</td>
</tr>
<tr>
<td>TEST</td>
<td>Triggered to record the success value of a test message.</td>
</tr>
<tr>
<td>TRUSTED_APP</td>
<td>The record of this type can be used by third party application that require auditing.</td>
</tr>
<tr>
<td>TTY</td>
<td>Triggered when TTY input was sent to an administrative process.</td>
</tr>
<tr>
<td>USER_ACCT</td>
<td>Triggered when a user-space user authorization attempt is detected.</td>
</tr>
<tr>
<td>USER_AUTH</td>
<td>Triggered when a user-space user authentication attempt is detected.</td>
</tr>
<tr>
<td>USER_AVC</td>
<td>Triggered when a user-space AVC message is generated.</td>
</tr>
<tr>
<td>USER_CHAUTHTOK</td>
<td>Triggered when a user account password or PIN is modified.</td>
</tr>
<tr>
<td>USER_CMD</td>
<td>Triggered when a user-space shell command is executed.</td>
</tr>
<tr>
<td>USER_END</td>
<td>Triggered when a user-space session is terminated.</td>
</tr>
<tr>
<td>USER_ERR</td>
<td>Triggered when a user account state error is detected.</td>
</tr>
<tr>
<td>USER_LABELED_EXPORT</td>
<td>Triggered when an object is exported with an SELinux label.</td>
</tr>
<tr>
<td>USER_LOGIN</td>
<td>Triggered when a user logs in.</td>
</tr>
<tr>
<td>USER_LOGOUT</td>
<td>Triggered when a user logs out.</td>
</tr>
<tr>
<td>USER_MAC_POLICY_LOAD</td>
<td>Triggered when a user-space daemon loads an SELinux policy.</td>
</tr>
<tr>
<td>USER_MGMT</td>
<td>Triggered to record user-space user account attribute modification.</td>
</tr>
<tr>
<td>Event Type</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>USER_ROLE_CHANGE</td>
<td>Triggered when a user's SELinux role is changed.</td>
</tr>
<tr>
<td>USER_SELINUX_ERR</td>
<td>Triggered when a user-space SELinux error is detected.</td>
</tr>
<tr>
<td>USER_START</td>
<td>Triggered when a user-space session is started.</td>
</tr>
<tr>
<td>USER_TTY</td>
<td>Triggered when an explanatory message about TTY input to an administrative process is sent from user-space.</td>
</tr>
<tr>
<td>USER_UNLABELED_EXPORT</td>
<td>Triggered when an object is exported without SELinux label.</td>
</tr>
<tr>
<td>USYS_CONFIG</td>
<td>Triggered when a user-space system configuration change is detected.</td>
</tr>
<tr>
<td>VIRT_CONTROL</td>
<td>Triggered when a virtual machine is started, paused, or stopped.</td>
</tr>
<tr>
<td>VIRT_MACHINE_ID</td>
<td>Triggered to record the binding of a label to a virtual machine.</td>
</tr>
<tr>
<td>VIRT_RESOURCE</td>
<td>Triggered to record resource assignment of a virtual machine.</td>
</tr>
</tbody>
</table>

[a] All Audit event types prepended with **ANOM** are intended to be processed by an intrusion detection program.

[b] This event type is related to the Integrity Measurement Architecture (IMA), which functions best with a Trusted Platform Module (TPM) chip.

[c] All Audit event types prepended with **RESP** are intended responses of an intrusion detection system in case it detects malicious activity on the system.
APPENDIX C. REVISION HISTORY

Revision 1-30  Thu Jul 27 2017  Mirek Jahoda
Document version for 7.4 GA publication.

Revision 1-24  Mon Feb 6 2017  Mirek Jahoda
Async release with misc. updates, especially in the firewalld section.

Revision 1-23  Tue Nov 1 2016  Mirek Jahoda
Version for 7.3 GA publication.

Revision 1-19  Mon Jul 18 2016  Mirek Jahoda
The Smart Cards section added.

Revision 1-18  Mon Jun 27 2016  Mirek Jahoda
The OpenSCAP-daemon and Atomic Scan section added.

Revision 1-17  Fri Jun 3 2016  Mirek Jahoda
Async release with misc. updates.

Revision 1-16  Tue Jan 5 2016  Robert Krátký
Post 7.2 GA fixes.

Revision 1-15  Tue Nov 10 2015  Robert Krátký
Version for 7.2 GA release.

Revision 1-14.18  Mon Nov 09 2015  Robert Krátký
Async release with misc. updates.

Revision 1-14.17  Wed Feb 18 2015  Robert Krátký
Version for 7.1 GA release.

Revision 1-14.15  Fri Dec 06 2014  Robert Krátký
Update to sort order on the Red Hat Customer Portal.

Revision 1-14.13  Thu Nov 27 2014  Robert Krátký
Updates reflecting the POODLE vuln.

Revision 1-14.12  Tue Jun 03 2014  Tomáš Čapek
Version for 7.0 GA release.